A Heuristic Inter-Satellite Fault Tolerant Routing Mechanism Based on A-Star Algorithm

Yuting Zhang[1], Yifan Yang[2], Neng Ye[1], and Jie Zeng[1(✉)]

[1] School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China
{1120200732,ianye,zengjie}@bit.edu.cn
[2] School of Information and Electronics, Beijing Institute of Technology, Beijing 100081, China
3220230923@bit.edu.cn

Abstract. As inter-satellite networks grow, reliable routing is critical. However, it is still challenged by rapidly changing topology and propagation delay. Unlike Internet protocols, space solution requires different approaches because of long distance spans and large propagation delays of inter-satellite links (ISLs). We propose a fault-tolerant routing mechanism using a modified heuristic function based on A-Star algorithm to reduce search space and determine optimal paths faster. Simulations at various failure rates and network sizes demonstrate the efficiency of routing and stability of time cost at various path lengths. Our mechanism can find the sub-optimal path quickly regardless of size, minimizing route time and guaranteeing short path length. This shows promise for applying this mechanism to future mega-constellation routing.

Keywords: Inter-Satellite Routing · A-Star Algorithm · Path-Finding Optimizing

1 Introduction

As the application and demand for inter-satellite routing continue to grow, the study of stable and reliable inter-satellite routing becomes increasingly important. Internet protocols are often unsuitable for use on links that operate intermittently or have long propagation delays, as reliable transmission and routing can be problematic. Solutions to these issues will likely differ significantly from Internet operation norms [1].

Burleigh S et al. proposed a packet routing algorithm for LEO satellite networks. This distributed algorithm can calculate the minimum propagation delay

This work was supported by the National Key Research and Development Program under Grant 308, the National Natural Science Foundation of China under grant 62001264, the Fundamental Research Funds for the Central Universities, and the Beijing Institute of Technology Research Fund Program for Young Scholars.

F. Gao et al. (Eds.): Chinacom 2023, LNICST 590, pp. 478–492, 2024.
https://doi.org/10.1007/978-3-031-67162-3_30

path in real-time without connection path switching, effectively avoiding congestion and failure [1]; Werner M et al. studied a routing scheme for LEO satellite systems to provide acceptable quality of service and efficient utilization of network resources by optimizing the routing scheme according to different objective function [2]; Ji X et al. proposed and evaluated an A-Star algorithm based on demand routing (ASOR) protocol with better convergence time and end-to-end delay performance, ensuring the optimal path and avoiding node breakage [3].

In path-finding, there are many related studies about A-Star algorithm, such as the usage of A-Star algorithm in big map navigation and how to find a shorter path [4]; adding binary tree and bidirectional search in A-Star algorithm to enhance search and constraints-handling efficiency [5]; some researchers expand the number of search directions in A-Star to significantly reduce the total path length and inflection points [6] or change the evaluation function and adjust the search directions from 8 to 5, generating paths that avoid collisions with obstacles effectively [7]. In the condition of UAV and ship routing, studies are performed to combine with RTT algorithm to generate a temporary path, enhancing real-time performance, and take the safety distance between the object and obstacles into concern [8–10]. Compared with local rerouting, full rerouting can avoid high local routing computation overhead as well as network performance degradation [11].

To this end, this paper proposes a fault-tolerant inter-star routing mechanism based on a heuristic shortest path algorithm. The main contributions are as follows:

- We developed a modified heuristic function that guides the path search mechanism and helps improve network performance. By adjusting the heuristic function, we effectively reduce the search space and the search time. Also, this mechanism can find a way to reduce the hop counts, approaching the minimum value.
- We verify the efficiency and stability of this mechanism by conducting simulation experiments at different failure rates and network sizes. The results show that the algorithm can quickly find the near-optimal path regardless of the network size and reduce the routing time. Based on this, we propose an outlook on the application of this mechanism in future mega-constellation routing.

The rest of the paper is structured as follows: Sect. 2 presents the basic model assumptions and the system architecture. In Sect. 3, we describe the specific implementation scheme. Section 4 shows the simulation with specific experimental results. Section 5 concludes the paper.

Notation: We use upper and lower case boldface to denote matrices and vectors. $\mathrm{L}(\bullet)$ denotes the path length. $(\bullet)^{\mathrm{pd}}$ denotes the propagation delay, $(\bullet)^{\mathrm{t}}$ denotes the transmission delay and $(\bullet)^{\mathrm{p}}$ denotes processing delay. $(\bullet)_0$ denotes parameter of the source node, and $(\bullet)_n$ denotes parameter of the destination node. $[\bullet]_{ij}$ denotes the element in the i^{th} row and the j^{th} column of a matrix of a 2D-mesh graph.

2 System Model

2.1 Network Model

A discrete-time topology-based method is adopted so that routing algorithms can be run like Fig. 1 [12]. Dynamic network topology is separated by step width $\Delta t = \frac{T}{K}$ to a sequence of K topology snapshots. Each snapshot at $t = k\Delta t$ is modeled as a graph $G(k) = (V, E(k))$. For $G(k)$, the topological situation encountered in the time interval $[k\Delta t, (k+1)\Delta t]$ can be considered as explicitly fixed, as shown in Fig. 2.

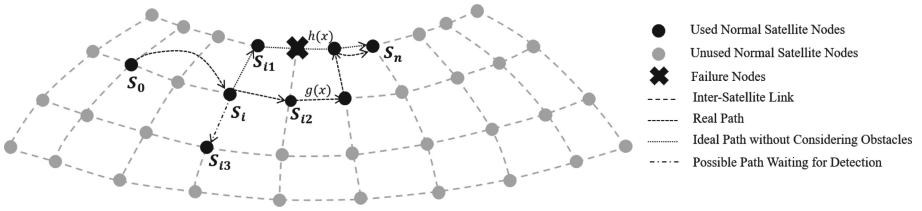


Fig. 1. A Satellites Routing Mechanism

For step width Δt, there are two conditions:

1) Δt is adapted to discrete-time link activation/deactivation behavior to ensure that $G(k)$ reflects the physical topology;
2) Δt is small enough, so that the slow continuous distance change during $[k\Delta t, (k+1)\Delta t]$ can simulate real physical topology changes. For monotonic cost changes, the condition becomes:

$$\frac{((c_{ij}((k+1)\Delta t - 0) - c_{ij}(k\Delta t)))}{(c_{ij}(k\Delta t))} \ll 1, \forall (i,j)_k \in E(k) \tag{1}$$

Although the first ideal condition is difficult to meet, the second condition leaves some freedom for an appropriate compromise.

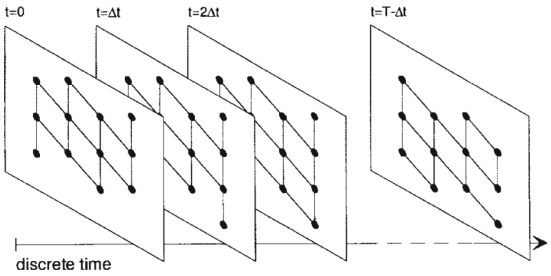


Fig. 2. Discrete-Time Topology

2.2 LEO Satellites Network Model

The polar orbit constellation model is used to implement the inter-satellite routing algorithm. The satellite networks have N polar orbital planes, and there are M satellites on each plane [13].

The angular distance between two adjacent planes is $\frac{360^\circ}{2N}$. The angular distance between two adjacent satellites in the same orbital plane is $\frac{360^\circ}{M}$.

Each satellite has four ISLs: 2 inter-plane ISLs and 2 intra-plane ISLs.

The intra plane ISLs exist all the time and the length of intra plane ISLs, L_v is equal. The length of L_v is calculated as:

$$L_v = \sqrt{2}R\sqrt{1 - \cos(\frac{360^\circ}{M})}; \tag{2}$$

To enable the exchange as in Fig. 3, the inter-plane ISLs are allowed to be turned off in the polar region, but the intra-plane ISLs remain in place. When leaving the polar region, the inter-plane ISLs can be re-established.

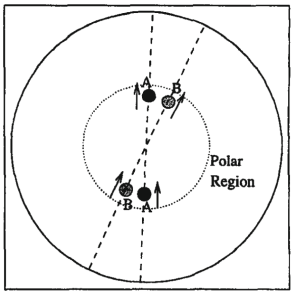


Fig. 3. Satellites in Polar Region

Given that: longitude direction distance = longitude difference × cos (latitude), the length of inter-plane ISLs, L_h can be calculated as:

$$L_h = \alpha \times \cos(lat); \tag{3}$$

where the longitude difference α can be calculated as:

$$\alpha = \sqrt{2}R\sqrt{1 - \cos(\frac{360^\circ}{2N})}; \tag{4}$$

2.3 Routing in Grid Map

Assume that the constellation network can be considered as a grid network as Fig. 4 [14].

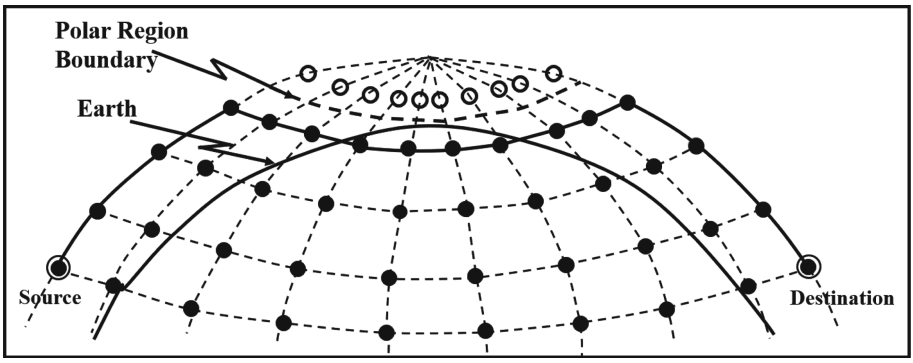


Fig. 4. Grid Satellites Network

There are logical links between adjacent satellite nodes, and the algorithm will choose the best route from every possible virtual connection route from source to destination satellites [15]. Let P_0 and P_n be outside of the polar regions and P_0 be at a higher latitude, lat, than P_n. Assume that P_0 is A ($A \geq 0$) hops away from the horizontal ring closest to the polar region and that the ring of P_0 is the k^{th} ($k > 0$) horizontal ring when counted from the closest pole. Let the number of horizontal jumps from P_0 to P_n be n_h.

Therefore, if the path from P_0 to P_n will pass through the pole area, the delay is:

$$L_{pr} = (N - n_h)\alpha\cos(lat_{min}) + L_v(2k + 1), \tag{5}$$

where lat_{min} is the latitude of the ring closest to the polar region.

If the path from P_0 to P_n does not pass through the polar region, the delay is:

$$L_{h+a} = n_h\alpha\cos(lat + a\frac{360^\circ}{M}) + 2aL_v; \tag{6}$$

where L_v and α are given by (2) and (4).

Since DRA causes packets to try to choose high-latitude links during link selection, inter-plane links at high latitudes can cause congestion. Therefore, sometimes it is necessary to spread some link pressure through the path of polar regions.

When $L_{pr} < L_{h+a}$,

$$(N - n_h)\alpha\cos(lat_{min}) + L_v(2k + 1) < n_h\alpha\cos(lat + a\frac{360^\circ}{M}) + 2aL_v; \tag{7}$$

Solve the inequation (7) and get the range of n_h:

$$n_h > \max_{1<a<A} \frac{N\cos(lat_{min}) + \frac{L_v}{\alpha}(2(k-a)+1)}{(\cos(lat + a\frac{360^\circ}{M}) + \cos(lat_{min}))}; \tag{8}$$

When n_h satisfies inequation (8), the time through the polar region is shorter.

It ascends to the high-latitude region first, moves to the longitude of the destination node on the horizontal ring closest to the polar region, then passes through the polar region, and reaches the destination node only through the vertical hops.

3 Proposed Mechanism

If the path does not pass through the polar region, which means the nodes in the polar region are not considered in routing, let the link cost be divided into three parts: transmission delay, propagation delay, and processing delay:

$$t = {t_i}^{\mathrm{pd}} + {t_i}^{\mathrm{t}} + {t_i}^{\mathrm{p}}; \tag{9}$$

where,

$${t_i}^{\mathrm{pd}} = \frac{\alpha\cos(lat)}{v}, v = 3 \times 10^8\,\mathrm{m/s}; \tag{10}$$

However, since a fixed topology within a fixed time slice cannot exactly match the physical situation, a node shown as valid in the virtual topology may fail or enter the polar region without being able to follow the expected path. When these happen, it can be considered as a failure.

When a node fails, the proposed mechanism can be used to help reroute. The proposed mechanism is based on the A-Star algorithm, which is commonly used in map path-finding to find the optimal path. It is more quickly than the Dijkstra algorithm, a greedy algorithm because it uses a heuristic function to estimate the distance to the destination.

In a grid map, each center of a grid can be regarded as a node, and grids connected with the top, bottom, left, and right of the current grid are called the neighbor nodes.

The core of the A-Star algorithm lies in the design of the estimated cost function $f(x)$:

$$f(x) = g(x) + h(x); \tag{11}$$

$f(x)$ is the estimated cost from the starting node to the destination node, and $g(x)$ is the dissipation function and also the actual cost from the starting node to the current node n. $h(x)$ is the estimated cost from the current node n to the destination node without considering obstacles.

In this mechanism, the failed node can be considered an impassable obstacle, and the dissipation function $g(x)$ consists of the propagation delay:

$$g(x) = \sum_{i=0}^{c} {t_i}^*, \tag{12}$$

where ${t_i}^*$ stands for the actual routing cost;

Adding ${t_i}^{\mathrm{t}}$ and ${t_i}^{\mathrm{p}}$ to the dissipation function can be analogous to adding available non-road terrain (e.g., swamps, lakes, etc.) to the map.

Since the interstellar network is similar to a 2D mesh network, in general, we can take the time of spreading as the adjusted Manhattan distance and use it as $h(x)$.

In a square map, let the coordinates of the source node be (x_0, y_0), the coordinates of the current node be (x_c, y_c), and the coordinates of the destination node be (x_n, y_n). The adjusted Manhattan distance can be calculated as:

$$h(x) = \frac{|x_c - x_n| + |y_c - y_n|}{v}; \tag{13}$$

Since the more information in $h(x)$, the fewer states are processed by the A-Star algorithm and therefore more efficient. Also, if $h(x)$ is sometimes larger than the actual distance from the current node c to the destination node n, A-Star may arrive at a non-optimal solution.

Therefore, the adjusted Manhattan distance can be improved for this scenario as follows:

Let the actual latitude of the source node be lat_0, then: the latitude of the current node lat_c can be calculated as:

$$lat_c = lat_0 + (y_c - y_0)\frac{360^\circ}{M}; \tag{14}$$

the latitude of the destination node lat_n can be calculated as:

$$lat_n = lat_0 + (y_n - y_0)\frac{360^\circ}{M}; \tag{15}$$

Therefore, we can calculate the vertical distance Δy and horizontal distance Δx as:

$$\Delta y = |y_n - y_c| L_v; \tag{16}$$

$$\Delta x = \alpha \cos(max\{lat_c, lat_n\})|x_n - x_c|; \tag{17}$$

where L_v and α are given by (2) and (4).

Finally, we get $h(x)$ as:

$$h(x) = \frac{\Delta x + \Delta y}{v}; \tag{18}$$

When updating $g(x)$, we prioritize whether the $g(x)$ of the node to be expanded to is greater than the value of $g(x)$ of the current node adds the t value needed to reach the vertical grid. If it is, the next hop is selected along the vertical direction, otherwise, the next hop is selected horizontally at the current latitude.

Algorithm: A Heuristic Inter-Satellite Fault Tolerant Routing Mechanism Based on A-Star Algorithm

---

Input: coordinates of the source node, destination node, and the failures; the value of transmission delay and processing delay.

Initialize: $f(x_0) = g(x_0) = h(x_0) = 0$; openlist and closelist as **null**;

Place start_node in openlist;

do{
Select the node with the smallest value of $f(x)$ from openlist as the current node.
Remove from current node from openlist and place it in closelist;
Put its neighbor nodes that are not in closelist into openlist;
Calculate the value of $g(x)$ and $h(x)$;
if (neighbor node in the openlist){
Update the value of $g(x)$ to a smaller value;
Decide to search vertically or horizontally;
Update the parent node to the current node;
}
}
until (the end point is found **or** openlist is empty)
end
Output:
if (end point is found)
Follow the pointer of the parent node from the end point,
until it goes back to the starting point to get the shortest path;
else
No solution.

4 Simulation Results and Analysis

4.1 Simulation Model

The simulation aims to evaluate the performance of the proposed routing recovery scheme. We have chosen the shortest path algorithm represented by Dijkstra and the original A-Star algorithm to compare with our scheme.

We perform the simulation experiments with Matlab. The constellation parameters are shown in Table 1. We use a polar orbit LEO satellite constellation similar to OneWeb. When the satellite crosses the polar region (ISL latitude over 70°), the ISL is turned off. And there is no link connecting the satellites in the reverse rotating orbit.

Table 1. Simulation Model Parameters

Parameter	Value
Type of Orbit	Polar Orbit
Satellites altitude	1200 km
Number of Planes	18
Number of Satellites per Plane	40
Inclination	87.9°
Number of ISLs in Intra-Plane	2
Number of ISLs in Inter-Plane	0 or 2

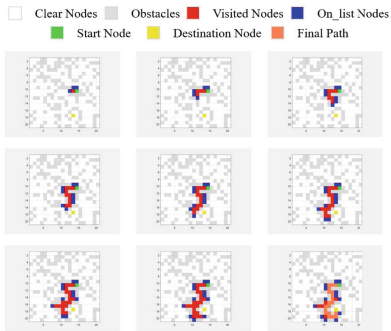
Meanwhile, to simplify the calculation, set the transmission delay and processing delay as fixed values.

We have done three sets of experiments. The first set is used to compare the time costs, search number, and path hops of Dijkstra, the original A-Star, and proposed mechanism without failure. The second experiment explores the recovery path hop count with recovery time and reliability under different algorithms. The third experiment analyzes the path-finding time of different algorithms with different path lengths and the average path-finding time with different network sizes.

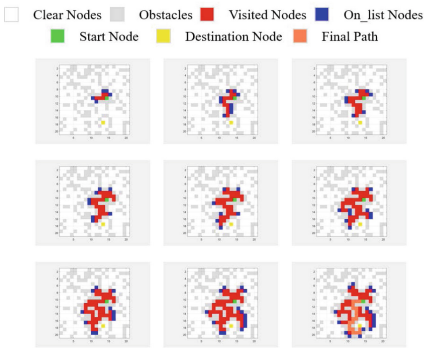
4.2 Time to Recover Path

Dijkstra, the original A-Star, and the proposed mechanism all require convergence time to obtain the link state at LEO layer and compute the optimal path.

When the network size is 20×20 and the failure rate is 12%, the path-finding process of the proposed mechanism and the original A-Star algorithm are obtained as they are shown in Fig. 5.



(a) Proposed Mechanism



(b) Original A-Star Algorithm

Fig. 5. Path-Finding Process of Different Algorithms

From the figure, the original A-Star algorithm directly unfolds a small local flood when it encounters an obstacle, to find the optimal path. While the proposed mechanism prioritizes path-finding in the longitudinal direction after considering that the destination node is above the starting node, and then performs a horizontal flood search when it basically reaches the height of the longitudinal coordinate of the destination node.

Run this 500 times and count TIMECOSTS and EXPANDED NUMBER. The results are shown in Fig. 6 and Fig. 7.

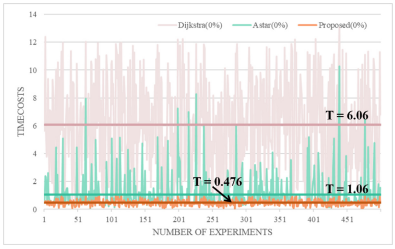


Fig. 6. Time Cost

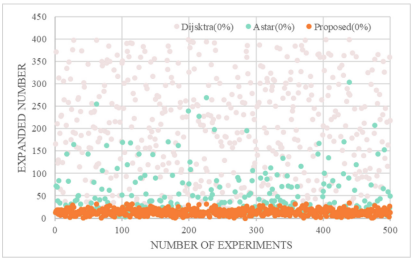


Fig. 7. Expanded Number

Since different computers have different arithmetic power, and that may lead to different path-finding time under the same condition, a vertical lattice in TIMECOSTS in the figure represents a unit time Δt_0. Thus, TIMECOSTS itself represents the time complexity of the algorithm. Expanded Number represents the number of lattices explored to reach the optimal path.

It can be seen from Fig. 6 that the average route establishment TIMECOSTS of our proposed mechanism is 0.47 units of time, which is lower than the average establishment time of LAOR protocol by about 1.1 units of time [3]. LAOR protocol is an on-demand routing protocol especially suitable for LEO networks, which uses minimum propagation delay to calculate the optimal path.

To measure the performance of the mechanism, we run the simulation for 10,0000 times and calculate the difference rate between the path hop count of the proposed mechanism and the hop counts of the other two algorithms.

It can be seen from Fig. 8 that the proposed mechanism can basically reach parity with the first two algorithms in terms of the number of hops for the same path, with deviations greater than 0.5 occurring only with a probability close to 10^{-3}. However, the convergence time of our proposed approach is 55.161% shorter than the original A-Star algorithm and 92.142% shorter than Dijkstra algorithm, and it can still accomplish more efficient path-finding in a more effective search.

4.3 Routing Recovery and Reliability

Routing Recovery Performance and Time. During the routing process, it is possible that the satellite itself may fail, or that a satellite shown as normal in

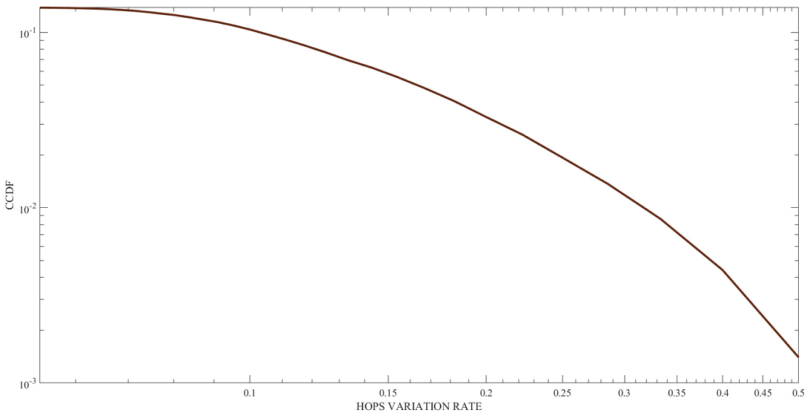


Fig. 8. Hop Count Difference Rate CCDF

the routing topology at this time may be turned off while passing through the polar region and cause the link to fail.

Figure 9 presents the recovery time in terms of TIMECOSTS and Fig. 10 presents the recovery performance in terms of the number of routing hops obtained by re-routing at failure rates of 4%, 8%, 12%, and 16%, respectively.

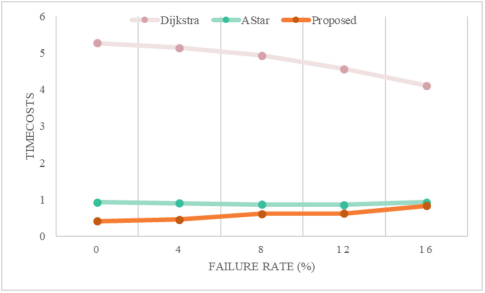


Fig. 9. Time Cost under Different Failure Rates

It can be seen from Fig. 9 that the proposed mechanism reduces the time overhead when processing the task, so the mechanism takes less time to execute. This means that the mechanism can provide efficient path-planning solutions in a shorter period.

In addition, it can be seen from Fig. 10 that even at higher link failure rates, there is only a probability less than 10^{-2} that our proposed mechanism will generate less than 0.2 more paths than the previous two algorithms; and 0.5 more paths at a probability of 10^{-5}. This probability is small enough and acceptable.

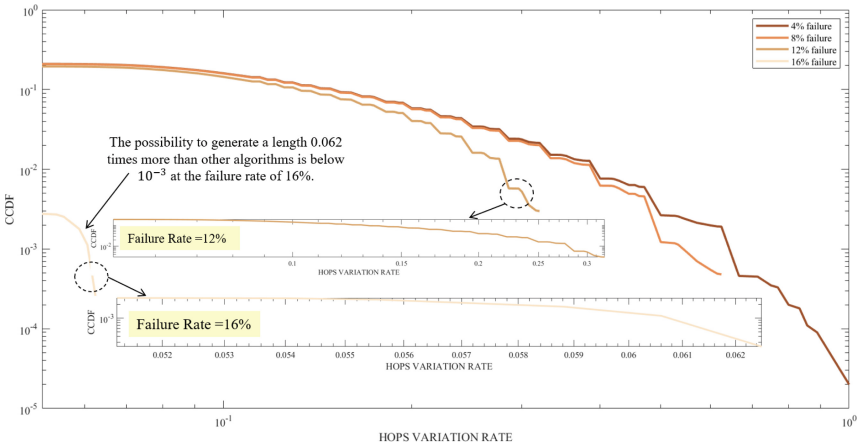


Fig. 10. Hop Count Difference Rate CCDF under Different Failure Rates

Reliability. In this set of experiments, we compare the reliability of the three sets of algorithms at different failure rates. Reliability means that the route can be re-established within 2 time units after a route failure is detected.

In Fig. 11, it can be seen that at a scale of 20×20, the proposed mechanism, has the highest reliability at all the above failure rates due to its advantage over the original A-Star algorithm in TIMECOSTS.

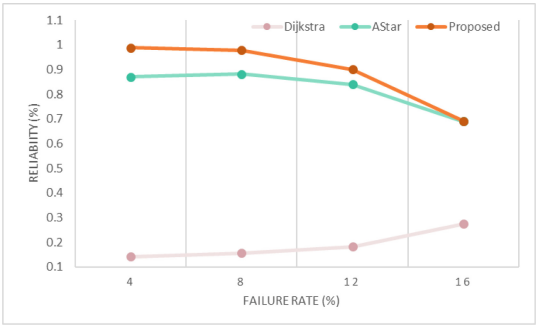


Fig. 11. Reliability under Different Failure Rates

As the number of faults increases, the number of lattices that need to be explored is decreasing, so Dijkstra's algorithm appears to have improved performance as the fault rate increases, while our proposed mechanism and the original A-Star algorithm have reduced reliability due to the number of lattices that need to be explored due to having to work around obstacles.

Network Scale. With the development of mega-constellations, the number of satellites is increasing and the network size is about to get bigger and bigger.

The expansion of network size is inevitably accompanied by the basic situation of multi-hop transmission and routing over longer distances with each hop. In this part of the simulation, we explore the time required to route three different algorithms for different path lengths, and the time complexity of our proposed mechanism compared to the original A-Star algorithm for different network sizes and different failure rates.

Fig. 12. Time Cost under Different Path Lengths

In Fig. 12, it is obvious that Dijkstra's algorithm increases the path-finding time near linearly as the distance increases, and the original A-Star algorithm has better performance at shorter distances, while it reflects stronger volatility when the distance is greater than about 15000km. In contrast, our proposed mechanism can basically stabilize the path-finding time below 2 time units regardless of the fault condition and the change in path length.

As the network scale increases, the pathfinding time is bound to increase as well. In Fig. 13, it is obvious that the TIMECOSTS of both algorithms increases as the failure rates increases, and our proposed mechanism always has some advantages over the original A-Star algorithm. At a small network size (20×20), the original A-Star algorithm and our proposed mechanism have basically the same performance, while the advantage of the proposed mechanism gradually expands as the network size increases.

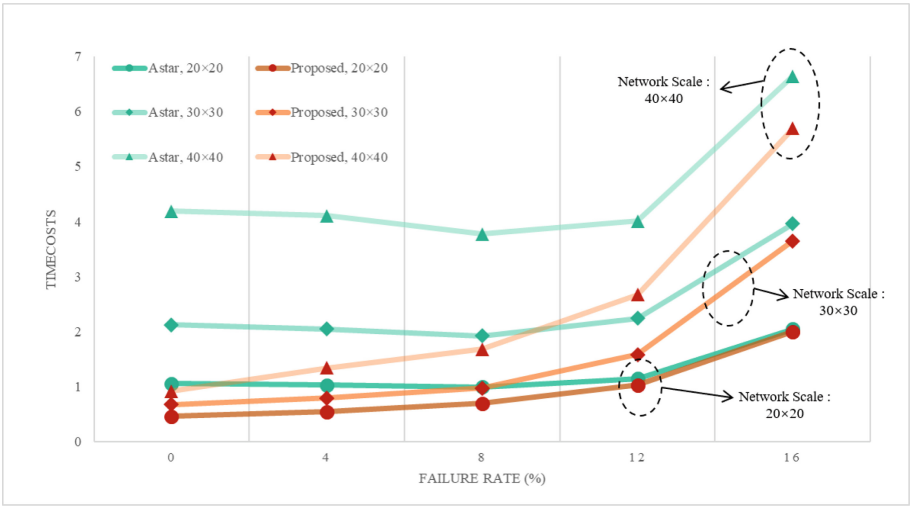


Fig. 13. Time Cost under Different Network Scales

5 Conclusion

In this work, we have proposed and evaluated an A-Star algorithm-based routing mechanism for large-scale satellite networks. The mechanism we have designed aims to reduce the number of satellites involved in the path discovery process by exploiting the deterministic dynamics of satellite motion and optimizing the heuristic function. Simulation results have shown that our proposed mechanism has outperformed the original A-Star algorithm in terms of path establishment time under different failure rates and network scales. Also, it can essentially maintain parity with it in terms of average end-to-end delay.

With the development of mega-constellation, the network scale becomes more enormous, and so is the topology to be more complex and variable. The complexity of interstellar links requires low-delay and high-efficiency routing, while our proposed fast routing mechanism can realize the timely interconnection of satellites in different orbital planes. In the future, our fast routing mechanism may help provide high-speed and reliable Internet access to remote areas without terrestrial base stations, supporting power inspection, emergency protection and other tasks.

References

1. Burleigh, S., et al.: Delay-tolerant networking: an approach to interplanetary internet. IEEE Commun. Mag. **41**(6), 128–136 (2003)
2. Werner, M., Berndl, G., Edmaier, B.: Performance of optimized routing in LEO inter-satellite link networks. In: 1997 IEEE 47th Vehicular Technology Conference. Technology in Motion, pp. 246–250. IEEE, Phoenix (1997)

3. Ji, X., Liu, L., Zhao, P., Wang, D.: A-star algorithm based on-demand routing protocol for hierarchical LEO/MEO satellite networks. In: 2015 IEEE International Conference on Big Data (Big Data), pp. 1545–1549. IEEE, Santa Clara (2015)
4. Wang, H., Zhou, J., Zheng, G., Liang, Y.: HAS: Hierarchical A-Star algorithm for big map navigation in special areas. In: 2014 5th International Conference on Digital Home, pp. 222–225. IEEE, Guangzhou (2014)
5. Xu, Z., Liu, X., Chen, Q.: Application of improved Astar algorithm in global path planning of unmanned vehicles. In: 2019 Chinese Automation Congress (CAC), pp. 2075–2080. IEEE, Hangzhou (2019)
6. Zhang, Z., Wang, S., Zhou, J.: A-star algorithm for expanding the number of search directions in path planning. In: 2021 2nd International Seminar on Artificial Intelligence, Networking and Information Technology (AINIT), pp. 208–211. IEEE, Shanghai (2021)
7. Chen, S., Ji, J., Su, H., Yang, Y.: Improved A-star method for collision avoidance and path smoothing. In: 2023 IEEE International Conference on Control, Electronics and Computer Technology (ICCECT), pp. 32–35. IEEE, Jilin (2023)
8. Fan, G., Xing, X., Han, Y., Chen, M., Gui, H.: Path planning for ground target reconnaissance based on improved Astar algorithm. In: 2021 China Automation Congress (CAC), pp. 2092–2097. IEEE, Beijing (2021)
9. Zhou, Q., Liu, G.: UAV path planning based on the combination of A-star algorithm and RRT-star algorithm. In: 2022 IEEE International Conference on Unmanned Systems (ICUS), pp. 146–151. IEEE, Guangzhou (2022)
10. Cao, Y., Huang, Y., Ma, Z.: Ship route planning method based on optimized A-star algorithm. In: 2022 International Conference on Cloud Computing, Big Data Applications and Software Engineering (CBASE), pp. 242–245. IEEE, Suzhou (2022)
11. Li, Z., Li, X.: Destruction-resistant routing protocol based on topology prediction and backup path. In: 2022 7th International Conference on Multimedia Communication Technologies (ICMCT), pp. 36–40. IEEE, Xiamen (2022)
12. Werner, M.: A dynamic routing concept for ATM-based satellite personal communication networks. IEEE J. Sel. Areas Commun. **15**(8), 1636–1648 (1997)
13. Zhang, L., et al.: A routing algorithm based on link state information for LEO satellite networks. In: 2020 IEEE Globecom Workshops (GC Wkshps), pp. 1–6. IEEE, Taipei (2020)
14. Ekici, E., Akyildiz, I.F., Bender, M.D.: Datagram routing algorithm for LEO satellite networks. In: Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies, pp. 500–508. IEEE, Tel Aviv (2000)
15. Werner, M., Delucchi, C., Vogel, H.J., Maral, G., De Ridder, J.J.: ATM-based routing in LEO/MEO satellite networks with intersatellite links. IEEE J. Sel. Areas Commun. **15**(1), 69–82 (1997)