TLR: A Traffic-Light-Based Intelligent Routing Strategy for NGEO Satellite IP Networks

Guanghua Song, Mengyuan Chao, Bowei Yang, and Yao Zheng

***Abstract*—We present TLR, a traffic-light-based intelligent routing strategy for NGEO satellite IP networks. In TLR, a set of traffic lights are used to indicate the congestion status at both the current node and the next node. When a packet travels along a pre-calculated route to the destination, it may adjust the route dynamically, according to the real-time color of traffic lights at each intermediate node. Through the combination of preliminary planning and real-time adjustment, each packet can eventually get an approximately optimal transmission path. The multi-path routing mechanism in TLR can help achieve a good distribution of traffics when the network traffic increases. The Public Waiting Queue scheme in TLR can fully utilize free spaces of the buffer queues and lower the packet drop rate.**

While the concept of TLR has many advantages, it may result in endless-loop of routing. To eliminate this phenomenon, a defense scheme is incorporated in the design of TLR. A set of simulations are conducted using the Network Simulator (version 2) to verify the good performance of TLR, in terms of lower packet drop rate, better distribution of traffics and higher throughput, over the entire satellite constellation.

***Index Terms*—NGEO satellite network, traffic light, intelligent routing, packet drop rate, load balance.**

I. Introduction

NON-geostationary (NGEO) satellite networks, with their inherent multicast capability and global coverage potential in comparison to terrestrial networks, and their advantages of offering services with lower latency and terminal power requirements over geostationary (GEO) satellites, have become an attractive infrastructure to accommodate the burgeoning communication demands in current networks. Major countries in the world are paying much attention to this field, regarding NGEO satellite networks as an indispensable part of the Next Generation Networks (NGN)[1].

In order to efficiently transmit serivce data in NGEO satellite networks, an effective routing strategy is essential. However, due to periodic rotation of the satellites around the Earth, NGEO satellite networks have some distinct characteristics, such as dynamic and predictable topologies, frequent link switching and interruption. These characteristics, different from those of traditional terrestrial networks, bring about a lot of special difficulties for route planning. Many network specialists and communication researchers have thus proposed some routing schemes for NGEO satellite networks and made this issue a subject of extensive research since the late 1990s[2].

Previously, most proposed routing strategies focus on finding a transmission route that has the shortest end-to-end propagation delay[3, 4]. These strategies are simple to operate, and can get rather good results if the network traffic is not heavy. However, as the communication traffic increases, they begin to show two defects: the packet drop rate at network layer becomes abnormally high, and the cumulative queuing delay during transmission gets non-ignorablely large. We consider that it is the neglect of congestion that should be blamed. As we know, most world population lives around the equator or in middle-latitude regions, so the communication demands there are much larger than those from other areas. This phenomenon directly leads to an unbalanced traffic distribution over the whole constellation: some satellites are congested while others remain underutilized[2]. When the communication traffic becomes heavier, if only the propagation delay is taken into account as the routing metric, pre-scheduled routes will inevitably contain some heavily congested nodes. If packets are continually sent to those congested nodes, high packet drop rate and cumulative queuing delay are inevitable.

To cope with issues mentioned above, researchers begin to take the expected queuing delay and congestion status into account when computing the optimal route for packet transmission[5–9]. The multi-path routing mechanism is also introduced to better utilize free Inter Satellite Links(ISLs) and realize load balance of the entire constellation[5, 6]. Based on this background, we propose a traffic-light-based intelligent routing strategy (TLR), hoping to relieve the situation of long queuing delay, high packet loss rate and unbalanced traffic distribution in NGEO satellite networks.

The remainder of this paper is organized as follows. Section II presents a detailed survey on the existing routing protocols for NGEO satellite networks. The key definitions and mechanisms involved in TLR are described in Section III. Section IV gives a detailed description on how TLR works. The endless-loop avoidance mechanism is discussed as well. Section V provides a simplified model for our scheme, and based on this model, we analyse the end-to-end delay, stability and routing convergence time of our scheme. In Section VI, a series of experimental results are given out and the performance of TLR is evaluated and compared with other algorithms. The paper concludes in Section VII with a summary recapping the main ideas and advantages of the proposed TLR strategy.

II. Related Work

Due to the dynamic topology characteristic of NGEO satellite networks, traditional terrestrial Internet routing strategies,

Manuscript received January 6, 2013; revised July 17 and December 2, 2013; accepted February 12, 2014. The associate editor coordinating the review of this paper and approving it for publication was A. Abouzeid.

The authors are with the School of Aeronautics and Astronautics, Zhejiang University, Hangzhou, 310027, P.R.China (e-mail: {ghsong, chaomengyuan2011, boweiy, yao.zheng}@zju.edu.cn). B. Yang is the corresponding author.

Digital Object Identifier 10.1109/TWC.2014.041014.130040

such as OSPF or RIP, cannot be directly applied to them. Researchers hence proposed lots of specific routing schemes that could deal with the routing complexity of NGEO satellite networks.

Generally, according to whether the routing schemes adopt periodicity-based or on-board computation, we could classify them into two categories[2]. Schemes of the former category make full use of the periodic and predictable variations of constellation topology and divide the system period into several slots. Routing table for each slot is calculated on ground and stored onboard in advance. When the topology changes, corresponding routing information will be retrieved to meet the routing demands. The advantage of this kind of schemes is their simplicity and easy operability. And the major drawbacks are their large storage requirements, weak fault tolerance, and poor adaptive capabilities. Among all these schemes, two concepts called Dynamic Virtual Topology Routing(DVTR)[3] and Virtual Node(VN)[10] deserve to be mentioned. In DVTR, the system period is divided into a series of time intervals. On-off operations of ISLs are supposed to be performed only at the beginning of each interval and the whole topology keeps unchanged during each interval. Under such assumptions, the complex dynamic topology is transformed into a group of simple static topologies and traditional Dijkstra Shortest-Path(DSP) algorithm could be utilized. In VN-based schemes, virtual nodes are supposed to be set above the surface of the Earth to represent certain physical satellites. A virtual node and a physical satellite have a one to one correspondence at any time and such correspondence will not change until a physical satellite flies out of and another flies into the coverage of a VN. The VN keeps state information, such as routing table entries or channel allocation situation for the physical satellite. And when handoff happens, the state information will be transferred from the former physical satellite to the latter. In this way, rotating physical satellites can be converted into fixed virtual nodes, and the dynamic topology is also transformed into an accordingly static one, topology changes thus can be hidden behind. Many latter schemes are more or less based on the idea of above two schemes.

Schemes of the latter category calculate the routing tables onboard according to the collected near-real-time state information like satellite state, link load and traffic condition. Unlike above periodicity-based schemes, most of onboard routing schemes exhibit strong adaptive capabilities. However, they impose significant challenges for the space devices as well, especially in terms of the required computational and processing capability. These years, a lot of onboard routing schemes have been proposed in the literature. For some examples, Jianjun $et\ al.$ in [11] proposed an onboard routing scheme which is based on a distributed hierarchical link state update mechanism: The defined plane speakers at first collect state information of all the links within the plane, and then exchange obtained information with other plane speakers to build a routing information base (RIB) for the network. Finally, the converged RIB can be distributed to all satellites through the intra-plane and inter-plane ISLs. Each satellite then can calculate the routing table based on this RIB for themselves. Another onboard routing scheme which proposed a similar state information collecting mechanism is involved in [12], and the only difference is that this scheme adopts a multi-layer topology architecture instead of defining a plane speaker. [13] is another example which proposes an on-demand computing and caching centralized routing strategy. The strategy is designed for satellite network topology dynamic grouping, its route calculation is divided into three phases: direction estimation, direction enhancement, and congestion avoidance. The strategy provides significant advantages of high efficiency, low complexity, flexible configuration and great potential in scalability.

Previously, in the context of satellite networks, since the traffic load is not so heavy, numerous researchers presume that the propagation delay is the dominating factor in the communication delay[14]. Therefore, they focused on developing routing mechanisms that find minimum propagation delay paths with minimal hop count for communication. However, in recent years, with the traffic load increasing in the NGEO satellite network, queuing delay has become an important factor that could not be ignored any more. The avoidance from congested node and traffic load balance are also taken into account when designing a routing scheme. The CEMR algorithm[5] periodically collects the expected queuing delay at each hop through an orbit speaker scheme. When it computes the routes, it takes both the expected queuing delay and propagation delay into account. However, since the expected queuing delays are collected in advance, they may not completely conform to the actual situation. So, there may be some potential congested nodes in the calculated routes, which will cause serious packet loss if packets are still sent there. Therefore, in order to decrease the packet loss rate and get better load balance, ELB[6] takes the state of next hops into account when it makes a choice of the best next hop. However, it still fails to anatomize some special cases, in which even if the state of the next hop shows “free”, the packet should not be sent there, for some packets may be dropped even before they are sent out if the current hop is overloaded. In [7], a distributed agent-based load balancing routing scheme for low-earth orbit (LEO) satellite networks is presented. Two kinds of agents are used there. Mobile agents migrate autonomously to explore the path connecting source and destination, to gather ISL cost, identifier and latitude of visited satellites. Meanwhile, stationary agents employ exponential forgetting function to estimate ISL queueing delay, calculate ISL cost using the sum of propagation and queueing delays; evaluate path cost considering satellite geographical position as well as ISL cost, finally update routing items. The scheme is shown to achieve good load balancing, and can especially decrease packet loss ratio efficiently, guarantee better throughput and end-to-end delay bound in case of high traffic load. However, it needs the favor of many agents, which means a lot of cost and complexity. In [8], a load balancing mechanism based on a new congestion-prediction method is devised. The author thinks that, since it is possible to know which LEO satellite is going toward the congested area in mesh constellations, the satellite in the congested area could preliminarily informs the neighboring satellite following itself with the coordinates of the congested area. The congested area then can be defined as a circle with its center at the informed coordinate. The radius of the circular field could be determined

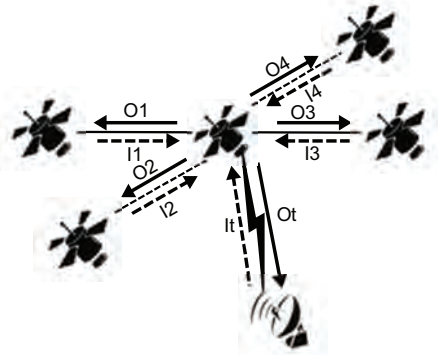


Fig. 1: Satellite linking relationships.

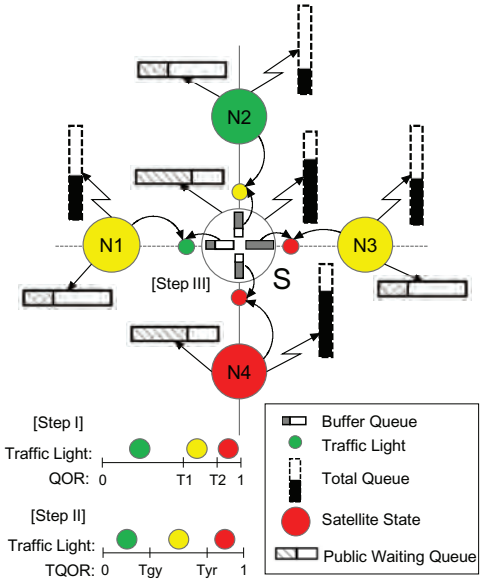


Fig. 2: Traffic light color setting scheme.

according to the configuration of the satellite constellation and the LEO satellites coverage area. By exchanging such information, the satellite approaching the congested area can predict network congestion and immediately begin traffic detouring upon entering the area without awaiting detection of actual network congestion events. In [15], to effectively resolve the problem of traffic congestion, the author proposes a new Multi-Layered Satellite Networks(MLSNs) model by envisioning a method to distribute the flow of packets between the two layers of the considered MLSNs for minimizing the packet delivery delay of the network. Moreover, they analyze the effect of the method on the packet delivery delay by considering propagation and queuing latencies.

In this paper, we aim at developing a routing strategy which considers both the expected and real-time queuing delays, concerns both current and next hop congestion, and adjusts the pre-calculated route according to the real-time situation, aiming at reducing packet drops, lowering queuing delay and distributing traffic burdens fairly.

III. Key Definitions and Mechanisms in TLR

This section describes some key definitions and mechanisms that make the TLR strategy work. The envisioned multi-hop NGEO satellite constellation is based on an Iridium-like backbone which consists of $S = M * N$ satellites, where M represents the number of orbits and N represents the number of satellites in each orbit. In this constellation, a satellite can set up ISLs with four neighbors at most, with two in its own orbit and the other two in the neighboring orbits. It can also establish several ground-satellite links (GSLs) with the terminals in its coverage. Fig. 1 shows the linking relationships among them, where all terminals are represented by a single ground antenna for simplicity. In the satellite, a buffer queue is allocated for each ISL to temporarily store packets to be sent out through this ISL to the next hop. A traffic light is also maintained for each ISL to indicate the traffic condition at this direction. A description about how to set the color of traffic lights is given progressively in section A. It starts from considering the congestion status of a buffer queue at the current hop. Then, the congestion status of the next hop is taken into account, for it will directly influence the packet drops if bad congestion happens there. After that, the congestion status of the current and next hops will be combined together to set the color of traffic lights, so as to provide more accurate information about surrounding traffic situations. Upon that, a periodic checking and updating mechanism will be introduced.

In addition, a public waiting queue will be constructed at each satellite, aiming at alleviating packet drops when traffic lights at all candidate directions show "RED". We construct the public waiting queue by borrowing free space from queues at each direction. Since the total memory space at a satellite is constant, constructing the public waiting queue in such a way actually means: When traffic at one direction is too heavy, it will borrow some space from other buffer queues to store packets. This strategy is quite applicable to the NGEO satellite networks, where congestion normally happens due to heavy traffic at one or two directions.

A. Setup and Update of Traffic Light Color

1) Step I, Consider the Current Hop: In TLR, when a satellite receives a packet, it first searches the routing table for candidates of next hop. Then, the packet is inserted into a buffer queue at one direction, waiting to be sent out. If the traffic at one direction is too heavy, accumulated packets will soon fill up the whole queue. Packets will then be dropped if the satellite continues inserting packets there. Therefore, to clearly indicate the state of a buffer queue and to effectively reduce unnecessary packet loss, a traffic light is set up for each direction, just as shown in Fig. 2. We denote $Q(n)$ as the buffer queue of current satellite at direction n. We define the queue occupancy rate (QOR) as the rate of number of packets in the queue to the length of the queue and denote QOR_n as the queue occupancy rate of $Q(n)$ at current satellite. When QOR is below the pre-set threshold T_1, the traffic light will be set "GREEN", which means the queue is quite free and packets are welcome to be inserted there. As QOR grows beyond T_1 but still below the other threshold T_2, the queue is considered to be relatively congested. Hence, the traffic light is set "YELLOW" which means new coming packets would further aggravate the congestion status. Therefore, a better way is to spare some packets to buffer queues at other directions,

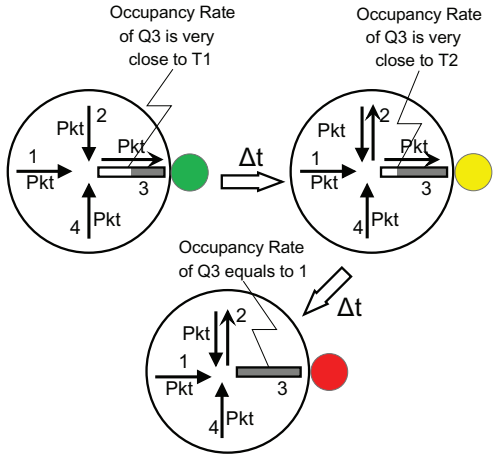


Fig. 3: Extreme situations of setting T_1 and T_2.

where packets can also be sent to the destination. When QOR continues growing to T_2, the traffic light turns "RED", which means "Stop inserting packets here, or they will be dropped".

Obviously, in order to gain the expected effects of aforementioned mechanisms, the setting of thresholds T_1 and T_2 is crucial. As an example shown in Fig. 3, T_1 and T_2 should be able to cope with the extreme situation below: When the satellite checks $Q(3)$, QOR_3 is very close to but still below T_1, so the traffic light remains "GREEN". Just after the checking, QOR_3 grows beyond T_1, but the satellite has to wait for Δt (the check interval) to recognize the change. To ensure that QOR_3 will not exceed T_2 during this Δt period, T_1 and T_2 should meet:

$$(T_2 - T_1) \cdot L \cdot APS \geq (I - O)_{max} \cdot \Delta t, \tag{1}$$

where L represents the size of buffer queue, APS represents the average packet size, and $(I - O)_{max}$ represents the maximum difference between input and ouput traffic rates of $Q(3)$ during this period.

Similarly, if QOR_3 exceeds T_2 just after a checking, the satellite also has to wait for Δt to realize the change. To avoid that a packet is dropped due to overflow of the buffer, T_2 should meet:

$$(1 - T_2) \cdot L \cdot APS \geq (I' - O')_{max} \cdot \Delta t. \tag{2}$$

From (1) and (2), we get

$$T_1 \leq 1 - \frac{[(I - O)_{max} + (I' - O')_{max}] \cdot \Delta t}{L \cdot APS}, \tag{3}$$

$$T_2 \leq 1 - \frac{(I' - O')_{max} \cdot \Delta t}{L \cdot APS}. \tag{4}$$

Considering the extreme situation, we set

$$T_1 = 1 - \frac{[(I - O)_{max} + (I' - O')_{max}] \cdot \Delta t}{L \cdot APS}, \tag{5}$$

$$T_2 = 1 - \frac{(I' - O')_{max} \cdot \Delta t}{L \cdot APS}. \tag{6}$$

2) Step II, Consider the Next Hop: The consideration above can help avoid the situation that packets are irresponsibly inserted into a buffer queue without considering the congestion there. A low packet drop rate and queuing delay can thus be achieved to some extent. However, if we want to achieve lower packet drop rate and queuing delay, the status of the next hop should never be ignored [6]. In Fig. 2, a satellite node checks its total queue occupancy rate (TQOR, considering all the buffer queues as a whole) periodically. It marks the state of itself as "GREEN", "YELLOW" or "RED", if its TQOR locates in [0,T_{gy}), [T_{gy},T_{yr}) or [T_{yr},1], respectively. The initial state of every satellite is set to "GREEN". If the value of TQOR increases to the threshold T_{gy}, the satellite state will turn to "YELLOW", and if TQOR increases to the threshold T_{yr}, the satellite state will turn to "RED". When the satellite state changes with the increase of TQOR, it will notice its neighbors, which finally may influence the color of relevant traffic lights. Thereby, every satellite can get to know the near-real-time congestion status of its neighbors.

Below, we will analyze how to set the value of T_{gy} and T_{yr}. Without loss of generality, we assume that, each satellite connects to k neighbors and stores p candidate next hops for any destination in the routing table. A packet will not be dropped as long as one candidate is available. To ensure the feasibility of our algorithms, below, we will consider some extreme scenarios.

Firstly, we define $Q(d_1, d_2, ..., d_k)$ as the integration of buffer queues at directions d_1,d_2,...,d_k. In general, if a packet is received by a satellite from one direction d_i, it will not be sent back from this direction right away. Therefore, we can separate $Q(d_i)$ from $Q(d_1, d_2, ..., d_k)$, and treat the rest buffer queues, namely $Q(d_1, ..., d_{i-1}, d_{i+1}, ..., d_k)$ as a whole. According to the theory of probability, the occupancy rate of both $Q(d_1, ..., d_{i-1}, d_{i+1}, ..., d_k)$ and $Q(d_i)$ equals to that of $Q(d_1, d_2, ..., d_k)$. So, if the packets occupying $Q(d_1, ..., d_{i-1}, d_{i+1}, ..., d_k)$ all concentrates at one direction, and the queue at this direction is already full, then any packets continued to be sent there will be dropped. At this time, the occupancy rate of both $Q(d_1, ..., d_{i-1}, d_{i+1}, ..., d_k)$ and $Q(d_i)$ equals $\frac{1}{k-1}$. Therefore, to ensure the satellite's absolutely passable characteristic (i.e."GREEN" state, which means any direction can accept packets), T_{gy} should meet:

$$T_{gy} \leq \frac{1}{k-1}. \tag{7}$$

Similarly, if packets occupying $Q(d_1, ..., d_{i-1}, d_{i+1}, ..., d_k)$ all concentrate at p directions, and queues at these p directions are all full, then some packets would have to be dropped if these p directions happen to be the p candidate next hop directions for a certain destination. And the occupancy rate of both $Q(d_1, ..., d_{i-1}, d_{i+1}, ..., d_k)$ and $Q(d_i)$ now equals $\frac{p}{k-1}$. Therefore, to ensure the satellite's passable characteristic (i.e."YELLOW" state, which means at least one direction can accept packets), T_{yr} should meet:

$$T_{yr} \leq \frac{p}{k-1}. \tag{8}$$

And besides, since T_{yr} should be greater than T_{gy}, T_{yr} should actually meet:

$$\frac{1}{k-1} < T_{yr} \leq \frac{p}{k-1}. \tag{9}$$

In our model(Iridium model), a satellite could directly connect to 4 neighbors at most. And when a satellite enters polar region, it will disconnect its inter-orbit ISLs(only have 2 neighbors at that time). In this context, 2 candidates is the best choice.

TABLE I: Traffic Light Color Setting Rules

QOR	TQOR	Final Traffic Light Color
[0,T_1)	[0,T_{gy})	GREEN
	[T_{gy},T_{yr})	YELLOW
	[T_{yr},1]	RED
[T_1,T_2)	[0,T_{gy})	YELLOW
	[T_{gy},T_{yr})	YELLOW
	[T_{yr},1]	RED
[T_2,1]	Any	RED

Except for the above extreme situations, we should also prevent the phenomenons below: (1) The state of a satellite has turned from "YELLOW" to "RED" before the notification "YELLOW" arrives at its neighbors; (2) The state of a satellite has turned from "RED" to "Absolutely Overfilled" before the notification "RED" arrives at its neighbors. Therefore, similar as in Step I, T_{yr} and T_{gy} should meet:

$$T_{gy} \leq 1 - \frac{[(I_g - O_g)_{max} + (I_y - O_y)_{max}] \cdot (\Delta t + d)}{N \cdot L \cdot APS}, \quad (10)$$

$$T_{yr} \leq 1 - \frac{(I_y - O_y)_{max} \cdot (\Delta t + d)}{N \cdot L \cdot APS}, \quad (11)$$

where I_g and O_g represent the total input and output of a satellite respectively, when the notification "YELLOW" is on the way; whereas I_y and O_y represent the total input and output of a satellite respectively, when the notification "RED" is on the way. N stands for the buffer queue number at a satellite. Δt represents the checking interval and d represents the average one-hop propagation time.

Synthesizing (7) to (11), we set the value of T_{gy} and T_{yr} using following formulas:

$$T_{gy} = Min(\frac{1}{k-1}, 1 - \frac{[(I_g - O_g)_{max} + (I_y - O_y)_{max}] \cdot (\Delta t + d)}{N \cdot L \cdot APS}), \quad (12)$$

$$T_{yr} = Min(\frac{p}{k-1}, 1 - \frac{(I_y - O_y)_{max} \cdot (\Delta t + d)}{N \cdot L \cdot APS}). \quad (13)$$

3) Step III: Consider Both the Current Hop and the Next Hop: At this step, we combine QOR of the buffer queue at the current hop with TQOR at the next hop, and determine the final traffic light color at each direction, as shown in Fig.2. The setting rules are listed in Tab. I.

In TLR, in order to decrease unnecessary additional transmission overheads, a satellite notifies its neighbors only when its state changes. That is to say, only when TQOR varies from one interval to another, will the satellite send a notification to its neighbors.

B. Public Waiting Queue

As mentioned above, in TLR, a satellite node stores two candidates of next hop for any destination in its routing table. The satellite chooses a next hop according to the color of relevant traffic lights. However, if both candidates are not able to accept packets at a moment, packets would have to be discarded if there is no special mechanism.

To cope with this issue, we provide the "Public Waiting Queue". It is based on a common situation in satellite networks: the congestion at a node is often caused by the heavy traffic at one or two certain directions. That is, buffer queues at certain directions may be overfilled while others remain free. To improve the utilization of the total queue space, we spare some part from each buffer queue to construct a public waiting queue. When all directions for a certain destination are congested, newly arriving packets can be inserted into the public waiting queue temporarily. In this way, a lower packet drop rate can be achieved. Then, how long should the public waiting queue be? We assume the total queue space in the satellite as TS. When there is no such a public waiting queue, the length of each buffer queue is $\frac{TS}{k}$. Now, we will spare the same portion from each queue to construct the public waiting queue. We assume event A: the state of the queue at current hop is RED; event B: the state of the next hop is RED; event A and event B are independent from each other. According to our rule, the light at one direction will be RED when either A or B happens, and the probability

$$P_{RED} = P(A + B) = P(A) + P(B) - P(A) \cdot P(B). \quad (14)$$

For a packet arriving from a certain direction, it has p next hop candidates. So, only when lights at all these p directions show RED, will the packet be inserted into the public queue, the probability

$$P(ToPubQue) = {P_{RED}}^p. \quad (15)$$

Otherwise, this packet will be inserted into an ordinary queue, the probability

$$P(ToOrdQue) = 1 - P(ToPubQue) = 1 - {P_{RED}}^p. \quad (16)$$

Therefore, for these p queues, the ratio of the part spared out and the part left should be:

$$\frac{L_{spare}}{L_{left}} = \frac{P(ToPubQue)}{P(ToOrdQue)} = \frac{{P_{RED}}^p}{1 - {P_{RED}}^p}. \quad (17)$$

To ensure fairness, for each queue, the ratio of the part spared out and the part left also equals this. Therefore,the length of the ordinary queue:

$$L_{ord} = \frac{TS}{k} \cdot (1 - {P_{RED}}^p). \quad (18)$$

The length of the public waiting queue:

$$L_{public} = k \cdot \frac{TS}{k} \cdot {P_{RED}}^p = TS \cdot {P_{RED}}^p. \quad (19)$$

Although a public waiting queue can be constructed to temporarily store those "passless" packets, it could not unrestrainedly accept them. In fact, the public waiting queue itself has a limitation of size. To avoid continued growing number of packets, a periodic checking mechanism is introduced. When a packet is inserted into the public waiting queue, a TTW (Time to Wait) field is allocated to it. Each time we check the public waiting queue, we will traverse packets in the queue to see if some could be sent out. If the traffic light at the expected outgoing direction shows "GREEN" or "YELLOW", the packet will be taken out and sent there. But if relevant traffic light remains "RED", we minus its TTW value by 1. When the TTW value of a packet is decreased to 0, it will be discarded from the waiting queue. Details about this algorithm are given in Algorithm 2 in the Appendix.

IV. Detailed Algorithm Description

A. Global Routing Pre-Calculation

The first step of the TLR strategy is to calculate two best routes for each source and destination pair periodically. Since the variation of satellite network topology is highly periodic

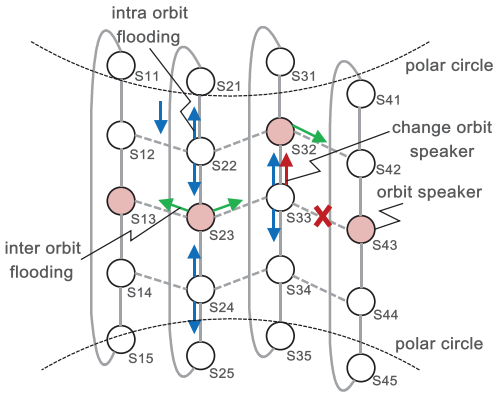


Fig. 4: Advanced orbit speaker mechanism.

and predictable, propagation delay between each two satellite node is easy to get. In the past, since on-board processing delay and queueing delay are rather short, propagation delay is considerd as the unique routing cost metric. However, as traffic in satellite networks becomes heavier, queuing delay, as a part of the end-to-end delay, could not be ignored anymore. In CEMR [5], B. Jianjun *et al*. propose an "Orbit Speaker" scheme, which can collect and exchange expected queuing delay at each hop before conducting global routing computing. We adopt a similar scheme here.

As shown in Fig. 4, each orbit has a speaker to collect the state information of intra orbit satellites and to change the collected information with orbit speakers at other orbits. Non-orbit speaker could only broadcast its state information to its intra orbit satellites; only the orbit speaker could broadcast state information to its inter orbit satellites. In our scheme, in order to reduce delay, we request that the orbit speakers at adjacent orbits had better to have a direct link. When the link breaks, the orbit speakers at the ends of the link will relinquish the role to their intra orbit neighbors. And at the same time, they will notify other orbits speakers about the change and let them realize similar role shift. In the end, all these orbit speakers could again be connected directly in a line. When an orbit speaker receives the state information from a neighboring speaker, except for delivering it to the neighboring speaker on the other side, it will also broadcast the information in its own orbit. In this way, all satellites could get to know the global state information in the end.

By the way, since the motion of satellites is deterministic, the position of satellites and their connectivity can be computed in advance according to the parameters of selected constellation. Consequently, only un-deterministic parameters such as the expected average queuing delays, accidental link breaks and so on, need to be distributed through the network, thus minimizing the signaling load. As long as the global state information is acquired, we could construct a graph $G(V, E)$, where V represents the set of satellites and E represents the set of "propagation delay+ expected queuing delay" between each pair of directly linked nodes. Then, the shortest-path algorithm is applied to identify two candidates of next hop from one node to any destination, and relevant next hop information will be stored in the routing table of each satellite.

B. Real-Time Adjustment

The above scheme seems considerate, but it still ignores an important fact: the expected queuing delay is often not accurate, especially when the traffic varies dramatically. In fact, pre-calculation can only provide a rough direction; further adjustment is needed to help make the best decision.

In TLR, we consider not only the expected queuing delay, but also the real-time queuing delay at each hop, as the packet traverses along the pre-calculated route. When a packet arrives at a satellite, it first searches the routing table for candidates of next hop according to its destination field. The next hop of the best route (BR.nxtHop) will be the first choice. If the traffic light at this direction shows "GREEN" (which means the real-time queuing delay is short), the packet will be directly sent there. However, if the traffic light shows "YELLOW" or "RED" (which means the real-time queuing delay is long or very long), the next hop of the second best route (SBR.nxtHop) will be chosen to share the transmission task. In the case that the traffic lights at both directions show "RED", the packet will be inserted into the public waiting queue temporarily. Detailed rules are given in Tab. II.

C. Endless-loop Avoidance

For a locally optimum routing algorithm, an inevitable problem is endless-loop: A packet starts from a node, travels through several nodes and finally goes back to the starting node to form a loop. Then, the "starting node" chooses the same next hop as before. So the packet may travel along the loop repeatedly. To avoid endless-loop, we propose a method which requires the packet to record the passed hops in its head as it travels in the network. When a packet arrives at a satellite, the ID of the satellite is inserted into the head of the packet. Then, it checks if the candidates of next hop have appeared in its head. If one appears, it will be ruled out from the candidates. If both have appeared in the head, the satellite will find the first position where the current satellite ID appears, and the packet will be sent back to the node whose ID appears just before the found position.

Fig. 5 gives an example to illustrate how this mechanism works. A packet starts from node "1" and arrives at node "D" after going through several nodes. Then, it finds node "F" has already appeared in its head. So, it chooses another next hop "E". After reaching there, it finds both "A" and "B" have existed in its head, so it has to find the first position where "E" appears in its head. Then, it goes back to node "D" that appears before "E". After arriving at "D" again, it finds both "E"and "F" have already been in its head, so it has to find the first position where "D" appears. Then, it goes to node "C" that appears just before "D". From "C", this packet chooses "G" as its new next hop and finally arrives at the destination node "2". From this example, we can find, by recording the passed nodes and backtracking to a former node when both next hops have been passed, we can effectively avoid Endless-loop. Detailed operations combined with TLR are given in Algorithm 1 in the Appendix and a rough proof of the Endless-loop-Free characteristic is provided.

Of course, when the historical list of visited nodes is included in the head of a packet, it will unavoidably cost

TABLE II: Detailed Rules of TLR

BR.nxtHop.TrafficLightColor	SBR.nxtHop.TrafficLightColor	Where to Go
GREEN	Any	BR.nxtHop
YELLOW	GREEN/YELLOW	Half BR.nxtHop, half SBR.nxtHop
	RED	BR.nxtHop
RED	GREEN/YELLOW	SBR.nxtHop
	RED	Public Waiting Queue

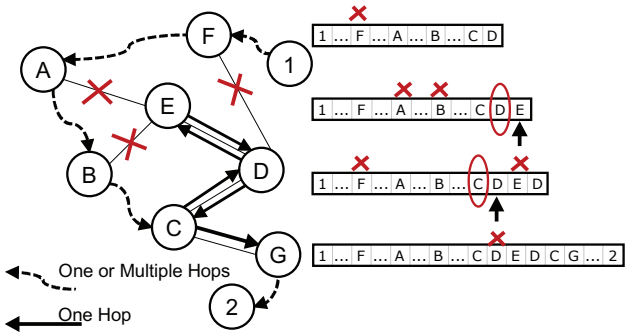


Fig. 5: Endless-loop-free mechanism.

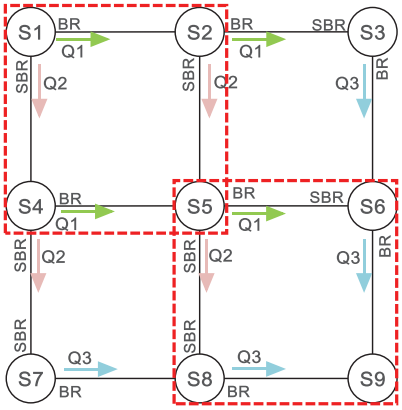


Fig. 6: A simplified model for the TLR scheme.

extra resource. In our model, the number of satellite nodes is very few (at present, the number is 66). Therefore, we could uniformly allocate a unique number (one Byte) to identify each satellite node. And in a satellite network with $66(6*11)$ nodes, a packet has to go through at most $11/2+5=10$ nodes to reach the destination, if we use the shortest path algorithm. In our algorithm, we allow a packet to take some detour to avoid long queuing delay at some heavy-loaded nodes. However, it does not mean that, we will let the path grow without restriction, because more intermediate nodes means more propagation delay. When the whole propagation delay goes beyond a certain level, even if the packet has arrived at the destination successfully, it has little meaning. So, we set an initial value TTL ($20=10*2$) for every packet. Once a packet goes through a node, the value of TTL minus 1. When the TTL value decreases to 0, the packet will be dropped. So, the length of the path will be at most 20 Bytes. It is not a very big cost because the IP-V4 head length could be as long as 60 Bytes.

V. Modeling and Analysis

A. End-to-end Delay in TLR

To evaluate the performance of our TLR scheme in theory, we propose a simplified model in Fig. 6 below. We assume S_1 as the source satellite and S_9 as the destination satellite. We will calculate the expectation of end-to-end delay from S_1 to S_9.

Firstly, we give some illustration to the symbols we use in the calculation: A_1,A_2,A_3 — The probability that the current hop is Green, Yellow, and Red; B_1,B_2,B_3 — The probability that the next hop is Green, Yellow, and Red; P_G,P_Y,P_R — The probability that the traffic light is Green,Yellow,Red; Q_1 — The probability to go to the best next hop; Q_2 — The probability to go to the second best next hop; Q_4 — The probability to go to the public waiting queue; Q_3 — The probability to go to the next hop when there is only one choice; Q_5 — The probability to go to the public waiting queue when there is only one choice; L_1,L_2,L_3 — the average queue length when the current hop is Green,Yellow,Red; $\bar{X_1}$ — The expectation of queue length when choosing the best route; $\bar{X_2}$ — The expectation of queue length when choosing the second best route; $\bar{X_3}$ — The expectation of queue length when there is only one choice; ΔT_1 — The waiting time at the public waiting queue; ΔT_2 —The waiting time at the public waiting queue when there is only one choice; APS — the average packet size; BW — band width of a link; ΔT — checking interval for the public waiting queue.

Among them,

$$P_G = A_1 \cdot B_1, \tag{20}$$

$$P_Y = A_1 \cdot B_2 + A_2 \cdot (B_1 + B_2), \tag{21}$$

$$P_R = A_3 + B_3 - A_3 \cdot B_3, \tag{22}$$

$$Q_1 = P_G + \frac{P_Y \cdot (P_G + P_Y)}{2} + P_Y \cdot P_R, \tag{23}$$

$$Q_2 = \frac{P_Y \cdot (P_G + P_Y)}{2} + P_R \cdot (P_G + P_Y), \tag{24}$$

$$Q_4 = P_R \cdot P_R, \tag{25}$$

$$Q_3 = P_Y + P_G, \tag{26}$$

$$Q_5 = P_R, \tag{27}$$

$$\begin{aligned}\bar{X_1} = [A_1 \cdot B_1 + \frac{A_1 \cdot B_2 \cdot (P_G + P_Y)}{2} + A_1 \cdot B_2 \cdot P_R]/Q_1 \cdot L_1 \\ +[\frac{A_2 \cdot (B_1 + B_2) \cdot (P_G + P_Y)}{2} + A_2 \cdot (B_1 + B_2) \cdot P_R]/Q_1 \cdot L_2,\end{aligned} \tag{28}$$

$$\begin{aligned}\bar{X_2} = [\frac{P_Y \cdot (A_1 \cdot B_1 + A_1 \cdot B_2)}{2} + P_R \cdot (A_1 \cdot B_1 + A_1 \cdot B_2)]/Q_2 \\ \cdot L_1 + [\frac{P_Y \cdot A_2 \cdot (B_1 + B_2)}{2} + P_R \cdot A_2 \cdot (B_1 + B_2)]/Q_2 \cdot L_2,\end{aligned} \tag{29}$$

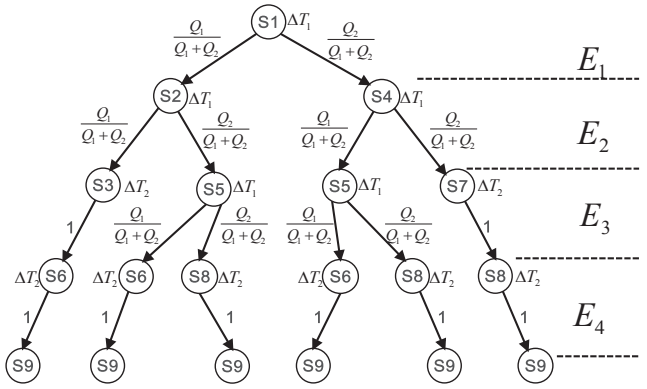


Fig. 7: Transmitting route from S_1 to S_9.

$$\bar{X_3} = (A_1 \cdot B_1 + A_1 \cdot B_2)/Q_3 \cdot L_1 + A_2 \cdot (B_1 + B_2)/Q_3 \cdot L_2, \quad (30)$$

$$\Delta T_1 = Q_4 \cdot \Delta T + Q_4^2 \cdot \Delta T + ... + Q_4^n \cdot \Delta T \approx \frac{Q_4}{1 - Q_4} \cdot \Delta T, \quad (31)$$

$$\Delta T_2 = Q_5 \cdot \Delta T + Q_5^2 \cdot \Delta T + ... + Q_5^n \cdot \Delta T \approx \frac{Q_5}{1 - Q_5} \cdot \Delta T. \quad (32)$$

As seen in Fig. 7, according to TLR scheme, there are six possible routes from S_1 to S_9. To calculate the end-to-end delay, we divide the whole route into four steps and calculate the expectation of each step respectively. By the way, to simplify the model, we assume that, no packets are dropped during the transmitting process. And since the propagation delays of all schemes in this model are the same, we only consider the queuing delay here.

For the first step, there are two choices; and since a packet will not be dropped, it eventually has the probability of $\frac{Q_1}{Q_1+Q_2}$ to go to S_2, and the probability of $\frac{Q_2}{Q_1+Q_2}$ to go to S_4. Therefore, with addition of the waiting time in the public queue, the expectation time for the first step

$$E_1 = (\frac{Q_1}{Q_1 + Q_2} \cdot \bar{X_1} + \frac{Q_2}{Q_1 + Q_2} \cdot \bar{X_2}) \cdot \frac{APS}{BW} + \Delta T_1. \quad (33)$$

Similarly, for the second step, the expectation time

$$\begin{aligned} E_2 &= \frac{Q_1}{Q_1 + Q_2} \cdot [(\frac{Q_1}{Q_1 + Q_2} \cdot \bar{X_1} + \frac{Q_2}{Q_1 + Q_2} \cdot \bar{X_2}) \cdot \frac{APS}{BW} + \Delta T_1] \\ &+ \frac{Q_2}{Q_1 + Q_2} \cdot [(\frac{Q_1}{Q_1 + Q_2} \cdot \bar{X_1} + \frac{Q_2}{Q_1 + Q_2} \cdot \bar{X_2}) \cdot \frac{APS}{BW} + \Delta T_1] \\ &= (\frac{Q_1}{Q_1 + Q_2} \cdot \bar{X_1} + \frac{Q_2}{Q_1 + Q_2} \cdot \bar{X_2}) \cdot \frac{APS}{BW} + \Delta T_1. \end{aligned} \quad (34)$$

And in the third step, we can see that, for S_3 and S_7, there is only one choice for the next hop(because the packet will not be sent beck right away to where it comes from). Therefore, a packet must go to the only next hop, i.e. the probability equals 1. So, the expectation time for the third step

$$\begin{aligned} E_3 &= (\frac{Q_1}{Q_1 + Q_2})^2 \cdot (1 \cdot \bar{X_3} \cdot \frac{APS}{BW} + \Delta T_2) + 2 \cdot \frac{Q_1}{Q_1 + Q_2} \\ &\cdot \frac{Q_2}{Q_1 + Q_2} \cdot [(\frac{Q_1}{Q_1 + Q_2} \cdot \bar{X_1} + \frac{Q_2}{Q_1 + Q_2} \cdot \bar{X_2}) \cdot \frac{APS}{BW} + \Delta T_1] \\ &+ (\frac{Q_2}{Q_1 + Q_2})^2 \cdot (1 \cdot \bar{X_3} \cdot \frac{APS}{BW} + \Delta T_2). \end{aligned} \quad (35)$$

Similarly, for the fourth step, the expectation time

$$\begin{aligned} E_4 &= (\frac{Q_1}{Q_1 + Q_2})^2 \cdot 1 \cdot (1 \cdot \bar{X_3} \cdot \frac{APS}{BW} + \Delta T_2) + 2 \cdot (\frac{Q_1}{Q_1 + Q_2})^2 \\ &\cdot \frac{Q_2}{Q_1 + Q_2} \cdot (1 \cdot \bar{X_3} \cdot \frac{APS}{BW} + \Delta T_2) + 2 \cdot (\frac{Q_2}{Q_1 + Q_2})^2 \\ &\cdot \frac{Q_1}{Q_1 + Q_2} \cdot (1 \cdot \bar{X_3} \cdot \frac{APS}{BW} + \Delta T_2) + (\frac{Q_2}{Q_1 + Q_2})^2 \cdot 1 \\ &\cdot (1 \cdot \bar{X_3} \cdot \frac{APS}{BW} + \Delta T_2). \end{aligned} \quad (36)$$

Consequently, the expectation from S_1 to S_9

$$\begin{aligned} E_{TLR} &= E_1 + E_2 + E_3 + E_4 = [2 + \frac{2Q_1Q_2}{(Q_1 + Q_2)^2}] \\ &\cdot [(\frac{Q_1}{Q_1 + Q_2} \cdot \bar{X_1} + \frac{Q_2}{Q_1 + Q_2} \cdot \bar{X_2}) \cdot \frac{APS}{BW} + \Delta T_1] \\ &+ [1 + \frac{{Q_1}^2 + {Q_2}^2}{(Q_1 + Q_2)^2}] \cdot (\bar{X_3} \cdot \frac{APS}{BW} + \Delta T_2). \end{aligned} \quad (37)$$

Accordingly, if we use DSP scheme, the expectation of total queuing time

$$E_{DSP} = 4 \cdot \bar{X} \cdot \frac{APS}{BW} = 4 \cdot (A_1 \cdot L_1' + A_2 \cdot L_2' + A_3 \cdot L_3') \cdot \frac{APS}{BW}. \quad (38)$$

We use L_1',L_2',L_3' here, because the length of queue L' at each direction in DSP is longer than that in TLR and ELB.

If we choose the ELB scheme:

$$\begin{aligned} {E_1}' &= (P_1 \cdot {\bar{X_1}}' + P_2 \cdot {\bar{X_2}}') \cdot \frac{APS}{BW} \\ &= \{[\beta + (1 - \beta) \cdot \chi] \cdot \bar{X} + (1 - \beta) \cdot (1 - \chi) \cdot \bar{X}\} \cdot \frac{APS}{BW} \\ &= \bar{X} \cdot \frac{APS}{BW} = (A_1 \cdot L_1 + A_2 \cdot L_2 + A_3 \cdot L_3) \cdot \frac{APS}{BW}, \end{aligned} \quad (39)$$

where β is the threshold, and χ is the percent of packets that remain to be transmitted through the best route.

$$\begin{aligned} {E_2}' &= [P_1 \cdot (P_1 \cdot {\bar{X_1}}' + P_2 \cdot {\bar{X_2}}') + P_2 \cdot (P_1 \cdot {\bar{X_1}}' + P_2 \cdot {\bar{X_2}}')] \cdot \frac{APS}{BW} \\ &= \bar{X} \cdot \frac{APS}{BW} = (A_1 \cdot L_1 + A_2 \cdot L_2 + A_3 \cdot L_3) \cdot \frac{APS}{BW}, \end{aligned} \quad (40)$$

$$\begin{aligned} {E_3}' &= [{P_1}^2 \cdot 1 \cdot {\bar{X_3}}' + 2 \cdot P_1 \cdot P_2 \cdot (P_1 \cdot {\bar{X_1}}' + P_2 \cdot {\bar{X_2}}') + {P_2}^2 \cdot 1 \cdot {\bar{X_3}}'] \\ &\cdot \frac{APS}{BW} = \bar{X} \cdot \frac{APS}{BW} = (A_1 \cdot L_1 + A_2 \cdot L_2 + A_3 \cdot L_3) \cdot \frac{APS}{BW}, \end{aligned} \quad (41)$$

$$\begin{aligned} {E_4}' &= [{P_1}^2 \cdot 1 \cdot 1 \cdot {\bar{X_3}}' + 2 \cdot ({P_1}^2 \cdot P_2 \cdot 1 \cdot {\bar{X_3}}' + {P_2}^2 \cdot P_1 \cdot 1 \cdot {\bar{X_3}}') \\ &+ {P_2}^2 \cdot 1 \cdot 1 \cdot {\bar{X_3}}'] \cdot \frac{APS}{BW} = \bar{X} \cdot \frac{APS}{BW} \\ &= (A_1 \cdot L_1 + A_2 \cdot L_2 + A_3 \cdot L_3) \cdot \frac{APS}{BW}. \end{aligned} \quad (42)$$

Therefore, the expectation of total queuing time

$$E_{ELB} = {E_1}' + {E_2}' + {E_3}' + {E_4}' = 4 \cdot (A_1 \cdot L_1 + A_2 \cdot L_2 + A_3 \cdot L_3) \cdot \frac{APS}{BW}. \quad (43)$$

According to the setting of our experiment,we calculate that:

$$E_{TLR} = 1.9367L \cdot \frac{APS}{BW}, \quad (44)$$

$$E_{DSP} = 2L' \cdot \frac{APS}{BW}, \quad (45)$$

$$E_{ELB} = 2L \cdot \frac{APS}{BW}. \quad (46)$$

Since L' is bigger than L, we could get the conclusion that,

$$E_{TLR} < E_{ELB} < E_{DSP} \quad (47)$$

To verify the accuracy of our model, we did some experiments based on 9 satellite nodes on the NS2 platform. The configuration of our experiment is: (1)The bandwidth of each link is 25Mbps; (2)background flows from S_i to $S_j(\forall i, j \in [1, 9], i \neq j)$ are set and the On/Off period of each flow obeys a Pareto distribution with a shape equal to 1.5; (3)A test flow from S_1 to S_9 is set, its transmission rate is 1Mbps. We calculated average end-to-end delay of packets from S_1 to S_9, when transmission rate of each background flow increases

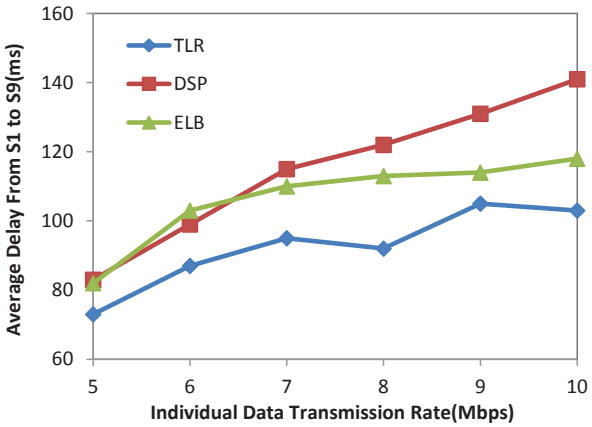


Fig. 8: Experiment for nine satellite node model.

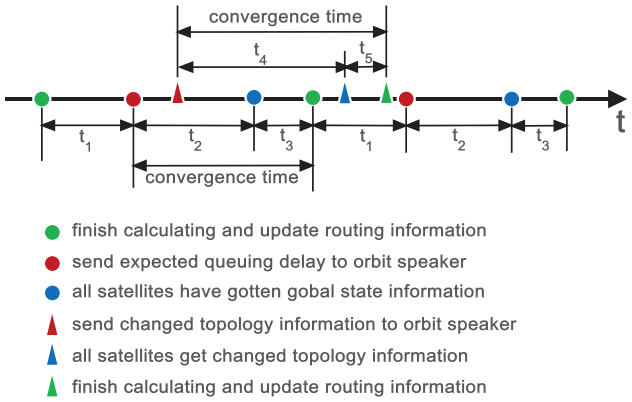


Fig. 9: Definition graph for convergence time.

from 5Mbps to 10Mbps. The experimental results are shown in Fig. 8, which is almost consistent with what we get from the analytical model.

B. Connectivity of TLR

Connectivity is an important parameter in satellite networks because the links here often suddenly break down. High connectivity means that there is more like to be a route from the source to the destination node, and thus could better ensure the packet transmission. We analyze the connectivity of TLR scheme basing on the simplified model used above. We assume the linking probability of horizontal links as p, and the linking probability of vertical links as q. We evaluate the connectivity of a routing policy according to the probability that there is at least a route from the source to the destination node. Firstly, let us analyze a simpler situation: form S_1 to S_5. Using DSP scheme, the stability

$$S_{DSP} = S_{125} = pq \tag{48}$$

Using TLR scheme, the stability

$$\begin{aligned} S_{TLR} = S_{1*5} &= p^2q^2 + 2(1-p)q^2p + 2(1-q)p^2q \\ &+ 2p(1-p)q(1-q) = pq(2-pq) \end{aligned} \tag{49}$$

Therefore,

$$S_{TLR} - S_{DSP} = pq(1-pq) > 0 \tag{50}$$

Actually, since TLR provides more choice than DSP, S_{TLR} is obviously bigger than S_{DSP}. And their ratio

$$S_{TLR}/S_{DSP} = (2-pq) > 1. \tag{51}$$

Then, we extend the situation to 4 hop situation in Fig. 6.

$$S_{DSP} = S_{12369} = S_{12569} = S_{125} * S_{569} = pq * pq = p^2q^2 \tag{52}$$

$$\begin{aligned} S_{TLR} &> S_{1*5*9} = S_{1*5} * S_{5*9} \\ &= pq(2-pq) * pq(2-pq) = p^2q^2(2-pq)^2 \end{aligned} \tag{53}$$

Therefore,

$$S_{TLR}/S_{DSP} > (2-pq)^2 > (2-pq) > 1 \tag{54}$$

From the above inequality, we get this conclusion: the worse the link condition is (i.e. the smaller p and q are) and the more hops from the source to the destination, the more stable TLR scheme is than DSP scheme. By the way, since ELB adopts the same multi-path routing idea with TLR, the analysis for its stability is the same as that of TLR.

C. Convergence Speed of TLR

We use the time from beginning to collect global state information to successfully calculating the routing table as routing convergence time. As shown in Fig. 9, normally, the convergence time equals $t_2 + t_3$, t_2 represents the time needed to collect the state information of all the satellites and t_3 represents the time needed for calculating and updating the routing information. Since the packets notifying state information have the highest priority, they will be inserted into the front of the queue directly, so their queueing delay nearly equals 0. Therefore, t_2 mainly consists of the propagation delay. In normal case, an orbit speaker needs at most $N/2$ intra orbit hop propagation time to collect the state information of all the satellites in an orbit. And to transmit the collected state information to the farthest counterpart orbit speaker, it at most needs $(M-1)$ inter orbit hop propagation time, for the cross-seam region has no links. And after the farthest orbit speaker receives the state information, it at most needs $N/2$ intra orbit hop propagation time to broadcast it to all satellites in its orbit. Therefore, t_2 equals 10 intra orbit hop propagation time + 5 inter orbit hop propagation time (about 280ms in total), according to our experiment setting. And since the scale of satellite network is very small and the processing ability of space device now is very strong, t_3 is negligible compared with t_2, so we ignore it here.

However, if a link breaks, which causes the change of topology, all the satellites should be notified about this change. The time needed to notify all the satellites about the changing topology equals t_4 and the new calculating and updating time equals t_5. In this case, the convergence time equals $t_4 + t_5$. If the broken link is the intra orbit link, t_4 at most equals $(N-1)$ intra orbit hops + $(M-1)$ inter orbit hops + $N/2$ intra orbit hops(about 355ms, according to our experiment setting). If the broken link is the inter orbit link and the link has nothing to do with the orbit speaker, t_4 equals t_2 (about 280ms). If the broken link is the inter orbit link and the ends of the link are orbit speakers, t_4 at most equals $N/2$ intra orbit hops + 1 intra orbit hops(role shift time)+ $(M-1)$ inter orbit + 1 intra orbit hops (to orbit speaker) + $N/2$ intra orbit hops (about 310ms, according to our experiment setting. Similar to t_3, t_5 could be ignored compared with t_4.

TABLE III: Distribution of Traffic Flows(%)

Source	Destination					
	NA	SA	EUR	Africa	Asia	Oceania
NA	60	10	15	2	10	3
SA	35	40	12	2	8	3
EUR	40	5	40	2	10	3
Africa	40	2	30	20	5	3
Asia	30	2	10	2	50	6
Oceania	40	2	10	2	12	34

VI. Experiments and Performance Evaluation

A. Simulation Setup

In this section, a series of experiments are carried out to evaluate the performance of TLR and to compare it with DSP and ELB, using Network Simulator (version 2)[16]. Our experiments are conducted on an Iridium-like constellation, with 66 satellites uniformly distributed over 6 orbits. Most satellites maintain four ISLs with neighbouring satellites, except those at high latitudes whose inter-orbit ISLs are turned off, and those along the seam whose cross-seam ISLs are switched off. For convenience, we set the capacity of each link as 25 Mbps. Besides, in order to eliminate the influence of channel error, we assume that all links are error-free so that we can focus our attention on the algorithm itself. In the experiments, the average one-hop ISL propagation delay is set to 14ms (d = 14ms) and the traffic light checking interval is set to 6ms (Δt = 6ms). The checking interval of public waiting queue is 30ms and the maximum waiting time for a packet is 90ms. Besides, global routing information is refreshed every 600ms. In TLR, the size of each buffer queue is set to 75 packets, and the public waiting queue size is set to 100 packets. In the DSP and ELB experiments, since there is no public waiting queue, each buffer queue size is set to 100 packets to keep the total queue space the same as that in TLR. In the experiments, the average packet size is set to 1 KB.

For traffic generation, similar to [6], we utilize 300 non-persistent On-Off flows. The On/Off period of each flow obeys a Pareto distribution with a shape equal to 1.5. The average burst and idle time are set to 500ms. The source and destination terminals are dispersed all over the Earth, following a distribution given in Tab. III [17]. The source terminals send data at rates varying from 1.6 Mbps to 2.4 Mbps.

In our experiments, we compare the performance between TLR, ELB and DSP, in terms of packet delay, packet drop rate, total throughput and traffic distribution. Since the ELB scheme can be implemented over any routing protocols [6], we utilize its key points and consider its implementations over Pre-Calculation to ensure the same base with TLR. In addition, Dijkstras Shortest Path (DSP) algorithm, as a "benchmark", also emerges as a comparison term. To investigate the impacts brought by queuing delay, we conduct two DSP experiments based on the routing metric of "Propogation Delay"(DSP-PD) and "Propogation Delay + Expected Queuing Delay"(DSP-PDQD), respectively. All the simulations are run for 20.51s.

TABLE IV: Six Pairs of Terminal Locations

Pair	Source Terminal	Destination Terminal
1	NewYork(NY, 40.7,-74.0)	ShangHai(SH, 34.5,121,4)
2	Tokyo(TY, 35.4,139.5)	London(LD, 51.3,0.1)
3	Paris(PA, 48.5,2.2)	Johannesburg(JN, -26.2,28.0)
4	Cairo(CR, 30.0,31.2)	Perth(PE, -32.0,115.8)
5	Sydney(SD, -33.5,151.1)	BuenosAires(BA, -34.4,-58.3)
6	SaoPaulo(SP, -23.3,-46.4)	LosAngeles(LA, 34.0,-118.2)

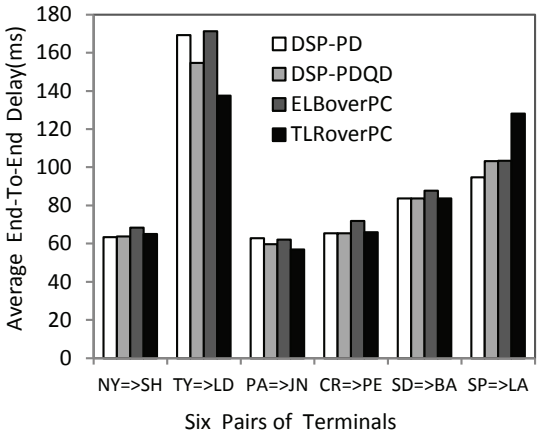


Fig. 10: Average end-to-end delay of six terminal pairs.

B. Simulation Results

1) Packet Delay: Firstly, we evaluate the performance of TLR in terms of packet delay. In TLR, packets are sometimes forced to traverse more hops than in the case of traditional algorithms, especially when most traffic lights along the shortest path show "RED". To verify whether TLR really increases the end-to-end delay, we set 6 pairs of terminals, let each source terminal send a packet to its destination every 0.5s, and compute the average end-to-end delay. As shown in Tab. IV, all the terminals are within metropolises on the continents. Therefore, the traffics around them are relatively heavier than those in other places.

Fig. 10 shows the average end-to-end delay of each pair, in the case of setting the individual data transmission rate to 2.4Mbps. In the figure, the average end-to-end delay of pair 1, 3, 4 and 5 in each routing strategy is almost the same; the end-to-end dealy of pair 6 in the TLR strategy is larger than that in other strategies. However, for pair 2, the TLR strategy achieves the smallest end-to-end delay. This fact proves that the TLR strategy does not always choose a route with a higher end-to-end delay than the DSP algorithms. The underlying reason lies in the abilities of TLR to avoid the congested satellites by traffic lights, to alleviate the congestion by allocating traffics to different routes, and to ultimately reduce the average queuing delay at each satellite. To verify this idea, we compute the average queueing delay at each satellite using four different strategies. Fig. 11 shows the result of this comparison. As we see, the DSP-PD algorithm gains a high queuing delay at some satellites, because it only considers the propagation delay when choosing the route. As a result, some packets may aggregate at some satellites and finally lead to high queuing delay. In DSP-PDQD, since it considers both the propagation delay and expected queuing delay, the peak points become lower. However, since there is only one route for each flow,

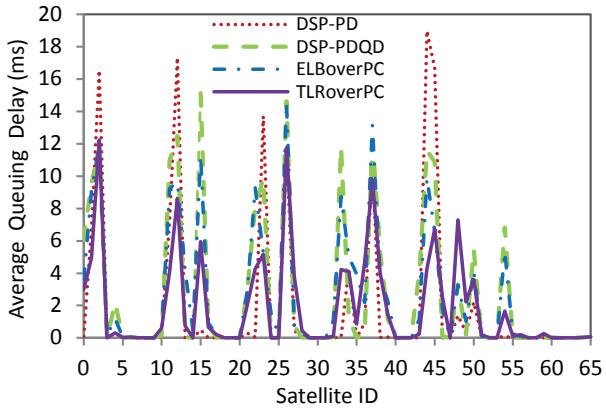


Fig. 11: Average queuing delay at each satellite.

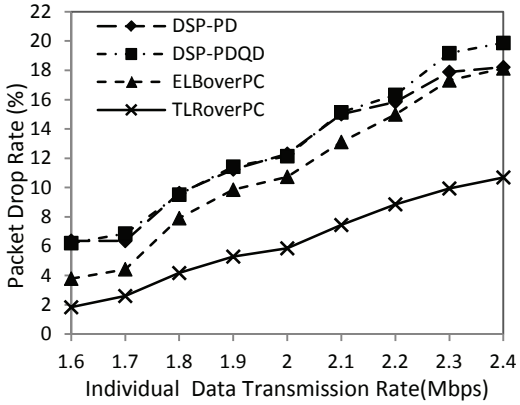


Fig. 12: Packet drops for different sending rates.

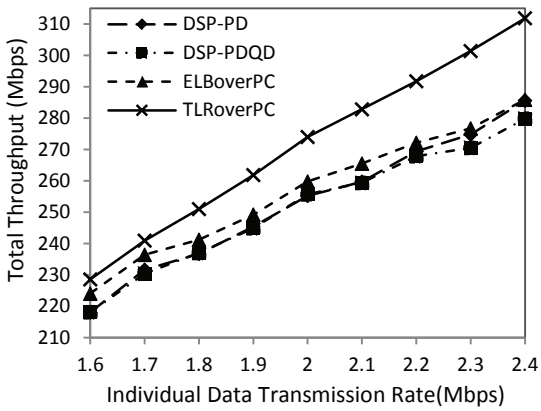


Fig. 13: Total throughput for different sending rates.

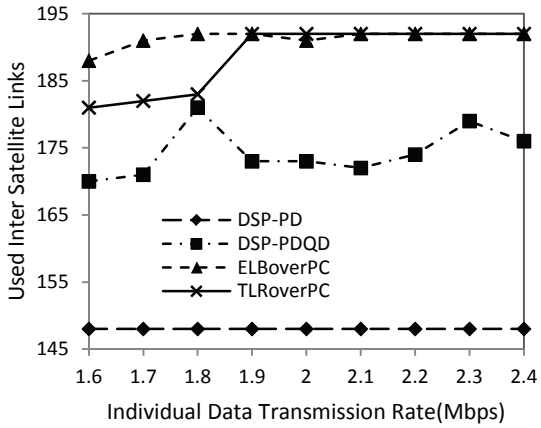


Fig. 14: Used ISLs for different sending rates.

the packets to be sent through a route do not decrease, so the queuing delay at some satellites remains high. In the ELB and TLR schemes, multiple routes are used for each flow when the traffic is heavy, packets may less likely aggregate. In addition, since TLR uses multiple routes more intelligently than ELB, the queuing delay peak values declined remarkably.

2) Packet Drop Rate and Total Throughput: Secondly, we evaluate the performance of TLR in terms of total packet drop rate. During each simulation, we record the total packets sent out by 300 source terminals and the total dropped packets. Fig. 12 graphs the variation trend of the total packet drop rate, with the data sending rate of each flow ranging from 1.6Mbps to 2.4Mbps. As we see, for all the rates, TLR achieves the lowest packet drop rate. Compared to the DSP algorithms, the ELB scheme also achieves lower packet drop rate because it considers the congestion at next hops and could possibly avoid overflow of the queue. Yet, it fails to consider the congestion at the current hop and to spare a public waiting queue for the extreme situation, therefore, some packets are discarded before they are sent out.

The good performance of TLR in lowering packet drop rate is also embodied in the high throughput it achieves. As shown in Fig. 13, the TLR strategy precedes a lot in the total throughput compared to the DSP algorithms. The advantage owes to the mechanisms TLR utilizes to avoid dropping packets as far as possible. Similar with the packet drop rate, TLR also achieves a better performance in the throughput than ELB, as it considers more factors that lead to packet drops.

3) Traffic Distribution: Besides the reduced packet drop rate and the improved total throughput, the TLR strategy yields a much balanced distribution of traffic over the entire constellation. To illustrate the idea, we first count the used inter satellite links (ISLs) in different strategies in each simulation. Fig. 14 shows the used ISLs for different individual sending rates. As we see, the DSP-PD algorithm uses the fewest ISLs regardless of the sending rate. In DSP-PDQD, although there is still only one route for each flow, the route itself may change as the expected queuing delay is taken into account. That is why the used ISLs increase a lot compared to the DSP-PD algorithm. In ELB and TLR, multiple routes are used when congestion happens. Those ISLs which are not used in the DSP algorithms are used in ELB and TLR to undertake the packet transmission tasks (all the 192 ISLs are used when the sending rate increases to 1.9Mbps). Admittedly, some ISLs seem not to be wise choices in terms of propagation delay, but they could indeed help balance the traffic load and decrease the packet drop rate.

To better investigate how well the traffic is distributed over the entire constellation, the following traffic distribution index is used:

$$Index = \frac{(\sum_{k=1}^{n} x_i)^2}{n \sum_{k=1}^{n} x_i^2}, \tag{55}$$

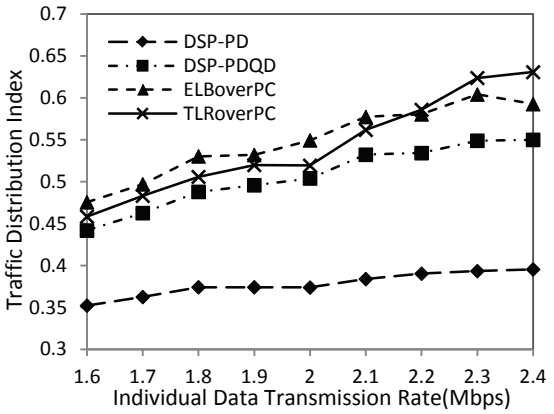


Fig. 15: Traffic distribution index for different sending rates.

where n is the number of ISLs and x_i denotes the actual number of packets that traversed the i^{th} ISL. This index value ranges from 0 to 1, and a higher value represents a better traffic distribution [6]. In Fig. 15, we plot the traffic distribution index for different sending rates in different strategies. The figure indicates that DSP-PDQD, ELB and TLR significantly outperform the DSP-PD algorithm. This result attributes to the fact that the DSP-PD algorithm only bases its routing on finding paths with the shortest propogation delay; packets belonging to a flow are transmitted over a single path during the entire transmission. The DSP-PDQD algorithm improves the situation by taking the expected queuing delay into account. Of course, compared to ELB and TLR which utilize multiple routes to transmit data between two terminal pairs, the DSP-PDQD algorithm underperforms. The performance of TLR is comparable with that of ELB, but is better if the individual data transmission rate increases.

VII. Conclusion

In this paper, we propose TLR, a traffic-light-based intelligent routing strategy for the satellite network, which can adjust the pre-calculated route according to the real-time congestion status of the satellite constellation. In a satellite, a traffic light is deployed at each direction to indicate the congestion situation, and is set to a relevant color, by considering both the queue occupancy rate at a direction and the total queue occupancy rate of the next hop. When a traffic light at a direction turns from "GREEN" to "YELLOW", some packets pre-routed to this direction should be re-routed to others to alleviate the congestion; if the light turns from "YELLOW" to "RED", no packets should be forwarded any more. The state of a satellite is evaluated by periodically calculating its total queue occupancy rate and is notified to its neighbors if changed. To further lower the packet drop rate, a public waiting queue is built to temporarily hold the passless packets, which will be sent out later when congestion is alleviated. In addition, an endless-loop-free mechanism is enforced in the design of TLR, which avoids unnecessary waste of bandwidth resources.

To verify the performance of TLR, a set of simulations are conducted. We compare the simulation results of TLR with DSP-PD, DSP-PDQD and ELB, in terms of packet delay, packet drop rate, total throughput and traffic distribution. The results show that the TLR strategy achieves better performance in avoiding congestion, reducing queueing delay, lowering packet drops and increasing total throughput. Besides, it achieves a balanced traffic distribution over the entire constellation. In terms of propagation delay, the TLR strategy may be inferior to other strategies in some cases, because packets may have to traverse additional hops to keep away from the congestion nodes. However, since TLR could reduce the queuing delay by alleviating congestion, the ultimate end-to-end delay is acceptable. Furthermore, considering the extra time that may be required to retransmit dropped packets, the total time required in TLR would be less than in others.

Finally, it should be pointed out that the obtained results are based on services which do not have special demands on end-to-end delay or bandwidth. The enhancement that considers diverse service types and their QoS demands is an interesting issue, which forms a basis for our future research work.

Appendix

Algorithm1:TLR over Pre-Calculation

$curHop$ receives a pkt
Add $curHop.id$ into $pkt.path$
Get $BR.nxtHop$, $SBR.nxtHop$ to $pkt.des$ from routingtable
if $BR.nxtHop.id = pkt.des$ **then**
  forward pkt to $BR.nxtHop$
else
  if $BR.nxtHop.id \notin pkt.path$ **and** $SBR.nxtHop.id \notin pkt.path$ **then**
    if $BR.nxtHop.color = GREEN$ **then**
      forward pkt to $BR.nxtHop$
    end if
    if $BR.nxtHop.color = YELLOW$ **then**
      if $SBR.nxtHop.color = GREEN||YELLOW$ **then**
        forward pkt to $BR.nxtHop$ or $SBR.nxtHop$ according to PTR (see notation)
      else
        forward pkt to $BR.nxtHop$
      end if
    end if
    if $BR.nxtHop.color = RED$ **then**
      if $SBR.nxtHop.color = GREEN||YELLOW$ **then**
        forward pkt to $SBR.nxtHop$
      else
        add pkt into $curHop.WL$
      end if
    end if
  end if
  if $BR.nxtHop.id \notin pkt.path$ **and** $SBR.nxtHop.id \in pkt.path$ **then**
    if $BR.nxtHop.color = GREEN||YELLOW$ **then**
      forward pkt to $BR.nxtHop$
    else
      add pkt into $curHop.WL$
    end if
  end if
  if $BR.nxtHop.id \in pkt.path$ **and** $SBR.nxtHop.id \notin pkt.path$ **then**
    if $SBR.nxtHop.color = GREEN||YELLOW$ **then**
      forward pkt to $SBR.nxtHop$
    else
      add pkt into $curHop.WL$
    end if
  end if
  if $BR.nxtHop.id \in pkt.path$ **and** $SBR.nxtHop.id \in pkt.path$ **then**
    $pos \Leftarrow$ the 1st position of $curHop.id$ in $pkt.path$

42: $preHop.id \Leftarrow pkt.path[pos-1]$
43: $preHop \Leftarrow$ find Node by $preHop.id$
44: forward pkt to $preHop$
45: **end if**
46: **end if**

Algorithm2:Periodical Checking for WaitingList(WL)

1: A new interval point comes
2: **for** $pkt \in WL$ **do**
3: $linkUp \Leftarrow FALSE$
4: **for** $neighbor \in curHop.neighbors$ **do**
5: **if** $pkt.nxtHop.id = neighbor.id$ **then**
6: $linkUp \Leftarrow TRUE$
7: **break**
8: **end if**
9: **end for**
10: **if** $linkUp = TRUE$ **then**
11: **if** $neighbor.color = GREEN||YELLOW$ **then**
12: move pkt out of WL
13: forward pkt to $neighbor$
14: **else**
15: $pkt.TTW \Leftarrow pkt.TTW - 1$
16: **if** $pkt.TTW = 0$ **then**
17: move pkt out of WL and drop it
18: **end if**
19: **end if**
20: **else**
21: move pkt out of WL
22: forward pkt to $curHop$ again
23: **end if**
24: **end for**

Theorem. *TLR over Pre-Calculation is an Endless-loop-Free algorithm, i.e., a packet can arrive at the destination or be dropped without wasting bandwidth resources.*

Proof: First, we choose a random node as the source node. Then, we divide all other nodes into several sets: $HOP(1), HOP(2), ..., HOP(MAX)$. In $HOP(N)$, any node has a shortest distance of N hops to the source. As shown in Algorithm 1, if the destination is only one hop away, packets will be directly sent there (lines 4-5). For convenience, we call it part1, and the rest part2.

Next, we use mathematical induction to prove that, by following TLR, a packet can reach the destination or be dropped without wasting bandwidth resources.

Step1: It is obvious that, nodes in $HOP(1)$ are reachable according to part1 of TLR.

Step2: We suppose that a packet can reach any node in $HOP(1)$-$HOP(K)$ or be dropped without wasting bandwidth resources during the transmission process.

Now, we prove: a packet can arrive at any node in $HOP(K+1)$ or be dropped without wasting bandwidth resources during the transmission process.

For any node in $HOP(K+1)$, there exist its precursors which belong to $HOP(1)$-$HOP(K)$. If a packet is dropped without wasting bandwidth resources at any of these precursors, obviously, the theorem is true. Therefore, we just need to prove: if a packet successfully reaches any node in $HOP(1)$-$HOP(K)$, the packet could reach any node in $HOP(K+1)$ or be dropped without wasting bandwidth resource.

We use $h(N)$ to represent a node which belongs to $HOP(N)$. Obviously, for any $h(K+1)$, there exists at least one $h(K)$ adjacent to it. According to our assumption, we can reach $h(K)$. Thus, an $h(K-1)$ can be reached by part2 of TLR. Then, we only need to prove: from $h(K-1)$, a packet can reach $h(K+1)$ or may be dropped without wasting bandwidth resources. As shown in Fig. 16, a packet arrives at D which is an $h(K-1)$ node:

Case 1: G is chosen as the next hop. Since H is the direct next hop of G, according to part1 of TLR, the packet will then be directly sent to H, therefore, $h(K+1)$ is reachable.

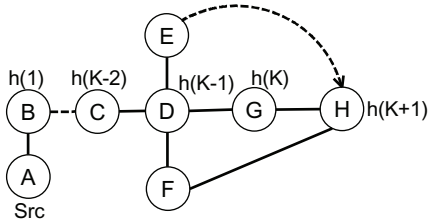


Fig. 16: A schematic proving that TLR is endless-loop-free.

Case 2: G is not chosen as the next hop, according to TLR, there may be several possibilities.

Case 2.1, G is already in the packet head. Since G is an $h(K)$ and D is an $h(K-1)$, if there exits a path in which G appears before D, G must be a node in $HOP(N)$ where N is smaller than $K-1$. This is impossible, because a node could not exist in two different sets.

Case 2.2, G is not the best route from D to H. If so, there must be a direct link between D and H. Then, H will be an $h(K)$ node, which is not consistent with the previous assumption. Therefore, it is an impossible case.

Case 2.3, G is not the only best route from D to H. That is to say, there exists a node between D and H, namely F. If the packet is sent to F, according to part1 of TLR, the packet will then be sent to H directly. If so, $h(K+1)$ is also reachable.

Case 2.4, G is the best route, but the traffic light at this direction shows "RED". In this case, the next hop of the second best route, for example, node E, will be chosen. From E, if the packet could arrive at H in the end, $h(K+1)$ is reachable. But if the packet could not reach H from this node, the packet will be sent back to D, just as we have described in Fig.5. Then, E has appeared in the packet head and would not be chosen as the next hop any more. If the traffic light at the direction of G has turned "GREEN" or "YELLOW", the packet would be sent there. Then, H would be reachable according to part1. If the traffic light still shows "RED", the packet would be inserted into the public waiting queue, waiting for the traffic light to turn "GREEN" or "YELLOW", or waiting to be dropped when its TTW value decreases to 0. Thus, no bandwidth resources would be wasted.

Therefore, TLR is endless-loop-free. ■

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China under grant No. 61272467. The authors wish to thank the center for engineering and scientific computation, Zhejiang University, for its computational resources, with which the research project has been carried out.

REFERENCES

[1] L. Wood, A. Clerget, I. Andrikopoulos, G. Pavlou, and W. Dabbous, "IP routing issues in satellite constellation networks," *International J. Satellite Commun.*, vol. 19, pp. 69–92, 2001.

[2] T. Taleb, A. Jamalipour, N. Kato, and Y. Nemoto, "IP traffic load distribution in NGEO broadband satellite networks," in *Proc. 2005 International Conf. Comput. Inf. Sciences*, pp. 113–123.

[3] M. Werner, "A dynamic routing concept for ATM-based satellite personal communication networks," *IEEE J. Sel. Areas Commun.*, vol. 15, no. 8, pp. 1636–1648, Oct. 1997.

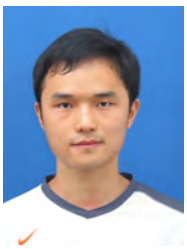
[4] T. Henderson and R. Katz, "On distributed, geographic-based packet routing for LEO satellite networks," in *Prof. 2000 IEEE Global Telecommun. Conf.*, vol. 2, pp. 1119–1123.

[5] B. Jianjun, L. Xicheng, L. Zexin, and P. Wei, "Compact explicit multi-path routing for LEO satellite networks," in *Proc. 2005 High Performance Switching Routing Workshop*, pp. 386–390.

[6] T. Taleb, D. Mashimo, A. Jamalipour, N. Kato, and Y. Nemoto, "Explicit load balancing technique for NGEO satellite IP networks with on-board processing capabilities," *IEEE/ACM Trans. Netw.*, vol. 17, no. 1, pp. 281–293, Feb. 2009.
[7] Y. Rao and R. Wang, "Agent-based load balancing routing for LEO satellite networks," *Comput. Netw.*, vol. 54, no. 17, pp. 3187–3195, Dec. 2010.
[8] H. Nishiyama, D. Kudoh, N. Kato, and N. Kadowaki, "Load balancing and QoS provisioning based on congestion prediction for GEO/LEO hybrid satellite networks," *Proc. IEEE*, vol. 99, no. 11, pp. 1998–2007, Nov. 2011.
[9] Y. Kawamoto, H. Nishiyama, N. Kato, N. Yoshimura, and N. Kadowaki, "Assessing packet delivery delay in multi-layered satellite networks," in *Proc. 2012 IEEE International Conf. on Commun.*, pp. 3311–3315.
[10] R. Mauger and C. Rosenberg, "QoS guarantees for multimedia services on a TDMA-based satellite network," *IEEE Commun. Mag.*, vol. 35, no. 7, pp. 56–65, July 1997.
[11] J. Bai, X. Lu, Z. Lu, and W. Peng, "A distributed hierarchical routing protocol for non-GEO satellite networks," in *Proc. 2004 International Conf. Parallel Process.*, pp. 148–154.
[12] I. Akyildiz, E. Ekici, and M. Bender, "MLSR: a novel routing algorithm for multilayered satellite IP networks," *IEEE/ACM Trans. Netw.*, vol. 10, no. 3, pp. 411–424, June 2002.
[13] X. Yi, Z. Sun, F. Yao, and Y. Miao, "Satellite constellation of MEO and IGSO network routing with dynamic grouping," *International J. Satellite Commun. Netw.*, 2013.
[14] J. Sun and E. Modiano, "Routing strategies for maximizing throughput in LEO satellite networks," *IEEE J. Sel. Areas Commun.*, vol. 22, no. 2, pp. 273–286, 2004.
[15] Y. Kawamoto, H. Nishiyama, N. Kato, and N. Kadowaki, "A traffic distribution technique to minimize packet delivery delay in multi-layered satellite networks," *IEEE Trans. Veh. Technol.*, 2013.
[16] Ucb/lbnl/vint network simulator—ns (version 2). Available: http://www-mash.cs.berkeley.edu/ns/
[17] M. Mohorcic, M. Werner, A. Svigelj, and G. Kandus, "Adaptive routing for packet-oriented intersatellite link networks: performance in various traffic scenarios," *IEEE Trans. Wireless Commun.*, vol. 1, no. 4, pp. 808–818, Oct. 2002.

Guanghua Song is an Associate Professor with School of Aeronautics and Astronautics, Zhejiang University, China. He received the Ph.D. degree in computer science from Zhejiang University, China, in 2003. His research interests include distributed systems and networking, satellite communications, mobile and embedded systems. He is a senior member of China Computer Federation (CCF) and a member of ACM.

Mengyuan Chao received the BS degree in software engineering from Huazhong University of Science and Technology, China, in 2011. He is currently a master student at the Center for Engineering and Scientific Computation, and School of Aeronautics and Astronautics of Zhejiang University, China. His research interests include wireless networking, software-defined networking, distributed computing and storage.



Bowei Yang received the BS and Ph.D. degrees in computer science from Zhejiang University, China, in 2006 and 2011, respectively. He is currently a Post-Doctoral Researcher at the Center for Engineering and Scientific Computation, and School of Aeronautics and Astronautics of Zhejiang University, China. His research interests include software-defined networking, satellite networking, distributed computing and storage.

Yao Zheng is a Cheung Kong chair professor with Zhejiang University, appointed by the Ministry of Education of China since 2001. He is the director of the Center for Engineering and Scientific Computation, and the founding deputy dean of the School of Aeronautics and Astronautics, both in Zhejiang University, China. He had been a Senior Research Scientist for NASA Glenn Research Center, Cleveland, Ohio, USA, from 1998 to 2002. His current research interests include aerospace computing engineering, and high performance computing.