KuVS Prize for the Best Master Thesis 2014: 'Optimal Virtualized In-Network Processing with Applications to Aggregation and Multicast'

Matthias Rost
TU Berlin
mrost@inet.tu-berlin.de

Stefan Schmid
TUBerlin and T-Labs
stefan@inet.tu-berlin.de

Andreas Bley
Universität Kassel
andreas.bley@mathematik.uni-kassel.de

Anja Feldmann
TU Berlin
anja@inet.tu-berlin.de

***Abstract*—Computer networks become more and more virtualized and software-defined, introducing new opportunities for in-network (a.k.a. in situ) processing and flexible traffic engineering. In order to make efficient use of in-network *node resources* (such as computation or storage) and bandwidth resources, the two resource types must be *jointly optimized*: traffic-steering mechanisms are needed to reroute traffic through the in-network node resources, potentially introducing longer paths. Matthias Rost's thesis takes a first look at this important tradeoff, which is not only relevant in the context of SDN/NFV, but can also be seen as an interesting generalization of classic multicast and aggregation problems, with applications in optical networks, sensor networks, or even network analytics. The main contribution of the thesis is VirtuCast, an exact single-commodity algorithm which is based on an interesting path decomposition algorithm. The thesis also stands out by its broad treatment of the topic as well as the in-depth analytical study and rigorous evaluation.**

I. Introduction

Virtualization has become a crucial ingredient for efficiently utilizing networks. Especially the advent of Software-Defined Networking (SDN) and Network Functions Virtualization (NFV) has recently opened up a multitude of new research questions considering the *joint* placement and stitching of (virtual) network functions. The research community has only begun in the last decade to consider the routing and placement problems *jointly*. One of the most studied and most general problem formulations is the one of the Virtual Network Embedding Problem (VNEP) [4]: given a virtual graph, describing computational requirements at the nodes and communication requirements on the edges, the virtual nodes and edges must be *embedded* on the physical host graph. Despite its generality, the VNEP fails to capture scenarios, where the *optimal* virtual network is not known in advance.

In his master thesis, Rost [18] considers a class of *service deployment* problems utilizing *in-network processing* to e.g. filter, aggregate or duplicate data. Concretely, the two communication schemes of aggregation and multicast are considered. Intuitively, in the aggregation scheme a set of nodes need to forward data to a single data sink, while processing nodes may be placed in the network to aggregate multiple data flows. In the multicast communication scheme, the task is to connect a single data source to a set of receivers, where processing nodes may be used to split and redirect the data flow towards multiple destinations. As installing processing functionality either induces monetary costs (e.g. in ISP networks for actually placing a server) or at opportunistic costs (e.g. in a data center, as computational ressources or energy is consumed), Rost considers the problem of *jointly* minimizing the number of processing locations and the bandwidth. Solving this problem is challenging, as (1) the number and locations of processing nodes is not known in advance, (2) the sub objectives of using as few bandwidth *while* using as few processing locations as possible dissent and (3) capacities both on processing nodes and the bandwidth on edges are considered.

Rost coins the above described problem the *Constrained Virtual Steiner Arborescence Problem* (CVSAP). While CVSAP is closely related to well-known combinatorial problems as minimum-cost flows and Steiner Trees, the respective problem has not been considered before. This is especially interesting, as the service deployment of aggregation or multicast communication schemes is reoccuring throughout various networking disciplines and CVSAP can be applied in as many applications as Big Data aggregation, network analytics, sensor networks, optical multicast and the geo-replication of services e.g. via caches. While these applications are quite specific with respect to the networking fabric and the type of processing applied, the combination of Software-Defined Networking and Network Functions Virtualization may allow to broaden the applications of CVSAP even more.

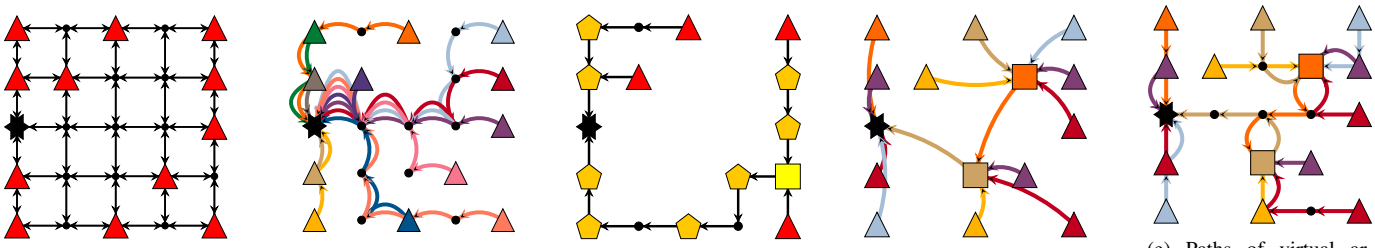
A. Overview of Contributions

The thesis stands out by providing multiple important contributions and its analytical rigour. In particular, the following shortly summarizes the four different parts of Rost's thesis [18][1].

1) Part I: The computational complexity of CVSAP and weaker variants is studied and the inapproximability of CVSAP (unless $P = NP$ holds) is shown. Furthermore three approximation algorithms for the weaker variants are obtained.

2) Part II: Based on the computational hardness of CVSAP, exact algorithms based on Mixed-Integer Programming are studied, yielding a multi-commodity and a single-commodity flow Integer Programming formulation. Together with a *novel*

[1]It must be noted that during the writing of the thesis, parts of it have already been published as a technical report and a conference paper [19], [20].



(a) Aggregation scenario (b) Unicast solution (c) Steiner arborescence (d) Virtual arborescence (e) Paths of virtual arborescence

Fig. 1: An aggregation example on a 5×5 grid. Senders are depicted as triangles while the receiver is depicted as star. Processing locations are pictured as squares or, in case that a processing location is collocated with a sender, pentagons.

flow decomposition scheme the single-flow formulation builds the foundation of the VirtuCast algorithm to solve CVSAP.

3) Part III: While the inapproximability result for CVSAP does not allow for deriving approximation algorithms of CVSAP, several linear and a purely combinatorial heuristic are developed.

4) Part IV: Lastly, in Part IV of his work, all presented algorithms for CVSAP are studied in an extensive computational evaluation. The results show that the VirtuCast algorithm - paired with the developed heuristics - can be applied to solve realistically sized instances with more than thousand nodes to within 6% of optimality.

B. Overview

In Section II first the problem investigated is illustrated using an aggregation example. The next section then underlies the importance of CVSAP by considering potential applications (see Section III). Section IV then summarizes the main contributions of the thesis[2].

II. Model and Definition of CVSAP

To illustrate the underlying communication model of the Constrained Virtual Steiner Arborescence Problem (CVSAP), consider the aggregation example depicted in Figure 1d. Given is a bi-directed 5×5 grids, where triangles represent senders and the star represents the single receiver. The black dots represent further nodes, which are neither receiver nor sender but on which processing functionality may be placed to merge multiple incoming data streams. If no such functionality is placed, then the node must forward all incoming data streams. The task is to connect all senders to the single receiver.

Figure 1b presents a simple unicast solution, where each sender forwards its information to the receiver directly, such that no processing functionality needs to be installed at all. Such a solution can be computed using known shortest paths algorithms, or minimum cost flow algorithms if capacities are given. Note that the solution uses 41 edges overall.

In contrast, when processing functionality can be placed at all nodes and the task is to minimize the bandwidth usage, an optimal solution can be computed using Steiner arborescence algorithms (see Figure 1c). Note that this solution uses 16 edges and 9 processing nodes.

Importantly, the question of *how to trade off bandwidth usage and the cost of installing processing functionality* was never studied before. Thus, when searching for optimal solutions to either the multicasting or aggregation problem, one could only activate in-network processing at all nodes or deactivate its usage globally. The main contribution therefore lies in answering the following questions:

1) Which structure do solutions have where in-network processing is only enabled at a subset of nodes?
2) How can such solutions be computed to optimize the overall cost, consisting of costs for bandwidth usage and installing processing functionality?

To outline the structure of solutions that is searched for, consider the solution presented in Figure 1d. In this solution the senders either connect to the receiver directly or to one of the two processing nodes. The upper processing node then forwards its aggregated result towards the lower processing node, which in turn sends its aggregated result towards the receiver. We denote the solution depicted in Figure 1d as virtual arborescence, as edges in this tree correspond to paths in the underlying network (see Figure 1e). Note that this solution only uses 2 processing nodes instead of 9 with respect to the Steiner arborescence solution and reduces the number of edges with respect to the unicast solution from 41 to 26. Furthermore, note that while the unicast solution was a directed acyclic graph and the Steiner arborescence solution was an arborescence, the paths of the virtual arborescence solution actually contain *cycles* (cf. also Figure 3b).

Given the notion of virtual arborescences, capacity constraints on *processing nodes and edges* can be easily modeled. The capacity of processing nodes will effectively limit the degree of activated processing nodes in the virtual arborescence and the links' capacities are enforced by summing up the number underlying paths using an edge (cf. Figure 1e). Furthermore, edges as well as processing locations can be attributed with usage and opening costs, such that the CVSAP asks for minimizing the joint costs. We refer the interested reader to the thesis [18], which contains a formal graph-theoretic definition.

[2]The thesis [18] is available online at www.matthias-rost.de.

III. Applications of CVSAP

In this section the definition of CVSAP is motivated by applications which are sensitive to the tradeoff between bandwidth usage and costs for installing processing functionality. We first discuss in-depth the problem of network analytics and then give a short outline over other related work.

A. CVSAP for Network Analytics

Handling monitoring traffic within ISPs is no trivial task. Already in 2003, the authors of [7] presented the Gigascope stream database that was deployed throughout the AT&T network to allow for stream based network analysis. Concretely, the task has been to trace errorneous configurations and unexpected events. Using Gigascope, the user can define (sources of) information and relation between them in an SQL-like fashion to continuously query the global network state.

With the advent of NFV and SDN, a similar aggregation scheme can also be used to merge data streams and discard information of no interest near the sources of information. This may reduce the traffic significantly. As all computations need to be executed in real-time, employing in-network processing at all nodes may not only induce unwanted delays but may itself cause a significant synchronization overhead. Thus, limiting the number of processing locations is at least as important as trying to minimize the bandwidth footprint. Furthermore, depending on the complexity of performed computations, the number of flows that can be processed at any location should be constrained. Lastly, to guarantee QoS for the monitoring traffic, the available bandwidth on each link may not exceeded by sending monitoring messages as this would again incurr latencies. CVSAP allows for the accurate modeling all of these properties, assuming that the functions are all commutative and associative, which in general does not need to be the case.

However, the multicast variant of CVSAP can also be employed to organize network analytics more efficiently by efficiently distributing monitoring data to *multiple data sinks*. Given that in ISP networks a variety of Key Performance Indicators (KPIs) need to be constantly monitored to ensure meeting Service Level Agreements (SLAs), it is easy to see that the same information may be used for *multiple* queries. To organize the delivery of such information in an efficient manner while minimizing the bandwidth consumed, the multicast scheme of CVSAP can be used to determine nodes to collect and forward information and *jointly* finding a suitable routing between these nodes. Within the EU FP7 project UNIFY [16] Rost and Schmid are currently working together with project partners on an extension of CVSAP to enable such an efficient monitoring information platform, including the *shared* usage of processing nodes for multiple aggregation / multicast schemes.

B. Other Related Work Pertaining to Aggregation

In the following two other possible applications of the aggregation scheme are outlined.

1) Sensor Networks: In the context of sensor networks, an important task is to minimize the energy consumption. As most energy is consumed during radio communication, models have been proposed in which multiple messages are first collected and then sent consecutively to minimize the time the radio has to be enabled [13]. Furthermore, when computing a commutative and associate function on measurements originating at sensor nodes, the aggregation scheme we propose can be directly adopted [8]. The objective to minimize both bandwidth usage and the cost of installing processing nodes applies to both these applications.

2) Big Data Applications: Big data applications make frequent use of aggregation and filtering, such that the question arises where to place such functionality. The authors of [6] propose for example a direct-interconnect network topology called Camdoop for executing MapReduce tasks. Using this network topology, during the shuffle phase intermediate results are locally merged to reduce the consumed bandwidth. The authors specifically take commutative and associative functions into account and argue that this approach can still significantly reduce traffic, even when the reduce function is not commutative and associative. Hence, CVSAP may be applied when considering data centers that provide MapReduce-as-a-Service [23], where MapReduce jobs can be spawned by customers and the provider may place aggregation functionality within the network to reduce link load.

C. Related Work Pertaining to Multicast

Considering the problem of in-network processing, the multicast scheme can be used for efficient broadcasting in virtual networks (see e.g. [10] for applications to content-addressable networks). Shi proposed in 2001 to consider virtual multicast trees to improve scalability of multicasting in the general context of overlay networks [22]. However, there are also two other distinct applications of the multicast variant of CVSAP, namely optical multicast and geo-replication of services.

1) Optical Multicast: A more specific application lies in the distribution of e.g. IPTV over optical backbones using technologies as Synchronous Digital Hierarchy (SDH) and reconfigurable optical add / drop multiplexer (ROADM) [11]. ROADMs are used to inject or extract information to or from a 'wavelength' such that the information can be distributed to the access network over e.g. ethernet. In the context of the in-network processing terminology, processing nodes would represent ROADMs, such that flows can be duplicated and destined to other destinations seamlessly. Finding the best locations for these expensive devices is an important network design problem [12]. Furthermore, dropping 'packets' at ROADMs and 'splitting' a wavelength induces a distinct power loss of the signal [21]. Therefore, such optical multicast enabled routers are only able to forward the stream towards a limited number of recipients. This necessitates the consideration of capacities for processing nodes.

2) Geo-Replication of Services / Caching: Another important application stems from the current trend in geo-replication of services and cache placement. In this context

the general task is to distribute copies of data throughout the network to balance load across servers and to reduce e.g. the users' latency of accessing for example video streams [14]. The relation to the multicasting communication scheme was already established by Oliveira and Pardalos [15], who defined the Flow Streaming Cache Placement Problem (FSCPP). In the FSCPP processing nodes represent caches connected in an arbitrary hierarchy and receivers are users accessing content. Even though they generally consider a similar problem, their problem definition is inherently flawed (see the proof of Rost and Schmid in [19]).

IV. Main Contributions

This section outlines the main contributions made in the thesis.

A. Part I: Computational Complexity

As a first step to approaching the CVSAP problem, the computational complexity of the problem is studied extensively in Part I of the thesis [18]. The main – and somewhat discouraging – result is that CVSAP is *inapproximable* unless $P = NP$ holds. This result is obtained by a set cover reduction, showing that finding *any feasible solution* is NP-complete. To better understand the inherent complexity of CVSAP computationally weaker variants are considered:

1) As CVSAPs communication is directed, the undirected variant, called the Constrained Virtual Steiner Tree Problem (CVSTP), is introduced.
2) Whereas CVSAP and CVSTP consider (processing) node capacities and edge capacities, weaker variants are introduced that drop only the node capacities or both capacities altogether. These variants are referred to as NVSAP and VSAP and NVSTP and VSTP respectively.

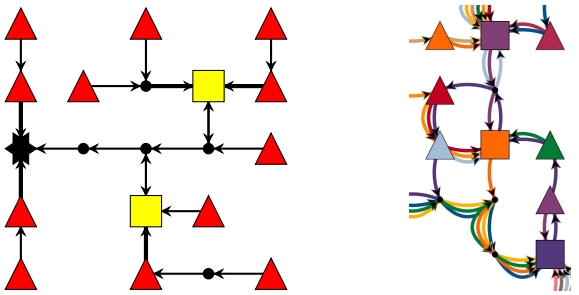
Given these six different variants, it is easy to observe that the undirected variants can be reduced to the directed ones and ones without capacities can be reduced onto the respective ones with capacities (solid black arrows in Figure 2).

Using three different reductions onto other known problems in the literature, namely the Steiner Arborescence Problem (SAP) [3], the Connected Facility Location Problem (CFLP) [9] and the Degree-Constrained Node Weighted Steiner Tree Problem (DNST) [17] the following results are obtained:

a) Theorem 3.17: The Virtual Steiner Arborescence Problem and the classical Steiner Arborescence Problem are equivalent and polynomial-time reductions exist for both directions. This construction yields an approximation algorithm for VSAP as any algorithm for SAP can be applied.

b) Corollary 3.11: There exists a polynomial-time reduction of VSTP to CFLP that preserves the cost up to a factor of 2. As a corollary, it is easy to derive a 8-approximation for VSTP based on a 4-approximation for CFLP.

c) Theorem 3.22: There exists a bi-criteria approximation algorithm for NVSTP based on a reduction onto DNSTP, such that the node-degree of any processing node is at most twice as high as the capacity allows and is logarithmic in the cost.



(a) single-commodity formulation (b) cyclic flows

Fig. 3: Visualization of the single-commodity approache. The single-commodity approach computes only a single flow towards the receiver and edges are only annotated with the bandwidth used. The single-commodity flow approach necessitates a novel decomposition scheme for the flow, as it may contain arbitrary cyclic flows.

B. Part II: Exact Algorithms

As proven in Part I of the thesis [18] CVSAP is inapproximable. To nevertheless tackle CVSAP, two exact algorithms based on Mixed-Integer Programming (see [2] for an introduction) are given.

d) Multi-Commodity Formulation: The multi-commodity flow formulation A-CVSAP-MCF [18] computes paths independently for each sender and processing node (cf. Figure 1e) and relies on Miller-Tucker-Zemlin constraints [5]. However, the number of variables and constraints in this formulation depends polynomially on the number of senders and processing nodes. This renders solving large instances hard, as a scenario with 500 processing nodes and 1,000 edges already requires more than 500,000 many variables.

e) Single-Commodity Formulation: To overcome this computational limitation, Rost has derived an intriguing single-commodity flow formulation for CVSAP, referred to as IP-A-CVSAP [18]. This single-commodity flow formulation abstracts from the concept of sources and destinations of flows and only computes how many flows pass a given edge. While the number of variables and constraints can be significantly reduced in this fashion, a novel decomposition scheme is required to obtain the underlying paths. Figure 3b illustrates that this decomposition scheme must be able to handle arbitrary cyclic flows. The Decompose Algorithm presented in the thesis [18] is a novelty, as it extends the standard flow decomposition approach for arbitrary directed graphs.

C. Part III: (Linear) Heuristics for CVSAP

The compactness of the single-commodity flow formulation IP-A-CVSAP allows not only for employing branch-and-cut techniques to solve CVSAP to optimality, but allows the derivation of efficient heuristics. Within the thesis [18] several different heuristic techniques for Mixed-Integer Problems [1] are employed to obtain six different linear heuristics.

f) Diving: The first three heuristics (GreedyDiving, GreedySteinerDiving and FastGreedySteinerDiving) are ob-

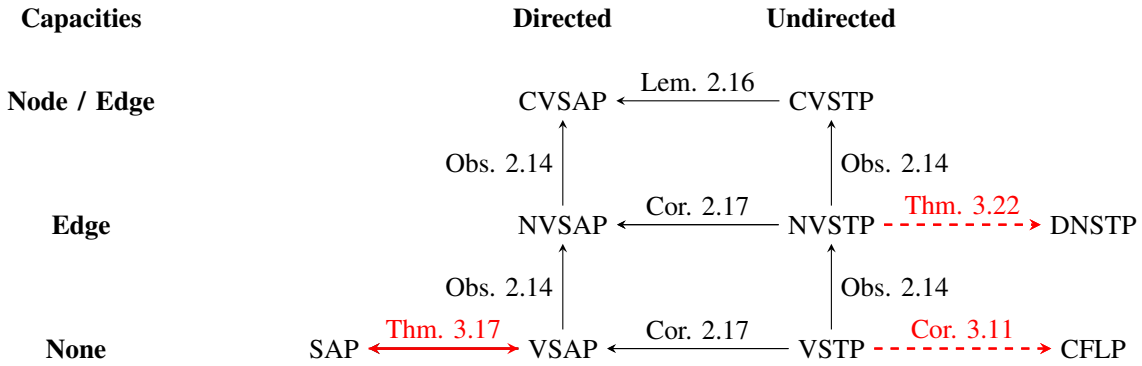


Fig. 2: Relation between CVSAP variants and related optimization problems. Solid edges represent strict cost preserving reductions while dashed edges represent reductions preserving the cost upto a constant factor. The numbering of theorems, lemmas, corollaries and observations refers to the one in [18].

tained by utilizing a diving scheme, such that a subset of variables are iteratively fixed to a certain value and LP-relaxations are re-computed in between iterations. Since only a few number of variables are fixed in each iteration, the runtime of these heuristics is comparatively high.

g) Multiple Shots: The second two heuristics (MultipleShots and MultipleShotsSquared) employ a common probabilistic scheme which fixes variables iteratively by considering their values as probabilities. As in each iteration many variables are fixed, the number of iterations necessitated for obtaining a solution is generally quite low. These heuristics therefore trade off accuracy with runtime (as shown in the computational evaluation).

h) Decomposition: The last heuristic, called FlowDecoRound, only requires a single LP-relaxation and heuristically (and probabilistically) decomposes the given single-commodity flow into paths to construct a solution. While this heuristic is the fastest, the obtained solutions are often times as much as 100% far from optimal and this heuristic rather often fails to find solutions at all.

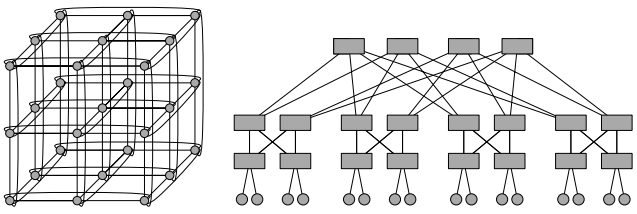
The above presented heuristics are bundled together with the single-commodity flow formulation to obtain the VirtuCast Algorithm.

D. Part IV: Computational Evaluation

The last part of the thesis [18] contains an extensive computational evaluation of the (1) multi-commodity flow formulation A-CVSAP-MCF, (2) IP-A-CVSAP and (3) the given heuristics.

Based on the ranges of different applications, three graph topologies are considered, namely the data-center topologies of 3D-tori and fat trees (see Figure 4 and synthetic ISP topologies generated by IGen (see Figure 5). Besides different graph sizes, generation parameters are chosen such that different ratios of edge and processing costs are considered. Overall 225 different instances are considered.

The main computational results of the thesis [18] are depicted in Figure 6. The VirtuCast algorithm, which was fully implemented by Rost using the framework SCIP [1], is executed for one hour on each of the 225 instances. Since during the execution lower bounds are computed, the (relative) gap to optimality is known. As can be seen, even for largest graph sizes (fat trees with 1584 nodes and 14680 edges; 3D tori with 1728 nodes and 10368 edges; IGen instances with 4000 nodes and 16924 edges), the solutions found always lie within 6% of optimality. Furthermore, for IGen and 3D Torus instances, the average gap lies beneath 2%. These results are furthermore substantiated, as the multi-commodity flow formulation does fail to compute solutions for any of these scenarios and the heuristics by Rost are key to finding high-quality solutions.



(a) A 3D torus of side length 3.

(b) A fat tree using 4-port switches.

Fig. 4: Data-center topologies

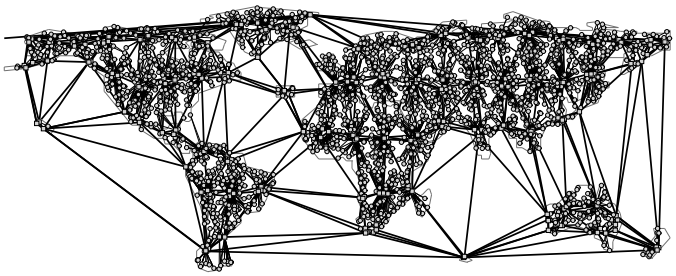


Fig. 5: An ISP topology generated by IGen with 2400 nodes.

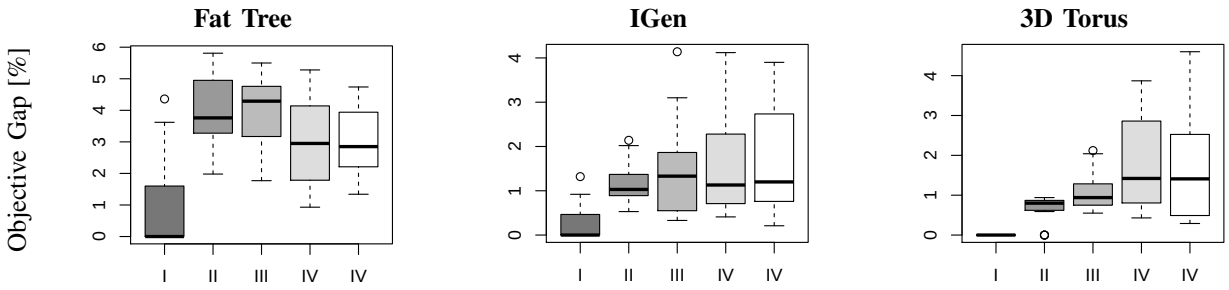


Fig. 6: Objective gap with respect to the optimal solution for different graph sizes I-V for the different topologies after one hour of computation.

V. Conclusion

The advent of Software-Defined Networking and Network Functions Virtualization has opened many interesting research questions. Rost has tackled the quite common service deployment problem of multicasting or aggregation in his thesis [18] in a novel fashion. While for particular applications some heuristic algorithms might exist, the Constrained Virtual Steiner Arborescence Problem subsumes multiple important applications and naturally extends upon classic problems, like e.g. the Steiner Tree Problem. The results on the computational complexity suggest that for many weaker variants only logarithmic approximations might exist. The single-commodity flow formulation and the corresponding novel decomposition scheme – i.e. the VirtuCast algorithm – is clearly the contribution of the most significance. It not only allows for efficiently computing lower bounds but also enables the application of well-thought out heuristics, to obtain high-quality solutions quickly. Lastly, the extensive computational evaluation has clearly demonstrated the applicability of the VirtuCast algorithm to solve large scale instances near-optimally within reasonable time.

References

[1] Tobias Achterberg. SCIP: solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41, 2009.

[2] Dimitris Bertsimas and Robert Weismantel. *Optimization over integers*, volume 13. Dynamic Ideas Belmont, 2005.

[3] Moses Charikar, Chandra Chekuri, To-yat Cheung, Zuo Dai, Ashish Goel, Sudipto Guha, and Ming Li. Approximation algorithms for directed steiner problems. In *Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms*, pages 192–200. Society for Industrial and Applied Mathematics, 1998.

[4] NM Mosharaf Kabir Chowdhury, Muntasir Raihan Rahman, and Raouf Boutaba. Virtual network embedding with coordinated node and link mapping. In *INFOCOM 2009, IEEE*, pages 783–791. IEEE, 2009.

[5] Alysson M Costa, Jean-François Cordeau, and Gilbert Laporte. Models and branch-and-cut algorithms for the steiner tree problem with revenues, budget and hop constraints. *Networks*, 53(2):141–159, 2009.

[6] Paolo Costa, Austin Donnelly, Antony Rowstron, and Greg O' Shea. Camdoop: Exploiting In-network Aggregation for Big Data Applications. In *Proc. USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2012.

[7] Chuck Cranor, Theodore Johnson, Oliver Spataschek, and Vladislav Shkapenyuk. Gigascope: A Stream Database for Network Applications. In *Proc. ACM SIGMOD International Conference on Management of Data*, pages 647–651, 2003.

[8] Min Ding, Xiuzhen Cheng, and Guoliang Xue. Aggregation tree construction in sensor networks. In *Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th*, volume 4, pages 2168–2172. IEEE, 2003.

[9] Friedrich Eisenbrand, Fabrizio Grandoni, Thomas Rothvoß, and Guido Schäfer. Connected facility location via random facility sampling and core detouring. *Journal of Computer and System Sciences*, 76(8):709–726, 2010.

[10] Ludovic Henrio, Fabrice Huet, and Justine Rochas. An optimal broadcast algorithm for content-addressable networks. In *Principles of Distributed Systems*, pages 176–190. Springer, 2013.

[11] Christian Hermsmeyer, Enrique Hernandez-Valencia, Dieter Stoll, and Oliver Tamm. Ethernet aggregation and core network models for effcient and reliable iptv services. *Bell Labs Technical Journal*, 12(1), 2007.

[12] X-D Hu, T-P Shuai, Xiaohua Jia, and Mu-Zhong Zhang. Multicast routing and wavelength assignment in wdm networks with limited drop-offs. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, volume 1. IEEE, 2004.

[13] Bhaskar Krishnamachari, Deborah Estrin, and Stephen Wicker. Modelling data-centric routing in wireless sensor networks. In *IEEE infocom*, volume 2, pages 39–44, 2002.

[14] Srinivas Narayana, Wenjie Jiang, Jennifer Rexford, and Mung Chiang. Joint Server Selection and Routing for Geo-Replicated Services. In *Proc. Workshop on Distributed Cloud Computing (DCC)*, 2013.

[15] CarlosA.S. Oliveira and PanosM. Pardalos. Streaming cache placement. In *Mathematical Aspects of Network Routing Optimization*, Springer Optimization and Its Applications. Springer New York, 2011.

[16] P. Sköldström et al. Towards unified programmability of cloud and carrier infrastructure. In *Third European Workshop on SDN*, 2014.

[17] R Ravi, Madhav V Marathe, SS Ravi, Daniel J Rosenkrantz, and Harry B Hunt III. Approximation algorithms for degree-constrained minimum-cost network-design problems. *Algorithmica*, 31(1):58–78, 2001.

[18] Matthias Rost. *Optimal Virtualized In-Network Processing with Applications to Aggregation and Multicast*, Master Thesis, Technische Universität Berlin,, 2014.

[19] Matthias Rost and Stefan Schmid. The Constrained Virtual Steiner Arborescence Problem: Formal Definition, Single-Commodity Integer Programming Formulation and Computational Evaluation. Technical report, arXiv, 2013.

[20] Matthias Rost and Stefan Schmid. Virtucast: Multicast and aggregation with in-network processing. In Roberto Baldoni, Nicolas Nisse, and Maarten Steen, editors, *Principles of Distributed Systems*, volume 8304 of *Lecture Notes in Computer Science*, pages 221–235. Springer International Publishing, 2013.

[21] George N Rouskas. Optical layer multicast: rationale, building blocks, and challenges. *Network, IEEE*, 17(1):60–65, 2003.

[22] Sherlia Shi. A proposal for a scalable internet multicast architecture. In *Washington Universtiy*, 2001.

[23] Yang Wang and Wei Shi. On scheduling algorithms for mapreduce jobs in heterogeneous clouds with budget constraints. In *Principles of Distributed Systems*, pages 251–265. Springer, 2013.