Energy-Aware Fog and Cloud Interplay Supported by Wide Area Software Defined Networking

Piotr Borylo, Artur Lason, Jacek Rzasa, Andrzej Szymanski and Andrzej Jajszczyk
AGH University of Science and Technology, Department of Telecommunications, Krakow, Poland
Email: {borylo, rzasa, szymans}@agh.edu.pl, {lason, jajszczyk}@kt.agh.edu.pl

***Abstract*—The paper focuses on dynamic resource provisioning in an all optical wide area Software Defined Network supporting energy-aware interplay between fog and cloud. The main contribution of this paper is a *latencyAware* policy for handling fog related traffic. The policy is combined with anycast strategies in order to form a complete approach able to handle fog and cloud traffic simultaneously. The proposed approach was compared with reference solutions with respect to latency of fog related requests, CO_2 emission and impact on network performance. It is shown that both latency and carbon footprint may be reduced without significant deterioration of network performance.**

I. Introduction

Fog computing is an emerging concept introduced and defined by Cisco in [1] as a highly virtualized platform that provides computing, storage, and networking services located at the edge of a network. The idea of fog computing is also supported by other companies introducing their own nomenclature: EdgeComputing (Akamai), Intel's Intelligent Edge or Microsoft's Cloudnet. Deploying computing resources at the network edge allows to meet the constraints imposed by modern applications, e.g. location awareness, low latency, support for mobility or geographical distribution of services. The most frequently referred use cases for the fog computing concept are related to the Internet of Things (IoT) as well as Smart Grids, Smart Cities (Vehicular Networks, smart traffic light systems, etc.) and Smart Buildings. For example, Vehicular Networks exchanging information with external systems (e.g. smart traffic lights) take advantage of the fog's geo-distributed manner and proximity to the customers, and further allow for low-latency and real-time interactions. Another example use case, Smart Grids, assume a huge number of sensors generating vast amounts of data that should be processed. Some Big Data preprocessing as well as control optimization might be, in an optimal case, performed by utilizing fog computing infrastructure. In [2] a platform to perform Big Data analysis in cooperating fog and cloud infrastructures was proposed along with energy use considerations regarding processing data in the fog vs. sending them to the cloud. The industry and researchers have taken numerous steps in order to widely deploy fog computing, example achievements are: modifications to mobile operating systems in order to optimize fog features, middlewares allowing for implementation of an application transparently to the final deployment infrastructure (local device, fog, cloud), optimization of web pages for fog purposes or fog resource provisioning algorithms.

Fog infrastructure is assumed to be highly heterogeneous as a wide variety of devices may contribute to the overall computing power. A set of possible devices spans not only sensors, wearable devices or smartphones, but also covers base stations, vehicles or network devices with extended functionality. For example, Cisco introduced an IOx platform that brings Cisco IOS and the open source Linux operated together in a single device to support fog computing. Additionally, the idea to utilize base stations and points of presence to deploy computing servers at the edge gains on popularity. Therefore, the currently available resources may vary as the whole infrastructure has a highly dynamic nature due to its heterogeneity. The effect is amplified by mobility of customers affecting both location of resources and traffic distribution. Additionally, in the fog computing concept, failures during accessing resources are considered to be normal states. As a consequence, unforeseen outages of available fog computing resources may occur and further lead to instabilities.

Properties of fog computing clearly indicate that fog computing was not designed to replace cloud computing, as the latter ensures practically unlimited computing resources and benefits from economies of scale. Instead, fruitful interplay between fog and cloud is expected to optimize resource utilization and improve services' Quality of Experience. Thus, fog computing is sometimes referred as a paradigm extending cloud to the edge of the network. For some of the applications, both cloud globalization and fog localization are beneficial. A Smart Grid use case might serve here as an illustrative example, where results of Big Data preprocessing performed in the fog are further passed to the cloud for long scale (days, months or even years), latency-insensitive analysis.

However, not only regular operations should be considered. As the aforementioned outages of fog resources may occur, it is reasonable to direct some of the latency-sensitive fog tasks to the cloud in order to maintain users' experience when local resources are unavailable. Unfortunately, data centers (DCs) provisioning cloud services are usually distant to the network edge which is not in line with the need for low latency. Wide Area Software Defined Networking (WA-SDN) tends to be a solution for this issue due to the separation of data and control planes. The concept of fog and cloud interplay was visualized in Fig. 1, where a WA-SDN controller communicates with fog and cloud orchestration software. Fog instances, which should be drawn next to each of the network nodes, were omitted for clarity. Centralized nature and complete knowledge

978-1-4799-6664-6/16/$31.00 ©2016 IEEE

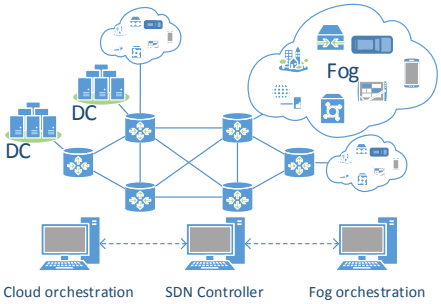

Fig. 1. The concept of cloud and fog interplay

about network state allows the controller to perform global traffic-engineering that may be aimed at meeting latency requirements of fog related tasks. The proposed solution should respect the coexistence of background, fog and cloud related traffic. Additionally, it shoud also take into consideration the problem of energy-awareness, which is a non-negligible issue when considering energy-hungry cloud infrastructure [3].

As stated in [4] a typical 500 square meter DC may consume 38 MWh of energy per day. Between 2000 and 2007, the worldwide energy consumption of DCs increased from 70 to 330 billions kWh, and this value is expected to grow to more than 1000 billion kWh by 2020. The enormous energy consumption obviously affects operational costs. In 2010 the global electricity bill for DCs was estimated to be over 11 billions dollars, and this amount is almost doubled every five years [5] while prices can vary strongly [6]. Furthermore, energy consumption entails carbon emission which impacts the natural environment and will probably introduce extra costs due to the forecasted law regulations. That is why one of the main areas of interests in the context of geographically-diverse DCs is to minimize overall carbon footprint by utilizing renewable energy sources which generate significantly less amount of CO_2 per each kWh of energy. We can only expect that increasing popularity of the IoTs concept may only pronounce the trends.

In this paper we propose and investigate a latency-aware policy for handling fog traffic directed to DCs. The proposed policy is expected to meet latency constraints of fog requests and simultaneously not to deteriorate performance of the infrastructure. Several realistic assumptions are taken. First of all, we consider a hybrid power cloud infrastructure, where selected DCs are powered using energy from renewable sources while the rest uses energy obtained from conventional sources. Second, an optical network (e.g. elastic optical network (EON), indicated as the most appropriate one to provide cloud computing services [7]) interconnecting such a hybrid power infrastructure is investigated under a dynamic traffic scenario in order to reflect an on-demand manner of fog and cloud services provisioning. Next, proportions of traffic types and contribution of cloud service to the overall DC traffic are based on traffic statistics and predictions provided in [8] and [7], respectively. What is more, investigated reference network topologies were casted on real locations in order to estimate realistic distances between network nodes. Finally, Markov Modulated Poisson Process (MMPP) with two states was used to model the unexpected and bursty nature of latency-aware fog traffic. The proposed approach is compared with several reference approaches with regard to network performance, CO_2 emission and average length of lightpaths handling fog requests. As all optical networks are buffer-free, reduction of lightpath length expressed in kilometers and decrease in latency are equivalent.

To the best of our knowledge it is the first work considering SDN-based WAN network support for energy-aware interplay between fog and cloud. The paper proves that it is possible to handle fog traffic in a latency-aware manner by decreasing average lightpath length without significant deterioration in terms of blocking probability and carbon footprint. The aim of the work, which is most related to this paper, is to reduce carbon dioxide emission in cloud and fog infrastructures delivering video streaming services [9]. However, in this work the authors neglect the networking aspect and present the offline approach while our work focuses on network issues in a network where requests are served in an on-demand fashion.

II. Problem Statement

The problem investigated in this work may be described as dynamic anycast routing in optical networks with latency-aware fog and energy-aware cloud service provisioning.

In the formal problem definition, graph $G(V, E)$ denotes the physical network, where V is the set of nodes, and E is the set of links. V_{DC} describes the set of nodes associated with DCs. DCs powered from renewable energy sources are called *green* and network nodes associated with *green* DCs are denoted by V_{gDC}. On the other hand, DCs powered from traditional energy sources are called *brown* and nodes directly connected to them are denoted by V_{bDC}. In all further equations the g and b lower indices will always denote *green* and *brown*, respectively. The same set of DCs is able to handle cloud and fog related traffic. Nodes that are not associated with any DC are termed client nodes, and denoted by V_C. The following relations are met: $V_{gDC} \cap V_{bDC} = \emptyset$, $V_{gDC} \cup V_{bDC} = V_{DC}$, $V_{DC} \cap V_C = \emptyset$ and $V_{DC} \cup V_C = V$. The V_F denotes network nodes associated with fog instances. It is assumed that fog infrastructure is present at every edge of the network, thus V_F = V. Therefore, fog related traffic is generated by all nodes in the network. However, for the set $V_{DC} \cap V_F$ this traffic is handled simply by the DC attached to the same node as a fog instance and as such it may be omitted from our studies. The network controller has full knowledge about the network topology and its state, including information about existing lightpaths and the location of DCs (V_{DC}).

Dynamic resource provisioning algorithms operate on a lightpath request (LR). Two types of LRs are assumed: (1) a unicast LR from a source node $s \in V$ to a destination node $d \in V$; (2) an anycast LR from a source node $s \in V_C$ to one from the set of possible destinations $d \in D \subseteq V_{DC}$.

Unicast LRs are served using alternate routing with three precomputed paths. The details of the algorithm for unicast

LRs may be found in [3]. In this work unicast lightpath requests are used solely for handling the background traffic. As anycast schema may be defined as *one-to-one-of-many* routing, for each anycast LR the resource provisioning algorithm needs to choose a destination $d \in D$ and then set up a lightpath from the source node s to the chosen node d. Anycast LRs are used to handle both cloud and fog traffic. For the purpose of simplicity, cloud request (CR) and fog request (FR) will denote an anycast LR handling cloud and fog traffic, respectively. Each CR is used to handle requests assigned to one of the three types of cloud services: Processing, Storage and Software as a Service (*PaaS*, *StaaS* and *SaaS*). Each CR carries information about such assignment, thus a service oriented network controller is able to take advantage of fitting different anycast strategies to different types of cloud services. A separate policy is applied to fog requests, FRs, in order to introduce latency awareness.

In order to estimate carbon footprint of the considered infrastructure it is assumed that *brown* DCs contribute to CO_2 emission proportionally to the consumed power, while *green* DCs do not contribute to CO_2 emission at all. Therefore, processing requests in *green* DCs creates opportunity to reduce CO_2 emission. Energy consumed by network nodes always comes from non-renewable sources. However, its amount is negligible in comparison to energy consumed by DCs (especially in the assumed case of all optical networks) and, as such, was ignored in the subsequent studies.

III. Fog and cloud interplay

In this section we present a detailed description of our approaches to solving the problem of fog request handling. However, we first present a brief introduction of associated components in order to facilitate understanding of the proposed solutions.

A. Anycast strategies for CRs and FRs handling

Three anycast strategies are considered in this paper for routing of lightpath requests (LRs) associated with both cloud and fog requests. In the *closest* strategy a lightpath is established between the source s and the closest reachable destination $d \in V_{DC}$. If none of $d \in D$ is available then the LR is rejected. The *closestGreenWithPenalty* strategy works analogously to the *closest* strategy but with one significant difference. Distance from s to $d \in V_{bDC}$ is multiplied by the *penalty* factor, where the *penalty* is an input parameter of the strategy. The *closestGreenWithPenalty* strategy with *penalty*=1.0 is equivalent to the *closest* strategy. Finally, in the strategy *closestGreen* the network controller performs operations analogous to the *closest* strategy for all $d \in V_{gDC}$. If none of $d \in V_{gDC}$ is available then the network controller repeats the same operation for all $d \in V_{bDC}$. If none of $d \in V_{DC}$ are available then the LR is rejected. To sum up, the *closestGreen* and *closestGreenWithPenalty* create an opportunity to reduce the carbon footprint by favorising *green* DCs over *brown* DCs. However, as a side effect, the average lightpath length and resource utilization may significantly increase in case of *closestGreen* strategy, where green DCs are firmly prioritized. Thus, the *closestGreenWithPenalty* strategy was proposed to flexibly balance the trade-off between carbon footprint reduction and the average lightpath length. The higher value is assigned to the *penalty* parameter the more *green* DCs are preferred over *brown* ones and, at the same time, the higher is the average lightpath length. The aforementioned anycast strategies were proposed and thoroughly investigated in our previous work [10].

B. Fitting schemas for CRs handling

Based on the properties of both cloud service types and anycast strategies we proposed the *compound* schema for fitting strategies to service types. *PaaS* services are characterized by the highest energy consumption. Thus, *PaaS* requests should be handled by the *closestGreen* strategy, which strictly favors *green* DCs over *brown* DCs, at a cost of increased average lightpath length and increased network resource utilization. On the other hand, the energy consumed in a DC by a *StaaS* request is negligible. That is why *StaaS* requests should be handled by the *closest* strategy. It does not provide any *green* nodes preference, but is expected to ensure the shortest average lightpath length and thus, the lowest network resource occupancy. Finally, *SaaS* services usually occupy network resources for a longer time than the two other service types while energy consumption required for their handling in a DC may strongly vary between the upper and lower bounds defined by the values assumed for *PaaS* and *StaaS* services, respectively. Therefore, *SaaS*, as the most volatile service type, should be handled by the *closestGreenWithPenalty* strategy and the *penalty* parameter should be used to adjust the aggressiveness of the strategy to the changing properties of *SaaS* requests. The rationale behind the proposed schema is that the energy intensive tasks should be performed in *green* DCs while for other requests it is better to balance network resource use and the carbon footprint reduction. In subsequent studies this solution will be denoted as the *compound* fitting schema. Additionally, *closest* and *closestGreen* fitting schemas describe cases when all cloud services are handled using the same anycast strategy, *closest* and *closestGreen* respectively. A more detailed discussion of those schemas may be found in our previous work [11].

C. The latencyAware policy for FRs handling

It is a common assumption to operate on distances expressed in a hop manner. All the aforementioned strategies and fitting schemas use such distances. However, a key feature of the *latencyAware* policy proposed in this paper is to utilize anycast strategies operating not only on hop distances but also on physical distances expressed in kilometers. In this way, the policy aims to decrease physical distance which directly reduces both latency in buffer-free all optical networks and utilization of costly long optical lines. However, as a side effect the length, expressed in a hop manner, may increase resulting in higher network resource utilization, as a given wavelength is occupied on more network links. Thus, only fog requests, requiring

low latency connections will be allowed to use this policy. A combination of anycast strategies operating on distances expressed in hops or kilometers with fitting schemas applied to cloud services forms a complete approach for CRs and FRs handling. Anycast strategies utilized in this paper operate on a set of precomputed paths, calculated before the first connection request arrives. In previous studies we assumed a set of three link-disjoint shortest paths between each pair of network nodes, calculated using hop-based distances. In order to implement the *latencyAware* policy, this set needs to be expanded with an additional set of three link-disjoint shortest paths calculated using physical distances.

The *latencyAware* policy in its preliminary steps utilizes a selected anycast strategy to find two reachable destinations d_{hop} and $d_{km} \in V_{DC}$: the former one chosen in a hop manner and the latter chosen in the physical distance manner. Along with the destinations, anycast strategies return also distances from source to those destinations expressed in corresponding metrics, denoted as $dist^{hop}_{d_{hop}}$ and $dist^{km}_{d_{km}}$. Next, an additional pair of distances is computed by running anycast strategies with set of possible destinations D_{tmp} limited to one node d_{km} and d_{hop} for hop and physical distance metrics, respectively. As a result, $dist^{hop}_{d_{km}}$ and $dist^{km}_{d_{hop}}$ distances are found. The motivation for such a crosswise approach is to answer the question: what would be the distance to the destination selected using one metric if we obtain it with an anycast strategy using the other metric. Such potential distances will be further used to express both latency improvements and an increase in lightpath length. In a case when none of the $d \in V_{DC}$ is reachable using anycast with either metrics, then the FR is rejected. When only one of the calculated destinations is reachable, d_{hop} or d_{km}, then the request is satisified using this destination. This might happen as we use precomputed paths and network resources might be fully utilized on those paths. For the same reason $dist^{hop}_{d_{km}}$ and $dist^{km}_{d_{hop}}$ may be equal to infinity.

Finally, when both destinations d_{hop} and d_{km} are reachable, the following equation is used to express the normalized reduction of physical distance achieved by selecting d_{km} instead of d_{hop}: $dist_{red} = (dist^{km}_{d_{hop}} - dist^{km}_{d_{km}})/dist^{km}_{d_{hop}}$. For the case when $dist^{km}_{d_{hop}} = \infty$ we assume $dist_{red} = 1$ This gain is further compared with the following threshold $dist_{th}^{dist^{hop}_{d_{km}} - dist^{hop}_{d_{hop}}}$, where $dist^{hop}_{d_{km}} - dist^{hop}_{d_{hop}}$ denotes that threshold value depends on the increase of the lightpath length expressed in hops and resulting from selection of d_{km} instead of d_{hop}. For example, $dist^1_{th}$ denotes threshold value used for comparison in cases when d_{km} is one hop more distant to s comparing to d_{hop}. Destination d_{km} is selected only if $dist_{red} > dist_{th}$, otherwise destination d_{hop} is selected. The rationale behind this approach is to apply increasing values to subsequent thresholds $dist^0_{th}$, $dist^1_{th}$, ..., $dist^n_{th}$ and in this way ensure that the higher the increase in hops the greater reduction in latency is required to select latency-aware destination. Such an approach is expected to balance the tradeoff between the blocking probability and the latency of FRs. The pseudocode presented in Algorithm 1 precisely describes the *latencyAware* policy.

Algorithm 1 latencyAware policy

Input: $(s, D) = (s, V_{DC})$
1: $(d_{hop}, dist^{hop}_{d_{hop}}) \leftarrow arwaHop(s, D)$ ▷ Perform anycast strategies
2: $(d_{km}, dist^{km}_{d_{km}}) \leftarrow arwaKm(s, D)$ ▷ with specific weights
3: $(d_{km}, dist^{hop}_{d_{km}}) \leftarrow arwaHop(s, D_{tmp} = \{d_{km}\})$
4: $(d_{hop}, dist^{km}_{d_{hop}}) \leftarrow arwaKm(s, D_{tmp} = \{d_{hop}\})$
5: **if** $d_{hop} \neq null$ OR $d_{km} \neq null$ **then**
6: **if** $d_{hop} = null$ **then**
7: Establish lightpath between s and d_{km}
8: **else if** $d_{km} = null$ **then**
9: Establish lightpath between s and d_{hop}
10: **else** ▷ Both destinations are reachable
11: $dist_{red} \leftarrow \begin{cases} 1, & if dist^{km}_{d_{hop}} = \infty \\ (dist^{km}_{d_{hop}} - dist^{km}_{d_{km}})/dist^{km}_{d_{hop}}, & otherwise \end{cases}$
12: **if** $dist_{red} > dist_{th}^{dist^{hop}_{d_{km}} - dist^{hop}_{d_{hop}}}$ **then**
13: Establish lightpath between s and d_{km}
14: **else**
15: Establish lightpath between s and d_{hop}
16: **end if**
17: **end if**
18: **else** ▷ None of destinations is reachable
19: Reject request
20: **end if**

D. Complete approaches

To compose a complete approach, an appropriate policy should be combined with an anycast strategy to form a solution for handling FRs and a fitting schema should be chosen to adjust anycast strategies to services. As latency is a crucial factor for FRs, the *latencyAware* policy was chosen to be combined with the *closest* strategy. The *compound* fitting schema will be used for provisioning cloud services, as it gives balanced CO_2 emission and network performance. In subsequent studies this solution will be denoted as the *proposed* approach.

For the purpose of comprehensive assessment several reference approaches are defined. The *latencyBaseline* approach is the only one that utilizes the *latencyAware* policy and combines it with the *closest* anycast strategy for FRs and the *closest* fitting schema for cloud services provisioning. This approach is expected to provide the highest reduction of FRs latency. The remaining reference approaches do not introduce any latency-awareness. Therefore, a simple *hopAware* policy for FRs is utilized where only d_{hop} is considered. The *blockingBaseline* approach assumes the *closest* strategy handling FRs and the *closest* fitting schema for CRs. This approach is expected to provide the shortest average lightpath length expressed in hops, and thus, will be interpreted as a baseline for network performance. The *carbonBaseline* is expected to significantly reduce carbon footprint. This approach strictly favours *green* DCs for handling the whole anycast traffic by assuming the *closestGreen* strategy for handling FRs and the *closestGreen* fitting schema for handling CRs. Finally, the *carbonBaselineCompound* assumes the *closest* strategy for handling FRs and the *compound* fitting schema for handling CRs. This approach originates from studies conducted in our previous work which proved that the *compound* fitting schema is very effective in carbon footprint reduction [11]. Table I summarizes information about the complete approaches.

IV. Simulation environment

To assess the presented approaches several simulations were performed in two reference networks: the 24 nodes US network and the 21 nodes Italian Mesh Network [12]. Each physical link is composed of two fibers, one in each direction. Each fiber carries 80 optical channels transporting data at a rate of 10 Gb/s. There are no wavelength converters in the networks and the optical channel assignment is done using the first-fit schema. In both topologies six DCs were deployed, $V_{DC} \in \{1, 5, 14, 15, 17\}$ and $V_{DC} \in \{1, 7, 11, 13, 15, 20\}$ in US and *ItalyNet*, respectively. Three of those DCs were assumed to be *green*. The topologies, location of DCs and the number of *green* facilities were chosen based on the assumptions provided in [12].

The offered traffic is composed of two traffic types: background and DC. The background traffic consists of unicast LRs and is generated based on a uniform traffic matrix. The DC traffic consists of anycast FRs and CRs generated only by $s \in V_C$ (with the same intensity for each s) to a single anycast group $D = V_{DC}$. Proportions of traffic types (background and DC) are obtained based on predictions for years from 2016 to 2020 performed by Cisco and summarized in [7].

The lightpath holding time (ht) is exponentially distributed independent of the traffic type and topology. For the background and fog traffic as well *PaaS* and *StaaS* requests the mean value of ht is always equal to 10 seconds. For *SaaS* it is equal to 360 seconds. The selected values originate from our previous work [11]. In the base case each lightpath request is a unidirectional request with exponentially distributed inter-arrival time iat. The mean intensity of background traffic is equal to 0.0797 and 0.1119 for US and *ItalyNet*, respectively.

Fog traffic intensity is modeled using Markov Modulated Poisson Process (MMPP) to reflect the suddenness and bursty nature this traffic. Two states of Markov Chain were assumed. In the *off* state the mean intensity of fog traffic is equal to 0. During the *on* period the mean fog traffic intensity is equal to 0.99 and 0.495 for US and *ItalyNet*, respectively. Transitions between states are equally probable, thus the average length of *on* and *off* periods is the same.

The contribution of different cloud services to the overall DC traffic was evaluated using traffic statistics provided by Google in [8]. Requests were classified according to duration and their requirements on computing resources, including CPU time and RAM size. The provided classification was easily mapped to three types of cloud services considered in our work. Therefore, the mean intensity of CRs is 0.7699 *PaaS*, 1.0633 *StaaS* and 0.051 *SaaS* requests per second for the US network and 0.7418 *PaaS*, 1.0245 *StaaS* and 0.049 *SaaS* requests per second for the *ItalyNet*.

In order to asses scenarios under different loads we multiplied the mean intensity of the fog traffic by a scaling factor ranging from 1.0 to 13.0 (US) and from 1.0 to 7.0 (*ItalyNet*). The mean intensity for background and cloud traffic remained unchanged. Scaling was done by changing the mean *iat*, while the mean ht of all traffic types remained constant. The increase in the fog traffic is motivated by growing number of devices contributing to the IoT concept. For the purpose of comprehensive assessment, the results are gathered for a very broad range of fog traffic intensity.

V. Results

Numerous simulation scenarios were investigated as various parameters are variable, including selection of different nodes from V_{DC} as *green* ones. Due to limited space, only selected simulation results will be presented: two scenarios for the US network with $V_{gDC} \in \{1, 14, 15\}$ and $V_{gDC} \in \{5, 11, 17\}$, and one case for the *ItalyNet* with $V_{gDC} \in \{7, 13, 20\}$. Energy consumption of *SaaS* requests was assumed to be 5.4, 2.7 and 2.7 kW/(Gb/s) in each simulation scenario, respectively.

To asses the *proposed* approach three indicators were used. The first one is the LR blocking probability, calculated as the ratio of rejected LRs to all LRs. The second indicator expresses the average length of a lightpath handling a FR which may be directly translated to the latency in all optical networks. Finally, the last one estimates the carbon footprint and is the average ratio of the power consumed from non-renewable sources to the amount of DC traffic switched in all DCs ($brownKiloWatts/(Gb/s)$). The simulation results were obtained using the OMNeT++ simulator and evaluated at the 0.95 confidence level using the batch means method. All results were presented with regard to the aforementioned scaling factor of fog traffic as a measure of traffic intensity.

A. Penalty and thresholds values

Based on the analysis of preliminary results some assumptions about *latencyAware* thresholds were taken. First of all, it is assumed that $dist_{th}^{0} = 0$. This assumption denotes the case of d_{hop} and d_{km} being equally distant to s in a hop manner. The network controller should always prefer d_{km} which is closer in the physical distance manner. This assumption is intuitive as we reduce latency for FRs without increase in utilization of network resources. On the other hand, we assume that $dist_{th}^{n} = 1$, where $n \geq 2$. This means that the controller will not prefer any d_{km} over d_{hop} if it results in increase of lightpath length from s by two or more hops. The rationale behind this assumption is that increasing lightpath length by two or more hops results in significant deterioration on network performance while the improvement achieved in the context of latency is negligible in most cases. It also results from the fact that in reference topologies a node being more distant by two hops from s is rarely much closer in a physical distance manner. Finally, $dist_{th}^{1}$ is assumed to vary from 0 to 1 in 0.25 steps. Simultaneously, in order to use the *closestGreenWithPenalty* strategy the value of the *penalty* parameter had to be determined. We assumed that the operator is minded to slightly deteriorate network performance in order to reduce latency of FRs and the carbon footprint. We assumed that for the cases where the *latencyAware* policy or *closestGreenWithPenalty* strategy was applied to any of service types, the total blocking probability may not increase by more than 0.3 of a percentage point in comparison to the

TABLE I
SUMMARY OF COMPLETE APPROACHES

approach	FR		CR			
	strategy	**policy**	***PaaS***	***StaaS***	***SaaS***	**schema**
blockingBaseline	*closest*	*hopAware*	*closest*	*closest*	*closest*	*closest*
carbonBaseline	*closestGreen*	*hopAware*	*closestGreen*	*closestGreen*	*closestGreen*	*closestGreen*
carbonBaselineCompound	*closest*	*hopAware*	*closestGreen*	*closest*	*closestGreenWithPenalty*	*compound*
latencyBaseline	*closest*	*latencyAware*	*closest*	*closest*	*closest*	*closest*
proposed	*closest*	*latencyAware*	*closestGreen*	*closest*	*closestGreenWithPenalty*	*compound*

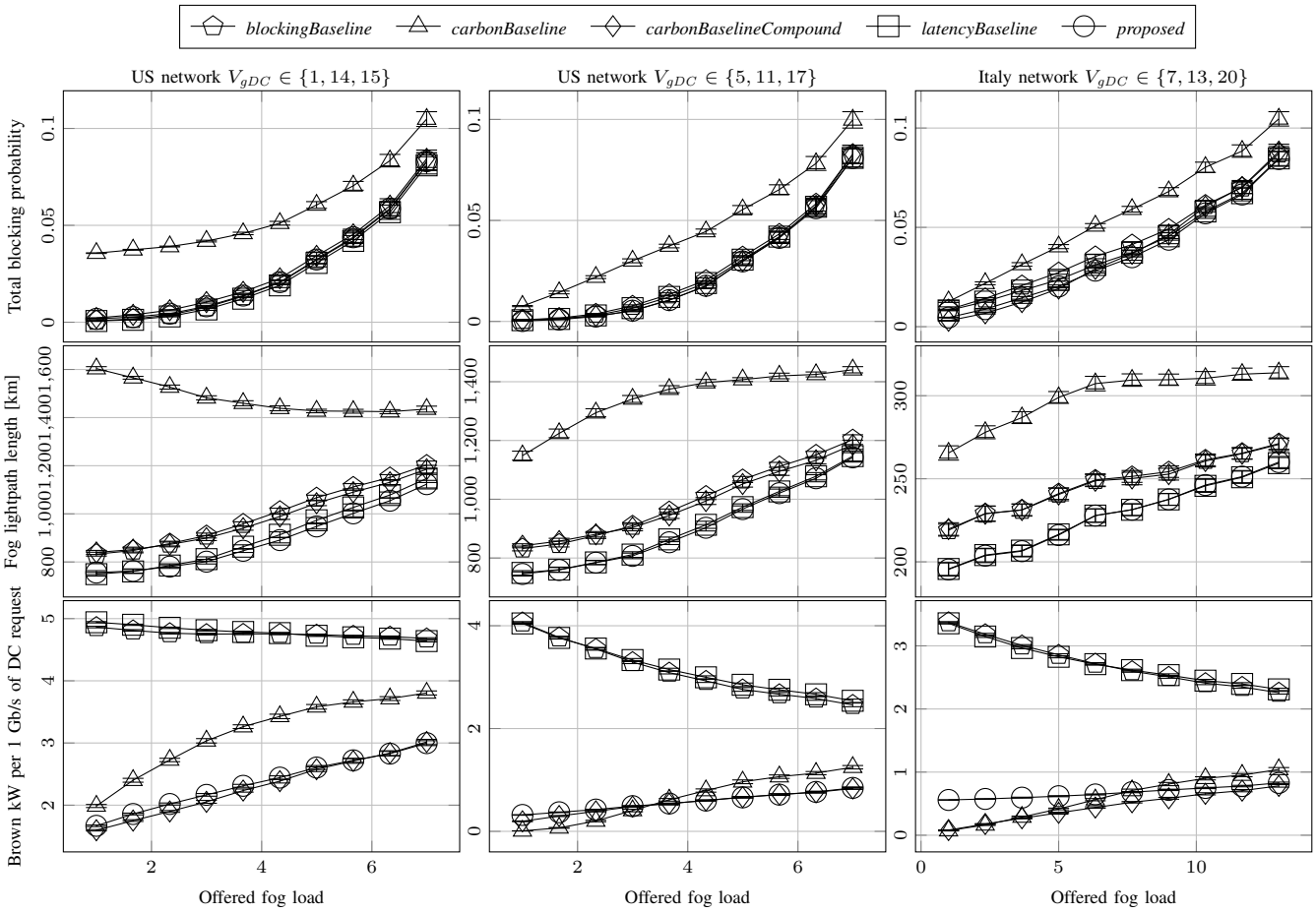

Fig. 2. Plots presenting results for the three simulation scenarios (columns), with regard to the three assessment indicators (rows).

blockingBaseline approach. Thus, for each simulation scenario the total blocking probability was measured with $dist^1_{th}$ ranging in the aforementioned scope and *penalty* values ranging from 1.0 to 4.0 in 0.1 steps. The tuples of $dist^1_{th}$ and *penalty* values that met the aforementioned limitation were denoted as *applicable*. It must be noted that there are cases when only $dist^1_{th}$ or *penalty* parameter are relevant (e.g. *latencyBaseline* approach). In such cases only the relevant parameter varies and is considered as the only decision factor. Next, the specific values from *applicable* set must be selected. If *latencyAware* policy is utilized alone, than the tuple that ensures the highest reduction in latency of FRs is selected. When a given case assumes *hopAware* policy only *penalty* values are investigated and the one that resultes in the highest reduction of CO_2 emission is chosen among the ones in the *applicable* set.

B. Assessment

Fig. 2 presents results obtained for the three aforementioned simulation scenarios (a separate column for each scenario) with regard to the three assessment indicators (a separate row per indicator). Due to the assumptions about thresholds and *penalty* parameters most of the approaches provide the total blocking probability at the level comparable to the *blockingBaseline* approach. What is more, approaches utilizing the *latencyAware* policy or the *compound* fitting schema are able to decrease total blocking probability in some simulation scenarios, which is a result of two aspects. First, the *latencyAware* policy distributes fog traffic more evenly in the network as modified metrics entails changes in preferred paths only for FRs. Second, *green* DCs are preferred in the *compound* schema and may happen to be placed in the

network areas where less background traffic is switched which further improves network performance. The only approach resulting in a significantly higher total blocking probability is the *carbonBaseline* which introduces strict preference of *green* DCs for both FRs and CRs.

From the latency point of view each approach that uses the *latencyAware* policy significantly decreases length of lightpaths handling FRs. Therefore, latency is decreased in comparison to the approaches utilizing *hopAware* policy. As a side effect, utilization of costly long optical lines is also reduced. In contrast with this, strict preference of *green* DCs for both FRs and CRs deployed by the *carbonBaseline* approach results in an unacceptable increase in latency of FRs.

Considering results related to carbon footprint it may be concluded that deploying *compound* or *closestGreen* fitting schema results in a significant reduction of CO_2 emission in comparison to approaches employing *closest* schema. The most interesting considerations regard the relations between approaches deploying *compound* and *closestGreen* schema. Intuitively, *closestGreen* schema should provide the lowest carbon footprint as preference of *green* DCs is strict. However, in some cases requests that require little energy may occupy a great deal of network resources near *green* DCs, which may divert energy-hungry requests to other DCs. As a consequence, this may result in worse *green* energy utilization than in the *compound* fitting schema, which balances *green* DCs preference with *SaaS* request energy requirements and network resource utilization.

To sum up, the following conclusions about complete approaches may be drawn. The *proposed* approach is able to provide results comparable to approaches separately considered as a baselines with regard to the three indicators. Thus, the *proposed* approach extracts the most valuable characteristics from all of the reference approaches. It is able to significantly reduce both latency of FRs and carbon footprint of DCs without significant deterioration on network performance. Elasticity ensured by the thresholds and *penalty* parameter make it possible to retain network performance at the level comparable to the *blockingBaseline* approach. This level is significantly lower then the one obtained by the *carbonBaseline* approach. In the same time, use of the *compound* fitting schema allows the *proposed* approach to provide carbon dioxide emission comparable to both *carbonBaseline* and *carbonBaselineCompound* approaches, and on much lower level compared to the *blockingBaseline* and *latencyBaseline* approaches. Finally, introduction of the *latencyAware* policy ensures a significant reduction in latency of FRs, which is the main goal of this policy. The improvement obtained by the *proposed* and *latencyBaseline* approaches is the most substantial when comparing them to the *carbonBaseline* approach but is also significant when comparing them to the *blockingBaseline* and *carbonBaselineCompound* approaches.

VI. Conclusions

In this paper we investigated the possibility to handle fog related traffic in a latency-aware manner. For this purpose we introduced SDN-based WAN network support for energy-aware interplay between fog and cloud infrastructures. It was proved that both latency of fog traffic and the overall carbon footprint of DCs interconnected via an optical network may be reduced without a significant deterioration of the network performance. The studies conducted under numerous simulation scenarios show that it is possible to achieve the improvement by applying the proposed *latencyAware* policy to handle fog traffic and combining it with the *compound* schema for fitting anycast strategies to cloud services. The *proposed* approach is not only effective but also flexible as it may be adjusted to current network conditions and the expected user traffic. An additional advantage of the proposed solution is the fact that it may be easily implemented in an SDN controller due to very limited requirement for input data. As the degree of observed improvements vary depending on location of green DCs in a network as well as network topology, it might be interesting to investigate the static problem of optimal DC placement and input parameters selection. However, this remains a subject for future research.

Acknowledgment

This work was funded by the NCBR and CHIST-ERA ERA-Net SwiTching And tRansmission project.

References

[1] F. Bonomi *et al.*, "Fog Computing and Its Role in the Internet of Things," in *Proc. the First Edition of the MCC Workshop on Mobile Cloud Computing*, Helsinki, Finland, Aug. 2012, pp. 13–16.

[2] P. P. Jayaraman *et al.*, "CARDAP: A Scalable Energy-Efficient Context Aware Distributed Mobile Data Analytics Platform for the Fog," in *Conference on Advances in Databases and Information Systems*, Sep. 2014, pp. 192–206.

[3] P. Borylo *et al.*, "Green Cloud Provisioning Throughout Cooperation of a WDM Wide Area Network and a Hybrid Power IT Infrastructure." *Journal of Grid Computing*, 2015 (online).

[4] T. Mastelic and I. Brandic, "Recent Trends in Energy-Efficient Cloud Computing," *IEEE Cloud Computing Magazine*, vol. 2, no. 1, pp. 40–47, Jan. 2015.

[5] Z. Xu and W. Liang, "Operational cost minimization of distributed data centers through the provision of fair request rate allocations while meeting different user SLAs," *Computer Networks*, vol. 83, pp. 59 – 75, June 2015.

[6] J. Rzasa *et al.*, "Dynamic Power Capping for Multilayer Hybrid Power Networks," in *Proc. IEEE GLOBECOM*, Austin, TX, USA, Dec. 2014, pp. 2563–2569.

[7] M. Klinkowski and K. Walkowiak, "On the Advantages of Elastic Optical Networks for Provisioning of Cloud Computing Traffic," *IEEE Network*, vol. 27, no. 6, pp. 44–51, Nov. 2013.

[8] A. K. Mishra *et al.*, "Towards Characterizing Cloud Backend Workloads: Insights from Google Compute Clusters," *SIGMETRICS Perform. Eval. Rev.*, vol. 37, no. 4, pp. 34–41, Mar. 2010.

[9] C. Do *et al.*, "A proximal algorithm for joint resource allocation and minimizing carbon footprint in geo-distributed fog computing," in *Proc. ICOIN*, Jan. 2015, pp. 324–329.

[10] P. Borylo *et al.*, "Anycast Routing for Carbon Footprint Reduction in WDM Hybrid Power Networks with Data Centers," in *Proc. IEEE ICC*, Sydney, Australia, June 2014, pp. 3720–3726.

[11] ——, "Fitting Green Anycast Strategies to Cloud Services in WDM Hybrid Power Networks," in *Proc. IEEE GLOBECOM*, Austin, TX, USA, Dec. 2014, pp. 2633–2639.

[12] M. Gattulli *et al.*, "Low-Emissions Routing for Cloud Computing in IP-over-WDM Networks with Data Centers," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 1, pp. 28–38, Jan. 2014.