Journal of Computers Vol. 29 No. 5, 2018, pp. 1-14
doi:10.3966/199115992018102905001

A Multicast-Tree Construction Algorithm for Efficient Data Collection over Mobile Networks of Military Vehicles

Tien-Yu Chang[1], Jichiang Tsai[1*], Wen-Tien Huang[1]

[1] Department of Electrical Engineering, National Chung Hsing University, Taichung 402, Taiwan
d9864006@mail.nchu.edu.tw, jichiangt@nchu.edu.tw, etbu5563@livemail.tw

Received 16 May 2017; Revised 12 October 2017; Accepted 30 October 2017

Abstract. Due to tanks moving, joining and leaving, the wireless network topology formed by them will continuously change. Hence, how to make moving tanks to form a smart networking system is an important research issue. Moreover, the information exchange between nodes in a networking system can be performed via a multicast tree structure, and the root node can collect information from all the nodes in the structure. So in this paper, an algorithm is proposed to build an efficient distributed multicast tree construction algorithm for a mobile network. Such a multicast tree can be continually adjusted according to the latest topology information carried on message packets propagated among nodes. In addition, in all possible routing paths, a path with the lowest weight will be dynamically selected to improve the overall system messaging throughput. The above technique can be applied to make military vehicles form a smart networking system, including tanks, armored vehicles and transport vehicles, etc. Furthermore, through this network, the fast-changing situation of the battle can be immediately sent back to the server for the commander to refer to.

Keywords: data collection, Internet of Vehicles (IoV), multicast tree

1 Introduction

Wireless Mesh Network (WMNs) is an emerging network technology as a wireless communications deployment. The line laying is inconvenience and construction costs are high in the traditional wired backbone network. If you encounter a wide range of monitoring, the deployment is not convenient. Compared to the traditional wired backbone network, wireless mesh network deployment costs are low, and it can provide lower cost of Internet services and applications.

Because of the popularity of the wireless network environment, users can use the wireless network to transmit and receive data at any time. Such as Wireless Mesh Networks (WMNs), Wi-Fi, ZigBee, Bluetooth Low Energy (BLE 4.0), and other applications are also attracting more and more people to develop [1-4]. In many wireless network technology applications, Wireless Mesh Networks has become an effective way to improve the service area coverage, and widely used in industrial control, natural environment monitoring, military operations and the Internet of Things and other related fields. In particular, the Internet of Things has become one of the key technologies of the monitoring system, and can be applied to power systems, factory rooms, fire alarm, flood forecasting, earthquake warning, community security and military operations in order to speed up the dissemination of information and contingency treatment [5].

There are a growing number of vehicles imported IoT [6] communication technology to form a vehicle network (IoV) [7-11]. Data collection is a key application of the Internet of Things. It is usually formed by a wireless mesh network formed in the device to establish a multicast tree structure. As shown in Fig. 1, the root node *S* can collect the data of each member in the system. In particular, the multicast tree must have the ability to dynamically self-organize, self-heal, and self-route as the node joins, leaves, or moves to provide the distributed architecture to quickly establish and maintain multi-hop wireless communication.

[*] Corresponding Author

As a result, the transmission of data will not be interrupted due to the loss or damage of the nodes [12-14].

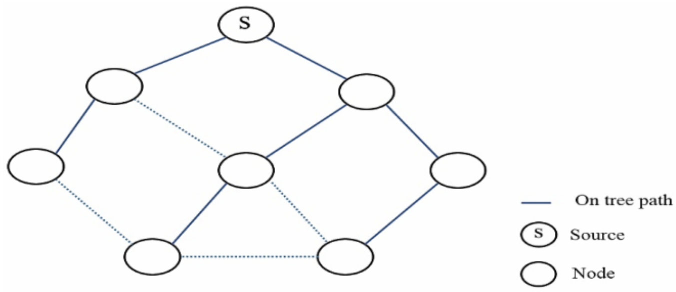

Fig. 1. Schematic diagram of multicast tree structure

The composition of the multicast tree structure is the backbone of a tree whose upstream nodes and downstream nodes may contain multiple hop points as relay nodes. The multicast tree can be automatically and automatically performed according to the latest topology information through information about the nodes themselves contained in the packets sent by each node adjustment. In particular, during the tree establishment, a path with the lowest weight is selected from all possible routing paths to reduce the overall system message delivery delay. The design of this algorithm is to effectively collect all the military vehicle information.

Based on the above discussion, the traditional routing method requires a longer time cost to search for a viable path. As a result of this motivation, we propose an algorithm for establishing an efficient multicast tree on the Internet of Things. Compared with the traditional MAODV, this method has fewer child nodes and lower transmission delay to the root node.

This paper is divided into six chapters: the first chapter is a brief introduction, the basic introduction to the research motive and the content of the paper; the second chapter is the basic introduction to the relevant multicast routing protocol of the wireless mesh network; the fourth chapter presents the result of our simulation experiment. The fifth chapter is conclusion and future prospect.

2 The Multicast Routing Protocol

Multicast routing protocol in wireless mesh networks can be divided into three types, namely Proactive, Reactive, and Hybrid.

Proactive Routing Protocol would be to establish a viable route for data transmission, and routing information is obtained via the node periodically updates. More specifically described, in such agreements, each node will store a routing table, and it is also known as Table Driven Routing Protocol [15]. This will be stored in the routing table of the node to the transmission path between all nodes. If they find a node in the network topology changes, the node must be recalculated each path, and sends the updated message to an adjacent node. When the neighbor receives the updated message, it will also recalculate each transmission path in order to update its own routing table. Such agreements are highly mobile features and have a good effect for small-scale mesh network [16]. You can quickly create a path to send a message, and the message has a lower latency. However, the drawback is that if the mobile network is low, the network bandwidth will be wasted. Because the nodes periodically transmit broadcast packets and flood to other nodes and thus cause waste. The flooding behavior is shown in Fig. 2.

Reactive Routing Protocol only the node sends a message to a destination, and the demand will be followed to establish a viable route. It can also be called On-demand Routing, common examples ODMRP (On-Demand Multicast Routing Protocol) [17] and MAODV (Multicast Operation of Ad Hoc On-Demand Distance Vector) [18], and the method mentioned in this paper also belongs to this protocol. In terms of performance, MAODV is superior to DSDV and DSR [19]. DSR is better than OLSR [20-21] and better than DSDV [22], so we use MAODV as the object of comparison in this paper.

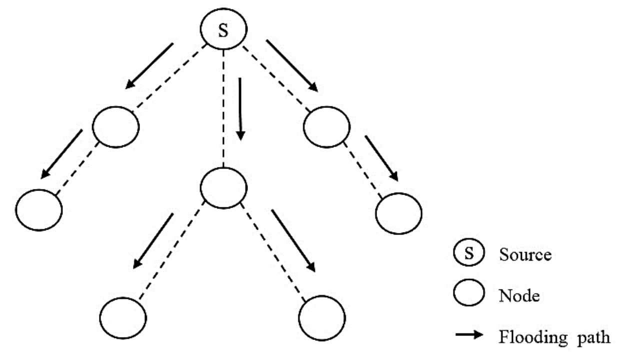

Fig. 2. Flooding

In particular, such agreements only when a node wants to send messages to other nodes, and it will dynamically establish routing path from the source to the end. Therefore, when the source needs to transmit the packet, it will find the appropriate path from the routing table to send the message, and if it cannot be transmitted in the path, the source will send a routing message to the destination side. When the destination receives, it will return a response to the message source. Thereafter, when the source receives the response message, it can immediately produce a valid path and added to the routing table. In particular, in the routing table, if the path within a certain period of time is not used and it will be removed, so you can effectively reduce the cost of bandwidth required to maintain routing tables. However, the disadvantage of this agreement is the need to spend more time in the search for a feasible path. The following is a brief description of how the MAODV routing protocol works:

Ad-hoc On-Demand Distance Vector Routing (AODV) [23] is a unicast routing protocol and MAODV is a multicast application for AODV. In the MAODV routing protocol, the routing path is established and the route path is maintained. Each node has three routing tables. The first routing table is the routing table of the node itself. If the node is used as a member of the multicast tree, the message can be routed to the multicast group to the destination. The second routing table is a multicast routing table that prepares the next hop point for each tree member of the multicast community. The third routing table for the request message table is used to achieve the goal for optimization. The MAODV protocol establishes a shared delivery tree to support multiple transport and receivers in a multicast group. The MAODV relies on flooding to discover the routing path and establish a multicast tree for the entire network. In the MAODV, the first group of multicast members is called the Group leader, which periodically broadcasts the Group HELLO (GRPH) message to update or maintain the information of the multicast group. When a node wants to join or send a message to the multicast group, which will broadcast a route request (RREQ) message. The intermediate node creates a reverse route and route request (RREQ) message. After receiving the route request (RREQ) message, the multicast group member replies to a route reply (RREP) message to set the delivery path. If the incoming node receives multiple Route reply (RREP) messages from the destination in time, the message sender with the maximum number of sequences and the minimum number of hops is selected and unicast a multicast activation (MACT) message to the next hop. Finally, it will begin transmitting the broadcast multicast message to perform the route, as shown in Fig. 3.

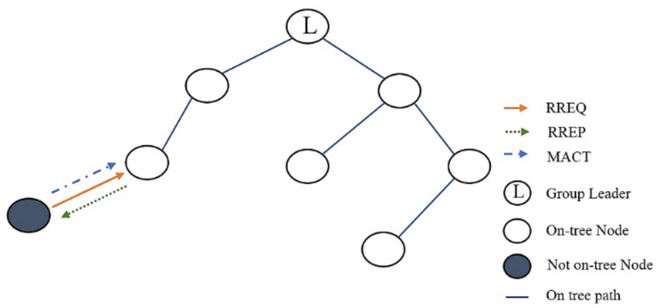

Fig. 3. MAODV operation

In the MAODV, when an On-tree node detects a link break, it starts routing recovery. First, the node needs to check whether the disconnected node is its upstream node. If the disconnected node is its upstream node, the node will remove it from the routing table, and the downstream node will send the multicast message and add the route request (RREQ) message of *Flag_J*, and then use these messages to rebuild a new branch. If the broken node is its downstream node, it is removed from the routed routing table, as shown in Fig. 4.

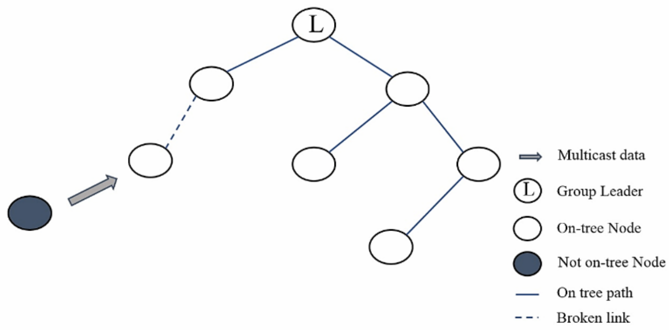

Fig. 4. MAODV link disconnected operation

Finally, the Hybrid Routing Protocol [24] is a Proactive Routing protocol and Reactive Routing mixed agreements, also known as Balanced-Hybrid Routing. More specifically described, the routing information of the node is maintained by its own router. When there is any change in network topology, the router will need to update its routing table. And according to their own routing, that is, active routing or reactive routing, to confirm the destination, and then try to find the best routing path to the destination. In particular, the advantage is that the power consumption can be effectively reduced and the number of times the node replaces the battery can be reduced, and can reduce the end to end delay of messages sent.

3 The Method of Establishing Multicast Tree

In this paper, the proposed algorithm is based on the multicast tree, and the tree-based multicast routing protocol has high transmission efficiency and low bandwidth consumption [25-27]. The packets sent by the nodes in the multicast tree contain the information of each node itself. The multicast tree can be adjusted according to the latest topology information, and dynamically find the efficient routing path for each node in the system in all possible routing paths.

The paper proposed the establishment of multicast tree algorithm, in order to effectively utilize radio characteristics, taking into account the following four benchmarks: *SrcDist* (the number of hops to the root node), *Timestamp* (the root node to the node message propagation delay), *ChildNum* (the number of child nodes of the current node is connected) and the *Request* (the node currently received his request to join the node number). Among them, the first two are respectively used to reduce the number of hops from each node to the root node and transmission delay. The latter two are due to the tree structure established, mainly used in data collection systems of the members, so that each child in order to reduce the message collision between child nodes and specifically to join them. In particular, when a node receives a large number of the requests from other nodes, it may have a greater number of sub-nodes in the future, so we will reference this into account together.

3.1 Control Message

Our proposed algorithms use the following types of control messages to efficiently build multicast tree:
Join Request (JREQ). In addition to the root node, each node will transmit such control message to the neighboring node as a request to join the above process as shown in Fig. 5.

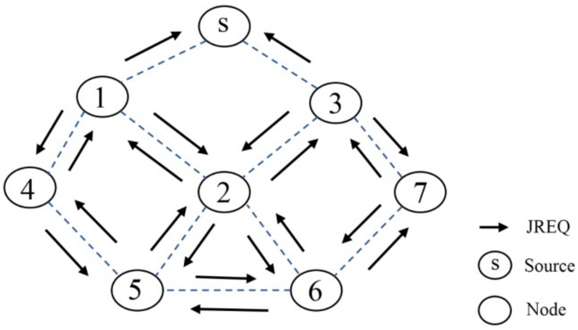

Fig. 5. The process of sending JREQ message

Join Reply (JREP). When a node receives a message JREQ, if this node is multicast tree, then return JREP message to the sender. Among them, this type of message contains information about the current construction of the sender node, namely: *SrcDist*, *Timestamp*, *ChildNum* and *Request*, the above process shown in Fig. 6.

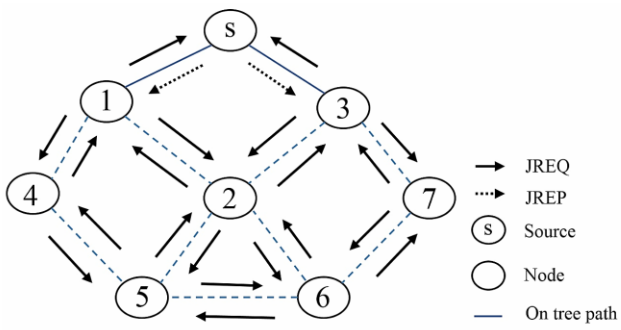

Fig. 6. The process of sending JREP message

Join Determinant (JDET). Such messages are used to inform the receiver node. The sender has selected its parent node, and you can let the receiver statistics on the number of its child nodes above process as shown in Fig. 7.

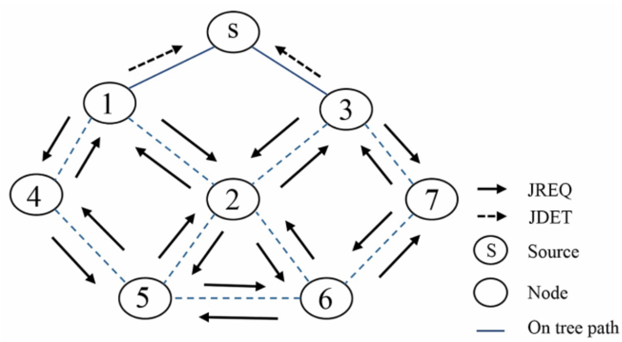

Fig. 7. The process of sending JDET message

Join Invalidate (JINV). This type of message is used when a node is disconnected from its parent. This message will tell all of its children to re-find the appropriate parent. When child node receives this message, child node also needs to re-find the appropriate parent node and transmits JINV message to their child nodes, the above process shown in Fig. 8.

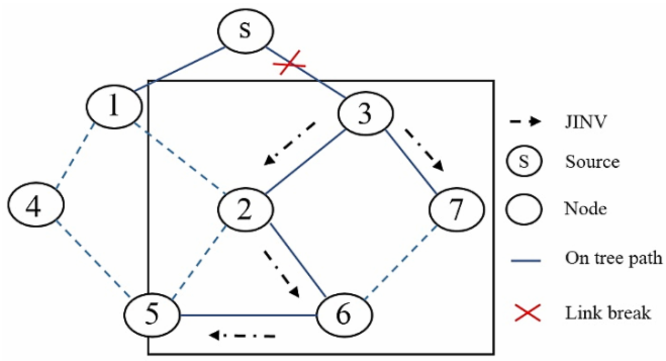

Fig. 8. The process of sending JINV message

3.2 Operating Procedures

Apart from the root node *S*, each node will send JREQ to its neighbor nodes. When *S* receives JREQ from its neighbor nodes, and then returns JREP to the sender. In addition, when other nodes of the non-root node receive the JREQ and receive the JREP within the waiting time if the node is not in the current tree structure and compare the weight of each JREP. After the waiting time is over, the JREP sender whose weight is the smallest is selected as the parent node, becomes the child node of the node, and sends the JDET message to its parent node.

On the other hand, after the multicast tree is created, each node checks whether the connection to the parent node or child node is broken at a fixed period. If the connection to the child node is disconnected, the child node set is updated. If it is disconnected from the parent node, it marks itself as not in the current tree structure and re-finds the appropriate parent node, and sends a JINV to all of its own children to inform them of the appropriate parent node. Finally, the operational flow described above is from Fig. 9 to Fig. 12.

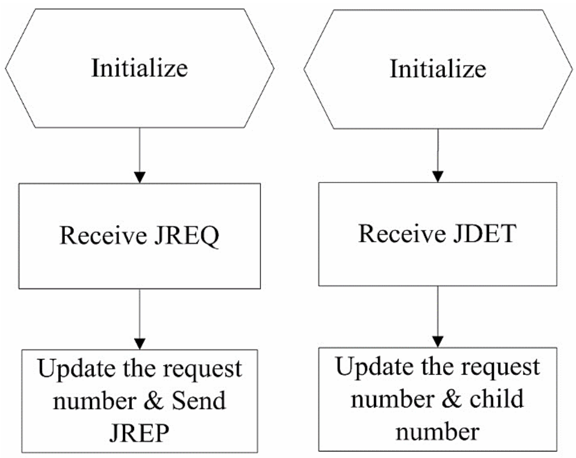

Fig. 9. A flowchart of the operation of the root node (static start)

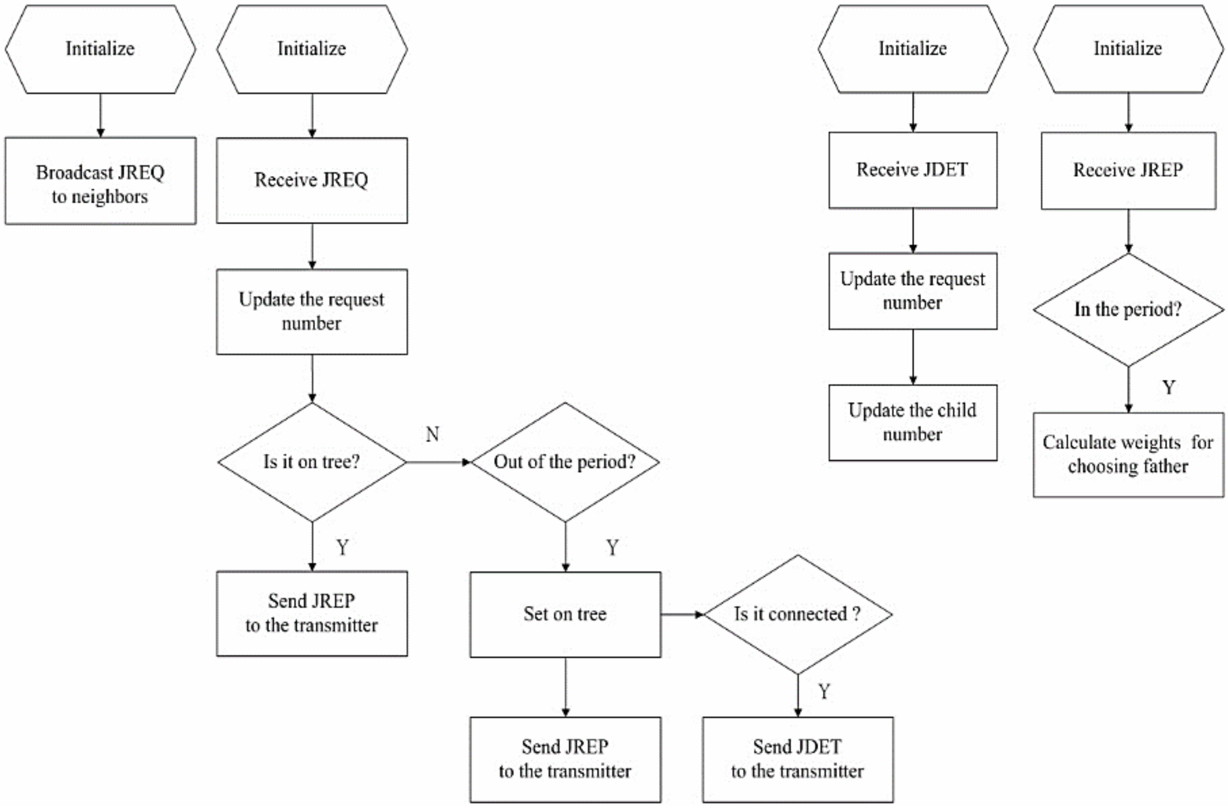

Fig. 10. A flowchart of the operation of the non-root node (static start)

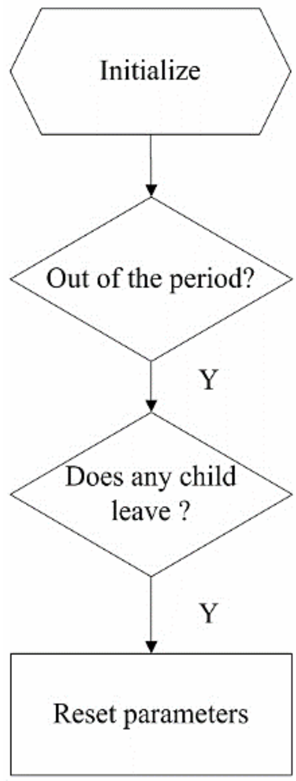

Fig. 11. A flowchart of the operation of the root node (dynamic process)

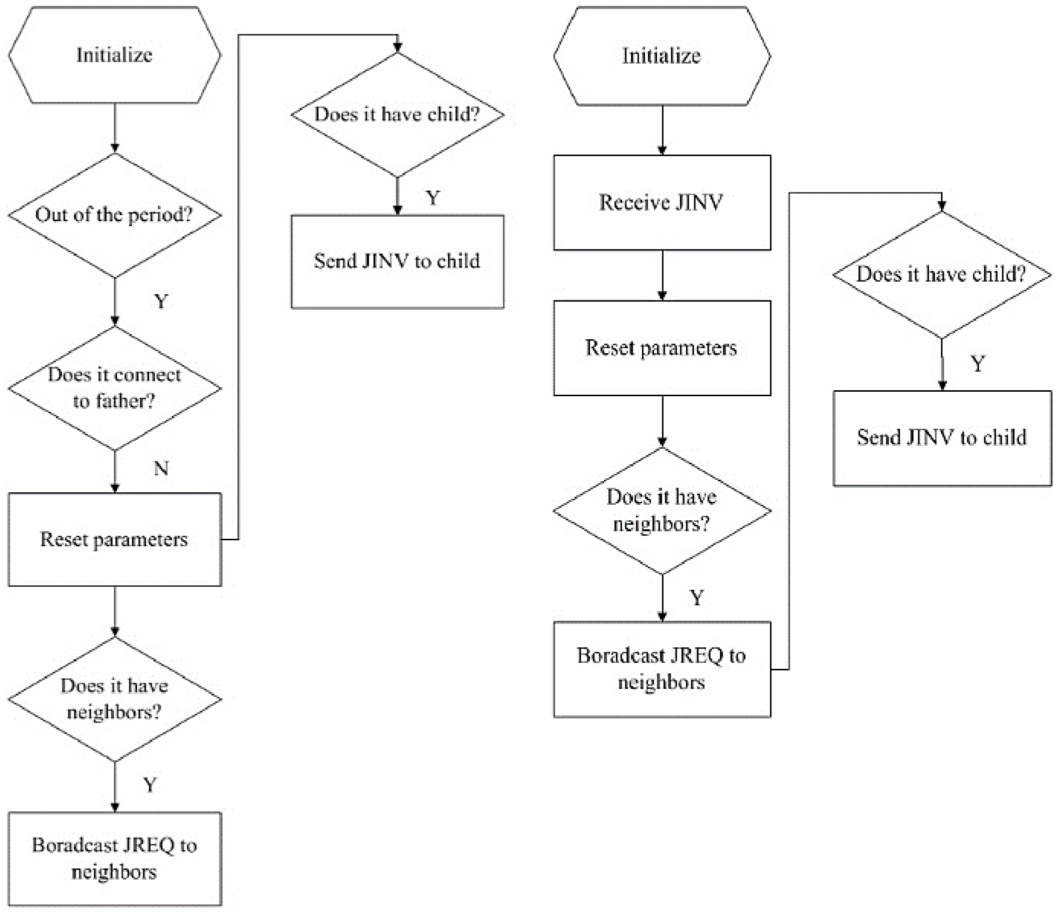

Fig. 12. A flowchart of the operation of the non-root node (dynamic process)

3.3 Weight Calculation

In the JREP message, there are four items for calculating the weights, namely *SrcDist* (the number of jumps from the source to the node), *timestamp* (the time delay for the message received by the node), *ChildNum* (the number of children to which the node is connected), and *request* (the number of requests received by the node).

The receiving end can select the lowest-weight party as its parent node, which is initialized at the beginning of the operation. When the node receives JREP, it calculates the delay between the sending of the message and the sender and adds it to its *Timestamp* field. In addition, each node after receiving JDET, it will update its *ChildNum* value. After receiving JREQ, Request value is updated. In particular, when a node determines its parent, it will be the parent node *SrcDist* value plus 1 as their *SrcDist* value. More importantly, when a node returns a JREP control message to other nodes, it will carry all the weight-related calculations described above to the receiver for reference by the receiver.

Choosing the above four benchmarks is to determine the weight of the decision to avoid a single benchmark. For example, if we choose only the transfer delay between nodes as a reference, we may discard the parent node that is temporarily congested but closer to the root node. In addition, if we did not consider the number of hops from the root to a node, then the probability of selecting a parent node with a large number of hops is increased. As a result, the quality of the transmitted message will be reduced due to the increase in the number of hops, and thus affect the size of the system throughput. On the other hand, if a node has a large number of child nodes or a larger number of received requests, it means that the node may connect with multiple child nodes in the future. Therefore, each child node will easily collide with the node when it returns data to the node. This point in the application of data collection must be included in the consideration, but also the method of this paper and the existing related technology differences.

To make a conclusion here, we use these benchmarks to build efficient multicast tree structure. When a node wants to join, you can dynamically select a minimum weight of the JREP message sender as the parent node, and you can improve the efficiency of the overall system data collection. In order to use the above parameters at the same time, we propose a weighting function as shown in equation (1), using it as a weight comparison method:

$$W(i) = 2 * SrcDist + \frac{timestamp}{5} + childNum + request \quad \textbf{(1)}$$

Wherein, in order to balance the influence of each reference, we set the *SrcDist* to 2. Experiments use the exponential function e^x to simulate the parameters of the time delay and define the range, record the time delay, and record the delay time of the previous node. In addition, the value of the transmission delay will be larger than the other reference value. In order to avoid loss of accuracy, its value is divided by 5, so that nodes can easily select the low-latency, high-efficiency path to the root node. Finally, the Fig. 13 gives an outline of the weights of the other nodes that each node needs to consider.

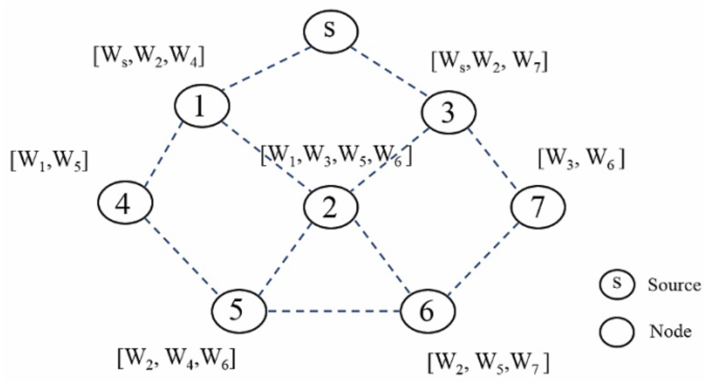

Fig. 13. A schematic of the weights considered by each node

4 The Experimental Results

In this paper, simulation software Parsec (PARallel Simulation Environment for Complex systems) [28] was used to simulate the experiment. First, in the simulation, we set up some relevant simulation parameters, as shown in Table 1. More specifically described, we will set the simulation environment 1000 × 1000 square area and the root *S* coordinates fixed in the environment directly above (500,999) at the coordinates, and then randomly generate additional 49 nodes, and the coordinates of these nodes are all within the range considered. In particular, we only consider the case of a network extension. The node represents a military vehicle, and the square area represents the battlefield.

We set up in accordance with the above, respectively static and dynamic process of starting two scripting simulation 100 times, and in each experiment, based on the average number of child nodes of all the nodes in the system, the transfer delay to the root node and the average number of hops are compared with MAODV.

Table 1. Experimental simulation parameters

Parameter	Value
Node number	50
Area	$1000*1000\ m^2$
Transmission range	200 m
Simulation time	300000 unit

4.1 Static Initial

The results of this experiment are shown from Fig. 14 to Fig. 16.

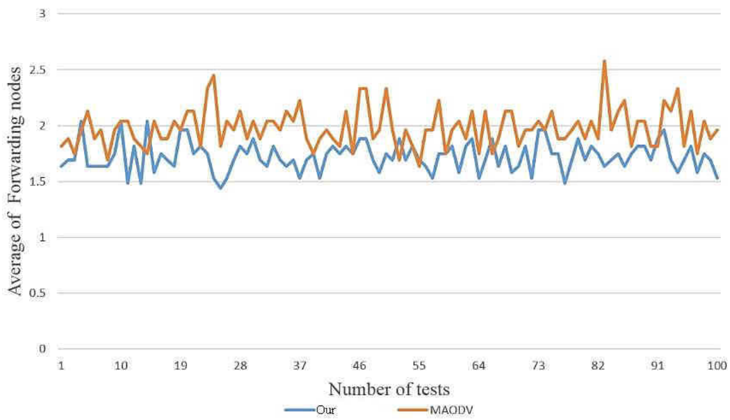

Fig. 14. Comparing the average number of child nodes (static)

First, we can see from Fig. 14 that the average number of nodes in each experiment is smaller than that of MAODV. The smaller the number of children, the smaller the chance of message collision. In particular, the average of the 100 experimental results of our method was 1.723017, and the standard deviation was 0.1313782. The mean value of MAODV was 1.9823614 and the standard deviation was 0.173466.

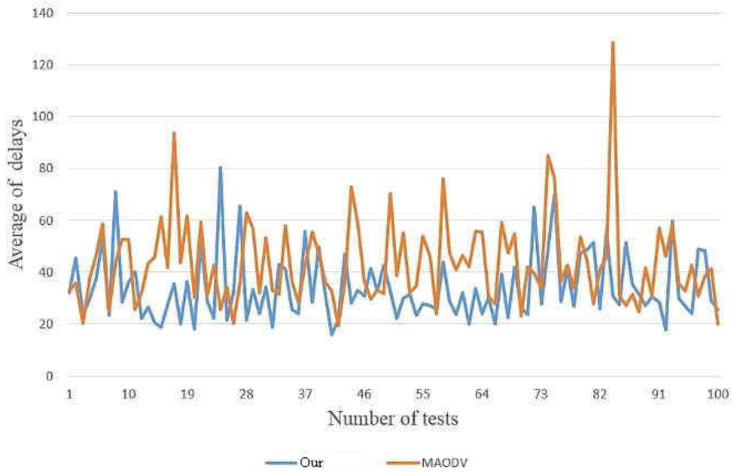

Fig. 15. Transfer delay comparison to the root node (static)

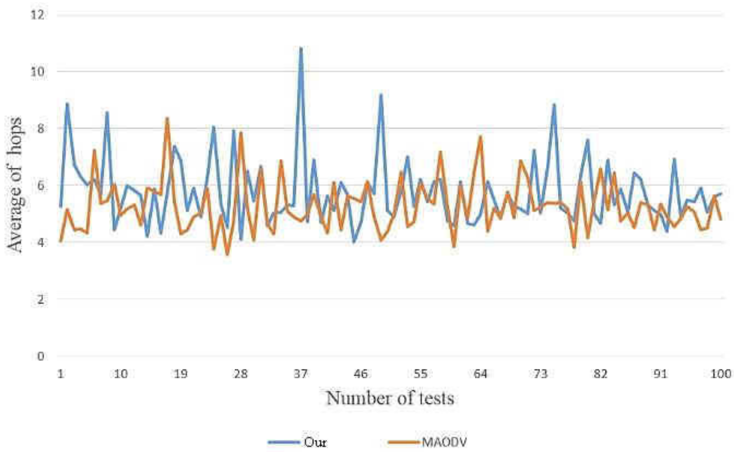

Fig. 16. The average number of hops comparison to the root node (static)

On the other hand, it can be deduced from Fig. 15 that the average of the 100 times nodes to root node transfer delay test results of our method is 34.076939 and the standard deviation is 13.024949, while the mean value of MAODV is 43.38388 and the standard deviation is 16.76644. In this paper, the transfer delay of the method is low. The longer the node's message delay, the worse the message transmission performance. In addition, we can deduce from Fig. 16. Our method to perform 100 times to the root node of the average number of jumps, and the experimental results of the average is 5.768163, and the standard deviation is 1.16521. The MAODV average is 5.251633, and the standard deviation is 0.897315. There is not much difference between the two methods. The number of hops reflects the structure of the multicast tree. The larger the number of hops, the higher the probability that nodes will transmit messages because of the longer paths.

4.2 Dynamic Process

In the simulated environment in which the nodes are moving, the same settings as in the stationary state are used, and the root node is fixed. The difference is that the non-root node will move once according to the setting in Table 2. Here, the moving distance and moving angle of the node are set to normal distribution in order to simulate the movement of the military vehicle. The experimental results of this script are shown in Fig. 17 to Fig. 18 respectively.

Table 2. Simulation parameters for node movement

Parameter	Value of normal distribution
Angle	Average: 0° Standard Deviation: 30°
Step	Average: 20 m Standard Deviation: 10 m

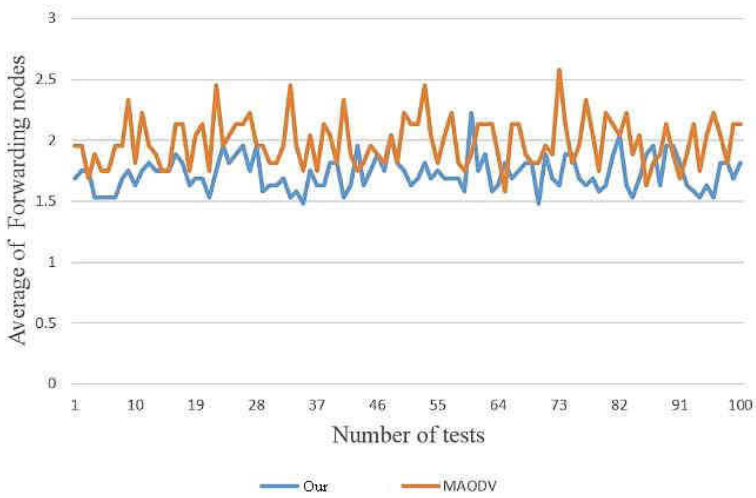

Fig. 17. Average number of child nodes (dynamic)

Similarly, we can see from Fig. 17 that the average number of child nodes of all the nodes generated by our method is almost less than the result of MAODV in each experiment. In particular, the mean value of the 100 experiments in our method is 1.7295171, and standard deviation is 0.1401298. The average value of MAODV is 1.986001 and the standard deviation is 0.196908. Therefore, the average number of child nodes of this method is also small.

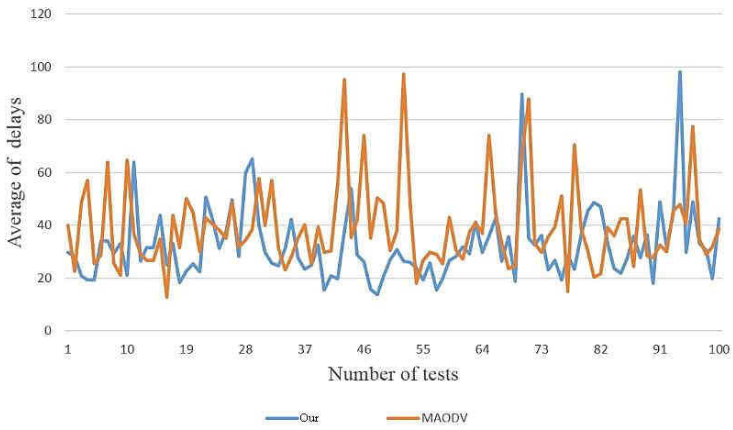

Fig. 18. Comparison of the transfer delay to the root node (dynamic)

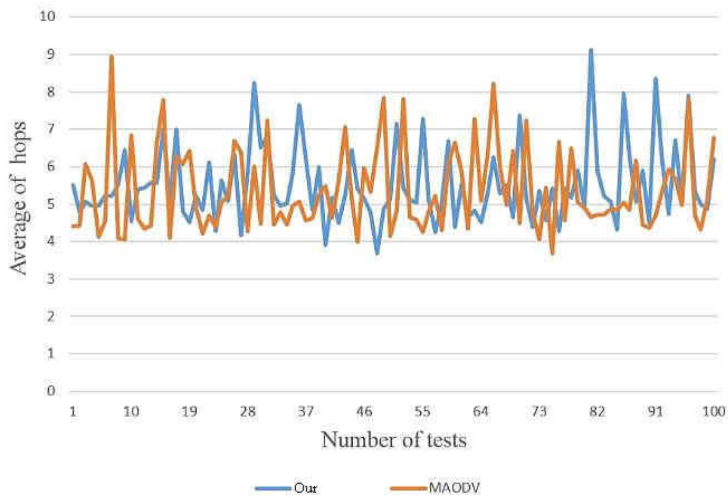

Fig. 19. Comparison of the average number of hops to the root node (dynamic)

On the other hand, it can be deduced from Fig. 18 that the average transfer delay of the 100 experimental results of our method was 32.33421 and the standard deviation was 13.659204, while the mean value of MAODV was 39.15983817 and the standard deviation was 15.983817. The transfer delay of this method is also low. In addition, from Fig. 19 can be deduced from doing our method 100 times the average number of hops to the root node experimental results is 5.5116991, and standard deviation is 1.0160507, while the MAODV average is 5.3622449, and standard deviation is 1.1088654. There is no difference between the two too much, and this is because MAODV root node is the average number of hops as the main consideration.

5 Conclusions and Future Work

In this paper, mainly for the military vehicle network data collection, we propose a new multicast tree construction algorithm for the data collection of mobile Internet of Things. It not only dynamically adjusts the multicast tree according to the current network topology information, but also takes the number of hops to the root node and the delay of the message transmission to the root node and the number of the connected child nodes to improve the whole system message delivery performance. In particular, we can see from the simulation results that compared with the traditional MAODV protocol, and our method has less number of child nodes and lower number of nodes in the case of the average number of hops to which the protocol focuses to root node transfer delay. This paper can be applied to the vehicle network system. The road vehicle group will be able to immediately return the relevant sensing data back to the back-end server database, and the battlefield on the rapid changes in the environmental information provided to the decision-makers reference.

In the future research, we can consider adding other kinds of algorithms, such as Multichannel Multicast, Ant Colony Optimization, and Particle Swarm Optimization, and compare the differences between the various algorithms to find the most appropriate way to route.

References

[1] M. Jahanshahi, A.T. Barmi, Multicast routing protocols in wireless mesh networks: a survey, Computing 96(11)(2014) 1029-1057.

[2] C. Kolias, G. Kambourakis, A. Stavrou, S. Gritzalis, Intrusion detection in 802.11 networks: empirical evaluation of threats and a public dataset, IEEE Communications Surveys & Tutorials 18(1)(2016) 184-208.

[3] A. Narmada, P.S. Rao, Zigbee based WSN with IP connectivity, in: Proc. 2012 International Conference on Computational Intelligence, Modelling and Simulation, 2012.

[4] B. Yu, L. Xu, Y. Li, Bluetooth low energy (BLE) based mobile electrocardiogram monitoring system, in: Proc. 2012 International Conference on Information and Automation, 2012.

[5] Z. Sheng, C. Mahapatra, C. Zhu, V. Leung, Recent advances in industrial wireless sensor networks towards efficient management in IoT, IEEE Access 3(2015) 622-637. doi:10.1109/ACCESS.2015.2435000

[6] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, M. Ayyash, Internet of Things: a survey on enabling technologies, protocols, and applications, IEEE Communications Surveys & Tutorials 17(4)(2015) 2347-2376.

[7] A. Al-Fuqaha, A. Khreishah, M. Guizani, A. Rayes, M. Mohammadi, Toward better horizontal integration among IoT services, IEEE Communications Magazine 53(9)(2015) 72-79.

[8] J.-J. Cheng, J.-L. Cheng, M.-C. Zhou, F.-Q. Liu, S.-C. Gao, C. Liu, Routing in Internet of Vehicles: a review, IEEE Transactions on Intelligent Transportation Systems 16(5)(2015) 2339-2352.

[9] K.M. Alam, M. Saini, A.E. Saddik, Toward social Internet of Vehicles: concept, architecture and applications, IEEE Access 3(2015) 343-357. doi:10.1109/ACCESS.2015.2416657

[10] L. Guo, M. Dong, K. Ota, Q. Li, T. Ye, J. Wu, J. Li, A secure mechanism for big data collection in large scale Internet of Vehicle, IEEE Internet of Things Journal 4(2)(2017) 601-610.

[11] Y. Fan, N. Zhang, A survey on software-defined vehicular networks, Journal of Computers 28(4)(2017) 236-244.

[12] L. Junhai, V. Danxia, X. Liu, F. Mingyu, A survey of multicast routing protocols for mobile ad-hoc networks, IEEE Communications Surveys & Tutorials 11(1)(2009) 78-91.

[13] S.K. Soni, T.C. Aseri, A review of current multicast routing protocol of mobile ad hoc network, in: Proc. the 2nd International Conference on Computer Modeling and Simulation (ICCMS' 10), 2010.

[14] A. Nargesi, M. Bag-Mohammadi, Efficient multicast tree construction in wireless mesh networks, Journal of Communications and Networks 16(6)(2014) 613-619.

[15] M.E.M. Campista, L.H.M.K. Costa, O.C.M.B. Duarte, WPR: a proactive routing protocol tailored to wireless mesh networks, in: Proc. 2008 Globecom Workshops (GC Wkshps), 2008.

[16] J. Xie, R.R. Talpade, A. McAuley, M. Liu, AMRoute: Ad hoc multicast routing protocol, Mobile Networks and Applications 7(6)(2002) 429-439.

[17] S.-J. Lee, M. Gerla, C.-C. Chiang, On-demand multicast routing protocol, in: Proc. 1999 Wireless Communications and

Networking Conference (WCNC), 1999.

[18] E.M. Royer, C.E. Perkins, Multicast operation of the ad-hoc on-demand distance vector routing protocol, in: Proc. the 5th annual ACM/IEEE international conference on Mobile computing and networking, 1999.

[19] D.T.M. Viet, N.H. Chau, W. Lee, H.Q. Thuy, Using cross-layer heuristic and network coding to improve throughput in multicast wireless mesh networks, in: Proc. 2008 International Conference on Information Networking (ICOIN), 2008.

[20] G.Z. Santoso, M. Kang, Performance analysis of AODV, DSDV and OLSR in a VANETs safety application scenario, in: Proc. the 14th International Conference on Communication Technology (ICACT), 2012.

[21] A. Kumar, R. Hans, Performance analysis of DSDV, I-DSDV, OLSR, ZRP proactive routing protocol in mobile ad hoc networks in IPv6, International Journal of Advanced Science and Technology 77(3)(2015) 25-36.

[22] T.-L. Lin, Y.-S. Chen, H.-Y. Chang, An ant colony optimization routing strategy for wireless sensor networks: a SDN-like perspective, Journal of Computers 28(4)(2017) 103-113.

[23] C.E. Perkins, E.M. Royer, Ad-hoc on-demand distance vector routing, in: Proc. the 2nd IEEE Workshop on Mobile Computer Systems and Applications, 1999.

[24] A.B. Mnaouer, L. Chen, C.H. Foh, J.W. Tantra, OPHMR: an optimized polymorphic hybrid multicast routing protocol for MANETs, IEEE Transactions on Mobile Computing 6(5)(2007) 503-514.

[25] H. Chen, Z. Yan, B. Sun, Y. Zeng, X. He, An entropy-based long-life multicast routing protocol in MAODV, in: Proc. 2009 International Colloquium on Computing, Communication, Control, and Management (ISECS), 2009.

[26] X. Chen, Q. Zhong, D. Liu, An improved MAODV based on mobility prediction and self-pruning flooding, in: Proc. 2009 International Conference on Communications and Mobile Computing (WRI), 2009.

[27] H.L. Nguyen, U.T. Nguyen, High-performance multicast routing in multi-channel multi-radio wireless mesh networks, in: Proc. 2010 Globecom Workshops (GC Wkshps), 2010.

[28] R. Bagrodia, R. Meyer, M. Takai, Y. Chen, X. Zeng, J. Martin, H.Y. Song, Parsec: a parallel simulation environment for complex systems. < http://pcl.cs.ucla.edu/projects/parsec/>, 2002.