An Approach to Energy-efficient Virtual Network Embeddings

Andreas Fischer*, Michael Till Beck*†, Hermann De Meer*
*University of Passau, Innstr. 43, 94032 Passau, Germany
†Stadtwerke Passau, Regensburger Str. 29, 94036 Passau, Germany
Email: {andreas.fischer,michael.beck,demeer}@uni-passau.de

***Abstract*—Network Virtualization is recognized as a key technology for the Future Internet. Energy-efficiency is one of the main challenges in future networking environments. Most algorithms for mapping virtual resources to substrate resources however do not consider energy as a factor for the mapping. In order to evaluate the energy-efficiency of such a mapping, an energy model and appropriate energy-aware metrics are needed. This paper discusses how an algorithm can be modified to take energy-efficiency into account. The modified algorithm is then evaluated, showing that energy-efficiency can be increased with only a minor impact on embedding quality regarding other metrics.**

I. Introduction

Network virtualization has been discussed as a solution to the perceived ossification of the current Internet [1], [2]. Several variants of network virtualization have been investigated [3] and it is already widely used in current Future Internet testbeds [4], [5]. It provides an abstraction from substrate resources, creating virtual resources that are expected to be more flexible and easier to manage for users.

One of the main incentives for deploying virtualization technology in the core network is the ability to consolidate resources. Rising energy costs lead to an increased focus on energy-efficiency of ICT equipment. Indeed, energy-efficiency is one of the main challenges in future networking environments. Network virtualization can be used to tackle this problem by sharing hardware, instead of requiring dedicated hardware for each instance. Thus, in order to save energy, unused equipment can be put into an energy-efficient sleep mode, or even turned off completely. To make use of these energy saving measures it is necessary to decide how the virtual network resources should be mapped onto hardware. This is complicated by the fact, that virtual resources often have performance requirements (e.g. a virtual link can demand a certain bandwidth), whereas substrate resources are performance-limited (e.g. a link has a maximum bandwidth).

Another application can be seen in the data centre. Here, virtual machines have to be mapped in an energy aware way while also considering bandwidth demands of communication connections between virtual machines. This has to take into account the tradeoff between Quality of Service and current energy prices (see for example the "Green SLA"/"Green SDA" concept of the All4Green project: www.all4green-project.eu).

Finding an optimal mapping of virtual resources onto substrate resources under a number of constraints is known

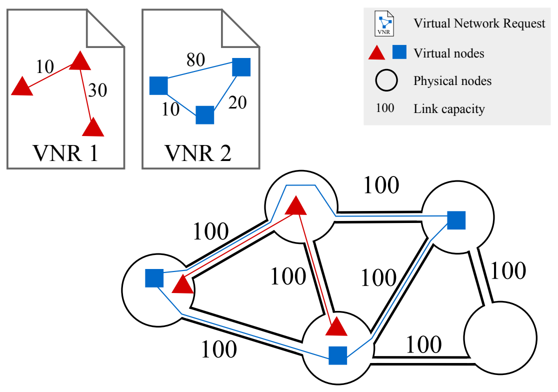


Fig. 1. The Virtual Network Embedding problem

as the Virtual Network Embedding (VNE) problem. Several algorithms to solve this problem have been discussed in literature, so far. Most algorithms for mapping virtual resources to substrate resources however do not consider energy as a factor for the mapping.

The modification of existing algorithms towards energy-efficiency, while still keeping performance at an acceptable level, is non-trivial. In this paper, the extension of VNE algorithms with energy-efficiency constraints is discussed. One well-known, exemplary VNE algorithm is extended, comparing embedding performance with the unmodified version and giving estimates regarding the possible energy-savings.

The rest of this paper is organized as follows: In Section II the VNE problem is presented in detail and energy-efficiency concepts in a virtual network environment are introduced. In Section III related work is discussed. Section IV presents an approach to modify an existing VNE algorithm to take energy-efficiency into account. The resulting algorithm variants are evaluated in Section V. Finally, Section VI concludes the paper and gives hints for future work in this area.

II. Background

A. The Virtual Network Embedding Problem

In order to realize a virtual network on top of a substrate network, each virtual node has to be mapped to a substrate node. Moreover, each virtual link has to be mapped to a substrate path (that is, a combination of links and nodes) in such a way that the desired virtual connections are appropriately represented. The problem is complicated by the fact, that each Virtual Network Request (VNR) is accompanied by a certain resource demand, but substrate resources are limited.

Figure 1 shows an example with two VNRs to be deployed over one substrate network. It can be seen that several virtual nodes have to be mapped to the same substrate node. Moreover, the topology of the virtual networks is different to the topology of the substrate network. In order to demonstrate resource demands, virtual links have been annotated with a bandwidth demand. This has to be matched by the substrate resources used to realize the virtual link. One of the nodes in the substrate network is idle and could be switched off.

Matching the provided resources to the requested resource demands in an optimal way is typically non-trivial. Even if one considers only the most basic resources, finding an optimal solution gets NP hard. The optimal matching of CPU capacity and demands is, for example, directly related to the well-known Bin-Packing problem, with CPU capacity representing bins and CPU demands representing items. As a result, most proposed VNE algorithms in the literature use heuristics to find a solution.

B. Energy-efficiency of Virtual Network Embeddings

Network virtualization can increase energy-efficiency of the substrate network in two different ways. On the one hand, virtualization enables consolidation. That is, several virtual resources can be hosted on the same substrate resource. On the other hand, virtual resources can be migrated to balance the overall load in an energy-efficient way, i.e., to reduce the total power consumption of the network. To improve VNE algorithms towards this kind of efficiency, it is first necessary to take a closer look at those components of the substrate network that will consume energy. The major part of the spent energy will likely be consumed by the substrate nodes (i.e. the routers of a network), as opposed to the links. Thus, efforts to significantly reduce the power consumption of a network should concentrate on minimizing the power consumption of the nodes of a network. As a first step towards a lower energy footprint of the network, one should therefore switch off as many substrate nodes as possible. Mapping one or more virtual nodes onto a substrate node requires the substrate node to be switched on. To save energy, therefore, as many nodes as possible should be mapped on the same set of active substrate nodes. However, substrate nodes may be required to be powered on, even if they don't host a virtual node. This is due to the fact that nodes that are connected inside the virtual network have to be connected inside the substrate network, too. Taking this into account, some substrate nodes may be required to serve as purely forwarding entities. These nodes are known as hidden hops [6]. Avoiding purely forwarding nodes is generally a good idea with respect to energy efficiency.

Moreover, substrate nodes can be further differentiated by the actual amount of power they consume. Some nodes may be more efficient in their use of energy. By preferring those energy-efficient nodes for the mapping, the overall consumption of the network can be further reduced.

III. Related Work

Work related to the approach presented here can be distinguished into two different areas: VNE algorithms and approaches to energy-efficient resource mappings. Multiple VNE algorithms have been discussed in the literature already [7]. Basic VNE algorithms tackle only the resource allocation problem. Algorithm performance is evaluated according to a number of different parameters, e.g. node and link stress [8], cost and revenue [9], [10], acceptance ratio [11], [12], or algorithm runtime [13]. Lischka and Karl [14] present a VNE algorithm that makes use of subgraph isomorphism detection in order to optimize the embedding. The presented algorithm is able to operate in an online manner, embedding Virtual Networks on demand. The algorithm is evaluated with regard to runtime and the ratio between revenue and cost (where "revenue" is the total amount of virtual resources mapped and "cost" is the total amount of substrate resources spent).

There has also been some work on extending the basic VNE problem into new areas. Some of those approaches are taking a different perspective on the VNE problem, e.g., discussing multipath routing [10] or embedding across different networks [15]. In other scenarios, new requirements are investigated, like distributed embedding [16] or resilience [17].

There are several papers that aim at energy-efficient mapping of resources. They perform energy aware resource management by using virtualization and consolidation. Work discussed in the literature ranges from consolidation of CPU workload [18] over data center consolidation [19], [20], up to consolidation in full-scale cloud environments [21]. All of these works focus either on the energy-efficiency of isolated hardware (as opposed to entire networks) or on the performance of VNE results in terms of cost, runtime or acceptance ratio. Only recently the topic of energy-efficiency has raised interest for this particular problem. In [22], Botero et al. provide a Mixed Integer Program formulation of the problem, showing energy savings compared to a cost-oriented approach. It focuses only on optimizing energy consumption.

This paper discusses how an existing VNE algorithm can be modified to take energy efficiency into account, while still achieving good results with regard to the original optimization criteria.

IV. Energy-efficient Embedding Algorithms

This section demonstrates how an existing VNE algorithm can be modified in order to take energy-efficiency as an additional optimization goal. First, an energy model is introduced. Next, the algorithm by Lischka and Karl is presented as an example. Finally, energy-efficient modifications to that algorithm are explained.

A. Modeling energy consumption of the substrate network

Energy consumption of a substrate network can be modeled in different ways. Here, the focus is on substrate nodes, as those are expected to be the major drivers in energy consumption. It is assumed that substrate nodes will consume energy only when they are active. I.e., a substrate node consumes

TABLE I
INPUT PARAMETERS OF THE ADVANCED SUBGRAPH ALGORITHM

Input Parameter	Description
G^V_{sub}	The parts of the virtual network that have already been mapped.
$M(G^V_{sub})$	The current mapping of the nodes and links.
G^V	The virtual network that should be mapped.
G^P	The residual graph of the substrate network.
G^P_{orig}	The graph of the substrate network.
E	Maximum value of ϵ.

energy only if either a virtual node is mapped to it or it is part of a path for a virtual link. The model used in this paper takes into account that energy consumption of nodes can vary:

$$C^{\text{het}}_{\text{energy}}(t) = \sum_{\substack{v \in V \\ \text{active}(t,v)=1}} \vartheta_v \qquad (1)$$

Here, energy consumption $C^{\text{het}}_{\text{energy}}(t)$ is approximated as the sum of the individual consumption values ϑ_{sn} over all active nodes v (*active*(t, v) returns 1 in case the node v is active at time t, 0 otherwise). This model is still feasible to calculate. It is also realistic to a certain extend, since it takes the differences in energy consumption of various nodes into account.

B. Advanced subgraph algorithm

The VNE algorithm presented by Lischka and Karl in [14] aims to find areas inside the substrate network that are similar to the topology of the virtual network. Table I describes the input parameters of the algorithm. G^P_{orig} is the graph of the substrate network with all nodes and links. The virtual network is represented by the G^V graph. G^V_{sub} contains the nodes and links of the virtual network G^V that already have been mapped successfully. In contrast, G^P is the residual graph of the substrate network. It contains the substrate nodes that have not yet been mapped. $M(G^V_{sub})$ is the current mapping of the substrate and virtual nodes and links.

In order to embed a virtual network, the algorithm chooses a random substrate node and tries to map a virtual node onto it. This virtual node is selected by the following procedure: One of the immediate neighbours of the virtual nodes that were embedded previously is chosen, if it has not been mapped yet (the first node is selected randomly). Fig. 2 describes the structure of the genneigh procedure that computes the list of possible mapping candidate nodes. It uses the subgraph $F_{G^V_{sub}}(G^V)$ of G^V (i.e. the set of nodes connected to G^V by a direct link) to select appropriate nodes for the next mapping.

Subsequently to each node mapping step, the corresponding links of each node are being processed: The algorithm tries to connect the virtual nodes that have already been mapped by reserving the communication paths between the corresponding substrate nodes. The length of each path must not exceed the current value of ϵ. If no further mapping is possible, previous decisions are reverted and alternative options are examined. Depending on ϵ, it is more or less likely that the algorithm

```
1: procedure GENNEIGH($G^P, G^V, G^V_{sub}, M(G^V_{sub})$)
2:     if $F_{G^V_{sub}}(G^V) = \emptyset$ then
3:         $c \leftarrow \text{nodes}(G^V) \times \text{nodes}(G^P)$
4:     else
5:         $c \leftarrow F_{G^V_{sub}}(G^V) \times (\text{nodes}(G^P) \setminus \text{nodes}(M(G^V_{sub})))$
6:     end if
7:     optimize($c$)
8:     sort $c$ by virtual node demand
9:     return $c$
10: end procedure
```

Fig. 2. The genneigh procedure

TABLE II
IMPROVEMENTS OF THE SUBGRAPH ALGORITHM

Algorithm	Description
AdvSubgraph	Unmodified algorithm
AdvSubgraph-MM	Allows cohosting of virtual nodes of the same network
AdvSubgraph-MM-EE	Prefers active and energy efficient substrate nodes for node mapping
AdvSubgraph-MM-EE-Link	Prefers active and energy efficient substrate nodes for node and link mapping.

can find a result. On the one hand, when choosing big ϵ values, the path length tends to be longer, too. Therefore, the advanced subgraph algorithm successively increments ϵ by one each time the embedding fails. This is done until ϵ gets bigger than a predefined upper limit E. Here, this approach is used as an example to show how a VNE algorithm can be extended to support energy efficiency goals.

C. Modifications of the algorithm

Table II outlines the different modifications of the algorithm. The *AdvSubgraph* variant is the advanced subgraph algorithm which increases the maximum allowed path length in each iteration, until a valid mapping has been found or the limit has been reached. Additionally, the *AdvSubgraph-MM* has a redefined node candidate selection approach. It allows to map several virtual nodes of the same network to the same substrate node. This means that several nodes can be consolidated onto one node. In that case, connecting links between those virtual nodes will not be mapped, because we assume that virtual machines on the same host can communicate directly without further expense. The *AdvSubgraph-MM-EE* variant prefers active nodes and nodes that consume (relatively to the maximum power consumption of all substrate nodes), less power, but only for node mapping. In addition to all other improvements, selection of energy efficient paths between the communicating substrate nodes is also done by the *AdvSubgraph-MM-EE-Link* algorithm.

The details of the modifications are as follows: After mapping a substrate node, the original algorithm removes that node from the list of nodes that can be assigned to further virtual nodes of the same virtual network. This means that a substrate node runs no more than one virtual node of the same virtual network. It is debateable whether this is a necessary restriction for the VNE problem. While forcing virtual nodes onto different substrate nodes may be advantageous for network resilience, situations are possible where such a constraint is not necessary. Since it is suboptimal for energy efficiency, the restriction is removed by changing line 5 of Fig. 2 as follows:

$c = F_{G^V_{sub}} \times \text{nodes}(G^P)$

Furthermore, the list of mapping candidates is sorted by preferring active nodes. Moreover, some nodes might be more energy efficient than others. This is taken into account, and the algorithm should prefer those nodes. So, the mapping candidates get pre-sorted first by preferring substrate nodes that are already active, and subsequently by preferring substrate nodes with lower power consumption. This requires changing the genneigh procedure. Line 8 is modified, so that the candidate list c is sorted first by active substrate nodes, then by energy efficient substrate nodes, then by virtual node demand.

Both modifications only affect the selection of nodes that should be mapped to virtual nodes. The sort function does not have any impact to the nodes that are used as part of a communication connection (i.e. hidden hops). Thus, further modifications of the algorithm are necessary to revise the link mapping procedure. The original approach uses Dijkstra's path algorithm to find paths between substrate nodes. It is assumed that every link l increments the path weight by one (so the total path weight of a path P is $\sum_{l \in P} 1$). However, obviously, this does not differentiate between energy efficient routes and non-energy efficient ones. Therefore, this weighting approach can be improved by also taking into account the properties of the involved substrate node sn the (directed) edge points to. This implies that energy efficient routes are preferred. Therefore, we recalculate the weight of the edges of the substrate network's graph by including the parameters (activity status and power consumption):

$$1 + \alpha \cdot (1 - \text{active}(t, sn)) + \beta \cdot \frac{\vartheta_{sn}}{\max_{u \in V} \vartheta_u}$$

Parameter α can be used to adjust to which extend the activity status of a link should be taken into account. The same applies for parameter β with respect to power consumption. By dividing ϑ_{sn} by $\max_{u \in V} \vartheta_u$, the absolute power consumption value ϑsn of a node is relativized with respect to nodes' maximum power consumption. Since Dijkstra's algorithm relies on positive edge weights, the following conditions should apply for weights α and β: $\alpha \in \mathbb{R}, \alpha \geq 0, \beta \in \mathbb{R}, \beta \geq 0$. Links pointing to nodes that are currently inactive are fined by incrementing the link weight by α. Independently of this, the weight also depends on the power consumption of this node: The higher the (relative) power consumption, the higher the penalty that is added. The total weight for a path p is $\sum_{l \in P} 1 + \alpha \cdot (1 - \text{active}(t, sn)) + \beta \cdot \frac{\vartheta_{sn}}{\max_{u \in V} \vartheta_u}$.

V. Evaluation

This section presents first the evaluation methodology, following up with the evaluation results.

A. Methodology

In order to evaluate VNE algorithms, one has to perform simulation and measure a variety of metrics. The ALEVIN VNE simulator [23] was used and extended to that end. Indeed, this simulator has already been used in previous VNE comparisons [24], [25]. It allows the user to compare different VNE approaches and provides a set of several metrics that can be used for this comparison. Thus, algorithms can be ranked against each other and the best one can be chosen regarding a given problem definition. The metrics used for this paper are:

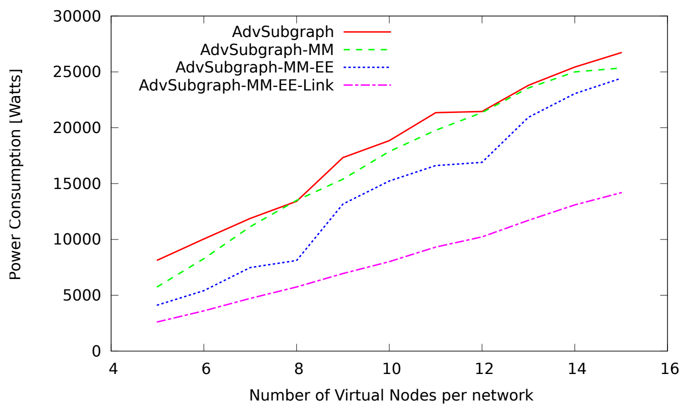
1) Number of active nodes: A substrate node is considered being active if a virtual node is mapped onto that node or the substrate node is part of a communication path.
2) Hidden hops: Counts the number of hidden hops.
3) Power consumption: Sum of the power consumption of all active substrate nodes.
4) Embedding cost: Sum of all resources that were used by a mapping (CPU and Bandwidth).
5) Average path length: Divides the sum of the length of all paths by the number of virtual links.
6) Execution time: Time spent calculating the embedding.

B. Results

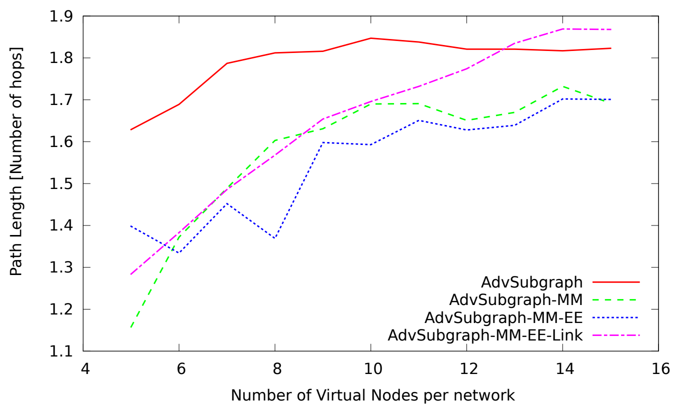
The four algorithm variants discussed before will be compared here to showcase the effects of each modification. Note, that in these experiments communication cost for co-hosted virtual nodes (i.e., two virtual nodes hosted on the same substrate node) is expected to be neglectable. Since in this case communication can be performed in-memory, both bandwidth and delay are orders of magnitude smaller, compared to a real network connection.

In order to ensure relevance and stability of results, a number of parameters have been varied. Experiments have been run five times with random parameters. All networks have been generated randomly each time by a Waxman Generator [26]. Link density and long link probability parameters have been set to 0.5. The parameters used during experimentation are as follows. Substrate networks have been chosen with a size of 100 nodes. Each substrate node provided a CPU capacity between 1 and 100 (chosen randomly). Likewise, each substrate link provided a bandwidth capacity between 1 and 100. The power consumption of a substrate node has been set to a value between 100 and 500 Watts (chosen randomly). Onto this substrate network, 5 virtual networks have been embedded. The number of nodes in each virtual network was varied between 5 and 15. Each virtual node poses a CPU demand between 1 and 50 (chosen randomly). Likewise, each virtual link requests bandwidth resources between 1 and 50 (chosen randomly).

In Section IV α and β have been presented as weights that influences energy efficiency of the mapping results. Figure 3 indicates the power consumption of the VNE algorithm with all modifications applied. For this figure a scenario with 15 virtual networks was used. One can see that, after an initial drop, the power consumption does not vary much with higher



(a) Power consumption



(b) Average path length

Fig. 4. Experimental results: Power consumption and average path length

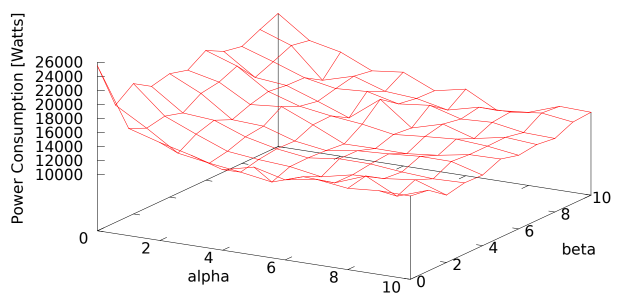


Fig. 3. Power consumption of AdvSubgraph-MM-EE-Links

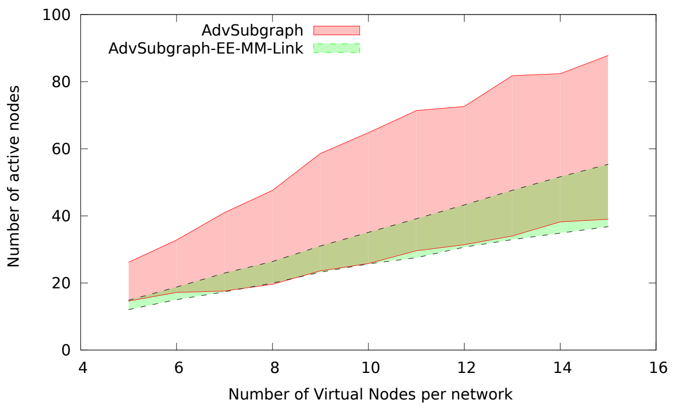


Fig. 5. Experimentation results: Active nodes and Hidden Hops

α and β values. In order to retain readability, the following figures have, been produced with a fixed value $\alpha = \beta = 5$.

Figure 4a shows the power consumption of all investigated algorithms, in dependence of the number of virtual nodes per network. While the simple extension of allowing multiple virtual nodes on one substrate node has only a minor influence, the weighting of the energy-efficiency of a node and the energy-aware link mapping lead to a significantly lower energy footprint.

Interestingly, these gains in energy-efficiency do not result in a significantly higher cost. All algorithms consume about the same amount of resources for the embedding of the virtual networks (apart from some slight variation). This is also verified by the fact that the average path length does not increase significantly. Taking into account that – with the multiple mappings modification – some links are not mapped to a substrate path at all, but are rather assumed to be realized in-memory, the average path length of our energy-efficiency modifications actually decreases in some cases, compared to the unmodified algorithm. This is depicted in Fig. 4b. Only the fully modified algorithm has been observed to result in higher path length, and only when a sufficiently high number of virtual nodes has to be embedded. Maximum path length has also been measured, but is omitted for brevity here, since it never exceeds a path length of three hops.

An explanation for the significant savings in energy, which the modifications produce, can be seen in Fig. 5. This figure compares the unmodified algorithm to the full set of the modifications with regard to active substrate nodes. For each algorithm, two lines are drawn. The top line shows the number of substrate nodes that have to be powered on (either because they host a virtual node or because they are part of the path realizing a specific virtual link). The bottom line shows the number of substrate nodes that host at least one virtual node of any network. The colored area between the two lines then indicates the number of substrate nodes that are only powered on in order to support a path for a virtual link. Clearly, there is a high potential for energy savings regarding these nodes. Indeed, one can see in the figure that, while the number of substrate nodes hosting a virtual node is about the same for both variants, the fully modified algorithm can significantly reduce the number of those hidden hops.

Finally, the execution time of each algorithm has been measured. For the investigated problem size the results are entirely within reasonable bounds, with the maximum being at a time of about 4.5 seconds for the fully energy-efficient embedding of 15 virtual networks.

VI. Conclusions and Future Work

Energy-efficiency and green networking will be a major driver for Future Internet research. As such, it is important to take the energy-efficiency of VNE algorithms into account. This paper presented an energy model and energy-aware metrics that allow to compare VNE algorithms with regard to their energy-efficiency. Moreover, it was demonstrated that the modification of one algorithm could lead to a significantly lower energy consumption with only minor impact on other performance metrics.

The authors are confident that there is still a great potential to further enhance the energy-efficiency of VNE algorithms. Future work includes, for example, extending our current energy-model in order to take the dependency between the load of a resource and its energy-consumption into account. Moreover, the evaluation can be expanded to include not only randomly generated, but also more realistic network topologies. Finally, scalability should be further evaluated, testing the algorithm in an overload situation, where the amount of virtual resources requested surpasses the amount of available substrate resources.

Acknowledgment

The authors would like to thank Xavier Hesselbach, Juan Felipe Botero, Michael Duelli, and Daniel Schlosser for their insightful discussions and close cooperation during the EuroNF VNREAL project.

This work has received support by the EU projects All4Green (FP7 STREP, grant no. 288674) and EINS (FP7 NoE, grant no. 288021).

References

[1] T. Anderson, L. Peterson, S. Shenker, and J. Turner, “Overcoming the internet impasse through virtualization,” Computer, vol. 38, pp. 34–41, April 2005.

[2] N. Feamster, L. Gao, and J. Rexford, “How to lease the internet in your spare time,” ACM SIGCOMM Computer Communication Review, vol. 37, pp. 61–64, 2007.

[3] N. M. K. Chowdhury and R. Boutaba, “A survey of network virtualization,” Computer Networks, vol. 54, no. 5, pp. 862 – 876, 2010.

[4] A. Bavier, N. Feamster, M. Huang, L. Peterson, and J. Rexford, “In vini veritas: realistic and controlled network experimentation,” SIGCOMM Comput. Commun. Rev., vol. 36, pp. 3–14, August 2006.

[5] D. Schwerdel, D. Günther, R. Henjes, B. Reuther, and P. Müller, “German-lab experimental facility,” in Future Internet - FIS 2010, ser. Lecture Notes in Computer Science, A. Berre, A. Gomez-Perez, K. Tutschku, and D. Fensel, Eds., vol. 6369. Springer Berlin / Heidelberg, 2010, pp. 1–10, 10.1007/978-3-642-15877-3_1.

[6] J. F. Botero, X. Hesselbach, A. Fischer, and H. De Meer, “Optimal mapping of virtual networks with hidden hops,” Telecommunications Systems: Special Issue ”Future Internet Services and Architectures: Trends and Visions”, vol. 52, no. 3, pp. 1–10, March 2013.

[7] A. Fischer, J. F. Botero, M. T. Beck, H. De Meer, and X. Hesselbach, “Virtual network embedding: A survey,” IEEE Communications Surveys and Tutorials, vol. PP, no. 99, pp. 1–19, 2013.

[8] Y. Zhu and M. Ammar, “Algorithms for assigning substrate network resources to virtual network components,” in Proc. IEEE INFOCOM, Apr. 2006, pp. 2812–2823.

[9] J. Lu and J. Turner, “Efficient mapping of virtual networks onto a shared substrate,” Washington University, Tech. Rep., 2006, available: http://cse.seas.wustl.edu/Research/FileDownload.asp?503.

[10] M. Yu, Y. Yi, J. Rexford, and M. Chiang, “Rethinking virtual network embedding: Substrate support for path splitting and migration,” ACM SIGCOMM CCR, vol. 38, no. 2, pp. 17–29, Apr. 2008.

[11] N. F. Butt, N. M. Chowdhury, and R. Boutaba, “Topology-awareness and reoptimization mechanism for virtual network embedding,” in Networking, 2010, pp. 27–39.

[12] I. Fajjari, N. Aitsaadi, G. Pujolle, and H. Zimmermann, “VNE-AC: Virtual Network Embedding Algorithm Based on Ant Colony Metaheuristic,” in 2011 IEEE International Conference on Communications (ICC), june 2011, pp. 1 –6.

[13] X. Cheng, S. Su, Z. Zhang, H. Wang, F. Yang, Y. Luo, and J. Wang, “Virtual network embedding through topology-aware node ranking,” SIGCOMM Comput. Commun. Rev., vol. 41, pp. 38–47, April 2011.

[14] J. Lischka and H. Karl, “A virtual network mapping algorithm based on subgraph isomorphism detection,” in VISA ’09: Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures. New York, NY, USA: ACM, 2009, pp. 81–88.

[15] I. Houidi, W. Louati, W. Ben Ameur, and D. Zeghlache, “Virtual network provisioning across multiple substrate networks,” Comput. Netw., vol. 55, pp. 1011–1023, March 2011.

[16] I. Houidi, W. Louati, and D. Zeghlache, “A distributed and autonomic virtual network mapping framework,” in Proceedings of the Fourth International Conference on Autonomic and Autonomous Systems. Washington, DC, USA: IEEE Computer Society, 2008, pp. 241–247.

[17] I. Houidi, W. Louati, D. Zeghlache, P. Papadimitriou, and L. Mathy, “Adaptive virtual network provisioning,” in Proceedings of the second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures, ser. VISA ’10. New York, NY, USA: ACM, 2010, pp. 41–48.

[18] C. Subramanian, A. Vasan, and A. Sivasubramaniam, “Reducing data center power with server consolidation: Approximation and evaluation,” in High Performance Computing (HiPC), 2010 International Conference on, dec. 2010, pp. 1–10.

[19] C. Rusu, A. Ferreira, C. Scordino, and A. Watson, “Energy-efficient real-time heterogeneous server clusters,” Real-Time and Embedded Technology and Applications Symposium, IEEE, vol. 0, pp. 418–428, 2006.

[20] D. Kusic, J. Kephart, J. Hanson, N. Kandasamy, and G. Jiang, “Power and performance management of virtualized computing environments via lookahead control,” Cluster Computing, vol. 12, pp. 1–15, 2009, 10.1007/s10586-008-0070-y.

[21] S. Srikantaiah, A. Kansal, and F. Zhao, “Energy aware consolidation for cloud computing,” in Proceedings of the 2008 conference on Power aware computing and systems, ser. HotPower’08. Berkeley, CA, USA: USENIX Association, 2008, pp. 10–10.

[22] J. Botero, X. Hesselbach, M. Duelli, D. Schlosser, A. Fischer, and H. de Meer, “Energy efficient virtual network embedding,” Communications Letters, IEEE, vol. 16, no. 5, pp. 756 –759, may 2012.

[23] A. Fischer, J. F. Botero, M. Duelli, D. Schlosser, X. Hesselbach, and H. De Meer, “ALEVIN - A Framework to Develop, Compare, and Analyze Virtual Network Embedding Algorithms,” Electronic Communications of the EASST, vol. 37, pp. 1–12, 2011.

[24] J. F. Botero, X. Hesselbach, M. Duelli, D. Schlosser, A. Fischer, and H. De Meer, “Flexible VNE Algorithms Analysis using ALEVIN,” in Proc. of the 11th Würzburg Workshop on IP: Joint ITG, ITC, and Euro-NF Workshop ”Visions of Future Generation Networks” (EuroView2011). ACM, 2011.

[25] M. Duelli, D. Schlosser, J. F. Botero, X. Hesselbach, A. Fischer, and H. De Meer, “VNREAL: Virtual Network Resource Embedding ALgorithms in the Framework ALEVIN,” in Proc. of the 7th Euro-NF Conf. on Next-Generation Internet (NGI 2011). IEEE, 2011, pp. 1–2.

[26] H. Tangmunarunkit, R. Govindan, S. Jamin, S. Shenker, and W. Willinger, “Network topology generators: degree-based vs. structural,” SIGCOMM Comput. Commun. Rev., vol. 32, pp. 147–159, August 2002.