


Article

SD-GPSR: A Software-Defined Greedy Perimeter Stateless Routing Method Based on Geographic Location Information

Shaopei Gao ¹, Qiang Liu ^{1,*} , Junjie Zeng ² and Li Li ³

¹ Beijing Key Lab of Transportation Data Analysis and Mining, School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China; 22120366@bjtu.edu.cn

² Institute of Telecommunication and Navigation Satellites, China Academy of Space Technology, Beijing 100094, China

³ Wuhan Maritime Communication Research Institute, Wuhan 430079, China; lilywork722@163.com

* Correspondence: liuq@bjtu.edu.cn

Abstract: To mitigate the control overhead of Software-Defined Mobile Ad Hoc Networks (SD-MANETs), this paper proposes a novel approach, termed Software-Defined Greedy Perimeter Stateless Routing (SD-GPSR), which integrates geographical location information. SD-GPSR optimizes routing functions by decentralizing them within the data plane of SD-MANET, utilizing the geographic location information of nodes to enhance routing efficiency. The controller is primarily responsible for providing location services and facilitating partial centralized decision-making. Within the data plane, nodes employ an enhanced distance and angle-based greedy forwarding algorithm, denoted as GPSR_DA, to efficiently forward data. Additionally, to address the issue of routing voids in the data plane, we employ the A^* algorithm to compute an optimal routing path that circumvents such voids. Finally, we conducted a comparative analysis with several state-of-the-art approaches. The evaluation experiments demonstrate that SD-GPSR significantly reduces the control overhead of the network. Simultaneously, there is a notable improvement in both end-to-end latency and packet loss rate across the network.

Keywords: Software-Defined Networking (SDN); routing computation; Mobile Ad-Hoc Networks (MANET); A^* algorithm



Citation: Gao, S.; Liu, Q.; Zeng, J.; Li, L. SD-GPSR: A Software-Defined Greedy Perimeter Stateless Routing Method Based on Geographic Location Information. *Future Internet* **2024**, *16*, 251. <https://doi.org/10.3390/fi16070251>

Academic Editor: Franco Davoli

Received: 17 April 2024

Revised: 11 July 2024

Accepted: 11 July 2024

Published: 17 July 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The characteristic separation of the control plane and data plane in Software-Defined Networking (SDN) facilitates network management through software programming, enhancing network resource utilization and driving the evolution and innovation of traditional networks [1,2]. In recent years, the growing demands of novel applications in SD-Mobile Ad-Hoc Networks (SD-MANETs) have progressively increased, encompassing aspects such as network resource monitoring, Quality of Service (QoS) for communication services, network security services, and routing paths [3,4]. This trend has led to an escalation in the complexity of control plane functionalities, resulting in heavier control overhead. In the architecture of SD-MANETs, control strategies are predominantly concentrated within the controller, with data forwarding relying on efficient and straightforward flow table matching methods. However, Mobile Ad-Hoc Networks (MANETs) exhibit complex and dynamic characteristics [5], including continuously changing topologies and weak wireless signals. These factors easily lead to issues such as flow table expiration and disconnection between data nodes and control nodes, significantly impacting network performance [6].

Routing protocols, serving as pivotal technologies for data transmission in SD-MANETs, currently predominantly employ a centralized, topology-based routing approach [7,8]. Controllers are tasked with protocol parsing, network resource monitoring, data storage, flow

table provisioning, and maintenance, leading to considerable control overhead. With advancements in satellite-based precision positioning systems (e.g., BeiDou, GPS), utilizing the geographical location information of nodes to aid in routing has emerged as a strategy to enhance node self-organization, consequently improving routing efficiency [9,10]. Among various geographically assisted routing protocols, Greedy Perimeter Stateless Routing (GPSR) stands out by utilizing only one-hop neighbor information through a greedy algorithm to obtain an optimal next-hop forwarding node [11], eliminating the need for the dynamic establishment of complete routes or the dynamic maintenance of flow tables, thereby achieving simplicity and low routing overhead.

Mitigating the issue of excessive control overhead in SD-MANETs, this paper proposes a Geographical Information-Based Software-Defined Greedy Perimeter Stateless Routing method (SD-GPSR). SD-GPSR utilizes the controller to provide location services, employing a distance and angle-based greedy forwarding algorithm in the data plane for data transmission. To tackle issues within the data plane such as routing voids and link interruptions, SDN technology is integrated to present a solution, rendering it more suitable for SD-MANET environments.

Specifically, we have made the following contributions:

- An SD-GPSR routing method is proposed. SD-GPSR decentralizes some of the routing functions in SD-MANET to the data plane and utilizes the node's geographic location information to achieve routing.
- An improved distance and angle-based greedy forwarding algorithm (GPSR_DA) is proposed. The controller plays a key role in providing location services and facilitating partial centralized decision-making, while the nodes in the data plane employ the GPSR_DA algorithm for data forwarding. This enhances node self-organization and overall routing efficiency and reduces control overhead.
- Addressing the issue of routing voids during the forwarding process, this paper introduces a solution by employing the A^* algorithm to calculate an optimal route that avoids these voids. Subsequently, the source routing concept is utilized to deploy flow tables, effectively reducing control overhead.
- We validate and analyze the SD-GPSR routing method.

The rest of this paper is organized as follows. Section 2 presents the related work. Section 3 presents the proposed SD-GPSR including architecture, control layer design, and data layer design. Section 4 describes the specific implementation process. Section 5 describes the simulation results and performance comparison. Section 6 summarizes the whole paper.

2. Related Work

In traditional MANETs, all routing processes are implemented directly on nodes, meeting the basic communication requirements of wireless environments [12]. S-DSR [13] represents a novel hybrid security routing system within MANETs, ensuring secure data packet transmission and optimal performance across network nodes. This protocol utilizes trust information from neighboring nodes to select the safest routes for file transmission. Nirmaladevi et al. [14] proposed a Trust-Optimized Cluster-Routing Algorithm based on Selfish Nodes (SN-TOCRP), adhering to the concept of layered clustering to create node clusters.

In recent years, the effective reduction in control overhead while enabling flexible network management through controllers has emerged as a crucial challenge in SD-MANETs. Dusia et al. [15] have proposed an SD-MANET architecture and designed three centralized communication protocols to meet the demands of active, passive, and layered routing strategies within MANETs, addressing the inherent challenges of the network. Maruthupandi et al. [16] introduced a unique routing protocol called Distinct Network Yarning (DNY) within the SD-MANET framework. In this protocol, the controller stores information about the entire SDN system's operational functionalities for congestion control. Gilani et al. [17] proposed a Geographic Routing Protocol suitable for Vehicular

Ad-Hoc Networks (VANETs) named SDN-Based Geographic Routing (SDGP). This protocol employs the controller to gather vehicle information within the network, enabling the computation of optimal paths through a network-wide view, thereby facilitating intelligent and secure routing in VANETs. Abbas et al. [18] introduced a performance-enhanced routing protocol specifically designed for Infrastructure-Assisted Vehicular Networks. This protocol explores the use of cellular networks to relay control information to the controller in a low-latency manner, exhibiting good availability under limited routing overhead.

Among all the proposed routing decisions mentioned above, there is a certain degree of reduction in the occurrence probability of issues such as flow table expiration and disconnection between data nodes and control nodes, thereby enhancing routing effectiveness. However, as these issues are tackled, there is a tendency for more control operations to be centralized at the controller, resulting in a rise in control overhead.

3. SD-GPSR Routing Method

This paper proposes an SD-GPSR routing method to reduce the control overhead in SDN routing technology while effectively improving the routing efficiency of SD-MANET.

3.1. SDN Architecture Integrating SD-GPSR

The SD-GPSR routing method requires the periodic exchange of beacons between SDN switches in the data plane for neighbor discovery and maintenance. It also employs flow table matching and greedy forwarding strategy for data forwarding. The SDN controller in the control plane dynamically discovers and maintains global position information.

As shown in Figure 1, we integrate the SD-GPSR routing method into the existing SDN architecture. Compared to traditional centralized routing methods, SD-GPSR reduces control overhead and leverages the advantages of the controller in resource monitoring, network security, and QoS. Based on flow table forwarding, it combines the GPSR routing protocol based on geographical location information to offload some routing functions to the data plane for most of the data forwarding.

The application layer mainly handles various requirements and runs upper-layer application services. The control layer mainly includes functions such as resource monitoring, load balancing, protocol parsing, and flow table installation. In the data layer, the network layer completes neighbor discovery and location information reporting and forwards data packets through flow table matching. In case of flow table matching failure, it continues routing using GPSR without the need for dynamic maintenance of the flow table.

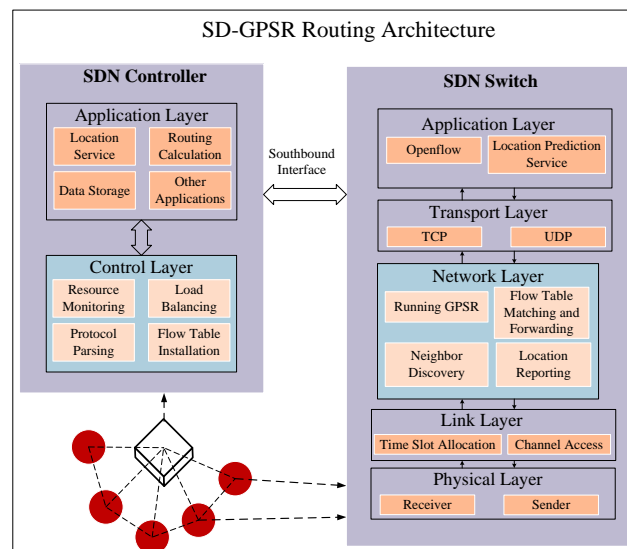


Figure 1. SD-GPSR routing architecture.

3.2. Design of the Control Layer

In the SD-GPSR routing architecture, the SDN controller can collect information from all nodes in the network for centralized control. Therefore, we design to use the SDN controller as a location server to provide node location information. In order to provide more accurate location information, the location server needs to complete the discovery and update of location information. Additionally, interruptions may occur during dynamic network operation, requiring a timely response mechanism.

3.2.1. Location Discovery and Update

We use an active approach for location discovery and update, where all nodes periodically use Packet-in messages to proactively report their own location information and neighbor relationships. During the initial operation of the network, all nodes encapsulate their own information (including IP address, MAC address, neighbor relationships, etc.) into Packet-in messages and use them to report to the controller. Upon receiving a Packet-in message, the controller identifies it as a location update packet based on the message type in the packet header. The controller then queries its local node information set. If the controller finds that the information of the node is already stored in the node information set, it updates the information. Otherwise, it adds the information of the node to the node information set.

3.2.2. Location Disruption

When the connection between a node and the controller is disrupted, the node cannot use the controller to obtain the location information of the destination node. In this case, the disconnected node sends a request message to the nearest node, which will be forwarded to the controller by the node's neighboring nodes. When the controller receives the request message, it identifies the disconnected node based on the source IP in the message and establishes a secure channel with the source node again.

When a data packet is being forwarded and approaches the vicinity of the destination node's location, but the destination node is not within the one-hop neighbor range of the forwarding node, it is essential to establish a suitable interrupt response to prevent packet dropping caused by the inability to locate the destination node. In such cases, the forwarding node C uses a Packet-in message to report the data packet to the controller. Upon receiving the data packet, the controller uses a centralized routing algorithm to calculate a route path and installs flow tables using Flow-mod messages. The controller also includes the data packet in a Packet-out message and sends it to the reporting node. After processing the data packet, considering that the location of the destination node has changed, subsequent nodes that continue to forward data based on the destination location stored in the source node may also need to request the controller again when the data reach node C. To prevent this situation, when the controller receives the request message, it includes the destination node's location information in a Packet-out message and sends it back to the source node, preventing subsequent data packets from using incorrect location information for forwarding.

3.3. Data Layer Forwarding

Finding an optimal path during the routing process is crucial for improving network performance. We combine flow table matching with greedy computation to achieve data forwarding in the routing process. When a node forwards data, it first attempts to match the flow table for data forwarding. If the flow table matching fails, the node employs a greedy forwarding strategy. If greedy forwarding also fails, the node needs to report to the controller, requesting the controller to calculate the route path and install a new flow table. The node then re-matches the flow table for data forwarding. Specifically, as shown in Figure 2, the entire routing process involves the following three stages:

(1) In the first stage, upon receiving a data packet, the node prioritizes flow table matching by default (considering that flow tables have better performance in terms of

forwarding rate, security, QoS, and load balancing). If the flow table matching is successful, the node executes the instructions in the action set to complete data forwarding, and the data packet processing process ends.

(2) In the second stage, if flow table matching fails, the node queries the Table-miss entry of the flow table. It then initiates the GPSR-DA algorithm according to the instructions. The node relies on neighbor information and destination node position information extracted from the data packet header to calculate the next-hop forwarding node using the greedy forwarding strategy. If a better destination node is found than itself, the node forwards the data packet, and the data packet processing process ends.

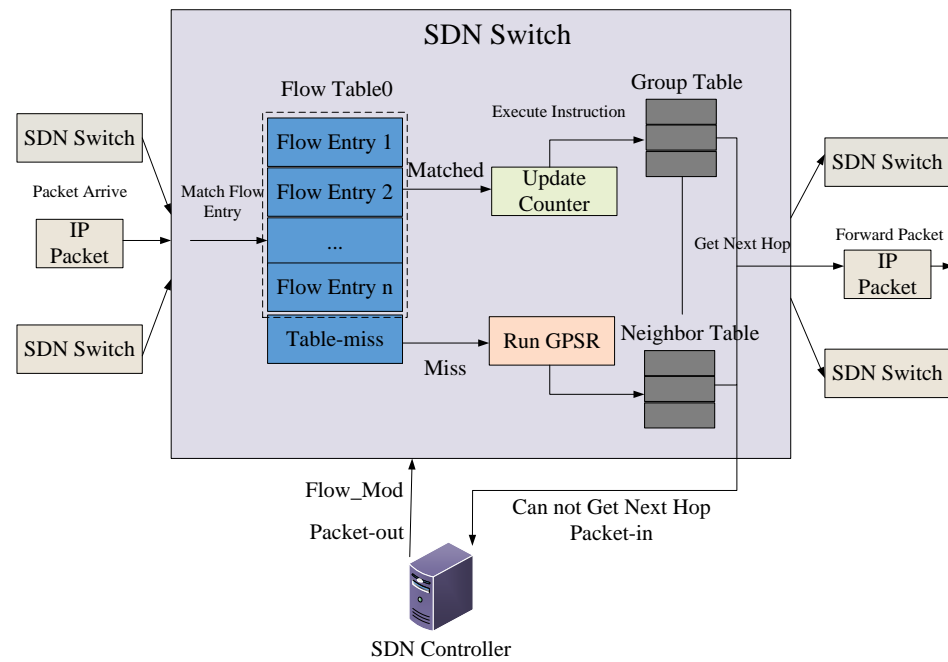


Figure 2. The process of SD-GPSR handling packets.

(3) In the third stage, when the node encounters a routing hole while forwarding the data packet using the greedy forwarding strategy, it reports the data packet to the controller through a Packet-in message. The controller utilizes the globally maintained position information and neighbor relationships to calculate a route path using a centralized algorithm. It then sends Flow-mod messages to the reporting node to install the flow table and resends the data packet to the reporting node using a Packet-out message. Upon receiving the Flow-mod and Packet-out messages, the node first parses the Flow-mod message to obtain the route path for node forwarding and updates the local flow table information. It then parses the Packet-out message to retrieve the data packet and adds the route path to the data packet header. Finally, the node re-matches the flow table to complete data forwarding. We will elaborate on the GPSR-DA algorithm in Section 4.

4. Implementation

In this section, we design the data layer using GPSR-DA algorithm.

4.1. GPSR-DA Algorithm

In the greedy forwarding process of GPSR, the nearest node to the destination node is selected as the next hop for forwarding. However, this approach may lead to two issues: (i) there may exist a shorter overall path between the source and destination nodes; (ii) nodes along a routing path may be excessively utilized, leading to high-frequency consumption of certain nodes in the network. Based on these issues, we design the GPSR-DA algorithm to select the next-hop forwarding node. Specifically, the forwarding node calcu-

lates the distances between itself and all neighboring nodes towards the destination node. Then, it selects all nodes that are closer to the destination node than itself and calculates the angles formed by these nodes with respect to the forwarding node and the destination node. Assuming the set of neighboring nodes of a forwarding node is denoted as $N = \{N_i, i \in N^*\}$, with S representing the forwarding node, D representing the destination node, and R representing the communication range of node S . The specific calculation methods for distance and angle are as follows.

4.1.1. Distance Calculation

We can use Equation (1) to calculate the position information $L_{(N_i,D)}$, representing the distance between each node in N and the destination node D . Let the coordinates of node N_i be (x_i, y_i) , and the coordinates of D be (x_d, y_d) .

$$L_{(N_i,D)} = \sqrt{(x_i - x_D)^2 + (y_i - y_D)^2}. \tag{1}$$

4.1.2. Angle Calculation

The set of neighboring nodes of node S , which are closer to the destination node than node S itself, is denoted as $N''' = \{N_k''', k \in N^*\}$. The coordinates of N_k''' are (x_k''', y_k''') . In this text, the relationship between node $N_k''' = (k = 1, 2, \dots, i + j)$ and the line SD is represented in polar coordinates, as shown in Figure 3.

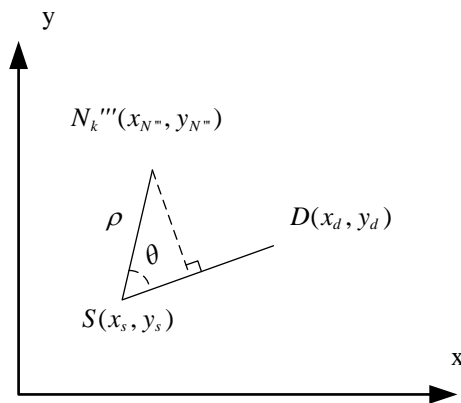


Figure 3. Calculation of relay node angle.

Taking node S as the origin of polar coordinates and SD as the x-axis, the distance between node N_k''' and node S is denoted as ρ , and the line connecting them is represented as SN_k''' . The angle between SN_k''' and SD is denoted as $\theta = \{\theta_k, k \in N^*\}$. Therefore, we have the following equation:

$$\rho = L_{(N_k''',S)} = d_{(N_k''',SD)} \sin\theta. \tag{2}$$

Therefore, we can obtain the following equation:

$$\theta = \arcsin\left(\frac{L_{(N_k''',S)}}{d_{(N_k''',SD)}}\right), \tag{3}$$

where $d_{(N_k''',SD)}$ represents the distance from point N_k''' to the line SD . The equation of the line SD can be expressed as:

$$(y_d - y_s)x + (x_d - x_s)y + (x_s y_d - x_d y_s) = 0. \tag{4}$$

The distance $d_{(N_k''',SD)}$ from point N_k''' to the line SD can be represented as follows:

$$d_{(N_k''',SD)} = \frac{((y_d - y_s) \times x_{N'} + (x_d - x_s) \times y_{N'} + (x_s y_d - x_d y_s))}{\text{sqrt}(\sqrt{(y_d - y_s)^2 + (x_d - x_s)^2})} \tag{5}$$

Based on the calculation of the distance and angle mentioned above, a shorter distance indicates that the neighboring node is closer to the destination node. A smaller value for θ increases the probability of a straight-line distance to the destination node. Therefore, considering both the distance and angle calculations, a smaller evaluation value for the neighboring node indicates that it is a better next-hop forwarding node. The comprehensive evaluation value is calculated as follows:

$$Vue = \alpha\theta + \beta L_{(N_i,D)}, \tag{6}$$

where α and β are the weights assigned to the angle and distance, $\alpha = 2/\pi$, $\beta = 1/(L_{(S,D)})$.

Due to the fact that the GPSR_DA algorithm only considers the angle and distance when selecting the next hop node, it is highly likely to select a node with a very small angle at a very close range from the source node, resulting in the minimum calculated Vue . This may cause the selected next hop node to be very close to the source node but very far from the destination node, thus increasing the number of forwarding hops. Therefore, we divide the communication range of node S into different regions, as shown in Figure 4.

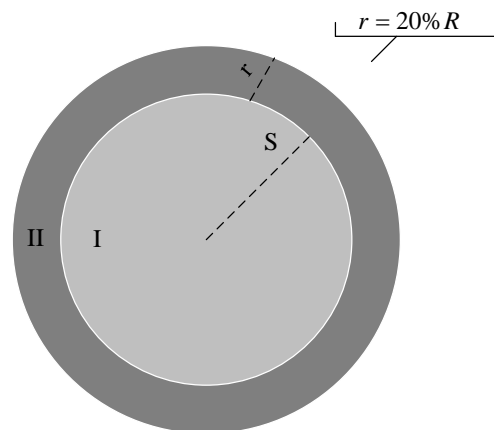


Figure 4. Division of the communication range of forwarding nodes.

Node S first selects a next-hop-forwarding node within Region II. If there are no available nodes in Region II, then the selection process moves to Region I. Experimental results have shown that the optimal range for Region II is 20% of the communication range at the edge of node S . We select all neighboring nodes of node S that are closer to the destination node and located within 20% R of the communication range of node S . These nodes form the first priority node set, denoted as $N' = \{N'_i, i \in N^*\}$. In other words, the nodes in set $N' = \{N'_i, i \in N^*\}$ must satisfy the following conditions:

$$R - L(N'_i, S) \leq 20\%R. \tag{7}$$

We consider all neighboring nodes of node S that are closer to the destination node and not present in the first priority node set as the second priority node set, denoted as $N'' = \{N''_j, j \in N^*\}$. Therefore, we have

$$N''' = N' \cup N''. \tag{8}$$

The specific implementation steps are as follows:

Step I: If $L_{(S,D)} \leq R$, search for the MAC address of the destination node in the neighbor table of node S , forward the data to D , and stop.

Step II: If $L_{(S,D)} \geq R$, calculate the distance between N_i and D based on Equation (1) and select the first priority node set. If the first priority set contains nodes, calculate the angle θ between the nodes in the set and the line connecting the forwarding node and the destination node according to Equation (3). Then, calculate the value Vue based on Equation (6), which combines distance and angle calculations. Select the node with the smallest as the next-hop-forwarding node.

Step III: If the first priority set does not contain any nodes, select all nodes that are closer to the destination node than the forwarding node as the second priority node set. Calculate the angle θ between the nodes in the set and the line connecting the forwarding node and the destination node according to Equation (3). Then, calculate the value Vue based on Equation (6), which combines distance and angle calculations. Select the node with the smallest as the next-hop forwarding node.

Step IV: The next-hop forwarding node repeats **Step I** and **Step II** until the data packet reaches the destination node. The specific execution algorithm is shown in Algorithm 1.

Algorithm 1 GPSR_DA

Require:

- 1: S_x, S_y : coordinates of node S
- 2: D_x, D_y : coordinates of destination node
- 3: N_x, N_y : coordinates of next-hop node
- 4: α, β : weights of angle and distance
- 5: N_i : neighbors of node S
- 6: R : radius of node S

Ensure:

- 7: N_{opt} : optimal next-hop node ID
 - 8: **while** neighbor table is not empty **do**
 - 9: **if** there is a next-hop node **then**
 - 10: **return** next-hop node ID {Return next-hop node if available}
 - 11: **else**
 - 12: Calculate distance to destination node using (1)
 - 13: Determine whether neighbor node belongs to set N' using (7)
 - 14: **if** $N_i \in N'$ **then**
 - 15: Add N_i to set N'
 - 16: **else**
 - 17: Add N_i to set N''
 - 18: **end if**
 - 19: Calculate the equation of the line SD using (5)
 - 20: Calculate the angle between each neighbor node in N' or N'' and the line SD using (3)
 - 21: Calculate the result Vue based on distance and angle using (6)
 - 22: **if** node belongs to N' and Vue is smaller than the minimum value in N' **then**
 - 23: Replace the node in N' with the current node and update Vue
 - 24: **end if**
 - 25: **if** node belongs to N'' and Vue is smaller than the minimum value in N'' **then**
 - 26: Replace the node in N'' with the current node and update Vue
 - 27: **end if**
 - 28: **end if**
 - 29: **end while**
 - 30: **if** N' has the best node **then**
 - 31: **return** its ID
 - 32: **else if** N'' has the best node **then**
 - 33: **return** its ID
 - 34: **end if**
 - 35: **return** N_{opt} {Return optimal node ID}
-

In order to avoid the occurrence of routing holes in the GPSR_DA algorithm, we propose a simple and effective solution by combining SDN controller capabilities. By leveraging the controller's global view, we use a centralized approach to perform routing calculations and update routing paths in the controller, using flow table matching to ensure seamless data forwarding.

$$D_{(N_i,D)} = |x_{N_i} - x_D| + |y_{N_i} - y_D|. \quad (9)$$

4.2. Routing Hole Response Mechanism

In this paper, we consider geographical location information comprehensively. The controller only knows node location information and neighbor relationship information. Therefore, we take path length and node stability into consideration.

Node Stability

Node stability can be measured by the Average Movement Distance (AMD) based on the updated node position information. In a given time period T , each node in a link will update $k + 1$ times. The set of updated position coordinates for each node within the time period T can be represented as follows:

$$\{(x_{t_0}, y_{t_0}), (x_{t_1}, y_{t_1}), \dots, (x_{t_k}, y_{t_k}), \}. \quad (10)$$

The average movement distance of the node within the past time period T is

$$AMD = \frac{\sum_{j=0}^{j=k} \sqrt{(x_{t_{j+1}} - x_{t_j})^2 + (y_{t_{j+1}} - y_{t_j})^2}}{T}. \quad (11)$$

Considering these two cost estimates, the smaller the Manhattan distance between the node and the destination node, the lower the cost incurred. Additionally, a smaller average movement distance of the node indicates a more stable routing path. Therefore, in this study, we select the node with the minimum comprehensive value and maximum benefit to join the link. The comprehensive cost estimate for the node is

$$c_i = AMD_i \times D_{(N_i,D)}. \quad (12)$$

Suppose that through the A^* algorithm, the routing path with the minimum cost is found, consisting of m nodes. When the A^* algorithm reaches the n -th ($n < m$) node during its search, the overall cost of the n nodes already added to the link is

$$G * (n) = \sum_{i=1}^{j=n} c_i. \quad (13)$$

After adding the $(n + 1)$ -th ($(n + 1) < m$) node, the cost estimate for the updated path is

$$F * (n) = G * (n) + H * (c_{n+1}). \quad (14)$$

The final algorithm implementation, as shown in Algorithm 2, obtains a path with the minimum cost based on the cost estimation function. This path is then sent to the reporting node, which carries the path information to complete the data-forwarding process.

Algorithm 2 A***Require:**

- 1: *openList*: set of nodes to be traversed
- 2: *closedList*: set of nodes that have been traversed
- 3: *destNode*: destination node
- 4: *startNode*: source node
- 5: *curNode*: current node
- 6: *routList*: list that stores the returned path nodes

Ensure:

```

7: routList
8: Push the neighbor list of startNode into openList
9: while openList is not empty do
10:   for  $i = 1$  to openList.length() do
11:     if openList[ $i$ ] == destNode then
12:       curNode  $\leftarrow$  openList[ $i$ ]
13:       while curNode is not startNode do
14:         Push curNode into routList
15:         curNode  $\leftarrow$  curNode.parent
16:       end while
17:       return routList
18:     end if
19:   end for
20:   Find the node with minimum value in openList using (7)
21:   curNode  $\leftarrow$  the node with minimum value
22:   Pop curNode from openList
23:   Add curNode to closedList
24:   nbrList  $\leftarrow$  neighbor list of curNode
25:   while nbrList is not empty do
26:     for  $j = 1$  to nbrList.length() do
27:       if nbrList[ $j$ ] is in closedList then
28:         continue
29:       end if
30:       if nbrList[ $j$ ] is not in openList then
31:         Push nbrList[ $j$ ] into openList
32:         nbrList[ $j$ ].parent  $\leftarrow$  curNode
33:         Calculate cost using (14)
34:       else if nbrList[ $j$ ] is in openList then
35:         Calculate new cost with curNode as parent
36:         if new cost < old cost then
37:           nbrList[ $j$ ].parent  $\leftarrow$  curNode
38:           Update cost of nbrList[ $j$ ]
39:         end if
40:       end if
41:     end for
42:   end while
43: end while

```

4.3. Location Preservation Method

In addition, in the SD-GPSR routing method, the source node needs to request the location information of the destination node from the controller. We design a location preservation method where the destination node can obtain its relatively accurate location information at a future time through location prediction services. Subsequently, the predicted location information can be sent to the source node through data packets sent by the destination node to update its location information, establishing a location preservation relationship.

Specifically, we use a cross-layer protocol interaction for location prediction. This design allows communication between different layers without necessarily passing through

intermediate layers. A flow can traverse from the link layer through the other four layers to the application layer.

When the destination node receives a data packet, it needs to dynamically update the predicted location information at the source node. Taking inspiration from TCP's acknowledgment mechanism, the location information is carried in the returning data packet and transmitted back to the source node. The implementation steps are as follows:

Step I: At the beginning of the network, the source node starts sending data packets to the destination node based on the requirements of the service flow.

Step II: The network layer of the destination node receives the first data packet of the service flow, calculates using (15), and triggers a remote interrupt to obtain the predicted location information after time PT .

$$PT = \frac{L_{D-1} - L_{(D-1,D)}}{v_D}, \quad (15)$$

where L_{D-1} represents the communication range of the last-hop node that received the most recent data packet. $L_{(D-1,D)}$ represents the distance between the last-hop node and the destination node D . v_D represents the velocity of the destination node D .

Step III: After the destination node waits for time PT , if the service flow has not ended yet, the destination node recalculates PT based on the most recently received data packet and calls the interface to obtain the predicted location information $Pre_Position$.

Step IV: When the destination node detects that there is a data packet being forwarded to the source node, it encapsulates $Pre_Position$ in the IP header of the data packet and sends it back to the source node.

Step V: Upon receiving the data packet, the source node extracts $Pre_Position$ from the IP header during the network layer's IP header parsing process and performs the necessary storage and update operations. Subsequently, the data packets will be routed using the updated destination node location information.

5. Simulation Implement

5.1. Simulation Setting

This paper divides the network scenarios into three different scales based on the number of nodes: small-scale network scenario with 16 nodes, medium-scale network scenario with 32 and 48 nodes, and large-scale network scenario with 96 nodes to study the performance variations in different scale network scenarios. Nodes in the network scenario move along set trajectories within different ranges: a $10 \text{ km} \times 10 \text{ km}$ area is suitable for simulating small-scale network scenarios, while a $40 \text{ km} \times 40 \text{ km}$ area is suitable for simulating large-scale network scenarios. For the medium-scale network scenario, the area is chosen between $20 \text{ km} \times 20 \text{ km}$ and $30 \text{ km} \times 30 \text{ km}$ based on the number of nodes and simulation requirements.

At the beginning of the simulation, the network starts broadcasting beacon messages, and the neighbor table entries of nodes are incomplete, leading to a high packet loss rate as many data packets are dropped due to the inability to find the next-hop forwarding node. Therefore, the network is unstable within the first 100 s. By 300 s, the network gradually stabilizes. The simulation is set to run from 600 s to 1000 s to evaluate network performance in a stable environment.

The data channel has a speed of 2 Mbps and a communication distance of 2 km–5 km, which effectively simulates real-world network-transmission scenarios. The communication distance is adjusted based on the size of the moving area. The control channel utilizes 96 Kbps with a communication distance of 10 km–40 km. Since control information has lower bandwidth requirements and needs to cover the entire moving area, the emphasis is on communication distance rather than bandwidth.

Node movement speed is set to normal vehicle speed. The channels include a control channel based on the TDMA protocol and a data channel based on the CSMA protocol. Specific parameter settings are provided in Table 1.

Table 1. Experiments parameters of settings.

Parameters	Values (Units)
Simulation Area	10 × 10 km–40 × 40 km
Number of Nodes	16, 32, 48, 96
Simulation Time	600 s–1000 s
Data Channel Speed Rate	2 Mbps
Control Channel Speed Rate	96 Kbps
Data Channel Distance	2–5 km
Control Channel Distance	10–40 km
Node Moving Speed	12 m/s
Services Flows	5 Kbps–25 Kbps

5.2. Evaluation Metrics

This paper compares and analyzes the proposed SD-GPSR routing method against traditional centralized SD-MANET routing methods, as well as distributed GPSR and AODV routing protocols, to verify the feasibility and effectiveness of the SD-GPSR approach. This paper collects the following performance metrics to analyze and evaluate the feasibility and effectiveness of the SD-GPSR routing method, as follows:

- **Total Control Overhead:** The total number of control packets transmitted in the network during the simulation time.
- **Control Plane Overhead:** The total number of control packets exchanged between the network control plane and the data plane during the simulation time.
- **Packet Loss Rate:** The ratio of lost data packets to the total number of data packets transmitted in the network, reflecting the reliability of network transmission.
- **Average End-to-End Delay:** The average time required for data packets to travel from source nodes to destination nodes, which is a key metric for measuring the real-time delivery of traffic flows.

5.3. Performance Analysis

5.3.1. Impact on Service Flows

The simulation duration for this experiment is 600 s. The simulation results, as shown in Figure 5, reflect the average performance trends of various metrics for the four routing methods with varying traffic loads.

In Figure 5a, the control overhead for all four routing methods increases with the increase in traffic load. AODV exhibits exponential growth in control overhead due to its broadcast-based route establishment, which becomes more pronounced with higher traffic loads. GPSR also shows exponential growth in control overhead, mainly because of the increased frequency of broadcasting location probe packets in the network. SD-GPSR and SD-MANET exhibit relatively smoother increases in control overhead.

In Figure 5b, as the traffic load increases from 15 Kbps to 25 Kbps, the control plane overhead for SD-GPSR remains relatively stable, while SD-MANET experiences a significant increase in control overhead. This is primarily due to the increased number of control messages for topology maintenance and route request messages in the routing process caused by higher traffic loads.

In Figure 5c, the packet loss rate trends for the three routing methods with varying traffic loads are shown. The packet loss rate of GPSR is higher than that of SD-GPSR and AODV when the traffic load is 5 Kbps, and there is little change in the packet loss rate as the traffic load increases from 5 Kbps to 10 Kbps. The packet loss rate is mainly influenced by changes in the position information of destination nodes and the presence of routing holes. As the traffic load gradually increases, congestion in nodes also contributes to packet loss, resulting in an increase in the packet loss rate. The packet loss rates of SD-GPSR and AODV gradually increase with the growth of the traffic load. During the transition from 0 Kbps to 15 Kbps, SD-GPSR shows a relatively moderate increase, but after reaching 15 Kbps, the packet loss rate increases significantly. This is because the increased traffic load may cause

a loss of data packets during position information reporting, and the delay in updating position information during the position-holding process leads to inaccurate destination node information at the source node, resulting in the dropping of some packets. However, compared to GPSR, which incurs additional packet loss due to re-sending probe packets to search for destination node positions, SD-GPSR experiences fewer packet losses.

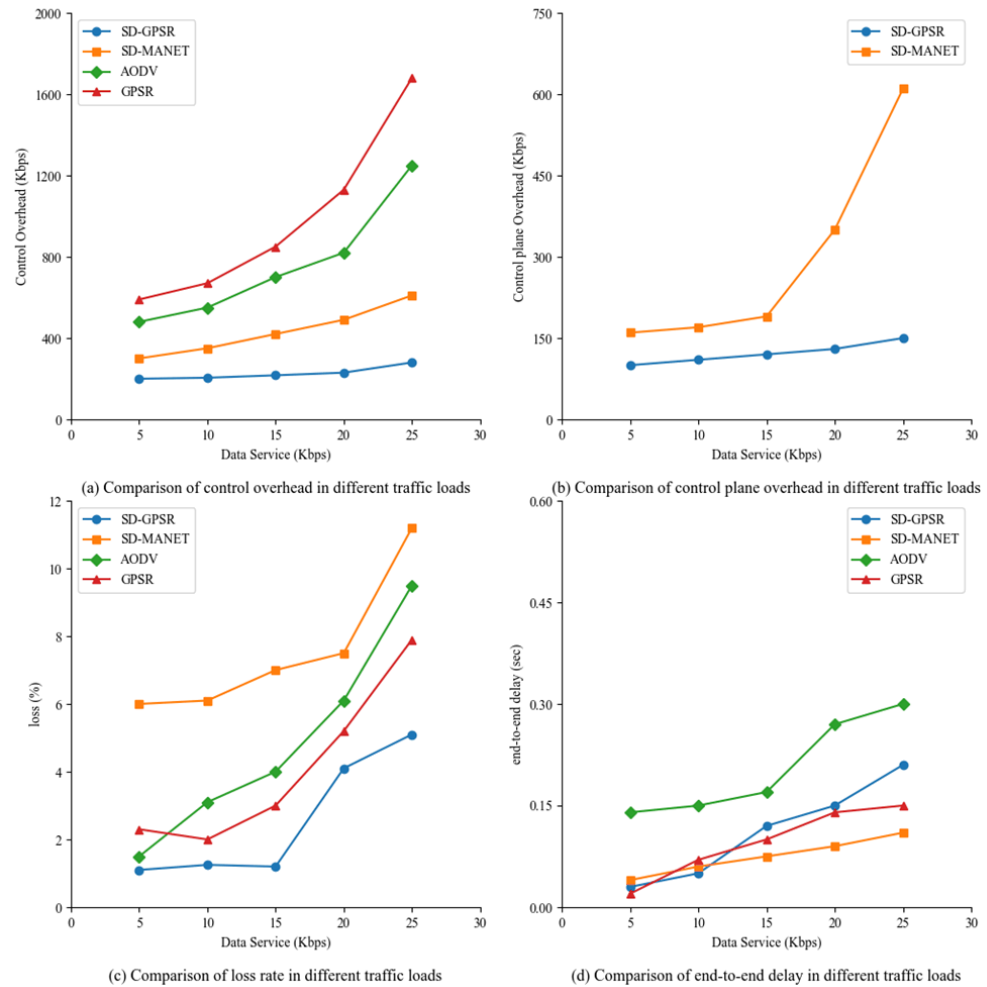


Figure 5. Comparison of the influence of four protocols on network performance with data service changes.

Figure 5d illustrates the trends of average end-to-end delay for the four routing methods with varying traffic loads. In the range of 5 Kbps to 10 Kbps, SD-MANET, GPSR, and SD-GPSR show a slight increase in average end-to-end delay. Once the traffic load exceeds 10 Kbps, all three routing methods experience a significant increase in average end-to-end delay. SD-GPSR exhibits the largest increase, primarily due to the higher computational load on the nodes, resulting in packets staying in buffers for longer durations. Additionally, as more data are forwarded, the occurrence of routing holes increases, requiring the controller to compute route paths, leading to additional delays. AODV’s delay increases with the growth of traffic load and network load. However, as the traffic load increases, routes are dynamically established, resulting in a larger number of routing entries and more opportunities for data forwarding, thus leading to a relatively smoother increase in delay.

5.3.2. Impact of Scenario Scale

In this section, four different scenario scales will be simulated to analyze the impact of the four routing methods on network performance in different scenarios. The scenario

scales are 16 nodes, 32 nodes, 48 nodes, and 96 nodes, with a constant traffic load of 20 Kbps and a simulation duration of 600 s.

(a) Control overhead in different scenario scales

Figure 6 compares the control overhead of the four routing methods in different scenario scales. From Figure 6a, it is evident that the control overhead of all four routing methods increases as the scenario scale grows. AODV's control overhead exceeds 6000 Kbps in the 96-node scenario. This is mainly because the number of nodes in the scenario increases, leading to more neighbor nodes for each node and consequently increasing the number of hops required from the source node to the destination node. The increased number of broadcasting RREQ messages significantly contributes to the generation of control overhead. GPSR incurs substantial control overhead due to the broadcasting of one-hop beacons and location probe packets, which also increase with the scenario scale. In terms of control overhead between the control plane and the data plane, SD-GPSR exhibits a relatively gradual increase, while SD-MANET experiences a significant increase in control overhead with the increase in the number of nodes, as evident from Figure 6b. SD-GPSR primarily incurs overhead from reporting node positions and obtaining destination node positions. On the other hand, SD-MANET needs to maintain the network topology, including reporting its own one-hop and two-hop neighbor information, resulting in a much larger overhead compared to SD-GPSR's position and neighbor information reporting. Additionally, SD-MANET constantly calculates route paths and issues flow tables during network operation, leading to higher overall control overhead compared to SD-GPSR.

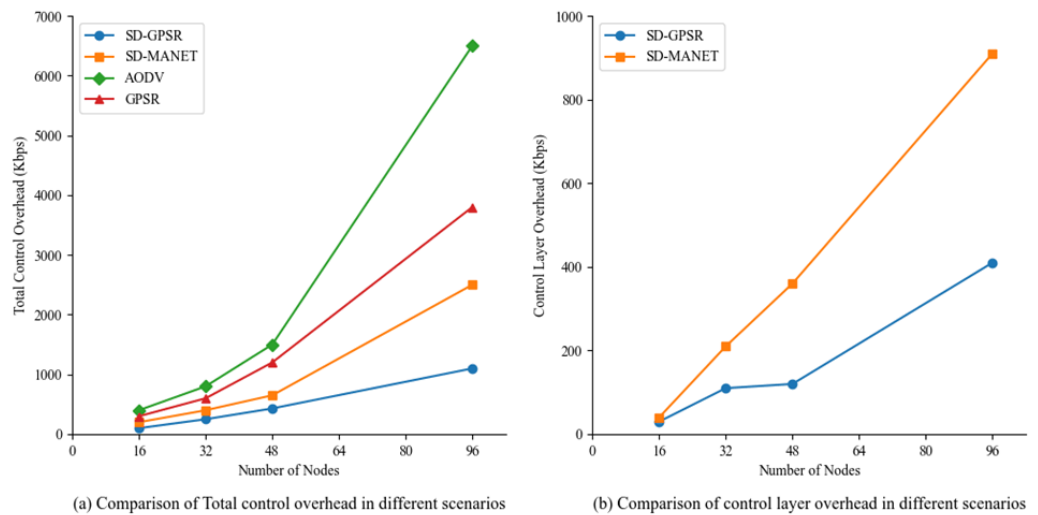


Figure 6. Comparison of control costs in different scenarios.

(b) Average packet loss rate

Figure 7 depicts the average packet loss rate of the four routing methods in different scenario scales. As the number of nodes increases from 16 to 32, the packet loss rate gradually decreases for all four routing methods. When the number of nodes increases from 32 to 48, the packet loss rates of SD-GPSR and GPSR continue to decrease, while AODV and SD-MANET start to experience an increase in packet loss rate. Finally, when the number of nodes increases from 48 to 96, the packet loss rates of all four routing methods increase again.

In smaller-scale scenarios, where the number of nodes is sparse and the available next-hop nodes for forwarding are limited, it is prone to data packet loss. GPSR and SD-GPSR employ a greedy forwarding strategy, which can easily result in packet loss due to the phenomenon of routing voids. On the other hand, the other two routing methods calculate from a global perspective, resulting in lower packet loss rates compared to SD-GPSR and GPSR. As the scenario scale gradually increases, the number of available next-hop nodes increases, making it easier to establish routing paths. Therefore, the packet loss rates of

SD-GPSR and GPSR decrease over time. However, in the case of SD-MANET, the loss of topology update messages during network operation leads to a higher packet loss rate. Similarly, AODV also experiences decreasing stability of routing paths due to an increase in the number of forwarding hops, resulting in an increase in packet loss rate.

Figure 8 compares the average end-to-end delays of the four routing methods in different scenario scales. As the scenario scale increases, the average end-to-end delay of GPSR is initially high at 16 nodes, but it decreases when the scenario scale reaches 32 nodes. However, it gradually increases with the increase in the number of nodes. On the other hand, the average end-to-end delays of AODV, SD-MANET, and SD-GPSR also increase gradually with the increase in the number of nodes.

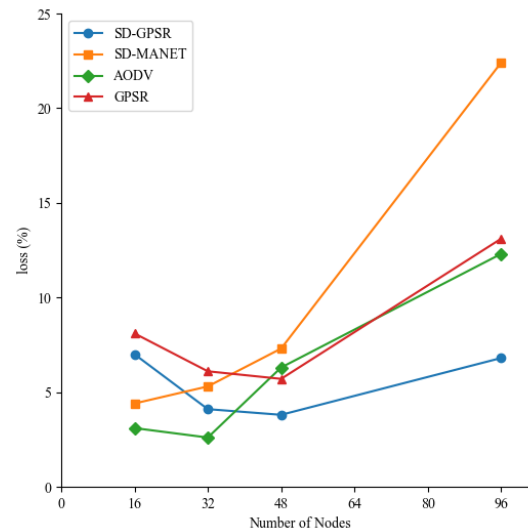


Figure 7. Comparison of average packet loss rate in different scenarios.

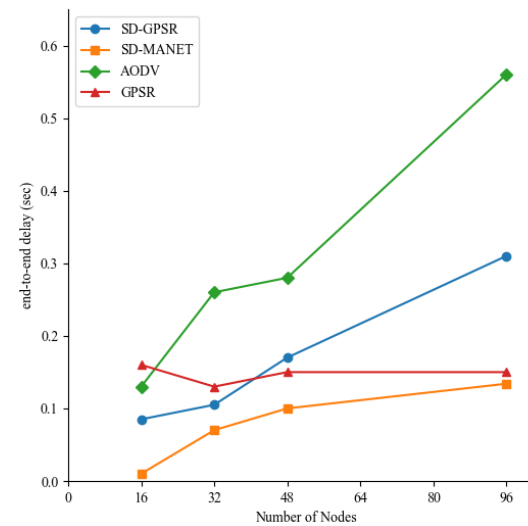


Figure 8. Comparison of average end-to-end delay in different scenarios.

With the increase in the number of nodes, all four routing methods experience an increase in forwarding hops. In the forwarding process of SD-MANET, data forwarding can be quickly accomplished through flow table matching. However, due to the increase in forwarding hops, the flow table becomes invalidated more frequently, resulting in a gradual increase in latency. In the scenario of 16 nodes, GPSR experiences higher end-to-end latency due to the occurrence of numerous routing voids caused by sparse node distribution during path selection. However, as the number of nodes increases, the available nodes in the forwarding process also increase, reducing the frequency of routing voids and enabling

fast packet forwarding. But as the scale of the scenario increases, the number of forwarding hops also increases, making it easier for packets to experience significant latency due to expired location information during the forwarding process. However, since the number of packets encountering this situation is relatively small, the average end-to-end latency is not expected to be very high. In SD-GPSR, with the increase in the number of nodes, there may be initial delays due to the inability to obtain accurate destination node location information, resulting in back-and-forth packet exchanges between the controller and nodes. Furthermore, an increase in the number of forwarding hops during route selection leads to a slight increase in the latency of path calculation in SD-GPSR. As a result, its average end-to-end latency gradually increases and remains higher than the latencies of SD-MANET and GPSR. In AODV, as the number of nodes increases, the number of forwarding hops along the routing path also increases. Once a node becomes unavailable, it requires rediscovering a new routing path, resulting in a gradual increase in average end-to-end latency.

5.4. Scalability and Practicality

With the evolution of MANETs towards vehicular networks, we need to focus on adapting to the unique environments and application scenarios of vehicular networks, such as high-speed mobility, frequent node connections and disconnections, and fluctuating communication quality. At the same time, considering the high requirements of vehicular networks for communication reliability and real-time performance, more in-depth optimization and research are needed on metrics such as data transmission delays, throughput, and energy efficiency. SD-GPSR has excellent scalability and practicality. From the experiments, we can see that SD-GPSR performs excellently in environments with high-speed node movement. With its location discovery and update mechanisms, as well as location interruption mechanisms, it can effectively handle frequent changes in network topology. Therefore, it shows good performance in addressing challenges such as high node mobility, frequent node connections and disconnects, and fluctuation in communication quality in vehicular networks. Consequently, SD-GPSR can be effectively used in vehicular networks.

6. Conclusions

This paper proposes an SD-GPSR routing method that decentralizes some routing functions of SD-MANET to the data plane by utilizing the geographic location information of nodes for routing. Additionally, an improved greedy forwarding algorithm based on distance and angle (GPSR_DA) is introduced, where the controller provides location services and partial centralized decision-making, and data plane nodes use the GPSR_DA algorithm to forward data. This approach enhances the self-organizing capability and overall routing efficiency of the nodes while reducing routing control overhead. To address the potential issue of routing voids during the forwarding process, the A^* algorithm is employed to calculate optimal routing paths that avoid voids. Flow tables are deployed using source routing principles, effectively reduces control overhead. Validation and analysis of the SD-GPSR routing method demonstrate superior performance advantages in terms of control overhead and packet loss rates in large-scale networks.

Author Contributions: S.G. wrote the main manuscript text, including study design, experiments and data analysis. L.L. provided computational resources and critically commented on the revised manuscript. J.Z. prepared the figures and completed and checked the English grammar and Q.L. revised the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding

Data Availability Statement: The data supporting this article are from previously reported studies and datasets, which have been cited.

Acknowledgments: The authors express their gratitude to the editor and reviewers for their work and valuable comments on the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Khalid, A.; Rehman, R.A.; Burhan, M. CBILEM: A novel energy aware mobility handling protocol for SDN based NDN-MANETs. *Ad Hoc Netw.* **2023**, *140*, 103049. [[CrossRef](#)]
2. Kalafatidis, S.; Skaperas, S.; Demiroglou, V.; Mamas, L.; Tsaoussidis, V. Logically-centralized SDN-based NDN strategies for wireless mesh smart-city networks. *Future Internet* **2022**, *15*, 19. [[CrossRef](#)]
3. Streit, K.; Rodday, N.; Steuber, F.; Schmitt, C.; Rodosek, G.D. Wireless SDN for highly utilized MANETs. In Proceedings of the 2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Barcelona, Spain, 21–23 October 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 226–234.
4. Srilakshmi, U.; Veeraiah, N.; Alotaibi, Y.; Alghamdi, S.A.; Khalaf, O.I.; Subbayamma, B.V. An improved hybrid secure multipath routing protocol for MANET. *IEEE Access* **2021**, *9*, 163043–163053. [[CrossRef](#)]
5. Saffre, F.; Hildmann, H.; Anttonen, A. Force-Based Self-Organizing MANET/FANET with a UAV Swarm. *Future Internet* **2023**, *15*, 315. [[CrossRef](#)]
6. Pundir, M.; Sandhu, J.K.; Gupta, D.; Gadekallu, T.R.; Juneja, A.; Gulzar, Y.; Nauman, A. Data rate aware reliable transmission mechanism in wireless sensor networks using Bayesian regularized neural network approach. *Phys. Commun.* **2023**, *59*, 102115. [[CrossRef](#)]
7. Qu, Y.; Zheng, G.; Ma, H.; Wang, X.; Ji, B.; Wu, H. A survey of routing protocols in WBAN for healthcare applications. *Sensors* **2019**, *19*, 1638. [[CrossRef](#)]
8. Zhao, C.; Ye, M.; Xue, X.; Lv, J.; Jiang, Q.; Wang, Y. DRL-M4MR: An intelligent multicast routing approach based on DQN deep reinforcement learning in SDN. *Phys. Commun.* **2022**, *55*, 101919. [[CrossRef](#)]
9. Chan, L.; Gomez Chavez, K.; Rudolph, H.; Hourani, A. Hierarchical routing protocols for wireless sensor network: A compressive survey. *Wirel. Netw.* **2020**, *26*, 3291–3314. [[CrossRef](#)]
10. Venkatasubramanian, S. Correlation distance based greedy perimeter stateless routing algorithm for wireless sensor networks. *Int. J. Adv. Netw. Appl.* **2021**, *13*, 4962–4970.
11. Houser, M.; Hasnaoui, M.L. An enhancement of greedy perimeter stateless routing protocol in VANET. *Procedia Comput. Sci.* **2019**, *160*, 101–108. [[CrossRef](#)]
12. Poularakis, K.; Qin, Q.; Marcus, K.M.; Chan, K.S.; Leung, K.K.; Tassiulas, L. Hybrid SDN Control in Mobile Ad Hoc Networks. In Proceedings of the 2019 IEEE International Conference on Smart Computing (SMARTCOMP), Washington, DC, USA, 12–15 June 2019.
13. Tej, D.N.; Ramana, K. MSA-SFO-based Secure and Optimal Energy Routing Protocol for MANET. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*. [[CrossRef](#)]
14. Nirmaladevi, K.; Prabha, K. A selfish node trust aware with optimized clustering for reliable routing protocol in manet. *Meas. Sens.* **2023**, *26*, 100680. [[CrossRef](#)]
15. Dusia, A.; Sethi, A.S. Software-defined architecture for infrastructure-less mobile ad hoc networks. In Proceedings of the 2021 IFIP/IEEE International Symposium on Integrated Network Management (IM), Bordeaux, France, 17–21 May 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 742–747.
16. Maruthupandi, J.; Prasanna, S.; Jayalakshmi, P.; Mareeswari, V.; Sanjeevi, P. Route manipulation aware Software-Defined Networks for effective routing in SDN controlled MANET by Disney Routing Protocol. *Microprocess. Microsyst.* **2021**, *80*, 103401.
17. Gilani, S.A.; Qayyum, A.; Rais, R.N.B.; Bano, M. SDNMesh: An SDN based Routing Architecture for Wireless Mesh Networks. *IEEE Access* **2020**, *8*, 136769–136781. [[CrossRef](#)]
18. Abbas, M.T.; Muhammad, A.; Song, W.C. SD-IoV: SDN enabled routing for internet of vehicles in road-aware approach. *J. Ambient. Intell. Humaniz. Comput.* **2020**, *11*, 1265–1280. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.