Unpublished: Length of the Shortest Distance in Triangular Sphere Grid Networks

Seung-Kyu Choi , *Member, IEEE,* Illsoo Sohn

***Abstract*—This study proposes an algorithm for efficiently calculating the shortest distance between two nodes in a triangular grid network based on a regular icosahedron. Previous research relied on computationally intensive methods for distance calculation, making them difficult to apply to large-scale networks. To address this, we designed an efficient algorithm with a computational complexity of O(1), aiming to maximize data transmission efficiency and resource utilization in spherical mesh networks. The algorithm's accuracy and efficiency were verified through experiments conducted using MATLAB, demonstrating superior performance across various values of n. The proposed algorithm shows potential applications in diverse fields such as satellite networks, weather modeling, and geographic data analysis, and is expected to contribute to optimizing network performance.**

***Index Terms*—Triangular Sphere, Satellite Mesh, Shortest Distance, Coordinate System.**

I. Introduction

In modern network systems, efficient data delivery and distance computation between nodes are critical research topics. Particularly in mesh networks, such as wireless communication networks, efficient data transmission plays a crucial role in determining network performance. This study addresses the problem of calculating the shortest distance between two nodes in a triangular grid-based network to evaluate and optimize network performance.

In satellite networks with spherical structures, such as Earth-like systems, triangular grid-based mesh structures are often employed due to their symmetry and efficiency. While previous implementations have utilized triangular grids based on regular octahedrons, this paper adopts a regular icosahedron as the foundation. The triangular grid is formed by subdividing each triangle into smaller segments and arranging nodes accordingly. The objective of this study is to propose an algorithm that efficiently computes the shortest distance between two nodes in such a spherical triangular grid and to provide metrics for evaluating network performance.

In particular, identifying the minimum Time-To-Live (TTL) value is essential for improving data transmission efficiency in mesh networks. TTL refers to the maximum number of hops a data packet can take, and minimizing this value is directly related to the efficient use of network resources. This study addresses the shortest distance computation problem on a spherical surface rather than on a simple 2D plane, aiming to overcome the limitations of existing research.

Existing studies on shortest distance computation between nodes in spherical mesh networks have primarily proposed algorithms based on octahedral or non-spherical structures. However, such structures often cause distortions in the triangular grid, leading to uneven node spacing and increased computational complexity. For instance, a 2009 study[1] introduced an algorithm with a computational complexity of $O(n \log n)$ to compute the shortest distances between a pair of nodes and $O(n)$ in [2]. While effective, this approach is inefficient for large-scale networks. In contrast, this study aims to overcome these limitations by proposing an algorithm with a computational complexity of $O(1)$.

This research utilizes a triangular grid based on a regular icosahedron to maximize structural symmetry and minimize network distortion. By doing so, the study simplifies distance computation in spherical mesh networks while achieving both computational accuracy and efficiency.

The main objectives of this study are as follows:

1) Design an algorithm for efficiently calculating the shortest distance between two nodes in a spherical triangular grid.
2) Minimize the algorithm's computational complexity to ensure scalability for large-scale networks.
3) Provide network performance evaluation metrics based on the minimum Time-To-Live (TTL) value, exploring practical applications such as satellite networks.

The proposed algorithm assigns 12-dimensional coordinates to each node in the spherical triangular grid, enabling precise computation of shortest distances between nodes. By leveraging the symmetry and repetitive patterns of the spherical mesh network, the algorithm aims to maximize efficiency. This approach contributes to enhancing data transmission efficiency in large-scale network environments.

II. Related Work

The main prior studies related to this research are as follows:

- **Kunszt et al.**[3], [4] proposed a mesh network structure using a triangular grid based on a regular octahedron, but it had the limitation of complex distance calculations due to structural distortions.
- **Balasubramanian et al.**[1] proposed a shortest distance calculation algorithm in a pseudo-spherical structure with a computational complexity of $O(n \log n)$, which, however, was not suitable for large-scale networks.

S. Choi was with the Department of Computer Engineering, Seoul National University of Science and Technology, Seoul, 01811 RoK e-mail: senkyu.choi@seoultech.ac.kr.

S. Choi and I. Sohn are with Seoul National University of Science and Technology.

Manuscript received April 19, 20–; revised August 26, 20–.

- **Bhalla et al.**[5] suggested a contour-based coordinate setting method in triangular grids, but there is a slight difference in approach.

To overcome the limitations of these prior studies, this research proposes an efficient shortest distance calculation algorithm based on a regular icosahedron triangular grid with a computational complexity of $O(1)$. The algorithm calculates the shortest distance within each cylinder and then combines the results to derive the final outcome. This approach is expected to significantly enhance the performance of mesh networks.

III. Main Body

Structure of the Spherical Triangular Grid

The spherical triangular grid is formed based on a regular icosahedron. Each triangular face is of equal size and is subdivided into smaller equilateral triangles. In this process, the position of each node is defined using spherical coordinates, with 12-dimensional coordinates assigned based on 6 symmetric polar points. This coordinate system simplifies distance calculations between all nodes in the network while maintaining structural symmetry and maximizing efficiency.

When contour lines are drawn from a pole, the number of nodes on each line can be represented in this way.
$n = 1$, 1+(5+5)+1
$n = 2$, 1+5+(10+10+10)+5+1
$n = 3$, 1+5+10+(15+15+15+15)+10+5+1
$n = 4$, 1+5+10+15+(20+20+20+20+20)+15+10+5+1
$n = 5$, 1+5+…20+(25+25+25+25+25+25)+20+15…
$n = 6$, 1+…+25+(30+30+30+30+30+30+30)+25+20…
The total number of nodes is $10n^2 + 2$. The area in parentheses is the gray area of Fig. 2.

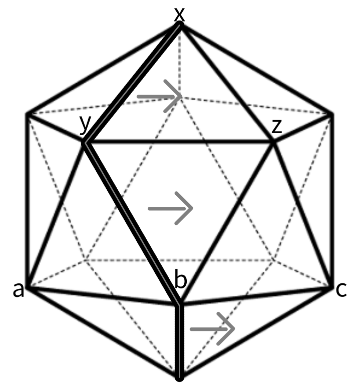

Fig. 1. Order of Coodinates Generation

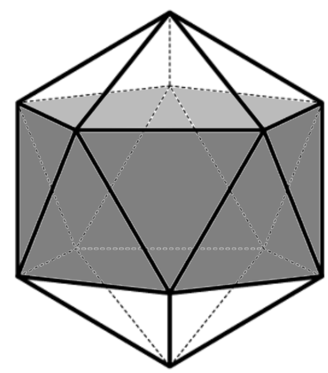

Fig. 2. Cylinder or Mid Earth Orbit(MEO) area

Shortest Distance Calculation Algorithm

The algorithm proposed in this study follows the steps below to compute the shortest distance in the spherical triangular grid:

1) **Initialization step:**
   - For each node in the network, draw latitude lines(row numbers) based on 6 types of polar points(Fig. 1) to create a 6-dimensional coordinate system.
   - Additional longitude coordinates(column numbers) are generated at the mid-orbit points based on the 6 types of polar points(Fig. 2), resulting in a total of 12-dimensional coordinates.
   - Each node belongs to at least one of the 6 cylinders, and distance calculations are performed within the corresponding cylinder.
2) **Shortest distance calculation within a cylinder:**
   - The distance between nodes within each cylinder is calculated based on row and column numbers.
   - The algorithm efficiently calculates the shortest distance by utilizing the repetitive patterns within the cylinder. The difference between row and column coordinates is compared to determine the shortest path length.
3) **Multiple cylinder comparison:**
   - Identify the cylinders to which the two nodes belong and compare the distances calculated within each cylinder.
   - The minimum value is selected to derive the final shortest distance.

As shown in the figure, the six polar points (x, y, z, a, b, c) are considered North Poles, with contour lines drawn from each north pole, which are row numbers starting from 1. When holding vertically for each set of poles, column numbers are assigned in order starting from 1 in right-hand direction, starting point being respectively, x to y North Pole, y to x North Pole, z to x North Pole, a to y North Pole, b to z North Pole, and c to a South Pole. Coordinates generator is in Github:https://github.com/senchoi/geotrigrid_coord_gen.

The shortest distance between two nodes must exist within at least one of the six cylinders. By checking each cylinder with conditional statements to calculate the shortest distance within that cylinder, we gather 2-4 values. Among these, the smallest value becomes the final shortest distance.

When calling the *alg_shortest* command, subtracting n from the row coordinate results in the first cylinder row starting from 1 regardless of the value of n. This gives the effect of slightly shifting even-numbered rows to the left, ensuring that the algorithm inside *alg_shortest* performs accurate calculations.

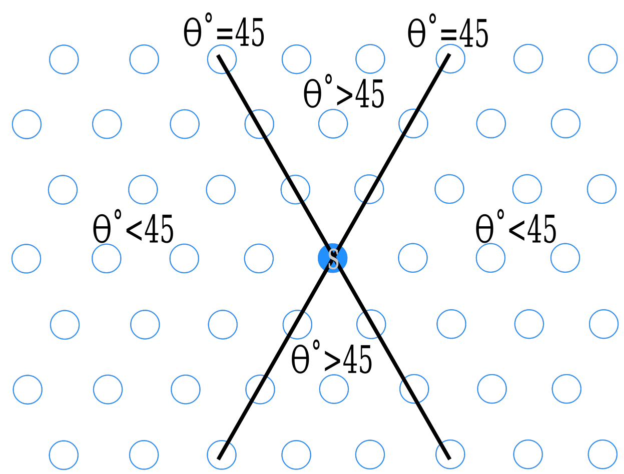

Fig. 3. Classification Angle depending on location of destination

Algorithm 1 Shortest Distance Algorithm

function $alg\_shortest(x_1, y_1, x_2, y_2, n)$
  $source \leftarrow [x_1, y_1]$
  $destination \leftarrow [x_2, y_2]$
  if $source[1]$ is even **then**
    $source[2] \leftarrow source[2] \times 2 - 1$
  else
    $source[2] \leftarrow source[2] \times 2$
  end if
  if $destination[1]$ is even **then**
    $destination[2] \leftarrow destination[2] \times 2 - 1$
  else
    $destination[2] \leftarrow destination[2] \times 2$
  end if
  $a \leftarrow |source[1] - destination[1]|$
  $b \leftarrow |source[2] - destination[2]|$
  $b_{cir} \leftarrow \min(b, |5n \times 2 - b|)$
  $\theta_{rad} \leftarrow \text{atan2}(a, b_{cir})$
  $\theta_{deg} \leftarrow \theta_{rad} \times \frac{180}{\pi}$
  if $\theta_{deg} < 45$ **then** //refer to Fig. 3
    $answer \leftarrow \frac{a+b_{cir}}{2}$
  else
    $answer \leftarrow a$
  end if
  return $answer$
end function

Algorithm Complexity Analysis

The main advantage of this algorithm is its $O(1)$ computational complexity, which means that distance calculations can be performed in real-time, even in large-scale networks. Previous studies computed shortest distances with complexities of $O(N^3)$ or $O(N \log N)$, but this research significantly improves upon these approaches by utilizing repetitive patterns and symmetrical structures.

Algorithm Verification and Performance Evaluation

The performance of the algorithm was verified using MATLAB. A node generator and distance calculation algorithm

Algorithm 2 Check all six cylinders

$Source \leftarrow (x_1, y_1, z_1, a_1, b_1, c_1,$
$xc_1, yc_1, zc_1, ac_1, bc_1, cc_1)$
$Destination \leftarrow (x_2, y_2, z_2, a_2, b_2, c_2,$
$xc_2, yc_2, zc_2, ac_2, bc_2, cc_2)$
$Answer\_vec \leftarrow [\infty, \infty, \infty, \infty, \infty, \infty]$
if $n+1 \leq x_1 \leq 2n+1$ and $n+1 \leq x_2 \leq 2n+1$ **then**
  $Answer\_vec[1] \leftarrow alg\_shortest(x_1 - n, xc_1, x_2 - n, xc_2, n)$
end if
if $n+1 \leq y_1 \leq 2n+1$ and $n+1 \leq y_2 \leq 2n+1$ **then**
  $Answer\_vec[2] \leftarrow alg\_shortest(y_1 - n, yc_1, y_2 - n, yc_2, n)$
end if
if $n+1 \leq z_1 \leq 2n+1$ and $n+1 \leq z_2 \leq 2n+1$ **then**
  $Answer\_vec[3] \leftarrow alg\_shortest(z_1 - n, zc_1, z_2 - n, zc_2, n)$
end if
if $n+1 \leq a_1 \leq 2n+1$ and $n+1 \leq a_2 \leq 2n+1$ **then**
  $Answer\_vec[4] \leftarrow alg\_shortest(a_1 - n, ac_1, a_2 - n, ac_2, n)$
end if
if $n+1 \leq b_1 \leq 2n+1$ and $n+1 \leq b_2 \leq 2n+1$ **then**
  $Answer\_vec[5] \leftarrow alg\_shortest(b_1 - n, bc_1, b_2 - n, bc_2, n)$
end if
if $n+1 \leq c_1 \leq 2n+1$ and $n+1 \leq c_2 \leq 2n+1$ **then**
  $Answer\_vec[6] \leftarrow alg\_shortest(c_1 - n, cc_1, c_2 - n, cc_2, n)$
end if
return $\min(Answer\_vec)$

were implemented, and experiments were conducted for various values of n. The results showed that the algorithm correctly derived the shortest distance for all test cases, with computation time greatly reduced compared to existing algorithms.

Applicability

This algorithm can be applied not only to spherical mesh networks like satellite networks but also to fields such as geographical data analysis and weather prediction modeling. It is expected to generate practical results by enhancing data transmission efficiency and maximizing resource utilization in large-scale networks.

Future Research

As satellites become increasingly difficult to orbit closer to the polar regions, three types of polar points must be carefully selected, and only mid-orbit satellites should be launched to cover the entire spherical area. This approach will create collision points between three types of satellite groups when all satellites are at the same altitude. To avoid this, satellites must be deployed at different altitudes, and the movement speeds of different satellites should be considered when assigning coordinates according to the proposed system in this study.

IV. Conclusion

This study proposes a new algorithm for efficiently calculating the shortest distance between two nodes in a spherical triangular grid-based network. By utilizing the symmetrical structure of a regular icosahedron, the computational complexity is reduced to $O(1)$ compared to previous methods, demonstrating its potential for real-time application even in large-scale network environments. The experimental results showed that the algorithm exhibited high accuracy and efficiency, with significantly improved computation speed compared to existing complex algorithms.

Moreover, the proposed algorithm shows potential applications in various fields such as satellite networks, geographical data analysis, and weather modeling. It can play a crucial role in network performance evaluation and optimization. Future research will focus on applying the proposed algorithm in real-world network environments to further verify and improve its performance. This is expected to further expand the potential of triangular grid-based networks.

Appendix A
Layout of a sphere when $n = 3$

Here is the link of a layout for readers to validate. https://github.com/senchoi/geotrigrid_coord_gen/blob/main/fig_last.pdf. Notice that just (x,y,z) coordinate is enough to locate each node, which is how the generator code at the end lists as two coordinates and distance at the last column that makes the total of seven columns.

Acknowledgment

The authors would like to thank...

References

[1] M. Balasubramanian, J. R. Polimeni and E. L. Schwartz, "Exact Geodesics and Shortest Paths on Polyhedral Surfaces," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, no. 6, pp. 1006-1016, June 2009, doi: 10.1109/TPAMI.2008.213.

[2] Biswas, R., Bhowmick, P. (2014). On Finding Spherical Geodesic Paths and Circles in Z^3 . In: Barcucci, E., Frosini, A., Rinaldi, S. (eds) Discrete Geometry for Computer Imagery. DGCI 2014. Lecture Notes in Computer Science, vol 8668. Springer, Cham. https://doi.org/10.1007/978-3-319-09955-2_33

[3] Kunszt, P.Z., Szalay, A.S., Thakar, A.R. The Hierarchical Triangular Mesh. In: Banday, A.J., Zaroubi, S., Bartelmann, M. (eds) Mining the Sky. ESO ASTROPHYSICS SYMPOSIA. Springer, Berlin, Heidelberg. https://doi.org/10.1007/10849171_83

[4] Barrett, Paul E.. "Application of the Linear Quadtree to Astronomical Databases." (1995).

[5] Bhalla, G., Bhowmick, P. (2016). DIG: Discrete Iso-contour Geodesics for Topological Analysis of Voxelized Objects. In: Bac, A., Mari, JL. (eds) Computational Topology in Image Context. CTIC 2016. Lecture Notes in Computer Science(), vol 9667. Springer, Cham. https://doi.org/10.1007/978-3-319-39441-1_24

PLACE PHOTO HERE

Michael Shell Biography text here.

John Doe Biography text here.

Jane Doe Biography text here.