Path determination for network slicing in wireless mesh disaster networks

Alexander Seng[1], Prof. Dr. -Ing. Ulrich Trick[1], Prof. Dr. Armin Lehmann[1], and Prof. Dr. Bogdan Ghita[2]
[1]Frankfurt University of Applied Sciences, Research Group for Telecommunication Networks, Frankfurt, Germany, {seng, trick, lehmann}@e-technik.org
[2]Centre for Security, Communications and Network Research, University of Plymouth, Plymouth, UK , bogdan.ghita@plymouth.ac.uk

Abstract

Network Slicing is one of the critical enablers for the upcoming 5G mobile networks. This approach allows the creation of different, separated virtual networks based on the same physical infrastructure. Wireless mesh networks are self-organizing and self-configuring, which qualifies them to supply larger areas with a communication infrastructure in a quick and less complicated way. As a result, there are different areas of application for the use of such networks, e.g. restoring communication infrastructure in a disaster area. One aspect of network slicing in wireless mesh networks is the determination of paths through the network. Optimising the path determination can lead to improvements of the overall network performance. This work is about the creation of a graph model of the wireless mesh network and the comparison between algorithms for finding the optimal path through that modelled network. It also looks at prior research and it's limitations.

1 Introduction

Network Slicing enables the possibility to provide different services with contradicting requirements on the same infrastructure. Based on the physical network infrastructure, the network provides a virtual network "slice" for different use cases. Network Slicing is one of the key enablers for the functionality of modern 5G networks [2]. The concepts proposed so far are only for application in 5G networks[9][8]; wireless mesh networks[13] (WMNs) were not subject of development and research. The approaches for WMNs that exist so far deal with the resource slicing approach in wireless networks and mostly describe network virtualisation. However, an end-to-end network slicing approach has to fulfil the complete functionality of a network that provides specific services. The network architecture of 5G networks and WMNs is entirely different. WMNs have a decentralised architecture and have radio-based connections to other nodes. Additionally, in this special case as a disaster network, all functionalities in the network must be provided by the WMN nodes. In a WMN, new nodes can also join or leave the network, resulting in topology changes [11]. In contrast to WMNs , in 5G Networks, only the access network part is radio based. The other parts of the network are wired connections. Data centers for the network functionality are also available in these networks. Because new base stations cannot simply join or leave the network, the network architecture of 5G networks is more static than in WMNs. Because of these reasons, it is not possible to transfer the principle of network slicing from 5G networks to wireless mesh networks without further research.
One of the key aspects of using network slicing is providing and guaranteeing different Quality of Service (QoS) levels to different tenants. Various existing approaches use different types of physical separation of the data streams [6] [17] [14], which has the additional benefit of complete isolation. However, working on physical layer often needs modification on the devices' drivers, making it more complicated to add new nodes to the network, especially if they are from other vendors. Another possibility is to create mechanisms in the higher OSI layers to provide QoS and isolation of the slice specific traffic. These have the benefit that they are mostly independent of the mechanisms in the physical layer, which leads to a better interoperability between different network devices. One possibility for example is to use software-defined networking (SDN) and network functions virtualisation (NFV). The advantage of this is that network slicing uses these methods already. This combination allows the provision of QoS requirements by selecting different paths through the network and flexible replacing the virtual network functions.
This study focuses on the path determination to create a virtualised infrastructure that provides nodes and links as base for a network slice. Three types of search algorithms are tested for path determination purpose: A* [12] as a greedy algorithm with heuristic, Djikstra [7] as greedy algorithm and Breadth-First Search [5] as an uninformed algorithm. The rest of this paper is organised as follows: Section two describes the architectural particularities of wireless mesh networks. Section three presents related work regarding to using NFV and SDN to improve the QoS in a Network. Section four describes the modelling of the wireless mesh network and the network slices based on it. Section five, compares algorithms that determine the optimal path between the nodes and virtual network functions (VNFs) of a slice. Section six concludes the paper and gives an outlook to further work in this area.

2 Architectural particularities of wireless mesh networks

Wireless mesh networks have a number of specific characteristics in contrast with the other network types, like 5G or other networks, which are built using dedicated core and access network parts. A technical and architectural peculiarity of wireless mesh networks is that the nodes can belong either to core, access or transport network, depending on the situation. For example, in Fig 1, a node with a connected client works in the access network but simultaneously in the transport network because it also forwards frames through the network. If this particular node hosts core functions, e.g. webserver, call server or proxy server, it is also working in the core network. To reach maximum flexibility, every node should be able to work in every part of the network. Therefore, there is no physical separation of the nodes that belong to the different network parts. The core network of the WMN describes the network functions (physical or virtual) and the logical connections between them, as, for example the connection between a webserver or call server and a database. An access network is only necessary on nodes with connections to clients, which could be on every node in the network. Access networks that serve multiple clients can be a bottleneck for the service performance because there is no alternative path between node and client available. The wireless links between the WMN nodes form the transport network, including the ingress nodes, the nodes that host the VNFs, intermediate nodes on the path, and eventual egress nodes. In this context, network slicing describes a number of aspects: the slicing of the core network, the transport network and the slicing of the access network. Slicing in the core network means creating the virtual network functions needed for the slice and their connections. These connections are virtual and necessary when multiple VNFs need to be traversed. The functions and connections are then transferred to the physical infrastructure of the WMN. Transport network slicing describes the processing of the slice specific traffic between the nodes from source to destination of the traffic. One key aspect is how the traffic is forwarded to ensure the QoS requirements of the slice specific services. Access network slicing describes the mechanisms for association and multiplexing of the radio resources in the access network part. Slicing in the access network is only required on nodes that have a connection to clients. This means that slicing in the access network is only necessary when the corresponding node has connections to end devices. Because this work focusses on path determination, the results can be applied to the area of transport network slicing.

3 Related Work

This section describes the related work on improving network QoS through path optimisation it the context of network slicing The work in [3] deals with the interference aspect in a sliced wireless network. The authors provide a management scheme that uses interference between the wireless links for topology decisions. The interference gets a weight based on the priority of the corresponding slice. Based on a weighted sum of the interferences, the proposed algorithm creates a path through the wireless network which has to minimize the interference sum of the other slices to create an optimised path.

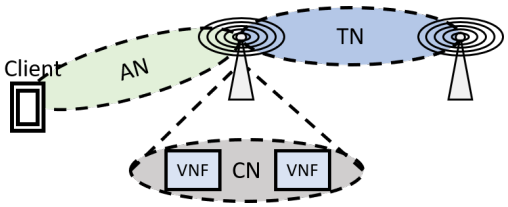

Fig 1 Schematic representation of access, transport and core network

In [16], the authors describe a replacing approach for service function chains (SFC). A SFC describes a combination of paths and virtual network functions. The focus of this work lies in the minimisation of resource consumption and as few adjustments as possible, through the relocation of VNFs to different network parts that can host them.

The work in [15] describes the so called "middlebox placement problem", which provides the optimal placing of network functions in a service chain. In this work, the authors use two heuristic algorithms; a greedy algorithm and a algorithm based on simulated annealing, which are used for replacing the network functions rather than changing the path through a network.

Another work that mainly focuses on the VNF replacing aspect is [4]. The main focus for optimisation in this work is the end-to-end delay of a SFC. The authors also use an heuristic algorithm for optimising this parameter by replacing the VNFs.

Building virtual networks on a shared physical infrastructure refers also to the virtual network embedding(VNE) problem. The authors of [10] created a survey to this problem. The VNE-problem describes algorithms to map a virtual network (nodes and links) to an underlying physical network. This includes the resources of the nodes and the links.

The papers outlined in this section provide an overview of the efforts to improve the network performance through virtualisation technologies and path determination. Performance stability is a crucial aspect, especially in wireless networks, because of their radio based nature. The combination of both approaches, path determination and VNF replacing, is also usable to locate a network slice in a specific area of a wireless mesh network. Therefore, it is possible to build the slice in the area of the WMN, where it is needed. As a first step, this work will compare the overall link utilisation in a WMN-Model when using different algorithms for path determination.

4 WMN and Slice Model

This section proposes a model of the wireless mesh network and the network slices. The model is necessary for describing optimisation approaches and for algorithmic processing. As highlighted by the previous section, wireless mesh networks require a different approach to modelling to presenting the network infrastructure.

4.1 Model of the overall network

The basis for the WMN model is an undirected graph $G(N,E)$, with the nodes N and the edges E [3]. The edges of the graph represent the connections between the WMN nodes, the nodes are the mesh nodes of the WMN. Fig. 2 gives an example of a graph representation of a wireless mesh network. The adjacency matrix $A = \{a_{i,j}\}_{NxN}$ describes how the nodes are connected. The matrix elements $a_{i,j}$ are one if there is a connection between the nodes i and j and zero if there is no connection. Every connection $e \in E$ has a maximum consumable bitrate $B(e)$, which represents the transmission speed of the link. Because of the wireless nature of the connections in a WMN, the bitrate is not a constant value and it depends mainly on the distance between the nodes, the interference, and obstacles in the environment. It is possible to combine the bitrate information with the adjacency matrix. In this case, the element $a_{i,j}$ contains the value of $B(e)$ if there is a connection and zero if there is none. An example for the extended definition shows (1) and (2)

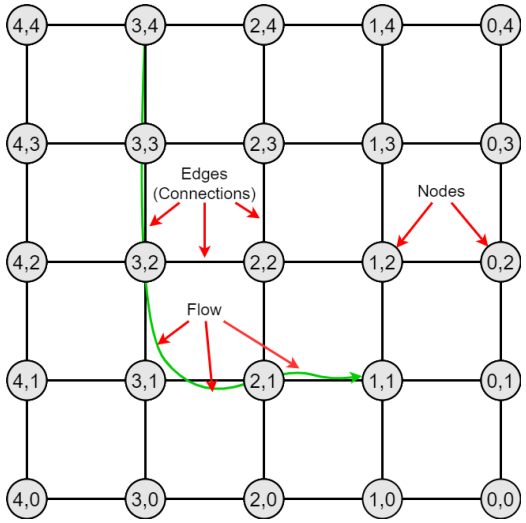

Fig 2 WMN architecture as undirected Graph

$$A = \begin{bmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{bmatrix} \quad (1)$$

$$A_B = \begin{bmatrix} B(a_{0,0}) & B(a_{0,1}) & B(a_{0,2}) & B(a_{0,3}) \\ B(a_{1,0}) & B(a_{1,1}) & B(a_{1,2}) & B(a_{1,3}) \\ B(a_{2,0}) & B(a_{2,1}) & B(a_{2,2}) & B(a_{2,3}) \\ B(a_{3,0}) & B(a_{3,1}) & B(a_{3,2}) & B(a_{3,3}) \end{bmatrix} \quad (2)$$

Any flow f on a connection e consumes a part of the maximum available bitrate $b_e(f)$. As a result, the number of flows that can work with a specific bitrate, e.g. 10 Mbit/s, a link can supply is limited. The resulting constraint shows (3). A flow f describes a communication relation between a start node, an end node, and intermediate virtual network functions if present.

$$\sum_{f \in F} b(f) \leq B(e) \quad (3)$$

Every WMN node has an amount of computing power C for providing virtual network functions (VNFs). The value of C is the CPU utilisation in per cent: $C \in [0\%, 100\%]$. A VNF v running on a node needs a part of the node's computing power $c(v)$ which increases the overall value of C. The maximum number of VNFs a node can provide without performance degradation due to CPU overload results from the constraint of (4):

$$\sum_{v \in V} c(v) \leq C_n \quad (4)$$

The last constraint of the WMN model is the link delay and the resulting overall delay of a flow. The overall delay of a flow L_f is the sum of all link delays $l(e)$ the flow passes through the network. Equation (5) describes this relation.

$$L_f = \sum_{e \in E_f} l(e) \quad (5)$$

As a result, every connection in the WMN model has two properties: Bitrate $B(e)$ and link delay $l(e)$. The WMN nodes have CPU utilisation C as the main property. It is also possible to define additional properties for these elements, e.g. packet loss on a connection or the energy consumption of a node. The defined model is, in principle, extensible.

4.2 Model of the slices in the network

A network slice consists of the following components: Virtual network functions, a priority value to define different slice priorities, and the flows belonging to the slice. Derived from the flows of the VNFs, a slice is a subgraph $G_s(N_s,E_s) \subseteq G(N,E)$ [3] of the WMN graph. N_s is the set of nodes that host the slice VNFs or forward the traffic of a slice flow. E_s is the set of edges that connect the nodes of N_s. To get the information necessary for N_s and E_s, the model uses two structures: The flow-edge-matrix and the node-VNF-matrix. The flow-edge-matrix R describes the relation between flows and edges. The definition is: $R = \{a_{f,e}\}_{|F|x|E|}, f \in F, e \in E$ were $a_{f,e} = 1$ if a flow f uses the edge e and $a_{f,e} = 0$ otherwise. As an extension, it is possible that the matrix also provides information about the bitrate of a connection. In this case $a_{f,e} = b_e(f)$ if a flow f uses edge e and $a_{f,e} = 0$ otherwise. The node-VNF-matrix P describes the positions of the VNFs in the network. The definition of this matrix is: $P = \{a_{n,v}\}_{|V|x|N|}, v \in V, n \in N$ where $a_{n,v} = 1$ if node n provides the VNF v and $a_{n,v} = 0$ otherwise.

5 Path determination

This section is about the path determination process in a sliced wireless mesh network. In the graph model of the WMN, a path consists of edges that a flow has to use from it's source to destination. To minimize the utilization of the paths through the network, algorithms for finding the shortest path between source node an destination node should be used. This section compares three different algorithms for path determination in a graph. The used algorithms are:

- A* Algorithm [12]
- Djikstra Algorithm [7]
- Breadth-First Search(BFS) [5]

The A* and the Djikstra algorithm are informed search algorithms and also belong to the category of greedy algorithms. A* and Djikstra are well-known search algorithms for the shortest paths in a graph. Both use cost information on the links of the graph which they use to minimise the cost from the start node to the destination node. A* additionally uses a heuristic function to estimate the costs until the target is reached. The BFS algorithm belongs to the uninformed search category. It does not use any additional information from the graph to determine the shortest path. It is used here as the default benchmark for comparison with the other two algorithms.

5.1 Test setup

The algorithms run on a graph that represents a wireless mesh network for testing, as described in the previous section. The graph consists of 25 nodes that are connected with 40 edges. The test setup graph shows Fig. 2. The edges have the two parameters: latency and consumed bitrate on the connection. The value of the consumed bitrate increases with every flow that goes over a specific edge. The simulation was implemented using Python with the Networkx [1] package for the graph processing.
The greedy algorithms need a cost value on the connections to work properly. This value has to increase with the consumed bitrate on an edge, so that the costs increase with the link utilisation. Also, the costs using a link with low utilisation should be lower than using a link with high utilisation when adding the same bitrate value. Therefore, the cost increases exponentially. Since the connections simulating wireless links, there is a value for the maximum consumable bitrate on a link before the performance of the connection decreases significantly. In the simulation, this value is set to 100 *Mbit/s*. If this value is reached, the cost value has to be infinity, so the greedy algorithms cannot use the corresponding edges anymore. The A* algorithm requires a definition of a heuristic. This setup uses the Manhattan-Distance and Euclidian Distance between the start and destination nodes as heuristic.
In the first test, the simulation adds a number of flows with fixed start and destination nodes and a specified bitrate value to the network and calculates the resulting paths with one of the algorithms. The added flows are shown in table 1 and table 2 with their corresponding bitrate values in *Mbit/s*. The tables show three different setups of the first test. In the first and second setup, the number of flows is equal and has the value ten. The changes are in the bitrate, as table 1 shows. Table 2 shows the third test setup. In this case, five additional flows are added to the initial ones from table 1.
The bitrate values are summed up and added to the corresponding edges as utilisation parameter. After that, the algorithms calculate a test path between the nodes 1,0 and 3,4. The simulation then calculates the maximum bitrate consumption on the edges of the test path. The higher this value, the higher is the utilisation of the test path because the performance of a path is as good as the worst link. This is done in all three cases of the first test with the three algorithms to compare the resulting values. Another important value to compare is the number of overloaded and therefore unavailable links.
A second test case checks how many paths these algorithms can determine before a link is overloaded. In this case, the simulation adds an increasing number of paths with random start and end nodes to the network. Afterwards, the simulation checks each connection of the graph whether the maximum value for the consumed bit rate has been exceeded. The simulation counts the number of flows until the first connection exceeds this value. This is done with every algorithm to compare how efficient the path determination works. The higher the counted value, the better the algorithm worked.

Table 1 Test setup 1 and 2 for test case 1

Flows	From	To	rate 1	rate 2
f1	4,0	1,4	15	30
f2	4,3	0,3	10	20
f3	3,0	3,4	15	30
f4	1,1	2,3	20	40
f5	0,4	3,1	10	20
f6	2,4	4,1	10	20
f7	1,0	3,4	15	30
f8	4,3	2,0	20	40
f9	2,2	4,1	10	20
f10	2,2	0,0	10	20

5.2 Test results

This part shows the results from the described test cases. Table 3 shows the results from the first test case that is defined by the tables 1 and 2. The values for the bitrate are the maximum bitrates on an edge out of all the edges in the network when using the specified algorithm. The lower this value, the better the result. The results of the first test setup show that there is no difference between the values of the maximum bitrate between A* and Djikstra, both have a value of 20 *Mbit/s*. In contrast, the BFS algorithm has already a value of 40 *Mbit/s*. In this case, no algorithm created an overloaded link. In the second test setup, the resulting values are 30 *Mbit/s* for A* and 40 *Mbit/s* for Djikstra.

Table 2 Test setup 3 for test case 1

Flows	From	To	rate
f1	4,0	1,4	30
f2	4,3	0,3	20
f3	3,0	3,4	30
f4	1,1	2,3	40
f5	0,4	3,1	20
f6	2,4	4,1	20
f7	1,0	3,4	30
f8	4,3	2,0	40
f9	2,2	4,1	20
f10	2,2	0,0	20
f11	2,4	2,1	15
f12	0,3	4,0	20
f13	4,1	2,3	15
f14	3,1	1,4	30
f15	3,4	0,1	15

Table 3 Results for bitrate consumption and overloaded links

Test setup	Algorithm	Max. bitrate	Overloaded links
Setup 1	A*	20	0
	Djikstra	20	0
	BFS	40	0
Setup 2	A*	30	0
	Djikstra	40	0
	BFS	80	1
Setup 3	A*	70	0
	Djikstra	60	0
	BFS	125	4

Table 4 Maximum possible paths before overload

Algorithm	Max. paths
A*	31
Djikstra	31
BFS	16

The value for BFS is 80 *Mbit/s* which makes a difference of 50 *Mbit/s* compared to A* and 40 *Mbit/s* compared to Djikstra. The BFS algorithm also creates the first overloaded link in this test setup. In the third test setup, the resulting values are 70 *Mbit/s* for A* and 60 *Mbit/s* for Djikstra. The value for BFS is 125 *Mbit/s* which makes a difference of 55 *Mbit/s* compared to A* and 65 *Mbit/s* compared to Djikstra. BFS creates four overloaded links in this setup, whereas this value is still zero with A* and Djikstra. Due to the small differences between A* and Djikstra, using Djikstra is the better choice because it needs no heuristic function to work. However, a different heuristic function for A* could lead to other results.

Table 4 shows the results of the second test. The values are the average of 100 iterations, because the simulation creates random paths in the network. The bitrate value is a random value between 10 and 30 *Mbit/s*. In this specific case, there is no difference between A* and Djikstra. Both algorithms have the first overloaded link after an average value of 31 paths. In contrast to this, the BFS algorithm reaches the first overloaded link at an average value of 16 paths.

As shown by the results of both test scenarios, the differences between the greedy algorithms and the uninformed algorithm increase as the number of paths in the network increases. Furthermore, the difference between A* and Djikstra is not significant.

6 Conclusion

This paper compared the A*, Djikstra, and BFS search algorithms to determine their performance in identifying a path between the nodes of a wireless network graph model: greedy algorithms with heuristic (A*), greedy algorithms without heuristic (Djikstra) and uninformed search algorithms (BFS). The tests compare these in terms of maximum path utilisation, the number of overloaded paths and how many path the network can provide until the first link is overloaded. The focus was only on path determination, the VNF placement is beyond the scope of this work. The results show a significant difference between the uninformed search algorithm and the greedy algorithms. Furthermore, there was no significant difference between the A* algorithm and the Djikstra algorithm. A reason for that could be the used heuristic for A*; it is possible that using a better heuristic leads to a different result. For this reasons, using a greedy algorithm like Djikstra is the pragmatic solution, because a better heuristic requires more knowledge about the network.

Further work will combine the results with an approach for VNF placement to get an optimisation mechanism for mapping the virtual resources of the slices to the underlying physical infrastructure. Another aspect for further work in this area is the wireless resource sharing in the access network. It is necessary to optimise the resource sharing between an access node and end devices because it is not possible to optimise a path on this connection.

7 References

[1] Networkx — networkx documentation, 27.07.2021.

[2] 5GPPP. View on 5g architecture: 5gppp architecture working group: Version 2.0, december 2017.

[3] Namwon An, Yonggang Kim, Juman Park, Dae-Hoon Kwon, and Hyuk Lim. Slice management for quality of service differentiation in wireless network slicing. *Sensors*, 19(12), 2019.

[4] C. Ouyang, Y. Wei, S. Leng, and Y. Chen. Service chain performance optimization based on middlebox deployment. In *2017 IEEE 17th International Conference on Communication Technology (ICCT)*, pages 1947–1952, 2017.

[5] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Al-*

gorithms, Third Edition. MIT Press, Cambridge, UNITED STATES, 2009.

[6] Yilong Deng and Akihiro Nakao. Mesh slicing: Improving robustness of a mesh network via multiple slices. 2011.

[7] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.

[8] ETSI. Ts 123 501 - v15.7.0 - 5g; system architecture for the 5g system (5gs) (3gpp ts 23.501 version 15.7.0 release 15).

[9] ETSI. Ts 128 530 - v15.3.0 - 5g; management and orchestration; concepts, use cases and requirements (3gpp ts 28.530 version 15.3.0 release 15).

[10] Andreas Fischer, Juan Felipe Botero, Michael Till Beck, Hermann de Meer, and Xavier Hesselbach. Virtual network embedding: A survey. *IEEE Communications Surveys Tutorials*, 15(4):1888–1906, 2013.

[11] G. Frick, A. Paguem Tchinda, B. Shala, U. Trick, A. Lehmann, and B. Ghita. Requirements for a distributed nfv orchestration in a wmn-based disaster network. *International Conference on Information and Communication Technologies for Disaster Management (ICT-DM)*, 2019.

[12] Peter Hart, Nils Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[13] IEEE, editor. *IEEE standard for information technology–telecommunications and information exchange between systems–local and metropolitan area networks–specific requirements: Part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications ; Amendment 10: Mesh networking*. Institute of Electrical and Electronics Engineers, New York, 2011.

[14] K. Nakauchi, Y. Shoji, and N. Nishinaga. Airtime-based resource control in wireless lans for wireless network virtualization. In *2012 Fourth International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 166–169, 2012.

[15] Jiaqiang Liu, Yong Li, Ying Zhang, Li Su, and Depeng Jin. Improve service chaining performance with optimized middlebox placement. *IEEE Transactions on Services Computing*, 10(4):560–573, 2017.

[16] Gustavo Miotto, Marcelo Caggiani Luizelli, Weverton Luis Costa Da Cordeiro, and Luciano Paschoal Gaspary. Adaptive placement & chaining of virtual network functions with nfv-pear. *Journal of Internet Services and Applications*, 10(1):1–19, 2019.

[17] S. Zehl, A. Zubow, and A. Wolisz. Hotspot slicer: Slicing virtualized home wi-fi networks for air-time guarantee and traffic isolation. In *2017 IEEE 18th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–3, 2017.