J Grid Computing (2016) 14:127–151
DOI 10.1007/s10723-015-9354-7

Green Cloud Provisioning Throughout Cooperation of a WDM Wide Area Network and a Hybrid Power IT Infrastructure

A Study on Cooperation Models

Piotr Borylo · Artur Lason · Jacek Rzasa · Andrzej Szymanski · Andrzej Jajszczyk

Received: 31 October 2014 / Accepted: 14 October 2015 / Published online: 24 October 2015

Abstract The paper focuses on cooperation between cloud and network operators, as well as on fitting particular routing strategies to various cloud services. Three cooperation models are presented, analyzed and compared in the paper: the proposed model and two widely used reference models. The main difference between the models is the set of information being exchanged between the involved parties. Additionally, we analyze the applicability of four fitting schemas for each considered model. It is shown that the proposed model, alongside with an appropriate fitting schema, is able to reduce the blocking probability of cloud services requests. At the same time, thanks to the use of *green* anycast strategies, it is able to significantly reduce carbon dioxide emission.

P. Borylo (✉) · A. Lason · J. Rzasa · A. Szymanski · A. Jajszczyk
AGH University of Science and Technology, Al. Mickiewicza 30, 30-059 Krakow, Poland
e-mail: borylo@agh.edu.pl

A. Lason
e-mail: lason@kt.agh.edu.pl

J. Rzasa
e-mail: rzasa@agh.edu.pl

A. Szymanski
e-mail: szymans@agh.edu.pl

A. Jajszczyk
e-mail: jajszczyk@kt.agh.edu.pl

Keywords Green cloud computing · Hybrid power · Carbon footprint reduction · Cloud architecture integration · All optical networks

1 Introduction

While considering the provisioning of cloud services one cannot neglect the importance of the network infrastructure which is necessary to ensure access to the IT resources. Due to the global character of modern cloud computing a great deal of effort is required not only to investigate networks within data centers, but also access, metro and wide area networks. However, cloud and network infrastructures are usually administered by different entities, with disparate optimization objectives. What is more, the involved parties are not usually willing to reveal the internal structure of their assets and often treat this information as confidential, which further complicates the case. As this behaviour may lead to suboptimal choices, we claim that in order to meet customers' needs and to effectively follow the concept of *greening the Internet* [1] some models of cooperation between network and IT infrastructure operators need to be introduced and thoroughly investigated.

The issue of integration of network infrastructure with IT resources was considered in numerous papers published in renowned journals and magazines. For example, authors in [2] provide a detailed definition of Software Defined Networking (SDN), its interfaces

as well as a list of its key attributes. One of the major features pointed in the paper is the ability of an SDN controller to cooperate with cloud orchestration software in order to quickly provision cloud applications and operate them in an automated manner. Thanks to the cooperation, networks may further support virtualization, migration of virtual machines and evolve towards application oriented networks. An additional effort is needed to bring those features to the Wide Area Networks (WANs). In [3], the authors present requirements, design alternatives and possible management architectures for a Wide Area SDN (WA-SDN). Again, communication of an SDN controller with cloud applications was mentioned as one of the most important use cases. In the wide area context cloud orchestration software may have to operate on more than one operators' IT infrastructure. Numerous middleware solutions aimed at using multiple infrastructures cooperating under different categories are available, e.g. [4]. Exhaustive survey about the existing solutions was provided in [5].

Not only scientific papers concern this issue. For example, B4 is an inter-datacenter private WAN connecting 12 Google's sites (data centers) across the planet [6]. B4 WAN, as well as networks within the sites, are controlled by a set of SDN controllers. All of them exchange information in order to fully utilize bandwidth available in the WAN and to control applications and site networks. However, only the largest cloud providers are able to build their own private WANs. Others must cooperate with various network operators to improve service quality and to reduce costs. This problem has been noticed by equipment vendors. For example, the concept of Seamless Virtual Cloud presented by Cisco at numerous conferences (e.g. IEEE Network Operations and Management Symposium 2014, Cisco Live - Milan 2014) assumes that IT resources and WAN infrastructure are seamlessly bounded together and are available for the tenants in an on-demand fashion. The presented framework provides an abstraction of relevant networking technologies in a WAN and in data centers. To sum up, it may be said that application driven wide area networks are becoming a part of the cloud instead of being just a way to access cloud applications and features.

The general idea of improving energy efficiency and decreasing negative impact on the natural environment is a hot topic and concerns almost every aspect of our lives. However, there are variuos reasons for introducing *green* mechanisms into IT and network infrastructure. The first, and the most natural reason is the will to decrease the negative impact of the infrastructure on the natural environment. Following the idea of *greening* the cloud also affects public relations and marketing. A cloud provider advertising itself as an environmental friendly one may improve its reputation and may seem to be more attractive for selected customers. However, it seems that the strongest driver is an economical one. Huge power requirements of the equipment directly translate to high operational costs related to electricity. On the one hand, the cost may be reduced by energy efficiency improvements. On the other hand, due to tax regulations introduced in numerous countries (e.g. United Kingdom, China), energy obtained from the renewable resources is cheaper. Thus, using renewable energy may also decrease the operational costs [7]. At the same time, network operators attract the attention of cloud providers by delivering *green* mechanisms and willing to cooperate in the field of energy efficiency. Such an additional value is currently especially important, due to the fact that the revenues from simple data transmission are decreasing. Thus, telecommunication operators must take advantage of emerging technologies, e.g. SDN, to provide more profitable services, including the *green* ones.

In the context of cloud computing, a great deal of previous works was focused on improving energy efficiency of data centers (DCs), as DCs are one of the main contributors to the overall energy consumption and carbon footprint of the ICT sector [8, 9]. The issue is of such importance, that the Green Grid Association, solely devoted to improve the efficiency of information technology and data centers throughout the World was founded [10]. The negative impact of DCs on the natural environment and the amount of energy they consume may be significantly decreased using some of the numerous available solutions. Among the other, the most popular approaches are as follows [7]:

- Placing DCs in locations where electricity is obtained from the sources which generate lower quantities of carbon dioxide and other greenhouse gases (e.g. hydropower, solar and wind power stations).
- Placing DCs in locations where Mother Nature may effectively help to improve energy efficiency,

for instance, where there is cold enough to use air side economizers that use outside air to chill a DC.
– Reusing heat being generated by the IT infrastructure and drawing upon recycled water rather than fresh one for cooling purposes.
– Optimizing airflows in DCs by, for example, isolating hot and cold airflows.

For the purpose of our research we assume that *green* DCs are those powered from renewable energy sources. This approach is a common practice of the largest IT players, like for example: Apple [11], Facebook [11, 12] or Google [11, 13]. However, our discussion and solutions may be easily adopted to other variants of *green* DCs, including the ones listed above.

Usually, it is not possible to power all DCs using renewable energy. *Green* energy is available only in selected locations and transmitting it from distant sites is not worthwhile due to high transmission loses. Thus, a hybrid power scenario, where a single cloud operator may have DCs powered from renewable energy mixed with those powered from conventional energy sources, is a very realistic one. In such a case it is reasonable to utilize those *green* DCs first, allowing the equipment in other DCs to enter the low energy consumption state, which reduces the overall carbon dioxide emission [14].

In this paper we focus on provisioning of cloud services throughout cooperation of the wide area all optical network interconnecting the sites equipped with the hybrid power IT infrastructure. We assume, that the network is controlled by the centralized controller consistent with the idea of the SDN. At the same time, we assume that the cloud orchestration software controls the cloud infrastructure and operates it in an automated manner. The architecture of such a system is described and systematized in this paper, followed by a discussion of different possible cooperation models between the WAN operator and the cloud provider. On the one hand, both entities are not willing to reveal their internal information. On the other hand, they want to cooperate in order to improve service quality and decrease carbon dioxide emission. Thus, we propose and evaluate a cooperation model allowing for improvement in performance of the infrastructure. Additionally, by joining cooperation models with *green* anycast strategies, carbon dioxide emission can be significantly decreased by utilizing *green* energy in the first place. As a side topic we investigate various anycast routing strategies as well as schemas for fitting those strategies to cloud services. To the best of our knowledge this work is the first one that analyzes deeply the problem of cooperation between network operator and cloud provider in order to provide cloud services with minimal carbon footprint. The issue is important as the network become an integral part of the cloud and, therefore, convergence of IT and telecommunication infrastructures is needed to effectively provide services to end users.

The rest of this paper is organized as follows. The next section provides a description and references to the related work. Section 3 contains a description of a cloud architecture being considered, including relationships between the network and cloud controllers. Additionally, it provides the rationale for numerous assumptions taken in the paper, as well as explanation of resource and requests models. Section 4 provides the formal problem statement, while Section 5 presents anycast strategies and schemas for fitting those strategies to different cloud services. Sections 6 and 7 describe cooperation models and simulation environment. Simulation results are presented in Section 8. Finally, Section 9 concludes the paper.

2 Related Work

Although the problem of reduction of greenhouse gases emission is a well known issue, only a limited set of papers deals with it in the context of networks supporting cloud architectures.

The problem of joint provisioning of network and IT resources was considered in [15, 16] and [17]. The authors of [15], apart from dimensioning issues, analyze three strategies for joint resource provisioning. Two of those strategies are conceptually similar to our reference models. The third one is an authors' original proposition to decrease the blocking probability by utilizing load balancing features in the network and the DCs, simultaneously. However, the paper focuses on performance issues only, while energy efficiency and impact on the natural environment are not investigated.

In [16] the authors address the problem of joint defragmentation of optical and computing resources in order to decrease the blocking probability. Reference cooperation models being considered in our work

have their equivalents in the context of defragmentation. Differences between our models and models described in [16] come from the fact that in the latter the holding time of each request is known in advance and the environmental aspect is not considered.

In [17], a control module is proposed to improve cooperation between cloud orchestration software and SDN controllers. The proposed idea of Software Defined Infrastructure (SDI) provides management of converged computing and networking resources. The SDI was implemented and assessed in a testbed. However, two important issues were not addressed: precise definition of information being exchanged between entities and the impact on the natural environment.

The authors in [18] proposed anycast heuristics to reduce CO_2 emission in IP-over-WDM networks connecting DCs under a dynamic traffic scenario. Their remarkable work was later extended in [19]. The proposed heuristics uses information about the availability of renewable energy, the amount of energy needed to serve a request in DC (the authors assume one specific service being provided) and network links occupancy. However, the authors assumed that network and cloud infrastructures are controlled jointly and without consideration of different cooperation models.

Utilization of anycast traffic to serve DC requests was also analyzed in [20]. Numerous modulation types and spectrum allocation strategies were investigated in elastic optical networks with regard to the blocking probability. However, the impact on the natural environment was not considered.

A strongly related issue was also addressed in [21]. The paper investigates a multilayer IP over WDM transport network. The most energy consuming parts of the infrastructure are network nodes switching the traffic in the electrical domain. A power capping technique, formerly used inside DCs, was adopted to effectively move those power hungry tasks to the network nodes powered from renewable resources.

Another approach was introduced in [22]. The authors proposed the migration of virtual machines (VMs) toward DCs powered from renewable energy sources. The incoming requests are directed to VMs through the network, therefore, the placement of VMs has a direct impact on traffic distribution in the network. Full cooperation between entities was assumed, but the exact range of information being exchanged between entities was not directly specified. VMs migration was also considered in [23]. VM allocation supported by dynamic voltage scaling and CPU pinning techniques was proposed in order to maximize resource utilization and energy efficiency while satisfying quality of service guarantees.

Numerous assumptions taken in this paper come from two of our previous works: [24] and [25]. In [24] we proposed three anycast strategies and compared them to three reference strategies with regard to network blocking probability and carbon dioxide emission. The main difference between [24] and this paper comes from the fact that in our previous work DCs had infinite computing resources and were assumed to offer only one type of cloud service. In [25] we introduced three types of cloud services and proposed schemas for fitting different anycast strategies to those services. The fitting was performed based on network resource requirements and energy consumption. The aim was to decrease carbon footprint as much as possible without a significant deterioration of network performance. But still, the only decision factor was associated with network resources utilization while the IT infrastructure had unlimited capacity. This assumption is no longer valid in this paper. Therefore, this paper is based on research conducted in both our previous works. However, it simultaneously extends those works and introduces original cooperation models between the entities.

3 Cloud and Network Architectures

One of the main paradigms of cloud computing is to offer numerous different services in an on-demand fashion. That is why incoming requests concerning different services are unpredictable and the cloud architecture must be investigated under a dynamic traffic scenario [26]. Rapid CRUD (create, read, update, delete) of resources is necessary in seconds. Software to automate provisioning and operation of a cloud is needed in order to meet the aforementioned requirement. Numerous cloud orchestration frameworks are available under open source as well as commercial licenses. A common purpose of those frameworks is to provide an abstraction of IT resources available in the cloud and to CRUD those resources in an automated manner. Exemplary solutions are: EC2 [27], OpenStack [28], OpenNebula [29], CloudStack [30]. Good scalability, independence from

underlaying hardware, versatility, modular architecture and user friendly interface are, among the others, the factors being considered while choosing a specific product. Another paradigm of cloud computing is to offer the same services in different DCs simultaneously. Thus, it is possible to choose one of many possible DCs to provide the requested service [26]. A corresponding routing scheme is called anycast. This assumption is realistic as numerous algorithms and solutions for data replication have already been proposed, e.g. [31].

Additionally, integration of a cloud with network infrastructure requires corresponding integration of cloud orchestration with network control. The aforementioned idea of an SDN perfectly suits the requirement as in the SDN architecture the control plane is separated from the switching plane. Such a control plane is realized in software, denoted as an SDN controller and deployed as an entity separate from the network devices. The controller communicates with hardware (OpenFlow enabled devices) and virtual devices through the OpenFlow Protocol in order to deliver forwarding rules [32]. Therefore, communication between the centralized controller and network devices must be ensured to allow proper operation of the Software Defined Network. Simultaneously, the SDN controller may be able to communicate through the Northbound-API with the aforementioned cloud orchestration software [2]. Examples of SDN controllers are: OpenDayLight [33], Floodlight [34], Beacon [35], Big Cloud Fabric Controller [36]. The relationships in the considered architecture are presented in Fig. 1.

Modeling of IT resources within a DC is a separate problem. It is common for a DC to have heterogeneous physical machines, with different amounts of RAM and different CPUs. However, this heterogenous infrastructure may be abstracted using cloud orchestration software responsible for resource allocation, virtualization and consolidation. Furthermore, numerous task schedulers may be used in order to effectively allocate such DC resources, e.g. [37, 38] or [39]. Therefore, IT resources available in each DC may be represented as a single value reflecting available RAM and computing power. A specific amount of those resources is consumed while serving a DC request and is released afterwards. A third kind of DC resources that needs to be considered is storage space, which is commonly provided using dedicated equipment. A shortage of storage space is generally unacceptable in a DC. Additionally, storage resources are not usually released after tearing down a request, further complicating the case. Thus, we decided to treat the storage resources as unlimited and omit them from abstraction.

In addition to modelling IT resources within a DC, it is necessary to model resource consumption of a particular request. Standardization of job submission process in cloud infrastructure is an emerging and still open issue (see [40]). A common practice is to divide all requests into a fixed number of classes. Such an idea of categorization of cloud service requests is applied, for example, by Google [41]. Three types of cloud services are being considered in the paper: *PaaS* (Processing as a Service), *StaaS* (Storage as a Service) and *SaaS* (Software as a Service). Those types

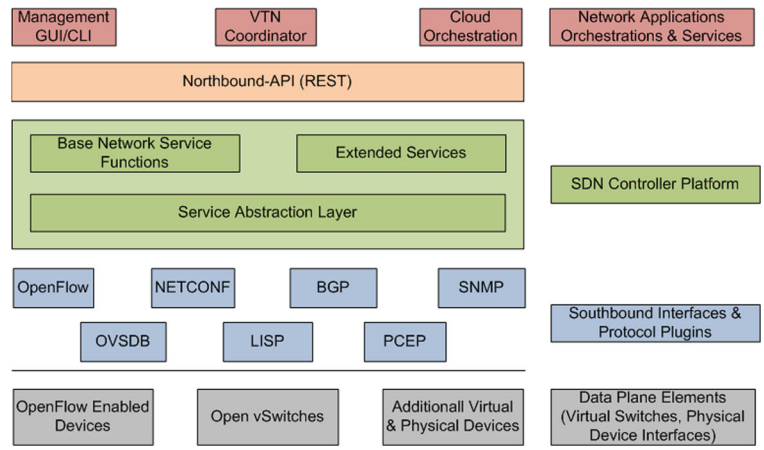

Fig. 1 Considered architecture (based on [33]), VTN: Virtual Tenant Network, OVSDB: Open vSwitch DataBase Protocol, LISP: Locator/Identifier Separation Protocol, BGP: Border Gateway Protocol, PCEP: Path Computation Element Communication Protocol, SNMP: Simple Network Management Protocol

of services have been described in detail in [42], but we focus solely on their requirements for network and data center resources as well as their energy consumption model. These three properties are crucial for appropriate fitting of anycast strategies to cloud service types and for proper evaluation of the proposed cooperation models. The presented assumptions are consistent with our previous work [25].

PaaS Refers to a group of services in which a cloud customer is able to process computationally demanding tasks in a DC. It means that a *PaaS* request occupies network resources only for the time needed to send task related data to the DC, and consumes huge amount of energy and IT resources inside the DC. According to the model proposed in [19], energy consumption of a DC handling a *PaaS* task may be expressed by the following equation: $E_{proc} = 1.5 \cdot T_{proc} \cdot P_s$, where P_s is the power consumption of the server (355 W), T_{proc} is the processing time (in hours) and the scaling factor represents the power consumption overhead (e.g., air conditioning). Thus, following considerations presented in [19], an exemplary service of encoding a video file requires $E_{proc} = 100$ Wh to process 10 Gb (1.25 GB) of data. The processing time usually exceeds the time of network resource occupancy, but if we assume that the whole task may be computed in a parallel way in the time equal to the network resource occupancy we get $E_{proc} = 36$ kW/(Gb/s). It must be emphasized that this assumption does not affect the total amount of the consumed energy, it only decreases the time in which the energy was consumed.

StaaS Represents cloud services that allow customers to store their data in the cloud. Thus, network resources are occupied for the time needed to transfer the data. Energy and IT resources needed to handle a *StaaS* request are related only to the Input/Output memory operations (requiring some CPU and RAM resources) and are marginal. This assumption is valid as, in general, power consumption of storage equipment is independent of the amount of currently stored data. The average *StaaS* energy consumption suggested in [18] is equal to 0.141 W/(Gb/s) and this value is reasonable from our point of view.

SaaS Spans numerous different cloud services, therefore, properties of *SaaS* requests are very hard to define. Some exemplary cloud services assigned to the *SaaS* group are: dedicated web application, virtual desktop, virtual machine with dedicated software, etc. As a result, a *SaaS* request may, for example, occupy network resources for a very long time while consuming only a little amount of IT resources and, subsequently, energy in a DC (as in case of virtual desktop service). However, at the same time, another *SaaS* request may also hold network resources for a similar time (required to make the use of dedicated software) and consume energy and IT resources comparable to the *PaaS* service by running computationally intensive tasks using the aforementioned dedicated software. The authors in [18] proposed an estimate of the average energy consumption in a DC for a *SaaS* request equal to 2.7 W/(Gb/s). This value is proper for *SaaS* services with very low energy requirements. Knowing a variety of *SaaS* services it seems reasonable to analyze different cases of energy consumption. Therefore, average energy consumption values assumed in our work are 1, 2.7, 5.4 and 10 kW/(Gb/s), i.e., much higher than the value proposed in [18], but still lower than for the *PaaS* services. However, the solution proposed in this paper is able to adjust even to the value suggested in [18].

A separate issue concerns the proportions in which the aforementioned services contribute to the total DC traffic. We assume that those proportions may vary in a great deal. In modern clouds, *PaaS* is probably the least popular as *PaaS* requires the cloud operator to strongly adjust to user needs. Also, security issues may decrease attractiveness of *PaaS*. On the other hand, *SaaS* appears to be the most frequently used as *SaaS* spans numerous services and offers them to clients in the most flexible way (for both cloud operators and users).

4 Problem Statement

The problem investigated in this paper may be briefly described as dynamic routing and wavelength assignment (RWA) merged with cloud services provisioning. A corresponding static problem that may be formulated is NP-hard. The complexity results from the fact that the RWA problem, which is a part of the considered problem, was proved to be NP-complete (for the proof see [43]). As we focus on the dynamic problem the static problem will not be formulated and

investigated in greater detail. The wide area network is assumed to be an automatically switched all optical network interconnecting hybrid power IT infrastructure. Optical data transmission is usually indicated as the most appropriate to provide cloud computing services. Elimination of electrical layer from the network architecture ensures low energy consumption of the network nodes. DCs are associated with selected network nodes. Network and IT resources are administrated by separate entities.

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. Each fiber $e \in E$ carries a fixed number of wavelengths indexed using w. Symbols $e_w = 1$ and $e_w = 0$ denote unoccupied and occupied wavelength w on link e, respectively. V_{DC} describes the set of nodes being associated with the DCs. DCs powered from renewable energy sources are called *green* and network nodes associated with the *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 g and b lower indices will always denote *green* and *brown*, respectively. Nodes that are not associated with any DC are termed client nodes, and denoted by V_C. The following relations are met:

$$V_{gDC} \cap V_{bDC} = \emptyset \tag{1}$$

$$V_{gDC} \cup V_{bDC} = V_{DC} \tag{2}$$

$$V_{DC} \cap V_C = \emptyset \tag{3}$$

$$V_{DC} \cup V_C = V \tag{4}$$

For each data center $d \in V_{DC}$, c_d and r_d denote total amount of resources and resources currently available, respectively. C_{DC} and R_{DC} describe the sets containing c_d and r_d for all $d \in V_{DC}$, respectively.

The centralized SDN controller has full knowledge about the network topology and its state, including information about existing lightpaths, the location of DCs (V_{DC}) and their power source. It means that for any set of DCs $d \in D \subseteq V_{DC}$ the SDN controller is able to distinguish *green* ones, $d \in D_g$, from *brown* DCs $d \in D_b$. At the same time, cloud orchestration software has knowledge about resources available in each DC and cloud operator's preferences.

Dynamic RWA algorithms operate on a lightpath request (*LR*). We assumed two types of *LRs*:

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 $d \in D \subseteq V_{DC}$. Anycast *LR* are used to handle DC traffic. When a cloud service request (*CR*) arrives to the cloud orchestration software it is further passed to the SDN controller as anycast *LR*.

From the SDN controller point of view, a path X between s and d must be found to serve any *LR*. Path X is composed of adjacent links and each fiber x being part of the path X, $x \in X$, must meet the wavelength continuity constraint (WCC):

$$\exists w \quad \forall x \in X \quad x_w = 1 \tag{5}$$

where w is a wavelength that may be chosen to serve the *LR* over the path X.

Unicast *LRs* are served using alternate routing with three link-disjoint, precomputed paths. The details of the algorithm for unicast *LRs* may be found in [44]. In this work unicast lightpath requests are used solely for handling the background traffic which does not interact with the cloud orchestration software. As the cloud infrastructure is not involved in serving unicast *LRs*, the path between s and d must only meet WCC (5).

As anycast scheme may be defined as *one-to-one-of-many* routing, for each anycast *LR* the RWA 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. The strategies for anycast *LRs* handling are described in detail in Section 5. The presented strategies use the $LAC(s, d)$ operation, which denotes lightpath availability check between the source node s and the destination node d. This operation is also based on alternate routing and checks if any of the three precomputed link-disjoint paths between s and d meets WCC. The result of $LAC(s, d)$ operation is the length (in the hop manner) of the shortest path meeting WCC. If WCC cannot be met by any of the precomputed paths, $LAC(s, d)$ returns infinity.

As mentioned earlier, each anycast *LR* is associated with a cloud request, *CR*, which consumes IT resources within a DC associated with the chosen node d. Thus, the following constraint must be met:

$$r_d \geq LR_r \tag{6}$$

where, LR_r is the amount of resources required to serve a cloud service request associated with the particular *LR*. After tearing down the request, both network as well as IT resources are released. We assume that cloud service requirements for IT resources are proportional to their energy consumption. Such an assumption seems to be reasonable in order to provide a consistent model of cloud services.

A subset of DCs meeting the constraint (6) is denoted as V_{rDC}. Again, the V_{rDC} subset may be further divided into two subsets: V_{rgDC} and V_{rbDC}, corresponding to *green* and *brown* DCs. The following relations exist:

$$V_{rgDC} \cap V_{rbDC} = \emptyset \tag{7}$$

$$V_{rgDC} \cup V_{rbDC} = V_{rDC} \tag{8}$$

$$V_{rDC} \subseteq V_{DC} \tag{9}$$

$$V_{rgDC} \subseteq V_{gDC} \tag{10}$$

$$V_{rbDC} \subseteq V_{bDC} \tag{11}$$

Each *CR* and a corresponding anycast *LR* is used to handle requests assigned to one of the three aforementioned types of cloud services: *PaaS*, *StaaS* or *SaaS*. For the purpose of simplicity, anycast *LRs* handling *PaaS* will be denoted as *PaaS* requests. The same nomenclature is applied to *StaaS* and *SaaS*. Each *LR* and *CR* carry information about the assignment. Thus, the cloud orchestration software and the SDN controller are able to use this information and perform actions adjusted to different services.

In order to estimate the carbon footprint of the considered infrastructure we assume 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.

5 Anycast Strategies and Fitting Schemas

In this section we present detailed definitions of all anycast strategies used in subsequent studies. It is important to note that the algorithms of the strategies may be easily implemented in a network controller. They utilize only fundamental control information that is to a great extent already present in the common unicast network controller. The first strategy, *closest*, is a well known, reference anycast strategy. The remaining two of them: *closestGreen* and *closestGreenWithPenalty* are the strategies proposed in our previous work [24] and aimed at minimizing carbon footprint.

5.1 Anycast Strategies

In the *closest* strategy the network controller performs $LAC(s, d)$ for all $d \in D$. The closest (in the hop manner) reachable destination d is chosen and the lightpath is established between s and d. If none of $d \in D$ is available then the *LR* is rejected. Pseudocode 1 formalizes the description of the strategy.

Strategy 1 closest [24]

```
Input: (s, D)
d ← null
dist ← ∞
for all i ∈ D do
    if LAC(s, i) < dist then
        d ← i
        dist ← LAC(s, i)
    end if
end for
if d ≠ null then
    Establish lighpath between s and d
    return d
else
    return null
end if
```

In the *closestGreen* strategy the network controller performs operations analogous to the *closest* strategy for all $d \in D_g$, where D_g is a subset of *green* DCs within D. If none of $d \in D_g$ is available then the network controller repeats the same operation for all *brown* DCs within D, $d \in D_b$. If none of $d \in D$

are available then the *LR* is rejected. Pseudocode 2 formalizes the description of the strategy.

Strategy 2 closestGreen [24]

Input: $(s, D) = ((s, \{D_g, D_b\})$

```
1: d ← null
2: dist ← ∞
3: for all i ∈ D_g do
4:     if LAC(s, i) < dist then
5:         d ← i
6:         dist ← LAC(s, i)
7:     end if
8: end for
9: if d ≠ null then
10:    Establish lighpath between s and d
11:    return d
12: else
13:    for all i ∈ D_b do
14:        if LAC(s, i) < dist then
15:            d ← i
16:            dist ← LAC(s, i)
17:        end if
18:    end for
19:    if d ≠ null then
20:        Establish lighpath between s and d
21:        return d
22:    else
23:        return null
24:    end if
25: end if
```

The *closestGreenWithPenalty* strategy works analogously to the *closest* strategy but with one significant difference. The distance from s to $d \in D_b$ is multiplied by the *penalty* factor, where the *penalty* is an SDN controller internal parameter. The *closestGreenWithPenalty* strategy with $penalty = 1.0$ is equivalent to the *closest* strategy. Pseudocode 3 formalizes the description of the strategy.

Strategy 3 closestGreenWithPenalty [24]

Input: $(s, D) = (s, \{D_g, D_b\})$

```
1: d ← null
2: dist ← ∞
3: for all i ∈ D do
4:     if i ∈ D_b then
5:         if LAC(s, i) · Penalty < dist then
6:             d ← i
7:             dist ← LAC(s, i) · Penalty
8:         end if
9:     else if i ∈ D_g then
10:        if LAC(s, i) < dist then
11:            d ← i
12:            dist ← LAC(s, i)
13:        end if
14:    end if
15: end for
16: if d ≠ null then
17:    Establish lighpath between s and d
18:    return d
19: else
20:    return null
21: end if
```

5.2 Fitting Schemas

In general, there are many possible choices regarding the fitting of an anycast schema to a particular cloud service type. However, based on properties of both cloud service types and anycast strategies we propose the following fitting schema which was thoroughly investigated in [25]. *PaaS* requests, which have the highest energy requirements, should be handled by the *closestGreen* strategy, which strictly favorises *green* DCs over *brown* DCs, at a cost of increased average lightpath length and increased network resource utilization. *StaaS* requests, which have the lowest energy requirements, should be handled by the *closest* strategy, which 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*, as the most undefined 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 carbon footprint reduction. In subsequent studies this solution will be denoted as the *compound* fitting schema to reflect the usage of different anycast strategies.

The *closest*, *closestGreen* and *closestGreenWithPenalty* fitting schemas are opposed and describe cases when all types of cloud services are handled

using the same anycast strategy, *closest*, *closestGreen* and *closestGreenWithPenalty*, respectively.

6 Cooperation Models

Models presented in this section concern the cooperation between the cloud operator (cloud orchestration software) and the network provider (an SDN controller). The models are applicable to cloud service requests which are served by both entities.

Some basic rules concerning the cooperation are common for all the models. A cloud service request, *CR*, arrives to cloud orchestration software and is further passed to an SDN controller as an anycast *LR*. In some cases, additional information is attached to the anycast *LR* being passed. Thus, cloud orchestration software provides some input for the SDN controller. The SDN controller chooses the destination d for the anycast *LR* and sets up the lightpath between s and d. Then, cloud orchestration software receives information about the selected d and tries to reserve resources and provision the requested service in that node. If the pointed d does not have enough resources, the cloud service request is blocked and the associated lightpath is automatically torn down by the SDN controller.

The difference between cooperation models concerns mainly the amount of information being exchanged between the cloud provider and the network operator. The less information is exchanged the more both entities are comfortable with cooperation. However, the cooperation is needed to effectively provide cloud services and reduce greenhouse gases emission. Thus, exchange of information through clearly defined interfaces is a compromise in this situation. Precisely defined interfaces ensure full control over the information being exchanged and therefore, improve the comfort of cooperation.

All the analyzed cooperation models have been presented in a formal way using pseudocodes. Each action described in them is performed by the cloud orchestration software with the exception of actions within *SDN*(*arguments*) blocks. The *SDN* block includes actions performed by the SDN controller, while *arguments* denotes the information passed to the controller by the cloud orchestration software. The SDN controller uses the $arwa(s, D)$ operation which denotes invoking the anycast strategy fitted to the cloud service associated with current *LR*. The s and D are the inputs for the anycast strategy: the source node and the set of possible destinations, respectively. At the same time, function name *arwa* denotes the anycast strategy invoked by the $arwa(s, D)$ operation. The chosen fitting schema associates each *LR* with an anycast strategy and *LR* and *CR* carry this information. Additionally, there are some operations that are specific to a particular anycast strategy. They are a part of the model and are explained in the pseudocode comments.

6.1 The Overlay Model

Pseudocode 4 formalizes the description of the model. In this model, the cloud orchestration software simply passes an anycast *LR* to the SDN controller and does not attach any additional information (1st line). Therefore, the *overlay* model assumes no cooperation between the network operator and the cloud provider. As the SDN controller does not have any knowledge about resources available in DCs it may choose a destination d by simply running anycast strategy on V_{DC} set (2nd line), but the chosen destination might not be able to handle the request. Subsequently, cloud orchestration software must examine the chosen d in terms of resource availability (conditional statement in 4th line). If the amount of available resources is sufficient the cloud request is handled by destination d (lines 5 and 6). The main drawback of this model is that the SDN controller may consequently choose destinations d which do not have sufficient resources. In such a case request is rejected (8th line). It may increase the blocking probability of cloud service requests despite the fact that it would be possible to handle rejected requests in a different DC.

Cooperation model 4 overlay

Input: *CR*
1: **SDN** (*LR*)
2: $d \leftarrow arwa(s, V_{DC})$
3: **end SDN**
4: **if** $d \neq null$ AND $r_d \geq LR_r$ **then**
5: Reserve resources in d
6: $r_d \leftarrow r_d - LR_r$
7: **else**
8: Reject request
9: **end if**

6.2 The Augmented Model

The *augmented* cooperation model is described using Pseudocode 5. In this model the cloud provider informs the SDN controller about the set of DCs that have enough resources to serve the request. This V_{rDC} set is created in the *for* loop between 2nd and 6th line and is further passed to the SDN controller, which uses it as an input for the anycast strategy (7th and 8th lines, respectively). Such a basic cooperation eliminates the main drawback of the *overlay* model, as only DCs with enough available resources are considered by the SDN controller. The *augmented* model is expected to significantly decrease the total blocking probability in comparison to the *overlay* model. The requests are rejected only if controller is not able to find a lightpath to any of $d \in V_{rDC}$ which is expressed by a conditional statement in lines from 10th to 15th. However, the model's decisions are based mainly on the network topology and state while the state of the cloud infrastructure has secondary impact on them.

Cooperation model 5 augmented

Input: *CR*

1: $V_{rDC} \leftarrow \emptyset$
2: **for all** $d \in V_{DC}$ **do**
3: **if** $r_d \geq LR_r$ **then**
4: $V_{rDC} \leftarrow V_{rDC} \cup \{d\}$
5: **end if**
6: **end for**
7: **SDN** (LR, V_{rDC})
8: $d \leftarrow arwa(s, V_{rDC})$
9: **end SDN**
10: **if** $d \neq null$ **then**
11: Reserve resources in d
12: $r_d \leftarrow r_d - LR_r$
13: **else**
14: Reject request
15: **end if**

6.3 The Peer Model

In this cooperation model, the cloud orchestration software provides the following information to the SDN controller: LR, V_{rDC}, *preferenceMode*, dc_{th}, C_{DC} and R_{DC}. Thus, the SDN controller has knowledge about DCs that have sufficient IT resources (V_{rDC}), the preference criteria (*preferenceMode* and dc_{th} will be further explained), as well as the total amount of resources in each DC (C_{DC}) and the amount of resources currently available in each DC (R_{DC}). In the first step, the SDN controller applies the anycast strategy to the set of possible destinations $D = V_{rDC}$, searching for the destination with enough resources to provide the service. Thus, the d_{real} obtained this way denotes the destination selected by the SDN controller assuming no preference of any DC.

A general idea of further steps is to limit the increase of the average lightpath length resulting from cloud operator's preferences. Therefore, the differences in distances between each $d \in V_{rDC}$ and d_{real} are computed and normalized by the $G(V, E)$ diameter. Subsequently, the normalized differences are compared with the net_{th} threshold which expresses the increase in the average lightpath length acceptable by the network operator. If net_{th} is not exceeded then preferences of the cloud operator expressed by three parameters: r_d (already introduced), dc_{th} and *preferenceMode* are considered. The dc_{th} is a threshold, expressed as a fraction of the total DC capacity (c_d). The *preferenceMode* may be assigned one of the two values: *mostOccupied* and *leastOccupied*. The *mostOccupied* mode denotes preferring DCs with available resources less than the assumed threshold which means that $d \in V_{DC}$ must met the $r_d \leq dc_{th} \cdot c_d$ constraint to be preferred. Analogously, *leastOccupied* mode denotes preferring DCs with available resources greater than the assumed threshold, thus $d \in V_{DC}$ must meet the $r_d \geq dc_{th} \cdot c_d$ constraint to be preferred.

The cooperation between the cloud operator and the network provider is quite tight and if only the net_{th} threshold is not exceeded then cloud provider's preferences are more significant. In this case, the most or the least occupied DC is selected for the *mostOccupied* or the *leastOccupied* modes, respectively. However, the cloud operator has to reveal internal information about its preference criteria and resources available in each of DCs. Pseudocode 6 formalizes the description of the model. The time complexity of the listed algorithm is considered to be O(N) as there are three independent loops over selected network nodes and the constant is neglected in the big O notation.

To sum up, the presented cooperation models differ from each other with regard to data being exchanged between a cloud operator and a network provider.

Cooperation model 6 Peer

Input: CR

$V_{rDC} \leftarrow \emptyset$
for all $d \in V_{DC}$ **do**
  if $r_d \geq LR_r$ **then**
    $V_{rDC} \leftarrow V_{rDC} \cup \{d\}$
  end if
end for
SDN (LR, V_{rDC}, $preferenceMode$, dc_{th}, C_{DC}, R_{DC})
  $d \leftarrow null$
  $d_{real} \leftarrow arwa(s, V_{rDC})$ ▷ Real case without preferences
  if $d_{real} \neq null$ **then**
    $dist_{real} \leftarrow LAC(s, d_{real})$
    if $arwa = closestGreenWithPenalty$ AND $d_{real} \in V_{bDC}$ **then**
      $dist_{real} \leftarrow dist_{real} \cdot Penalty$ ▷ Distance to *brown* node must be penalized
    end if
    for all $i \in V_{rDC}$ **do**
      $dist_{tmp} \leftarrow LAC(s, i)$
      if $arwa = closestGreenWithPenalty$ AND $i \in V_{bDC}$ **then**
        $dist_{tmp} \leftarrow dist_{tmp} \cdot Penalty$ ▷ Distance to *brown* node must be penalized
      end if
      if $(dist_{tmp} - dist_{real})/diameter \leq net_{th}$ **then**
        if $arwa = closest$ OR ▷ Only *green* DC may preempt *green* DC
        $d_{real} \in V_{bDC}$ OR
        ($d_{real} \in V_{gDC}$ AND $i \in V_{gDC}$) **then**
          if $preferenceMode = mostOccupied$ **then**
            if $r_d \leq dc_{th} \cdot c_d$ **then**
              if $d = null$ OR $r_i \leq r_d$ **then**
                $d \leftarrow i$
              end if
            end if
          end if
          if $preferenceMode = leastOccupied$ **then**
            if $r_d \geq dc_{th} \cdot c_d$ **then**
              if $d = null$ OR $r_i \geq r_d$ **then**
                $d \leftarrow i$
              end if
            end if
          end if
        end if
      end if
    end for
    if $d = null$ **then**
      $d \leftarrow d_{real}$
    end if
  end if
end SDN
if $d \neq null$ **then**
  Reserve resources in d
  $r_d \leftarrow r_d - LR_r$
else
  Reject request
end if

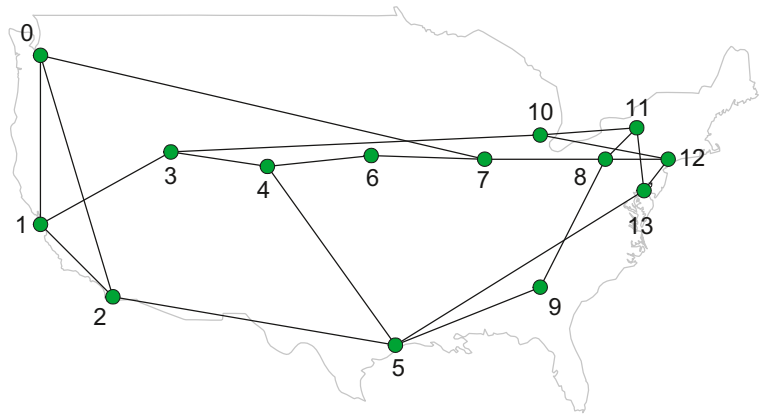

Fig. 2 The topology of the NSF network

On the one hand, the cloud operator and the network provider prefer to reveal as little information as possible. However, they also want to effectively provide services and reduce CO_2 emission. Therefore, a kind of compromise is necessary. Well defined interfaces for information exchange between those two entities and a precisely specified range of exchanged information seem to be the solution.

The cooperation models described in this section are just initial solutions and not the complete frameworks. In order to create the latter, a well defined protocol for communication between the cloud orchestration software and an SDN controller is first needed. A Northbound-API seems to be able to perform this function. However, according to [2] it is not standardized. For the simulation purposes lack of the communication protocol between cloud orchestration software and SDN controller was not the case. Both entities were implemented within a single application instance and exchange the information through software variables within this instance. Furthermore, the implemented centralized controller is authors' original implementation designed to cooperate with simulator environment and not the instance of any existing SDN controller. Any kind of Northbound-API will be indispensable in realistic distributed scenarios where SDN controller and cloud orchestration software are fully separated.

7 Simulation Environment

To asses the presented fitting schemas and cooperation models several simulations were performed in two reference networks: the 14 node National Science Foundation (NSF) network shown in Fig. 2 and the 21 node Italian Mesh Network (*ItalyNet*) shown in Fig. 3. Detailed parameters of the topologies are provided in Table 1. Each physical link is composed of two fibers, one in each direction. Each fiber carries 80 wavelengths transporting data at a rate of 10 Gb/s. Additionally, there are no wavelength converters in the

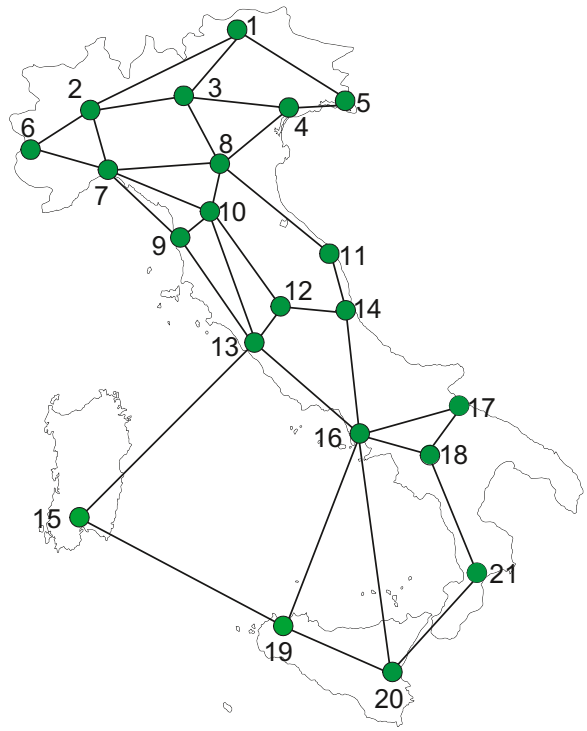

Fig. 3 The topology of the ItalyNet

Table 1 Topology parameters for investigated networks

Parameter	NSF	ItalyNet
Nodes	14	21
Links	21	34
Connectivity index (links/nodes)	1.5	1.62
Average nodal degree	3	3.24
Diameter [Hop]	3	6
No. of *green* DCs	2	3
No. of *brown* DCs	3	3

network and the wavelength assignment is done using the first-fit scheme.

In the NSF network five DCs were deployed. Their location was chosen based on results presented in [45] ($V_{DC} \in \{2, 4, 7, 9, 11\}$). In consecutive simulation studies different pairs of DCs from the mentioned set were assumed to be green. Analogously, six DCs were deployed in *ItalyNet* and were located following the assumption made in [18] ($V_{DC} \in \{1, 7, 11, 13, 15, 20\}$). Numerous sets of three DCs were assumed to be green, in this assumption we are also consistent with [18].

The offered traffic is composed of two traffic types: background and DC traffic. The background traffic consists of unicast *LRs* and is generated based on a uniform traffic matrix. The DC traffic consists of cloud service requests generated only by $s \in V_C$ (with the same intensity for each s) to a single anycast group $D = V_{DC}$.

The request holding time (*ht*) is exponentially distributed. For the background traffic the mean value of *ht* is always equal to 10 s. In case of DC traffic *ht* depends on service type: for *PaaS* and *StaaS* the mean value is equal to 10 s, while for *SaaS*, it is a variable parameter, where the considered values are 100, 360 and 1800 s. For subsequent considerations about interarrival time (*iat*) let us assume that the mean value of *ht* for *SaaS* is equal to 360 s.

In the base case each request is a unidirectional request with exponentially distributed *iat* and the mean intensity of 0.3795 (background traffic), 1.0956 (*PaaS* and *StaaS*) and 0.0297 (*SaaS*) requests per second for the NSF network and 0.132 (background traffic), 1.1778 (*PaaS* and *StaaS*) and 0.0319 (*SaaS*) requests per second for the *ItalyNet*. For *SaaS* and *ht* values other than 360 s, *iat* is scaled so that the total amount of generated traffic (product of mean values of *iat* and *ht*) remains constant.

As a consequence, the total network load in the base case is $14 \cdot 13 \cdot 0.3795 \cdot 10 + 9 \cdot 1 \cdot (1.0956 \cdot 10 + 1.0956 \cdot 10 + 0.0297 \cdot 360) = 690.69$ Erl $+ 293.44$ Erl$= 984.13$ Erl for the NSF network and $21 \cdot 20 \cdot 0.132 \cdot 10 + 15 \cdot 1 \cdot (1.1778 \cdot 10 + 1.1778 \cdot 10 + 0.0319 \cdot 360) = 554.40$ Erl $+ 525.60$ Erl $= 1080.00$ Erl for the *ItalyNet*. The total traffic in each network is a sum of two components. In the following explanation we will use numbers corresponding to the NSF network. The first component reflects the amount of background traffic generated by each node (14) to every other node (13). The traffic between each pair of nodes has a given mean intensity (0.3795) and a given mean holding time (10). The second component describes the amount of DC traffic generated by each client node (9) to a single anycast group (1) with the assumed mean intensities (1.0956 and 1.0956 and 0.0297) and, respectively, holding times (10 and 10 and 360) for different cloud services.

In order to asses the assumed scenarios under varying loads we multiplied the mean intensity of the DC traffic by a scaling factor ranging from 1.0 to 2.35 (NSF) and from 1.0 to 1.63 (*ItalyNet*). The mean *iat* for background traffic and the mean *ht* of all traffic types remained unchanged. The resulting contribution of DC traffic to the overall traffic varies from 30 to 50 % in the NSF network and from 49 to 60 % in the *ItalyNet*. In this way we try to adjust traffic conditions to the predictions presented in [46]. However, more traffic classes were provided in [46] and contribution of DC traffic to the overall traffic is constant. Therefore, we cannot directly apply traffic proportions found in this paper to our work.

For each traffic type (unicast, *PaaS*, *StaaS* and *SaaS*) separate pseudo-random number generators were used to generate interarrival times and holding times. A Mersenne-Twister generator was used as the one with good randomness in up to 623 dimensions and period length of $2^{19937} - 1$. The generator implementation was provided by the Boost Random library.

In the aforementioned considerations the three types of services contribute to the total DC traffic in 1:1:1 proportions (*PaaS*:*StaaS*:*SaaS*). However, in subsequent studies also 1:2:4 and 1:5:25 proportions were investigated in order to reflect considerations provided in Section 3. Changing the proportion between service types was achieved by

scaling *iat*, while *ht* for each service type remained unchanged.

As already mentioned in Section 4, we assume that requirements for IT resources of cloud services are proportional to their energy consumption. We also assume that in *green* DCs all IT resources are powered from renewable energy sources. Therefore, a *PaaS* request consumes 250000 units of IT resources, a *StaaS* request consumes 1 unit of IT resources and a *SaaS* request consumes 7000, 18750, 37500 or 67500 units of IT resources for different values of energy consumption, respectively. The capacity of each DC (c_d) is assumed to be the same for all $d \in V_{DC}$ and is adjusted to the investigated scenario. Its value depends on the contribution of cloud services to the overall DC traffic and the assumed consumption of IT resources by *SaaS* requests. The exact value was experimentally scaled to the network resources with the aim to make neither IT nor network resources a single bottleneck of the architecture. Instead, we aimed to get a situation where both IT and network resources are approximately equally utilized and both get saturated simultaneously when DC traffic increases. Table 2 presents the capacity of a single DC for each considered case.

To sum up, the architecture was investigated under a heavy traffic scenario. It is a realistic assumption only during unexpected traffic peaks. During low traffic hours there is no need for any online optimization technique since an overprovisioned architecture should be able to efficiently handle all the requests.

8 Results

Numerous simulation scenarios were investigated as many parameters are variable. However, only illustrative and representative simulation results will be presented in a graphical form. Additionally, during the preliminary assessment of the results we found out that applying different values of the *ht* parameter for the *SaaS* service resulted in similar network behavior. Thus, we present results only for the mean value of the *ht* parameter for the *SaaS* service equal to 360 s and omit the remaining ones. This conclusion is consistent with our previous work [25].

To asses the fitting schemas and the cooperation models, two indicators were used. The first 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 ($brownKiloWatts/(Gb/s)$). Thanks to that normalization we obtain comparable results under different traffic loads. The second indicator is the blocking probability, calculated as the ratio of rejected requests to all requests. The simulation results were obtained using the OMNeT++ simulator (implemented in C++, for details see [47]) and evaluated at the 0.95 confidence level using the batch means method. The

Table 2 IT resources in DCs

Service types' proportions	SaaS energy consumption	SaaS resources consumption	DC resources
1:1:1	1	7000	10639289
	2.6	18750	11125713
	5.4	37500	11901923
	10	67500	13143858
1:2:4	1	7000	4932293
	2.6	18750	5766164
	5.4	37500	7096810
	10	67500	9225842
1:5:25	1	7000	1702673
	2.6	18750	2879507
	5.4	37500	4757434
	10	67500	7762117

number of independent observations was chosen to be 31 as this assumption satisfies the rule according to which a sample is considered large. Therefore, based on the Central Limit Theorem, distribution of the sample mean is approximately normal. Additionally, in order to dispose the data gathered during a transient phase the warm-up period was estimated and validated based on the Central Limit Theorem. All results were presented with regard to the aforementioned scaling factor of DC traffic as a measure of traffic intensity.

For the sake of clarity the following systematization of provided results was introduced. Section 8.1 demonstrates the way in which values were assigned to the aforementioned input parameters. Section 8.2 presents the results obtained by different fitting schemas assuming a specific cooperation model. Based on results presented in Section 8.2 we were able to choose the best fitting schema for each cooperation model. Such a choice allows, finally, for a fair comparison of different cooperation models, which is provided in Section 8.3.

8.1 Input Parameters Values

In order to use the *closestGreenWithPenalty* strategy the value of the *penalty* parameter needs to be determined. Similarly, in order to use the *peer* cooperation model the values of *preferenceMode*, dc_{th} and net_{th} input parameters need to be specified. In both cases we assume that the operator is minded to slightly deteriorate network performance in order to reduce carbon footprint. Based on that assumption the required input parameters were determined using the methodology described below.

We assume that for the case where the *closestGreenWithPenalty* strategy is applied to any of the service types, the total blocking probability may not increase by more than 0.3 of a percentage point in comparison to the *closest* fitting schema, which is expected to utilize the least network resources. Thus, for each simulation scenario the total blocking probability was measured with *penalty* values ranging from 1.0 to 4.0 in 0.1 steps, and the *penalty* values that met the aforementioned limitation were denoted as *applicable*. For the *closestGreenWithPenalty* fitting schema, the highest *penalty* from the *applicable* set was chosen, as this prefers *green* DCs as much as admissible. At the same time, for the *compound* fitting schema the *penalty* value that resulted in the highest reduction of CO_2 emission was chosen among the ones in the *applicable* set. The *penalty* was determined separately for each cooperation model and for each fitting schema that uses the *closestGreenWithPenalty* strategy. Additionally, in the case of the *peer* cooperation model, the *penalty* must also be determined separately for each investigated combination of input parameters.

In order to determine the values of input parameters for the *peer* cooperation model we run preliminary simulations spanning numerous combinations of values. The first conclusion is related to the net_{th} value. The obtained results suggest that cloud operator preferences should be taken into account only when considering DCs equally distant to the request source. Choosing more distant DCs in order to satisfy cloud operator preferences results in an unacceptable increase of the request blocking probability. Therefore, $net_{th} = 0$ is a reasonable value as it forces comparing only equally distant DCs.

Second, we concluded that links connected to highly utilized DCs also have a great number of wavelengths occupied. A resulting small number of available wavelengths may quickly get occupied by background traffic, especially if the DC is associated with the node being traversed by numerous shortest paths connecting other nodes. Therefore, it is reasonable to fully utilize those highly occupied DCs as long as there are still network resources available to achieve that. As a result the *mostOccupied* value is assigned to the *preferenceMode* parameter. Thanks to that, IT resources are utilized more effectively and anycast mechanisms do not have to direct traffic to more distant DCs, which would result in the increase of the average lightpath length and in higher network occupancy.

Intuitively, directing traffic to most utilized DCs instead of distributing it throughout the whole DC infrastructure may create congested network areas. However, one must note that for a particular request only DCs equally distant to the source node are considered. Therefore, there is no globally preferred DC that serves all the requests. Additionally, in case of the *augmented* cooperation model the SDN controller also has to make some arbitrary decisions when DCs are equally distant to the source node. In this case it is more likely to always choose the same destination when considering the same set of equally distant DCs. Thus, preferring most occupied DCs should not

result in creation of congestion regions. Finally, the consideration of results leads to the conclusion that all equally distant DCs should be compared with regard to available resources. Therefore, $dc_{th} = 1$ was assumed as it does not exclude any DC from the set of equally distant ones.

Verification of the aforementioned assumptions was performed experimentally. Numerous simulations were run with different values of *preferenceMode*, dc_{th} and net_{th} parameters for different V_{gDC} sets in NSF and *ItalyNet* networks. For both possible values of *preferenceMode* various combination of changing dc_{th} and net_{th} parameters was investigated. The dc_{th} varied between 0 and 1 with 0.2 step providing a reasonable density of mesurement points. The net_{th} was changed with a step that reflects the network diameter. For example, the diameter of the NSF network is equal to 3, therefore net_{th} was varying from 0 to 1 with $1/3 = 0.33$ step.

From the presented set of results the combination of parameters that led to the lowest total blocking probability was found. The results obtained for the proposed set of parameter values were verified whether they do not exceed the lowest obtained blocking probability by more than 0.3 of a percentage point. The verification was positive, and, what is more, in some scenarios the proposed parameter values were found as the ones providing the lowest blocking probability. The results also show that the proposed parameter values provide one of the lowest carbon footprint among all considered input parameter combinations. It is the result of the fact that when *green* DCs are even slightly preferred, they become the most utilized ones.

Figure 4 presents blocking probability and carbon dioxide emission as a function of dc_{th}, net_{th} and *preferenceMode* parameters. This auxiliary figure proves and visualizes the aforementioned considerations about selecting values for the parameters. The presented results were obtained in the NSF network under exemplary simulation scenario where the *penalty* parameter and anycast traffic scaling factor were assigned values of 1.2 and 1.75, respectively. Please note, that plots for *preferenceMode* set to *mostOccupied* (upper row) and *leastOccupied* (lower row) are viewed from different angles. This is necessary to ensure proper readability of the figures, but it results in dc_{th} and net_{th} axis swapping places.

To sum up, it is reasonable that for all subsequent studies the *peer* cooperation model is investigated with the following input parameters: *preferenceMode* is set to *mostOccupied*, $dc_{th} = 1$ and $net_{th} = 0$.

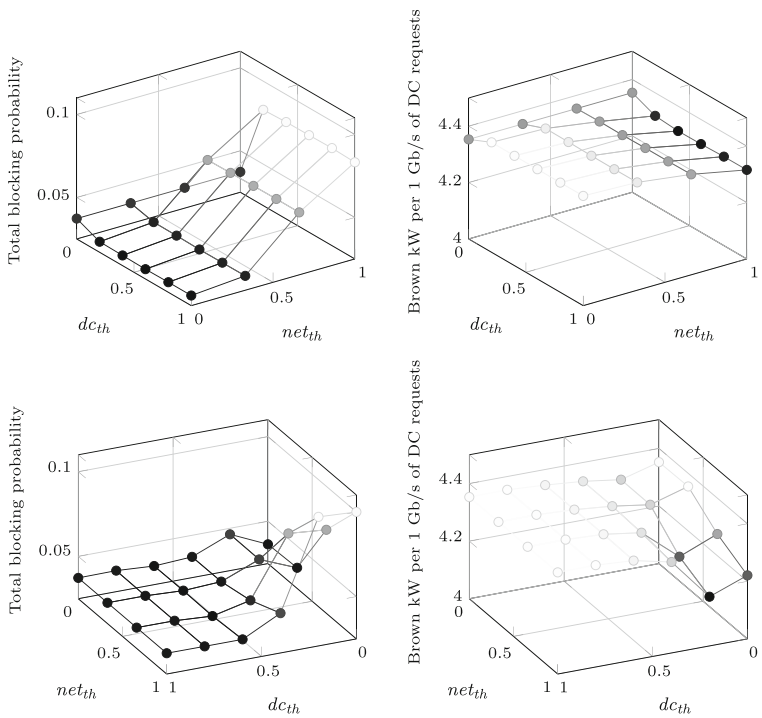

Fig. 4 Total blocking probability of all traffic types and brown kilowatts needed to handle 1 Gb/s of DC requests in the NSF network with $V_{gDC} \in \{9, 11\}$, service types' proportions 1:2:4 and assumed SaaS energy consumption equal to 5.4 kW/(Gb/s) and anycast traffic scaling factor equal to 1.75, *penalty* = 1.2 and *preferenceMode* set to *mostOccupied* (*upper row*) and *leastOccupied* (*lower row*)

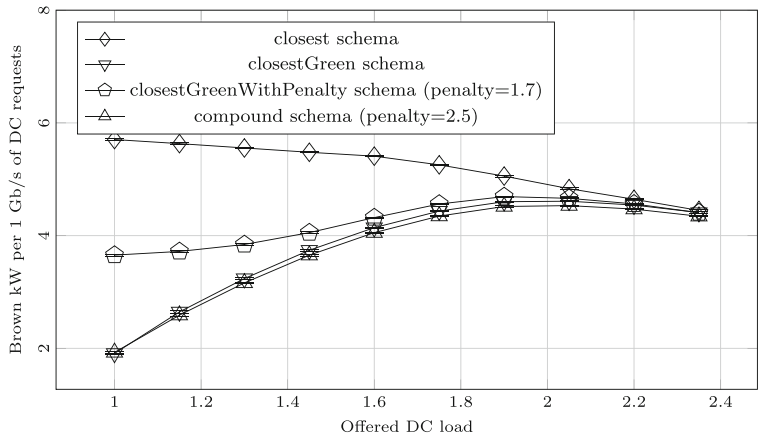

Fig. 5 Brown kilowatts needed to handle 1 Gb/s of DC requests in the NSF network with $V_{gDC} \in \{9, 11\}$, service types' proportions 1:2:4, assumed SaaS energy consumption equal to 5.4 kW/(Gb/s) and augmented cooperation model

However, changing those values introduces flexibility to the *peer* cooperation model. Thanks to that flexibility, the *peer* model may be adjusted to totally different conditions, e.g. other scaling of network and IT resources as well as larger or much more dense topologies.

8.2 Fitting Schemas

To compare different cooperation models the most prominent fitting schema must be selected for each of them. In case of the *overlay* model only the *closest* fitting schema is reasonable as this model does not assume any cooperation between a cloud provider and a network operator. Therefore, from the SDN controller's point of view the best approach is to direct cloud service requests to the closest DC. The *augmented* and *peer* models assume cooperation, so all of the fitting schemas must be considered. Figures 5 and 6 present the carbon footprint, while Figs. 7 and 8 show total blocking probabilities of the analyzed fitting schemas under the *augmented* and *peer* cooperation models, respectively.

The *closest* fitting schema has consistently the highest carbon footprint among the analyzed fitting schemas, as it provides no direct preference of *green* nodes. In the same time, its blocking probability may be considered as a baseline for other results, as it effectively uses the shortest available paths to route all DC traffic in the network.

Applying the *closestGreen* fitting schema results in strict preference of *green* DCs for each service type. This results in much lower carbon footprint compared to the *closest* fitting schema. However, this comes at a cost of an unacceptable deterioration of performance, as this solution blindly tries to direct each kind of cloud service requests to *green* DCs, even if those DCs are distant. Additionally, in some cases

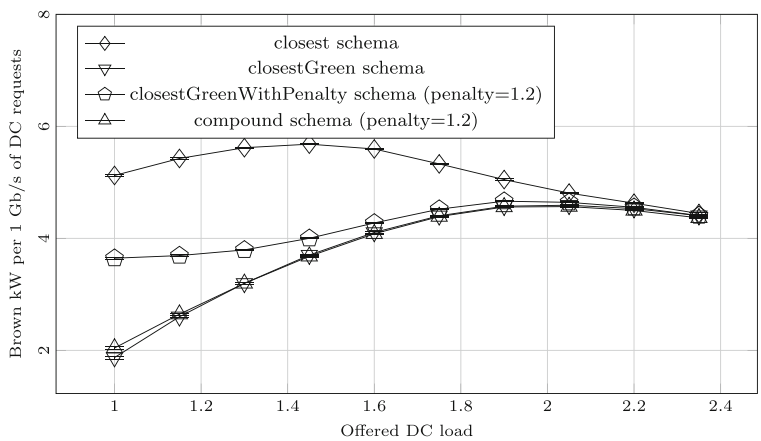

Fig. 6 Brown kilowatts needed to handle 1 Gb/s of DC requests in the NSF network with $V_{gDC} \in \{9, 11\}$, service types' proportions 1:2:4, assumed SaaS energy consumption equal to 5.4 kW/(Gb/s) and peer cooperation model

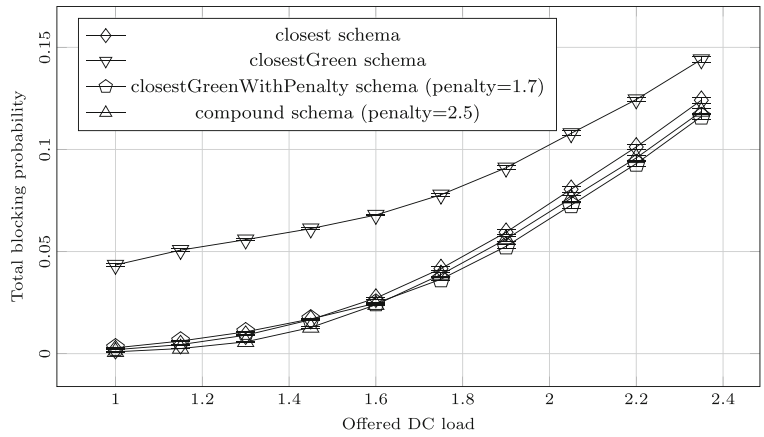

Fig. 7 Total blocking probability of all traffic types in the NSF network with $V_{gDC} \in \{9, 11\}$, service types' proportions 1:2:4, assumed SaaS energy consumption equal to 5.4 kW/(Gb/s) and augmented cooperation model

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 the *green* DCs preference with *SaaS* request energy requirements and the network resource utilization.

The *closestGreenWithPenalty* fitting schema yields a higher carbon footprint than *closestGreen*, as *green* DCs preference is less firm in this fitting schema. On the other hand, its blocking probability is comparable to the blocking probability of the *closest* fitting schema.

The *compound* fitting schema yields substantial improvements in the carbon footprint and achieves a similar blocking probability compared to the *closest* fitting schema. The obtained balance is attributed to two facts. First, in this fitting schema we do not blindly try to direct *SaaS* requests from distant clients to *green* DC nodes and, second, we always direct *StaaS* requests to the closest DC, saving network resources for other requests. Moreover, in some cases this schema provides the lowest carbon footprint and the lowest blocking probability among all considered ones. The additional decrease in the blocking probability seems to be a side effect of location of green DCs in parts of the network where less background traffic is switched. As a consequence, DC traffic is load balanced more evenly and the overall network congestion is limited.

Based on the presented results, we decided to use in subsequent studies the *closest* fitting schema with the *overlay* cooperation model and the *compound* fitting schema with the *augmented* and the *peer* cooperation models.

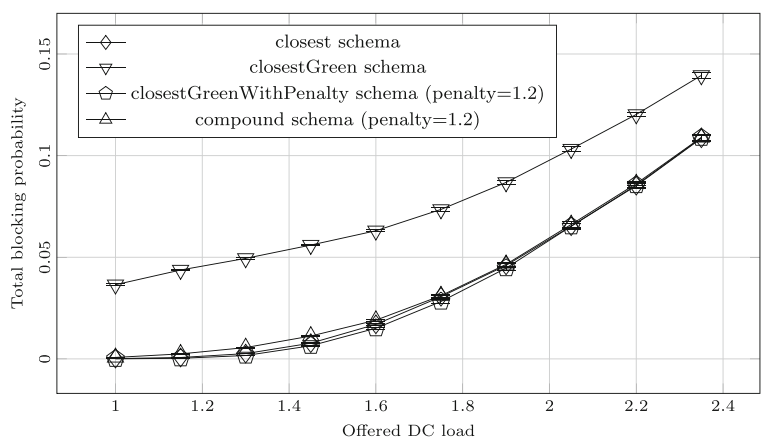

Fig. 8 Total blocking probability of all traffic types in the NSF network with $V_{gDC} \in \{9, 11\}$, service types' proportions 1:2:4, assumed SaaS energy consumption equal to 5.4 kW/(Gb/s) and peer cooperation model

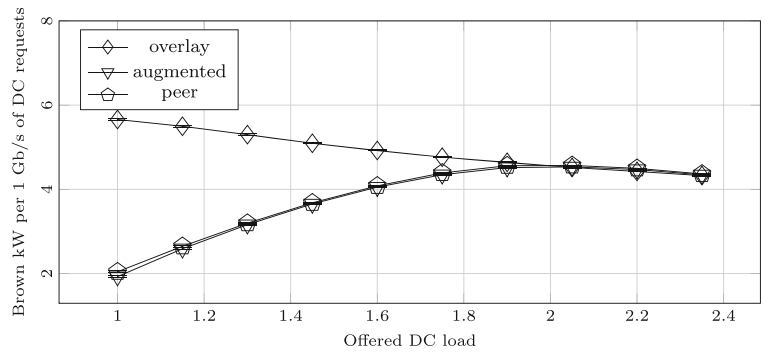

Fig. 9 Brown kilowatts needed to handle 1 Gb/s of DC requests in the NSF network with $V_{gDC} \in \{9, 11\}$, service types' proportions 1:2:4 and assumed SaaS energy consumption equal to 5.4 kW/(Gb/s)

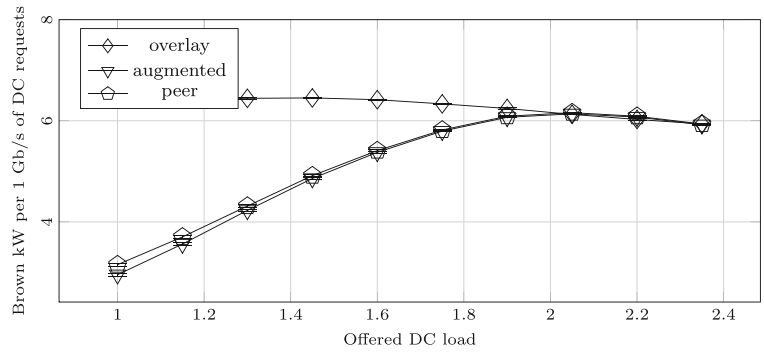

Fig. 10 Brown kilowatts needed to handle 1 Gb/s of DC requests in the NSF network with $V_{gDC} \in \{2, 9\}$, service types' proportions 1:2:4 and assumed SaaS energy consumption equal to 10 kW/(Gb/s)

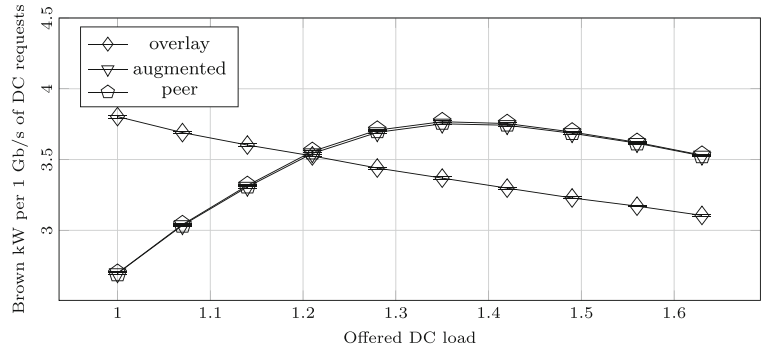

Fig. 11 Brown kilowatts needed to handle 1 Gb/s of DC requests in the ItalyNet with $V_{gDC} \in \{7, 15, 20\}$, service types' proportions 1:2:4 and assumed SaaS energy consumption equal to 5.4 kW/(Gb/s)

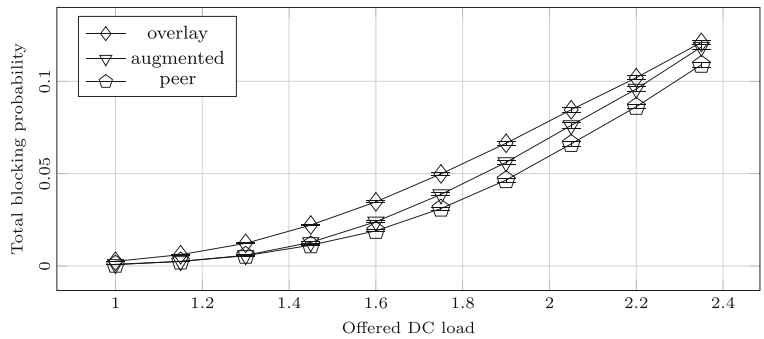

Fig. 12 Total blocking probability of all traffic types in the NSF network with $V_{gDC} \in \{9, 11\}$, service types' proportions 1:2:4 and assumed SaaS energy consumption equal to 5.4 kW/(Gb/s)

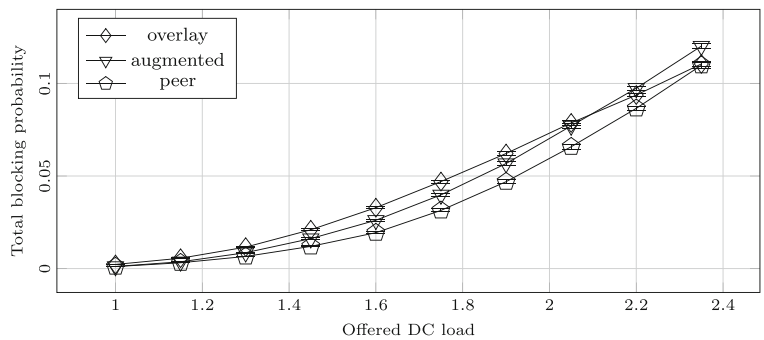

Fig. 13 Total blocking probability of all traffic types in the NSF network with $V_{gDC} \in \{2, 9\}$, service types' proportions 1:2:4 and assumed SaaS energy consumption equal to 10 kW/(Gb/s)

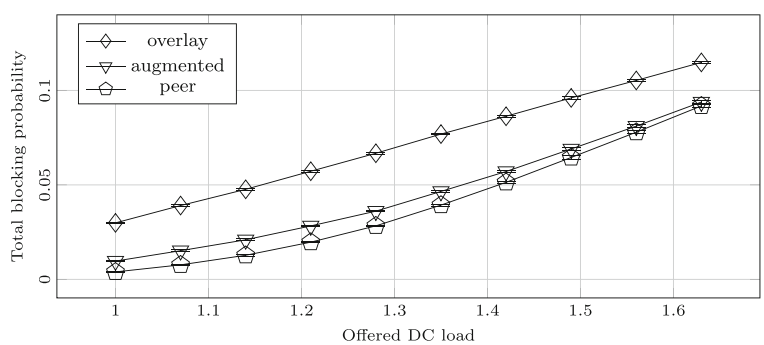

Fig. 14 Total blocking probability of all traffic types in the ItalyNet with $V_{gDC} \in \{7, 15, 20\}$, service types' proportions 1:2:4 and assumed SaaS energy consumption equal to 5.4 kW/(Gb/s)

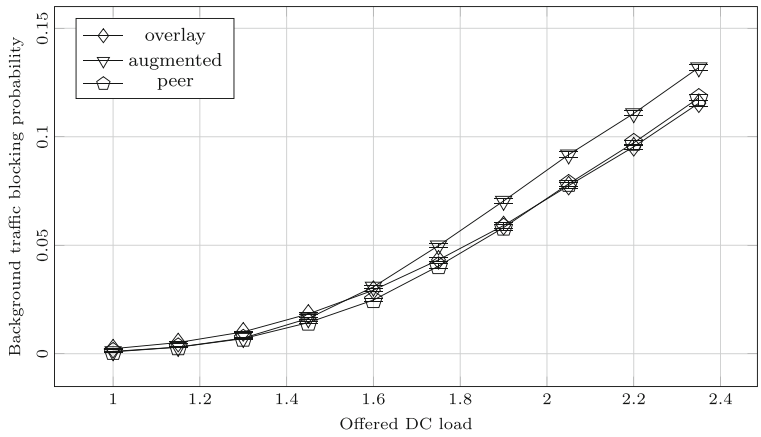

Fig. 15 Background traffic blocking probability in the NSF network with $V_{gDC} \in \{9, 11\}$, service types' proportions 1:2:4 and assumed SaaS energy consumption equal to 5.4 kW/(Gb/s)

8.3 Cooperation Models

Figures 9, 10 and 11 present the carbon footprint of the analyzed cooperation models in different simulation scenarios. The aforementioned metric and energy models were used. Figures 12, 13 and 14 show the corresponding total blocking probabilities.

The *overlay* model has constantly the highest total blocking probability among the analyzed cooperation models. This is a direct consequence of the fact that it assumes no cooperation between the network controller and the orchestration software. A destination DC is chosen solely based on network resource availability without any consideration for the availability of IT resources. Thus, it may result in choosing a DC which cannot handle the request. On the other hand, other DCs may still be able to accept the request, but they will not be taken into account due to current network conditions. Thus, DC requests may be blocked even if a great deal of IT and network resources is still available.

Additionally, in most simulation scenarios the *overlay* model resulted in the highest carbon footprint, as using the *closest* fitting schema provides no direct preference of *green* nodes and use of other schemas is impossible due to lack of necessary information about green DCs. However, in some cases the carbon footprint of the *overlay* cooperation model is lower than that provided by the *peer* or the *augmented* models. It happens when *green* DCs are located statistically closer to client nodes than *brown* DCs. Therefore, traffic is directed to *green* DCs if only network resources are available, as the *overlay* model does not take into account DC resources availability. When a chosen *green* DC is fully utilized then the request is rejected without considering any *brown* DCs. Figure 11 presents this case.

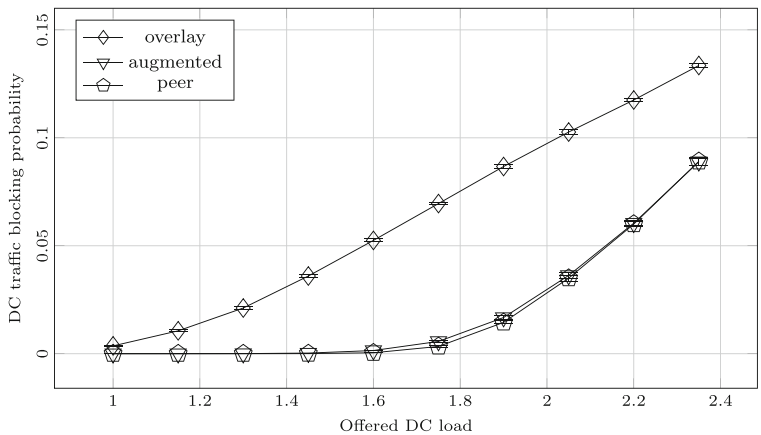

Fig. 16 DC traffic blocking probability in the NSF network with $V_{gDC} \in \{9, 11\}$, service types' proportions 1:2:4 and assumed SaaS energy consumption equal to 5.4 kW/(Gb/s)

Applying the *augmented* cooperation model results in a decrease of the total blocking probability in comparison to the *overlay* model. It is a result of the fact that the SDN controller tries to establish lightpaths only to DCs that have available IT resources. Therefore, the *augmented* model excludes situations where a cloud service request is blocked despite reachability of a DC with sufficient IT resources. However, Fig. 13 presents the scenario where the *overlay* model provides lower blocking probability than the *augmented* model for high traffic load. This phenomenon appears because the total blocking probability is a superposition of the background and DC traffic blocking probabilities. In the *augmented* model the controller tries to direct traffic to DCs with sufficient IT resources even if those DCs are distant. This results in the lower blocking probability for anycast traffic but consumes more network resources. As a result, the blocking probability of the background traffic is much higher. On the other hand, the *overlay* model always tries to direct traffic to the closest reachable DC, no matter if enough IT resources are available. This results in the higher blocking probability for anycast requests but the lower blocking probability of background traffic. This issue is addressed later in the text and illustrated in Figs. 15 and 16.

The *peer*, proposed cooperation model, constantly provides the best total blocking probability among all the considered models. The additional information provided by the cloud orchestration software limits the number of situations where available IT resources are unreachable due to network congestion. Thus, the model allows for efficient utilization of IT resources. Simultaneously, the carbon footprint is maintained at the same level as in the *augmented* model.

In order to more precisely explain the differences between the models additional results are provided. Figures 15 and 16 present the blocking probabilities of background and DC traffic, respectively. Anycast, compared to unicast traffic, has an additional degree of freedom related to the choice of a destination node. Therefore, in the presence of network congestion anycast requests have less likelihood of being affected. In both the *augmented* and the *peer* models, DC traffic is blocked with similar probability, lower than in case of the *overlay* model. However, in the *augmented* model DC traffic is routed in a less efficient way. This results in a greater overall congestion of the network and is reflected in the blocking probability of background traffic, which is higher in the *augmented* model, compared to both the *overlay* and *peer* models.

In the paper we limited the set of presented results to the most exemplary and illustrative ones. However, the provided conclusions are valid for all of the obtained results. Additionally, we observed that changing the proportions between DC service types from 1:1:1 through 1:2:4 to 1:5:25 results, in most cases, in better reduction of the carbon footprint while preserving blocking probability at the same level. Increasing the assumed energy consumption of the *SaaS* service results in similar conclusions.

9 Conclusions

Current cloud architectures are composed of two clearly defined entites: a network interconnecting data centers and IT infrastructure within them. We state, that in order to effectively provide cloud services with minimal negative impact on the natural environment a cooperation is necessary between those two entities. However, it is necessary to remember, that close cooperation comes at the expense of some information disclosure between cooperating parties and is the additional potential for security risks.

In the paper, we investigated models of cooperation between a network operator and a cloud service provider. The models are conceptually simple and are well placed in the control plane of modern cloud architectures. Our aim was to analyze their impact on the request blocking probability and carbon footprint.

The proposed *peer* model provides consistently the lowest blocking probability in comparison to the *overlay* and *augmented* models. Additionally, its carbon footprint remains at the level comparable to the *augmented* model. Therefore, the *peer* model may improve service quality and, when joined with *green* anycast strategies and an appropriate fitting schema, it may significantly decrease carbon dioxide emission of the cloud. Numerous simulations performed in various scenarios support this conclusion. Additionally, the results show that the *compound* fitting schema, introduced and analyzed in our previous work, is the best one for this model.

The proposed cooperation model is flexible and adjustable to the network conditions thanks to programmable thresholds and DC preference modes.

However, an appropriate choice of values for those parameters is not an easy task, and requires further investigation. For real deployments a method to quickly estimate the values of those parameters is necessary in order to adapt the behaviour of the model to the current state of IT resources and network conditions. This subject is, however, left for further studies.

Despite the fact that we focused on the all optical WDM networks our solutions may be adopted to other networking technologies that utilize the concept of an end-to-end path, such as Multiprotocol Label Switching (MPLS) or Elastic Optical Networks (EON). MPLS is very popular in contemporary WANs while EON is an emerging technology that seems to be the future of optical networks. A modification of the proposed solutions to such technologies is also an interesting area for future work.

The investigated problem of cooperation between a network controller and cloud orchestration software is one of the most important issues that needs to be dealt with for efficient and environmentally friendly cloud services provisioning. We believe that the presented architecture and the proposed solutions will be a step forward in the process of solving this problem and that they will become a part of the green cloud computing concept.

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

References

1. Gupta, M., Singh, S.: Greening of the internet. In: Proceedings of the ACM SIGCOMM 2003, pp. 19–26, Karlsruhe (2003)
2. Jarschel, M., Zinner, T., Hossfeld, T., Tran-Gia, P., Kellerer, W.: Interfaces, attributes, and use cases: a compass for SDN. IEEE Commun. Mag. **52**(6), 210–217 (2014)
3. Ahmed, R., Boutaba, R.: Design considerations for managing wide area software defined networks. IEEE Commun. Mag. **52**(7), 116–123 (2014)
4. Yangui, S., Marshall, I.J., Laisne, J.P., Tata, S.: Compatibleone: the open source cloud broker. J. Grid Comput. **12**(1), 93–109 (2014)
5. Petcu, D.: Consuming resources and services from multiple clouds. J. Grid Comput. **12**(2), 321–345 (2014)
6. Jain, S., Kumar, A., Mandal, S., Ong, J., Poutievski, L., Singh, A., Venkata, S., Wanderer, J., Zhou, J., Zhu, M., Zolla, J., Hölzle, U., Stuart, S., Vahdat, A.: B4: experience with a globally-deployed software defined Wan. In: Proceedings of the ACM SIGCOMM 2013, pp. 3–14 (2013)
7. Alger, D.: Grow a greener data center. Cisco Press, Indianapolis (2010)
8. Lewis, A.: The four key environmental factors of ICT: Energy, carbon, e-waste and water. http://www.sustainability-perspectives.com/article/the-four-key-environmental-factors-of-ict-energy-carbon-e-waste-and-water/(2014). Cited 07 Oct 2014
9. Lannoo, B.: Energy consumption of ICT networks - TREND final workshop. http://www.fp7-trend.eu/system/files/content-public/502-final-trend-workshop-brussels-24-october-2013-presentations/energyconsumptionincentives-energy-efficient-networks.pdf (2013). Cited 07 Oct 2014
10. The Green Grid: Carbon, Water and Energy Efficiency Metrics, Measurements and Trends for Data Center Planning. http://www.thegreengrid.org/ (2014). Cited 07 Oct 2014
11. Greenpeace: How Companies are Creating the Green Internet. http://www.greenpeace.org/usa/Global/usa/planet3/PDFs/clickingclean.pdf (2014). Cited 07 Oct 2014
12. Facebook: Green On Facebook - Carbon and Energy. https://www.facebook.com/green/app_439663542812831?ref=page_internal (2014). Cited 07 Oct 2014
13. Google: Data Centers Renewable Energy. www.google.pl/about/datacenters/renewable (2014). Cited 07 Oct 2014
14. Kliazovich, D., Arzo, S., Granelli, F., Bouvry, P., Khan, S.: Accounting for load variation in energy-efficient data centers. In: Proceedings of the IEEE International Conference on Communications (ICC 2013), pp. 2561–2566, Budapest (2013)
15. Wang, X., Razo, M., Tacca, M., Fumagalli, A.: Planning and online resource allocation for the multi-resource cloud infrastructure. In: Proceedings of the IEEE International Conference on Communications (ICC 2014), pp. 2944–2949, Sydney (2014)
16. Liu, X., Zhang, L., Zhang, M., Zhu, Z.: Joint defragmentation of spectrum and computing resources in inter-datacenter networks over elastic optical infrastructure. In: Proceedings of the IEEE International Conference on Communications (ICC 2014), pp. 3295–3300, Sydney (2014)
17. Lin, T., Kang, J.M., Bannazadeh, H., Leon-Garcia, A.: Enabling SDN applications on software-defined infrastructure. In: Proceedings of the IEEE Network Operations and Management Symposium (NOMS 2014), pp. 1–7, Krakow (2014)
18. Gattulli, M., Tornatore, M., Fiandra, R., Pattavina, A.: Low-Carbon routing algorithms for cloud computing services in IP-over-WDM networks. Proceedings of the IEEE International Conference on Communications (ICC 2012), pp. 2999–3003, Ottawa (2012)
19. Gattulli, M., Tornatore, M., Fiandra, R., Pattavina, A.: Low-emissions routing for cloud computing in IP-over-WDM

networks with data centers. IEEE J. Sel. Areas Commun. **32**(1), 28–38 (2014)
20. Walkowiak, K., Kasprzak, A., Klinkowski, M.: Dynamic routing of anycast and unicast traffic in elastic optical networks. In: Proceedings of the IEEE International Conference on Communications (ICC 2014), pp. 3313–3318, Sydney (2014)
21. Rzasa, J., Borylo, P., Lason, A., Szymanski, A., Jajszczyk, A.: Dynamic power capping for multilayer hybrid power networks. In: Proceedings of the IEEE Global Communications Conference (GLOBECOM 2014). Austin (2014)
22. Mandal, U., Habib, M., Shuqiang, Z., Mukherjee, B., Tornatore, M.: Greening the cloud using renewable-energy-aware service migration. IEEE Netw. **27**(6), 36–43 (2013)
23. Rodero, I., Viswanathan, H., Lee, E., Gamell, M., Pompili, D., Parashar, M.: Energy-efficient thermal-aware autonomic management of virtualized HPC cloud infrastructure. J. Grid Comput. **10**(3), 447–473 (2012)
24. Borylo, P., Lason, A., Rzasa, J., Szymanski, A., Jajszczyk, A.: Anycast routing for carbon footprint reduction in WDM hybrid power networks with data centers. In: Proceedings IEEE International Conference on Communications (ICC 2014), pp. 3720–3726, Sydney (2014)
25. Borylo, P., Lason, A., Rzasa, J., Szymanski, A., Jajszczyk, A.: Fitting green anycast strategies to cloud services in WDM hybrid power networks. In: Proceedings of the IEEE Global Communications Conference (GLOBECOM 2014), Austin (2014)
26. Develder, C., Leenheer, M.D., Dhoedt, B., Pickavet, M., Colle, D., Turck, F.D., Demeester, P.: Optical networks for grid and cloud computing applications. Proc. IEEE **100**(5), 1149–1167 (2012)
27. Amazon: Amazon ec2. http://aws.amazon.com/ec2/ (2014). Cited 07 Oct 2014
28. Open Stack: Open Source Cloud Computing Software. https://www.openstack.org/ (2014). Cited 07 Oct 2014
29. Open Nebula: Flexible Enterprise Cloud Made Simple. http://opennebula.org/ (2014). Cited 07 Oct 2014
30. Cloud Stack: Open Source Cloud Computing. http://cloudstack.apache.org/ (2014). Cited 07 Oct 2014
31. Boru, D., Kliazovich, D., Granelli, F., Bouvry, P., Zomaya, A.: Energy-Efficient data replication in cloud computing datacenters. In: Proceedings of the IEEE Global Communications Conference Workshops (GLOBECOM Wkshps 2013), pp. 446–451, Atlanta (2013)
32. McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J.: Openflow: enabling innovation in campus networks. ACM SIGCOMM Comput. Commun. Rev. **38**(2) (2008)
33. OpenDaylight: A Linux Foundation Collaborative Projects. http://www.opendaylight.org/ (2014). Cited 07 Oct 2014
34. Floodlight: Open Source Software for Building Software Defined Networks. http://www.projectfloodlight.org/ (2014). Cited 07 Oct 2014
35. Stanford: Beacon. https://openflow.stanford.edu/display/Beacon/Home (2014). Cited 07 Oct 2014
36. Big Switch: Big Cloud Fabric. http://www.bigswitch.com/sdn-products/big-cloud-fabric (2014). Cited 07 Oct 2014
37. Dorronsoro, B., Nesmachnow, S., Taheri, J., Zomaya, A.Y., Talbi, E.G., Bouvry, P.: A hierarchical approach for energy-efficient scheduling of large workloads in multicore distributed systems. Sustainable Computing: Informatics and Systems. doi:10.1016/j.suscom.2014.08.003 (2014)
38. Nesmachnow, S., Dorronsoro, B., Pecero, J., Bouvry, P.: Energy-Aware Scheduling on multicore heterogeneous grid computing systems. J. Grid Comput. **11**(4), 653–680 (2013)
39. Khajemohammadi, H., Fanian, A., Gulliver, T.: Efficient workflow scheduling for grid computing using a leveled multi-objective genetic algorithm. J. Grid Comput. **12**(4), 637–663 (2014)
40. Trger, P., Merzky, A.: Towards standardized job submission and control in infrastructure clouds. J. Grid Comput. **12**(1), 111–125 (2014)
41. Google: Cluster Data Trace of Google Workloads. https://code.google.com/p/googleclusterdata/ (2014). Cited 07 Oct 2014
42. Baliga, J., Ayre, R., Hinton, K., Tucker, R.S.: Green cloud computing: balancing energy in processing, storage, and transport. Proc. IEEE **99**(1), 149–167 (2011)
43. Chlamtac, I., Ganz, A., Karmi, G.: Lightpath communications: an approach to high bandwidth optical WAN's. IEEE Trans. Commun. **40**(7), 1171–1182 (1992)
44. Szymanski, A., Lason, A., Rzasa, J., Jajszczyk, A.: Performance evaluation of the grade-of-service-based routing strategies for optical networks. In: Proceedings of the IEEE International Conference on Communications (ICC 2008), pp. 5252–5257, Beijing (2008)
45. Dong, X., El-Gorashi, T., Elmirghani, J.M.H.: Green IP over WDM networks with data centers. IEEE J. Lightwave Technol. **29**(12), 1861–1880 (2011)
46. Klinkowski, M., Walkowiak, K.: On the advantages of elastic optical networks for provisioning of cloud computing traffic. IEEE Netw. **27**(6), 44–51 (2013)
47. Varga, A.: Using the OMNeT++ discrete event simulation system in education. IEEE Trans. Educ. **42**(4), 11 (1997)