C. Openflow Group Table for Multipath Action

Multipath routing can be implemented in SD-IoT networks using the OpenFlow protocol and Fat-Tree topology because the Fat-Tree topology has many paths that can be used as data transmission scenarios [28]. In an SDN network, the controller establishes a Transport Layer Security Protocol (TLS) connection with each switch where the switch, as a data plane, installs the flow rules in the flow table and group table so that the controller can control the data transmission path[29]. A detailed example of the group table component can be seen in Table 1 to handle packet transmission with group_id=2644423182. The switch can select bucket actions based on the path weights calculated during the path calculation process. In Table 1, the switch sends packets through ports 2, 4, and 5. The probability on each port is 0.8 for port 2, 0.15 for port 4, and 0.05 for port 5. The probability value is adjusted based on the final cost for each generated delivery path formed in the path calculation process. A weight of 0.8 is the highest probability of being chosen by the switch, meaning that the path has the lowest cost among other paths. Based on the OpenFlow rules, the bucket weight variable becomes a benchmark in the selection of paths using the group type method as select.

TABLE I
THE GROUP TABLE EXAMPLE

Group_identifier	Group_type	Action_buckets
group_id=2644423182	Select	bucket_weight=80, actions=output:2 bucket_weight=15, actions=output:4 bucket_weight=5, actions=output:5

After the delivery path is found from h1 to h2, the path is installed on each switch. By default, Openflow will give the switch device the freedom to choose actions that are defined using the scheduling algorithms that are already available. However, by defining bucket_weight, the probability of path selection is based on the value specified for this variable.

III. Results and Discussion

A. Path Discovery Results

The results of the path discovery from h1 to h2 using the DFS and the Dijkstra multipath algorithm, which are carried out five times, are presented in Tables 2 and 3. Based on five trials using the multipath DFS algorithm, a path was chosen with the smallest cost value in the 5th experiment with paths [13, 6, 2, 3, 11, 20] and a cost of 1206.88. The alternative path was path [13, 6, 2, 8, 3, 11, 20] with a cost value of 1440.84, and path [13, 6, 2, 9, 3, 11, 20] with a cost value of 1444.02. Meanwhile, from five trials using the Dijkstra multipath algorithm, there was a path with the smallest cost value in the 4th experiment with path [13, 6, 2, 3, 11, 20] and a cost value of 1263.05. Other alternative paths were on the path [13, 6, 2, 8, 3, 11, 20] with a cost value of 1498.42 and path [13, 5, 1, 2, 3, 11, 20] with a cost value of 1521.09.

TABLE II
THE RESULTS OF THE PATH DISCOVERY ON THE DFS AND DIJKSTRA MULTIPATH ALGORITHM

Algorithm	Experiment Number	Path 1	Path 2	Path 3
DFS Multipath	1	[13, 6, 2, 3, 11, 20]	[13, 6, 2, 8, 3, 11, 20]	[13, 6, 2, 9, 3, 11, 20]
	2	[13, 6, 2, 3, 11, 20]	[13, 6, 2, 8, 3, 11, 20]	[13, 5, 1, 2, 3, 11, 20]
	3	[13, 6, 2, 3, 11, 20]	[13, 6, 2, 8, 3, 11, 20]	[13, 6, 2, 9, 3, 11, 20]
	4	[13, 6, 2, 3, 11, 20]	[13, 6, 2, 3, 4, 12, 20]	[13, 6, 2, 9, 3, 11, 20]
	5	[13, 6, 2, 3, 11, 20]	[13, 6, 2, 8, 3, 11, 20]	[13, 6, 2, 9, 3, 11, 20]
Dijkstra Multipath	1	[13, 6, 2, 3, 11, 20]	[13, 6, 2, 9, 3, 11, 20]	[13, 6, 2, 8, 3, 11, 20]
	2	[13, 6, 2, 3, 11, 20]	[13, 5, 1, 2, 3, 11, 20]	[13, 6, 2, 8, 3, 11, 20]
	3	[13, 6, 2, 3, 11, 20]	[13, 6, 2, 8, 3, 11, 20]	[13, 6, 2, 9, 3, 11, 20]
	4	[13, 6, 2, 3, 11, 20]	[13, 6, 2, 8, 3, 11, 20]	[13, 5, 1, 2, 3, 11, 20]
	5	[13, 6, 2, 3, 11, 20]	[13, 6, 2, 9, 3, 11, 20]	[13, 5, 1, 2, 3, 11, 20]

TABLE III
THE RESULTS OF THE PATH DISCOVERY COST ON THE DFS AND DIJKSTRA MULTIPATH ALGORITHM

Algorithm	Experiment Number	Path 1	Path 2	Path 3
DFS Multipath	1	1229.81	1466.54	1481.00
	2	1314.52	1552.12	1572.22
	3	1213.55	1439.96	1454.42
	4	1223.35	1457.81	1460.72
	5	1206.88	1440.84	1444.02
	Average	1237.62	1237.62	1482.48
Dijkstra Multipath	1	1320.44	1568.62	1583.20
	2	1333.09	1583.99	1598.33
	3	1332.51	1590.12	1592.46
	4	1263.05	1498.42	1521.09
	5	1296.31	1547.33	1551.53
	Average	1309.08	1557.70	1569.32

B. Path Generation Time Results

The results of the path generation time for the DFS and Dijkstra's multipath algorithm are presented in Tables 4 and 5. The evaluation was done to determine the time required for the path-searching algorithm that has been developed to be able to find all paths in the Fat-Tree network topology. The results of time readings from each experiment had very different values. This was due to the unstable nature of the emulation using mininet [30]. The average path generation time from the five trials on the DFS multipath algorithm was 450,597 ms, and the Dijkstra multipath algorithm was 1087,962 ms. The multipath DFS algorithm had a much different average time difference from the multipath Dijkstra algorithm.

TABLE IV
THE RESULTS OF THE PATH GENERATION TIME ON THE DFS MULTIPATH ALGORITHM

Experiment Number	Path Generation Time (ms)	Average Time (ms)
1	443.095	
2	644.23	
3	181.574	450.597
4	723.966	
5	260.12	

TABLE V
THE RESULTS OF THE PATH GENERATION TIME ON THE DIJKSTRA MULTIPATH ALGORITHM

Experiment Number	Path Generation Time (ms)	Average Time (ms)
1	1081.162	
2	1001.268	
3	1014.267	1087.962
4	1320.178	
5	1022.934	

In accordance with the results in research [17], the results of the execution time testing carried out on the same three algorithms, namely the modified DFS multipath algorithm was 0.0903 ms, the multipath DFS algorithm was 0.0858 ms, and the Dijkstra multipath algorithm was 0.901 ms. The test results in these two studies showed that the multipath DFS algorithm had a faster path generation time than the Dijkstra multipath algorithm.

C. QoS Extraction Results

The bandwidth variable was extracted by statistical analysis on the network being tested by collecting statistical measurement data on Wireshark, such as the number of packets received in bytes. The results of data extraction were converted into KiloBytes (KB). Based on the results listed in Table 6, it can be seen that the results of bandwidth testing using the MQTT protocol on the DFS and Dijkstra multipath algorithms resulted in 825.372. While bandwidth testing using CoAP and HTTP protocols has the same results at 708 KB. Each protocol produced similar results for both algorithms.

TABLE VI
THE BANDWIDTH RESULTS

Algorithm	Bandwidth Average (KB)		
	MQTT	CoAP	HTTP
DFS Multipath	825.372	708	708
Dijkstra Multipath	825.372	708	708

The other QoS experiment was the throughput variable. The emulation was carried out to know a network's ability in actual data transfer. Throughput is a calculation of the value of a packet that has been successfully received at the destination for a certain period and divided by the length of the time interval. The throughput calculation was manually processed by taking Wireshark's statistical data in bytes and period (s). The results of throughput testing using three different protocols that have been carried out are presented in Table 7. The obtained values showed the same pattern for each protocol. MQTT, CoAP, and HTTP protocols had the same throughput value in each algorithm.

TABLE VII
THE THROUGHPUT RESULTS

Algorithm	Throughput Average (KB)		
	MQTT	CoAP	HTTP
DFS Multipath	81.60885	59.37517	48.1166
Dijkstra Multipath	81.60983	59.37401	48.1146

The third QoS test was delay and jitter, which was done by extracting the reception time between the received packets using the Wireshark application, which was then processed manually using the delay and jitter formula. The delayed test was also carried out using three different protocols, MQTT, CoAP, and HTTP, within five trials, and 100 packets were accumulated for each trial. The results of calculating the 100 packets are calculated on an average, as presented in Table 8. Jitter testing was performed to determine the delay variation between data transmission on the network that could occur due to the influence of traffic load on the network itself. Jitter testing was done by using the data from the calculation of the delay, and then all the resulting values were divided by the number of packets received. Referring to the delay and jitter values, all tested protocols produced relatively the same average values. However, there was a tendency for better delay variations in the DFS algorithm because the preprocessing (path generation time) was not too significant.

TABLE VIII
THE DELAY AND JITTER RESULTS

Algorithm	MQTT		CoAP		HTTP	
	Delay Average	Jitter Average	Delay Average	Jitter Average	Delay Average	Jitter Average
DFS Multipath	1.00136	0.00012	1.17234	0.00379	1.48629	2.97236
Dijkstra Multipath	1.00135	0.00013	1.01026	0.11952	1.48635	2.97247

Packet loss testing was calculated from the percentage of transmitted packets that were not received by the server described in Table 9. Based on the test results, both algorithms can effectively determine the path, as evidenced by the packet loss value equal to 0%.

TABLE IX
THE PACKET LOSS RESULTS

Algorithm	Packet Loss Average (%)		
	MQTT	CoAP	HTTP
DFS Multipath	0	0	0
Dijkstra Multipath	0	0	0

IV. CONCLUSION

Significant results were obtained on the path discovery experiment's path generation time variable between the DFS and Dijkstra multipath algorithm. In the QoS variable, the value obtained gives the same pattern in both algorithms because the generated path for sending data on the network is approximately the same as the overall multipath routing algorithms. From the results of the path discovery using two different algorithms, the DFS multipath algorithm can find delivery paths faster than Dijkstra. This is because the path search process using the DFS algorithm is carried out by expanding to nodes in a graph.

After the node expansion stage, it is followed by a backtracking process to get the entire path from the source node to the destination. In Dijkstra's algorithm, the path-searching process uses a greedy strategy where each traversed node is selected based on the smallest weight between the nodes selected in the path and other nodes not selected to produce the shortest path. From the results that have been analyzed, it can be concluded that multipath DFS is superior to Dijkstra's multipath algorithm when applied to SD-IoT with reference to the path generation time value. In addition, processing delay in calculating routing metrics does not impact the delay or packet loss value in the data transmission process. In future research, the authors plan to develop a routing model based on Artificial Intelligence (AI) that can automatically select a routing algorithm according to the real-time conditions on the network.

ACKNOWLEDGMENT

The authors are grateful to the University of Muhammadiyah Malang and the UMM Informatics Laboratory for supporting the implementation of this research.

REFERENCES

[1] N. Hossein Motlagh, M. Mohammadrezaei, J. Hunt, and B. Zakeri, "Internet of Things (IoT) and the Energy Sector," *Energies (Basel)*, vol. 13, no. 2, p. 494, Jan. 2020, doi: 10.3390/en13020494.

[2] S. N. Swamy and S. R. Kota, "An Empirical Study on System Level Aspects of Internet of Things (IoT)," *IEEE Access*, vol. 8, pp. 188082–188134, 2020, doi: 10.1109/ACCESS.2020.3029847.

[3] M. Pang, X. Yao, and M. Geng, "A computing resource scheduling strategy of massive IoT devices in the mobile edge computing environment," *The Journal of Engineering*, vol. 2021, no. 6, pp. 348–357, Jun. 2021, doi: 10.1049/tje2.12040.

[4] R. Thamilselvan, K. T. Selvi, R. R. Rajalaxmi, and E. Gothai, "Multipath Routing of Elephant Flows in Data Centers Based on Software Defined Networking," *Int J Eng Adv Technol*, vol. 9, no. 2, pp. 2714–2717, Dec. 2019, doi: 10.35940/ijeat.B3258.129219.

[5] G. A. Mutiara, N. Suryana, and O. Mohd, "WSN nodes power consumption using multihop routing protocol for illegal cutting forest," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 18, no. 3, p. 1529, Jun. 2020, doi: 10.12928/telkomnika.v18i3.14844.

[6] P. A. Y. and R. Balakrishna, "Implementation of optimal solution for network lifetime and energy consumption metrics using improved energy efficient LEACH protocol in MANET," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 17, no. 4, p. 1758, Aug. 2019, doi: 10.12928/telkomnika.v17i4.12004.

[7] N. Sultana, N. Chilamkurti, W. Peng, and R. Alhadad, "Survey on SDN based network intrusion detection system using machine learning approaches," *Peer Peer Netw Appl*, vol. 12, no. 2, pp. 493–501, Mar. 2019, doi: 10.1007/s12083-017-0630-0.

[8] M. M. Azmi and F. D. S. Sumadi, "Low-Rate Attack Detection on SD-IoT Using SVM Combined with Feature Importance Logistic Regression Coefficient," *Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control*, Jun. 2022, doi: 10.22219/kinetik.v7i2.1405.

[9] W. D. Nanda and F. D. S. Sumadi, "LRDDoS Attack Detection on SD-IoT Using Random Forest with Logistic Regression Coefficient," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 6, no. 2, pp. 220–226, Apr. 2022, doi: 10.29207/resti.v6i2.3878.

[10] R. Wazirali, R. Ahmad, and S. Alhiyari, "SDN-OpenFlow Topology Discovery: An Overview of Performance Issues," *Applied Sciences*, vol. 11, no. 15, p. 6999, Jul. 2021, doi: 10.3390/app11156999.

[11] F. Rhamdani, N. A. Suwastika, and M. A. Nugroho, "Equal-Cost Multipath Routing in Data Center Network Based on Software Defined Network," in *2018 6th International Conference on Information and Communication Technology (ICoICT)*, May 2018, pp. 222–226. doi: 10.1109/ICoICT.2018.8528730.

[12] N. Ahmed and S. Misra, "Collaborative Flow-Identification Mechanism for Software-Defined Internet of Things," *IEEE Internet Things J*, vol. 9, no. 5, pp. 3457–3464, Mar. 2022, doi: 10.1109/JIOT.2021.3099822.

[13] J. Ali and B. Roh, "An Effective Approach for Controller Placement in Software-Defined Internet-of-Things (SD-IoT)," *Sensors*, vol. 22, no. 8, p. 2992, Apr. 2022, doi: 10.3390/s22082992.

[14] T. Modi and P. Swain, "FlowDCN: Flow Scheduling in Software Defined Data Center Networks," in *2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, Feb. 2019, pp. 1–5. doi: 10.1109/ICECCT.2019.8869180.

[15] X. Zhong, L. Zhang, and Y. Wei, "Dynamic Load-Balancing Vertical Control for a Large-Scale Software-Defined Internet of Things," *IEEE Access*, vol. 7, pp. 140769–140780, 2019, doi: 10.1109/ACCESS.2019.2943173.

[16] Md. S. Hossen, Md. H. Rahman, Md. Al-Mustanjid, Md. A. Shakil Nobin, and Md. A. Habib, "Enhancing Quality of Service in SDN based on Multi-path Routing Optimization with DFS," in *2019 International Conference on Sustainable Technologies for Industry 4.0 (STI)*, Dec. 2019, pp. 1–5. doi: 10.1109/STI47673.2019.9068057.

[17] S. Syaifuddin, M. F. Azis, and F. D. S. Sumadi, "Comparison Analysis of Multipath Routing Implementation in Software Defined Network," *Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control*, May 2021, doi: 10.22219/kinetik.v6i2.1228.

[18] C. Chen, F. Xue, Z. Lu, Z. Tang, and C. Li, "RLMR: Reinforcement Learning Based Multipath Routing for SDN," *Wirel Commun Mob Comput*, vol. 2022, pp. 1–12, Feb. 2022, doi: 10.1155/2022/5124960.

[19] S. L. Aljohani and M. J. F. Alenazi, "MPResiSDN: Multipath Resilient Routing Scheme for SDN-Enabled Smart Cities Networks," *Applied Sciences*, vol. 11, no. 4, p. 1900, Feb. 2021, doi: 10.3390/app11041900.

[20] D. Y. Setiawan, S. Naning Hertiana, and R. M. Negara, "6LoWPAN Performance Analysis of IoT Software-Defined-Network-Based Using Mininet-Io," in *2020 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS)*, Jan. 2021, pp. 60–65. doi: 10.1109/IoTaIS50849.2021.9359714.

[21] Md. T. Islam, N. Islam, and Md. al Refat, "Node to Node Performance Evaluation through RYU SDN Controller," *Wirel Pers Commun*, vol. 112, no. 1, pp. 555–570, 2020, doi: 10.1007/s11277-020-07060-4.

[22] S. Julius Fusic, P. Ramkumar, and K. Hariharan, "Path planning of robot using modified dijkstra Algorithm," in *2018 National Power Engineering Conference (NPEC)*, Mar. 2018, pp. 1–5. doi: 10.1109/NPEC.2018.8476787.

[23] M. Singh and G. Baranwal, "Quality of Service (QoS) in Internet of Things," in *2018 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU)*, Feb. 2018, pp. 1–6. doi: 10.1109/IoT-SIU.2018.8519862.

[24] D. Silva, L. I. Carvalho, J. Soares, and R. C. Sofia, "A Performance Analysis of Internet of Things Networking Protocols: Evaluating MQTT, CoAP, OPC UA," *Applied Sciences*, vol. 11, no. 11, p. 4879, May 2021, doi: 10.3390/app11114879.

[25] N. Nikolov, "Research of MQTT, CoAP, HTTP and XMPP IoT Communication protocols for Embedded Systems," in *2020 XXIX International Scientific Conference Electronics (ET)*, Sep. 2020, pp. 1–4. doi: 10.1109/ET50336.2020.9238208.

[26] T. Tun, "A Forensics Analysis of ICMP Flooded DDoS Attack using WireShark," *Transactions on Networks and Communications*, vol. 8, no. 3, pp. 08–15, Jun. 2020, doi: 10.14738/tnc.83.8250.

[27] R. Jawaharan, P. M. Mohan, T. Das, and M. Gurusamy, "Empirical Evaluation of SDN Controllers Using Mininet/Wireshark and Comparison with Cbench," in *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, Jul. 2018, pp. 1–2. doi: 10.1109/ICCCN.2018.8487382.

[28] P. Qiao, X. Wang, X. Yang, Y. Fan, and Z. Lan, "Joint Effects of Application Communication Pattern, Job Placement and Network Routing on Fat-Tree Systems," in *Proceedings of the 47th International Conference on Parallel Processing Companion*, Aug. 2018, pp. 1–10. doi: 10.1145/3229710.3229747.

[29] S. H. Mohammed and A. D. Jasim, "Evaluation of Firewall and Load balance in Fat-Tree Topology Based on Floodlight Controller," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 17, no. 3, p. 1157, Mar. 2020, doi: 10.11591/ijeecs.v17.i3.pp1157-1164.

[30] Y. Ergiz, A. M. Demirtas, and T. Girici, "Joint multipath flow and layer allocation for scalable video streaming," *Computer Networks*, vol. 191, p. 107995, May 2021, doi: 10.1016/j.comnet.2021.107995.