Fast and Cost-Efficient Virtualized Network Function Placement Algorithm in Wireless Multi-hop Networks

Zahra Jahedi and Thomas Kunz(✉)

Carleton University, Ottawa, ON K1S 5B6, Canada
zahrajahedi@cmail.carleton.ca, tkunz@sce.carleton.ca

Abstract. Network Function Virtualization (NFV) can lower the CAPEX and/or OPEX for service providers and allows the deployment of services quickly. The main challenge in the use of Virtualized Network Functions (VNF) is optimally placing them in the physical network in terms of deployment cost and resource consumption. The critical problem of VNF placement is inherently NP-hard and the available optimal solutions do not scale with respect to the network size. The problem of NFV placement is even more challenging in wireless networks as we are facing the issue of scarcity of BW due to the presence of interference. Therefore, this paper aims to solve the problem of VNF placement in wireless multi-hop networks by considering BW limitations and scalability. We tackle both issues at once by limiting the search space to the shortest paths. We search for the placement solution along shortest paths to minimize the BW consumption and at the same time reduce the search space to the nodes and links along the shortest path. The results are compared to a mathematical optimization model and a comparable heuristic model. They show that our proposed heuristic greatly decreases the execution time in comparison to the mathematical model and the alternative heuristic while keeping the acceptance ratio close to the optimal solution.

Keywords: Network function virtualization · Wireless multi-hop network · Network function placement · Integer linear programming

1 Introduction

Network Function Virtualization (NFV) brings new opportunities and challenges. One of the main challenges is the optimized placement of the virtualized functions based on the characteristics and available resources of the network [6]. Placement of Network Functions (NF) can affect the path traffic flows take and consequently bandwidth usage in the network [8]. A chain of NFs with predefined parameters is referred to as a Service Graph (SG). The placement of all NFs of an SG is a Network Function Embedding Problem (NFEP): mapping the Virtual Network Functions (VNF) and the links between them to the physical network [6]. NFEP can be modeled as a mathematical optimization problem that can be solved using different Linear Programming (LP) solvers/tools [6]. Mathematical optimization models are proven to be NP-hard and are not scalable. The solution for this issue is to design a heuristic with lower complexity that can provide a near-optimal solution. In this paper we propose a Fast And Cost-Efficient (FACE) heuristic that achieves two key objectives: minimizing the consumption of network resources while at the same time accepting as many placement requests as possible.

Compared to wired networks, multihop wireless networks such as MANETs, VANETs, or wireless sensor networks suffer from severe bandwidth (BW) limitations. That is due to a number of reasons: typical wireless technologies operate at lower transmission rates, compared to wired technologies such as Ethernet, etc. Also, when multihop wireless networks are built up from devices using a single radio, flows interfere with themselves (a node that is a relay between source and destination can only either receive or transmit, but not both at the same time). Finally, wireless technologies typically experience significant interference (either from other flows or due to the above self-interference), significantly lowering the available BW for each link. Our designed heuristic is based on solving the problem of NF placement faster than the optimization models while minimizing the BW consumption. Limiting our search to the shortest paths between the source and destination of the request will lower the size of the search space, which reduces the execution time while reducing the BW consumption.

We use a Breadth First Search (BFS) method to calculate all shortest paths for all possible pairs of nodes in the network prior to running the placement algorithm. Upon the arrival of a request we extract these pre-computed shortest paths based on the request's source and destination. Among all shortest paths we start with the one that increases the chance of successful placement of the request. Then the NF that has the fewest options for placement will be chosen (based on the nodal resources of nodes along the shortest path and the resource request of that NF). We will then iteratively place other NFs, backtracking if necessary when an unfeasible solution is encountered. The process will be repeated until all NFs are being placed.

We compare our results against a mathematical optimization model and a similar heuristic. The collected results show the effectiveness of our approach in lowering the execution time and providing near-optimal acceptance ratio. It can be seen from the comparisons that although the execution time has been decreased drastically, the acceptance ratio is close to the acceptance ratio of the optimal approach. The recorded results shows that FACE can solve the placement problem much faster than either the optimization model or the alternative heuristic. Due to these properties, FACE is particularly able to solve the NFEP in larger networks in real-time, compared to the alternatives.

The remainder of this paper is organized as follows: Section 2 discusses some of the most recent related work on NFEP and the characteristics of the heuristics in the related papers. Section 3 introduces our heuristic model and its different stages. Section 4 describes the modeling environment and results. We conclude the work in Section 5.

2 Related Work

The amount of work on NFEP is considerable. This problem can be modeled by using mathematical methods or by designing a heuristic algorithm. The mathematical methods will provide optimal results (based on a defined objective function) but are proven to be NP-hard. They are not applicable to large scale networks and it is common to develop a heuristic algorithm. The mathematical methods for solving the optimization problem can be different forms of Linear Programming (LP), Non-Linear Programming (NLP), etc. Constraints can be defined based on the limitations of the physical network, and NFs, and objectives are defined to minimize one or multiple parameters.

There are various optimization models proposed for placement in wired and wireless networks. As our aim here is to provide a fast and cost-efficient heuristic applicable to wireless multi-hop networks, we focus our review on heuristic algorithms and only briefly introduce the optimization model we employ to compare the performance of our heuristic model. In our previous work [5] we proposed a mathematical optimization model for placement of the SGs in wireless multi-hop networks. That model uses Integer Linear Programming (ILP) to place a chain of NFs and includes interference as a BW constraint. The objective of the optimization model is to minimize the mapping cost based on the requirements of the NFs and available resources in the network. We use the interference model introduced in [5] and considered the effect of interference in calculating BW consumption by the request placement. As our results showed, the solution time for even smaller networks grew fast, making this approach not attractive for larger scenarios. However, for smaller networks we can use the results from this model to evaluate the performance of any proposed heuristic.

Designing a heuristic algorithm can be an alternative solution for NFEP in wired and wireless networks with less computational demand and near optimal performance. Here we review the recent proposed heuristics that took unique approaches and provided novel methods for mapping SGs' NFs to a physical network.

The authors of [2] broke the problem of an SG placement into sub-problems of placing each NF of an SG and the link connected to the NF. The authors showed that the multi-stage algorithm can reduce the execution time in comparison to the optimization model. However the lower execution time is reached by sacrificing the number of accepted requests. As the placement problem is being broken into smaller parts, the proposed algorithm does not have information about the whole problem. It optimizes placement of each NF, not the whole SG. [3] uses Dynamic Programming (DP) to organize the problem into smaller interdependent sub-problems of placing each VNF and the virtual link connected to it towards the next VNF. The solutions for the sub-problems are then aggregated to compose the overall chain placement. [3] compared its method of dynamic programming with the multi-stage method. They showed that both methods' execution times are similar, since both can find solutions in polynomial time. The proposed heuristic in [3] only optimizes the placement of each NF, not the whole SG, which lowers the execution time and also decreases the number of accepted requests.

The proposed heuristic in [11] consists of 3 parts. First, it computes the list of physical node candidates for each VNF, then sorts them based on the number of physical node candidates for placement in increasing order. In the last step, the heuristic computes the placement cost of that VNF and its virtual link to the physical network and chooses the one with the lowest cost. Prioritizing the placement of NFs with lower options for placement will improve the acceptance ratio, but lack of considering the whole SG during the placement is a shortcoming of this method. [12] places NFs one by one based on their order in the SG. [12] exploits the intuition of finding the nearest server which supports the first NF in the chain of NFs for each flow. After this step, the algorithm removes the VNF under consideration from the chain and finds the nearest server that supports the next VNF of the chain and so on. The proposed heuristic in [12] is fast and simple but only considers optimization for each NF not the whole SG.

Some heuristics, such as the one proposed in [9], focus on designing an algorithm which can be combined with the optimization model to reduce the execution time of the model. A sampling-based Markov approximation (MA) approach is proposed in [9] to solve the NP-hard problem which requires a long convergence time. The method begins with a random feasible solution, and iterates the process of transformation from the current solution to another feasible solution until the steady-state distribution of the Markov chain appears. To reduce the execution time, the solution space is reduced to a subset of randomly chosen nodes that satisfy the resource demands of a request. It is been stated that the problem can be solved in polynomial time but the execution time of the algorithm is not being mentioned or compared with other proposed heuristics with similar time complexity.

[10] narrows the target search space of VNF placement by introducing a smaller accessible scope where the locations of VNFs are confined. The requests are categorized based on their source and destination. The nodes with lowest sum of distance from source and destination are in the accessible scope of the request. The size of each accessible scope for each set of requests is proportional to the total traffic volume of those request. It is shown in [10] that the size of the accessible scope will impact the time efficiency and performance of the NF placement. Considering all nodes to be in the accessible scope will not reduce the execution time but will provide the acceptance ratio of the optimization model. On the other hand, a very small accessible scope will decrease the execution time but also the acceptance ratio. This approach, unlike the previous heuristics, considers the whole SG and its source and destination. In the design of our heuristic we adopted this idea to narrow the search space, discussed in more detail in the next section.

In the design of a heuristic model for wireless networks, the scarcity of bandwidth should be considered and given priority. We saw that many heuristics first place the NFs and then connect them, which is not efficient in terms of bandwidth consumption. Keeping a balance between reducing the execution time and increasing the acceptance ratio is another factor that should be considered. We can not oversimplify our heuristic model and select nodes randomly without considering its impact on future requests and expect to achieve a high acceptance ratio. One of the interesting methods we reviewed here was the one proposed in [10]. We will be using the idea behind their heuristic in our design to reduce the search space for the nodes that are along shortest paths between the source and destination of a request. This reduction in the search space will decrease the execution time. At the same time, it constrains the placement of NFs to be on paths that minimize BW consumption. We believe it is beneficial to give priority to those NFs that have lower number of candidates for placement, as discussed in [11], and will consider this factor in our placement too.

3 Fast and Cost-Efficient(FACE) Heuristic Model

As we mentioned earlier, some methods use a reduced search space idea. For example, we could consider only nodes that are on the shortest path, easily identified by the fact that the sum of their distance from source and destination equals the hop count of the shortest path(s). However, we also need to identify the links over which the data will flow, and not all links among this subset of nodes will be links that belong to the shortest paths. We therefore explicitly look for all shortest paths between the source and destination of the request and limit our search for an efficient placement to these paths. With the use of a shortest path, we are reducing the BW consumption and at the same time reducing the execution time of the placement algorithm. The placement algorithm starts with searching for all possible shortest paths between all possible source-destination pairs in the network. Requests arrive one at a time and our algorithm will attempt to place them. As a request arrives, the shortest paths will be extracted based on the request's source and destination and the search for a cost-efficient placement will be limited to these shortest paths. The requests have BW demand and nodal resource requirements for each NF. The physical network consists of nodes that have nodal resource and links that have available BW.

In this section, we provide a detailed description of the algorithm that searches for all shortest paths and then will provide a detailed description of the placement algorithm.

3.1 Search for All Shortest Paths

The placement algorithm starts with searching for all possible shortest paths between all possible pairs of source and destination in the network. We use a search method similar to Breadth First Search (BFS). Assume the physical network is a graph where its nodes are the vertices of the graph and the links are the edges. The BFS explores the edges of the graph to discover the vertex that is reachable from the source node. It computes the shortest distance from the source to each reachable vertex in the graph. We made some changes to the BFS to start from the source and end when it reaches the destination node. Also, in addition to the distance, we record the shortest paths themselves. In our search for shortest paths we define one array and one matrix for each node u in the physical network:

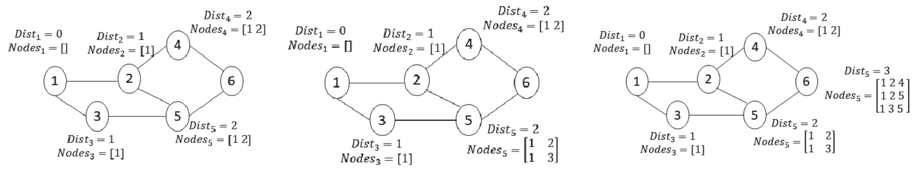
- $dist_u$: An array that represents the shortest distance in terms of the number of hops from the source node.
- $nodes_u$: A matrix which records nodes involved in each different shortest paths found from source node to node u.

The initial value of $dist$ for all nodes is infinity, except for the source node which is equal to 0. The initial matrix of $nodes$ for all nodes is empty. The search algorithm starts traversing the physical network graph and while visiting neighbor y of node x it compares the value of $dist_y$ with $dist_x + 1$. If $dist_y$ is greater than $dist_x + 1$ it means that $dist_y$ describes a path longer than the shortest path. So we decrease $dist_y$ to $dist_x + 1$ and assign $nodes_x$ to $nodes_y$. If $dist_y = dist_x + 1$ then it means we found another shortest path to node$y$. In this case $nodes_y$ is the union of $nodes_x$ and $nodes_y$. The pseudo-code of this search algorithm is presented in Algorithm 1. This algorithm can find all possible shortest paths for all pairs of nodes. The output of the algorithm is p, which is a set of shortest paths p_{ij} for each pair of nodes in the physical network.

Algorithm 1: Finding all shortest paths

```
Result: $nodes_{dest}$ that contains all shortest paths
x is the source node;
y are the neighbors of node x;
while $y \sim destination$ do
    if $dist_y > dist_x + 1$ then
        $dist_y \leftarrow dist_x + 1$;
        $nodes_y \leftarrow nodes_x$;
    else if $dist_y = dist_x + 1$ then
        $nodes_y \leftarrow [nodes_y; nodes_x]$;
    $x \leftarrow y$;
    $y \leftarrow neighbors - of - y$;
end
```

The following example demonstrates how we update the parameters of each node during the search for all shortest paths. As it is shown in Fig. 1, we consider a network of 6 nodes and want to find all shortest paths from node 1 to 6. In the first stage, we update the parameters of the source node's neighbors, which are nodes 2 and 3. Figure 1a shows the second stage and updated parameters of the neighbors of node 2 and Fig. 1b shows the third stage, after we updated the parameters for neighbors of node 3. In the third stage, when processing node 5, $dist_5 = dist_3 + 1$. So we update $nodes_5$ to the union of $nodes_3$ and $nodes_5$. Figure 1c shows the final stage and all shortest paths from 1 to 6 can be found in $nodes_6$.



a) Second step b) Third step c) Final step

Fig. 1. Second, third, and final stages of finding all shortest paths.

3.2 Placement Algorithm

The requests arrive one at a time and are placed separately. Each request has a duration, once an accepted request expires, it will be removed from the network and the associated used resources will be released. Our placement algorithm can be divided into three main parts.

1. Selecting a candidate shortest path.
2. Selecting the NF to place.
3. Selecting the node for NF placement.

Select a Candidate Shortest Path. To keep our algorithm fast and eliminate the shortest paths with insufficient resources, we first check the availability of bandiwdth (BW) and nodal resources. To check the availability of BW in shortest paths L_p we consider the BW consumption by virtual links and the effect of interference. We use the protocol interference model widely used in the literature [4] and our own prior work [5] which defines an interference set for each link in the physical network. The interference set for each link consists of all the links that are connected to the nodes in the transmission range R of the sender or receiver. $d_{u'u}$ represents the distance between node u and u'. The $intset_{E_{uv}}$ captures that transmission on the link (E_{uv}) between node u and v will affect the BW usage of all the links whose transmitter is within the transmission range of the sender u or the receiver v.

$$\forall E_{uv} \in L_p :$$
$$intset_{E_{uv}} = \{E_{u'v'} | d_{u'u} \vee d_{v'v} \vee d_{v'u} \vee d_{u'v} \leq R\}$$

The BW check limits the search to the shortest paths with sufficient available BW. The nodal resource check depends on the placement but we perform an easy check to eliminate the shortest paths that cannot be used for the placement of the SG. We consider the case for the placement of the most problematic NF, which is the one with the highest nodal resource demand. If we cannot find any node along the shortest path with enough nodal resource to place that NF, that shortest path will be eliminated.

To choose the shortest path which is more likely to have sufficient nodal resources, we sort the shortest paths based on their minimum nodal resources in decreasing order. The shortest path with the maximum-minimum nodal resource will be chosen for the placement. If the placement at any stage was not successful, we return to this list and choose the next shortest path with maximum-minimum nodal resource. At every stage of our placement algorithm, we give priority to the options that increase the probability of successful placement in order to place as many requests as possible in a speedy manner.

Select a NF to Place. Now that we have chosen a candidate shortest path we can start placement of the NFs along the path. We start from the NF that is hardest to place. The NFs are sorted based on the number of possible candidate nodes in increasing order. A candidate node parameter $candid_{f_i}$ is defined for each NF of the SG and is equal to the number of nodes along the shortest path that can be used for the placement of that specific NF. In choosing the nodes along the shortest path we consider two parameters: a node has to provide sufficient nodal resources, and the NFs that previously were placed. The order of the NFs in the SG is fixed and we can not re-order them. Furthermore, we do not want to have a placement that passes a physical link more than once. E.g. if the third NF of the SG is being placed in the second node of the shortest path, subsequent NFs in the SG can not be placed in the first node. The candidate nodes are being chosen based on the placement of previous NFs to avoid loops and backtracking in the placement. If there are no candidate nodes for any of the NFs at any stage of placement, the chosen path is infeasible and the placement process will choose the next shortest path with maximum-minimum nodal resource and repeat the process of NF placement.

Select a Node for Placement. To place the chosen NF in one of the nodes along the shortest path we sort its candidate nodes based on their index difference and choose the node with the lowest index difference. The index of the nodes along the shortest path is equal to their order in the shortest path e.g. the source node's index is one. The index of a NF is equal to its order in the SG, e.g. the index of the first NF of the SG is one and the index of second NF is two. We compare the index of the chosen NF with the index of the candidate nodes and choose the one with the minimum index difference with the chosen NF. In the end, the available resources of the nodes, BW of the links, and the list of candidate nodes for the remaining NFs will be updated.

4 Modeling Environment and Results

Two platforms are being used to solve the placement problems: MATLAB to implement our heuristic algorithm, and AMPL to solve the mathematical optimization model and the alternative heuristic proposed in [10]. AMPL is a modeling language designed to be used for solving optimization problems such as linear and non-linear programming problems [1]. We used AMPL to solve the optimization model and the compared heuristic as it works with a wide range of solvers. We used BARON for solving our optimization model in AMPL as described in more detail in [1]. Unlike AMPL, which is designed for solving optimization models, MATLAB allows us to develop our heuristic algorithm for VNF placement.

The wireless topologies are generated with the use of the method proposed in [7]. where the nodes are randomly deployed in a square area, based on a uniform distribution. We generate topologies with 20, 30, and 40 nodes, the network area grows with the number of nodes. We keep the average node density constant, consequently the network size ranges from $490 * 490\,\mathrm{m}^2$ for the 20 node network to $980 * 980\,\mathrm{m}^2$ for the 40 nodes network [7]. Nodes in the wireless network are directly connected if their distance is less than or equal to the transmission range of the nodes. This transmission range is constant for all nodes and we verified that all of the generated topologies are connected.

We used the same parameter value as [6] in order to be able to compare our results. Nodal resources of nodes and the bandwidth of links are values uniformly distributed between 100 and 150 in all network scenarios. The flows arrive over time following a Poisson process with an average rate of four flows per 100 time units. Each flow has a lifetime, exponentially distributed with an average of $\mu = 500$ time units and is accompanied by a SG, defining the required NFs and their interconnection to handle this flow. There are 6 NFs per request. The nodal resource demands of each NF follows a uniform distribution between 1 and 20. The bandwidth requirement of all links of the request is the same and chosen uniformly from between 1 and 50 units.

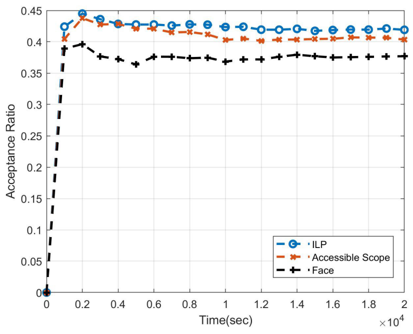
4.1 Measurement Metrics

To measure the performance of our proposed heuristic and compare its performance with the other models we used the following parameters.

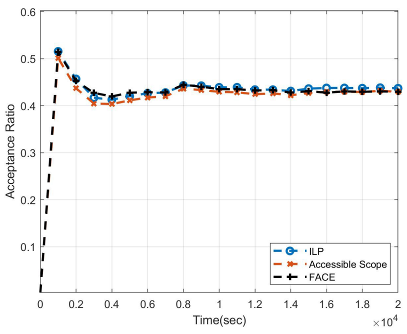
- Acceptance ratio: The total number of accepted requests divided by the total number of requests.
- Average BW Cost: Average of the BW units used for the deployed requests that are not expired. Note that this includes the bandwidth of links actually used by flows, as well as the bandwidth consumed on adjecent links due to interference.
- Execution time: The total time that it takes to place all requests in the course of an experiment simulation 20,000 s.

4.2 Results

We applied our heuristic model to wireless multi-hop networks of increasing size to evaluate its performance in terms of the time it takes to solve the placement problem and its success in placing the requests. To benchmark our results, we applied the Integer Linear Programming (ILP) model for wireless multi-hop networks introduced in [6] to the same topologies and the same set of requests as our heuristic model. Finally, we compared our results with the accessible scope heuristic proposed in [10] that we reviewed earlier.



20 nodes networks



40 nodes networks

Fig. 2. Comparing acceptance ratios of three models, 20 and 40 nodes network.

Table 1. Execution times in seconds for networks of different size

Network size	20	30	40	100
ILP model	142.6	427.3	967.22	–
The accessible scope heuristic	49.9	66.5	96.80	–
FACE	5.70	6.4	10.3	143.82

Figure 2 shows the acceptance ratios recorded for wireless networks of 20 and 40 nodes for all three approaches. It can bee seen from Fig. 2a that our heuristic model's acceptance rate is lower than the recorded acceptance ratios for both the ILP model and the accessible scope heuristic model for networks of 20 nodes. We expected to have a lower number of accepted requests than the other models as we strictly limit our search space to the shortest paths between source and destination of the request. The accessible scope heuristic imposes fewer limitation on its search space than ours and considers all the nodes that are involved in the shortest paths, which provide more options for placement of SGs than our model. Figure 2b shows the acceptance ratios recorded for all three models for larger networks of 40 nodes. It can be seen from the figures that the recorded acceptance ratios of all three models become closer to each other as the size of networks grows (a trend we also observed with the intermediate network size of 30 nodes, not shown here for space reasons). As shown in Fig. 2b, for wireless networks of 40 nodes, the acceptance ratio of all three models are less than 1% apart. As our goal is to apply the placement method to larger networks, our heuristic seems to successfully provide a performance similar to the mathematical model for such networks.

The advantage of our model is that it can provide an acceptance ratio close to the mathematical model in a timely manner. Table 1 shows the average recorded execution time of the three models for different size networks. The recorded time of our heuristic model includes the time it takes to find all shortest paths and the time consumed for providing a placement solution for all requests, while the other models' execution time only measures the time it takes to provide a placement solution for all requests. Putting strict limitations on the search space resulted in the low execution time for our model. It can be seen that, for 20 node networks, the execution time of our model is a few seconds, whereas the execution time of the ILP model is in the order of 100 s. The execution time of the accessible scope heuristic is lower than the ILP model but still much higher than our model's execution time as it is using an ILP for finding an optimal placement for a SG, constrained to a smaller subnetwork. By considering the recorded acceptance ratios and the execution times we can see that in networks of 40 nodes our proposed heuristic can solve the placement problem much faster than the other two (by one or two orders of magnitude) while accepting almost the same number of requests as the optimal model. Table 1 also shows that the proposed heuristic model can be applied to much larger networks of 100 nodes and provides solutions in a timely manner. As we run the experiment for 20,000 s and 4 requests arrive per 100 s, all models processed a total of 800 requests. If it takes on average 143 s for the complete run in wireless network of 100 nodes, we can place a single request in approximately 0.17 s (some of the 143 s is taken up with the pre-processing, determining all shortest paths). Which implies that we can provide a placement solution almost in real-time, something neither of the other two approaches are able to. The mathematical optimization model and the accessible scope heuristic take hours to solve the problem and provide an optimal placement solution.

We recorded the average BW cost for placement of requests. Figure 3 shows the recorded average BW cost of the FACE heuristic model for 20, 30, and 40 nodes networks. The average BW cost increases as the size of networks grows. That is expected as larger networks have a bigger diameter and therefore longer shortest paths between randomly selected source-destination pairs, consuming more BW. Figure 4a and Fig. 4b compare average BW costs recorded for the FACE heuristic model and the ILP model for networks of 20 and 40 nodes respectively. The lower average BW cost achieved by the ILP model is due to the fact that the ILP explicitly considers the impact of interference, and so would choose, among available placements, not only the one that minimizes the BW

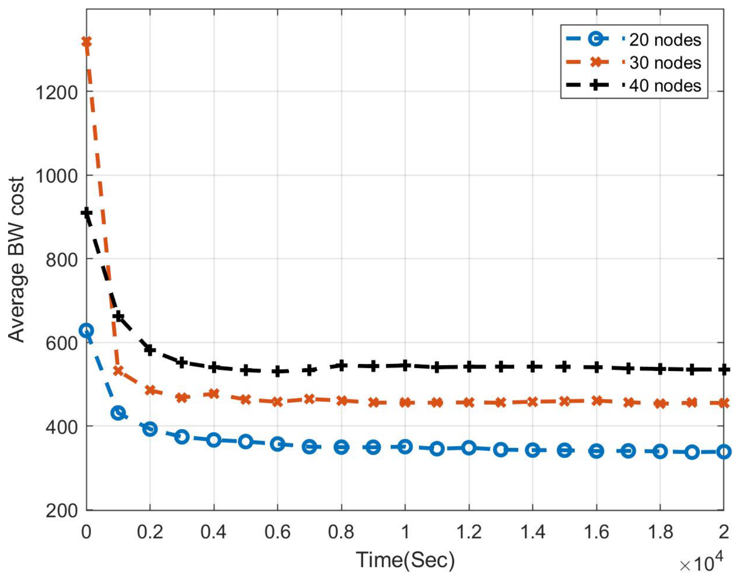
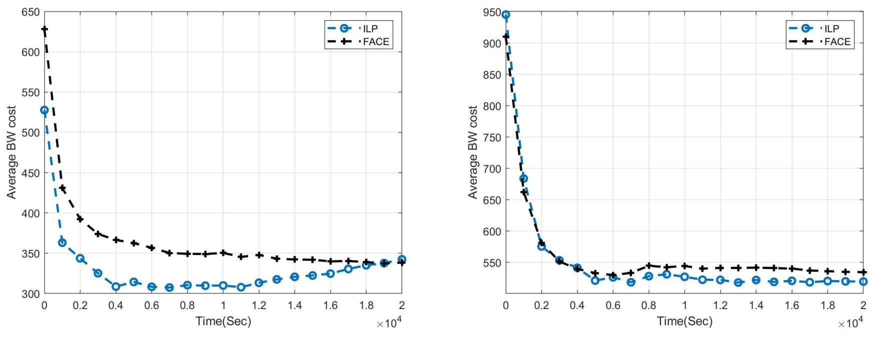


Fig. 3. FACE average BW cost for different size wireless networks



20 nodes networks 40 nodes networks

Fig. 4. Comparing average cost of FACE and ILP, 20 and 40 nodes networks

along that path, but also the one that reduces interference on neighboring links. Our proposed heuristic does not explicitly model this cost, using interference primarily to exclude certain candidate paths. However we can see that the difference between the recorded average cost of both models again seems to be bigger for the smaller network. For the network of 40 nodes, the difference is substantially reduced, again indicating that, for larger networks, the performance of FACE approaches that of the optimal solution.

5 Conclusion

The main challenge in the use of NFVs is optimally mapping the SG requests to the physical network. The placement problem is more challenging in wireless multi-hop networks as we are facing the problem of interference between wireless links, which causes higher BW consumption. As optimal placement methods are NP-hard and can not be applied to large size networks, we provide the FACE heuristic algorithm that can find a placement for SGs with low execution time. The FACE heuristic model minimizes BW consumption and provides solutions in a timely manner.

We compared our results with a similar heuristic and an optimization model, formulated as an ILP. The results show that the proposed heuristic reduces the execution time in comparison to the ILP model dramatically while it does not have a great impact on the number of accepted requests, especially for larger size networks. We can see from the results that although the accessible scope heuristic can provide better acceptance ratio in smaller networks, its execution time is much higher than FACE and it can not be applied to larger size networks.

In the future, we will more thoroughly study the performance benefits of the proposed FACE heuristic. For example, while we consider interference, the evaluations here only consider a fixed network density. We will explore whether the explicit considerations of interference in the placement decisions will be important for varying network densities. We also will vary the request arrival rate and/or the availability of network resources, to explore how the heuristic performs in networks that are more lightly of highly loaded.

In our previous work [6] we already showed that placing requests one at a time is not optimal and we may do better by selecting, among multiple possible choices, a solution that increases the chances of placing future requests. Our goal is to incorporate the factors we identified in [6] in making decision between multiple shortest paths and the process of placing NFs through the chosen shortest path.

Finally, one insight from this work is that a relatively simple heuristic can perform quite competitively, in terms of costs, with more complex heuristics or solving the embedding problem in an optimal manner, while providing answers/solutions to placement requests almost in real-time. We will therefore explore how far we can simplify the heuristic without sacrificing placement performance.

References

1. AMPL: a modeling language for large-scale optimization. OR/MS Today **36**(2), 68 (2009)
2. Bari, F., Chowdhury, S.R., Ahmed, R., Boutaba, R., Duarte, O.C.M.B.: Orchestrating virtualized network functions. IEEE Trans. Netw. Serv. Manage. **13**(4), 725–739 (2016)
3. Ghribi, C., Mechtri, M., Zeghlache, D.: A Dynamic Programming Algorithm for Joint VNF Placement and Chaining, pp. 19–24. ACM (2016)

4. Gupta, P., Kumar, P.R.: The capacity of wireless networks. IEEE Trans. Inf. Theory **46**(2), 388–404 (2000)
5. Jahedi, Z., Kunz, T.: Virtual network function embedding in multi-hop wireless networks. In: Proceedings of the 15th International Joint Conference on e-Business and Telecommunications, ICETE 2018 - vol. 1: DCNET, ICE-B, OPTICS, SIGMAP and WINSYS, Porto, Portugal, July 26–28, 2018. pp. 199–207 (2018). https://doi.org/10.5220/0006887401990207
6. Jahedi, Z., Kunz, T.: Optimal VNF placement: addressing multiple min-cost solutions. In: Obaidat, M.S. (ed.) ICETE 2018. CCIS, vol. 1118, pp. 1–23. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-34866-3_1
7. Kunz, T., Mahmood, K., Li, L.: Broadcasting in multihop wireless networks: the case for multi-source network coding. In: IEEE International Conference on Communications (ICC), pp. 5157–5162. IEEE (2012)
8. Mohammadkhan, A., Ghapani, S., Liu, G., Zhang, W., Ramakrishnan, K.K., Wood, T.: Virtual function placement and traffic steering in flexible and dynamic software defined networks. vol. 2015, pp. 1–6. IEEE (2015)
9. Pham, C., Tran, N.H., Ren, S., Saad, W., Hong, C.S.: Traffic-aware and energy-efficient vnf placement for service chaining: joint sampling and matching approach. IEEE Trans. Serv. Comput. **13**(1), 172–185 (2020)
10. Qi, D., Shen, S., Wang, G.: Towards an efficient vnf placement in network function virtualization. Comput. Commun. **138**, 81–89 (2019)
11. Riggio, R., Bradai, A., Rasheed, T., Schulz-Zander, J., Kuklinski, S., Ahmed, T.: Virtual Network Functions Orchestration in Wireless Networks, pp. 108–116. IFIP (2015)
12. Tajiki, M.M., Salsano, S., Chiaraviglio, L., Shojafar, M., Akbari, B.: Joint energy efficient and qos-aware path allocation and VNF placement for service function chaining. IEEE Trans. Network Serv. Manage. **16**(1), 374–388 (2019)