A Cloud-Oriented Content Delivery Network Paradigm: Modeling and Assessment

Chrysa Papagianni, Aris Leivadeas, and Symeon Papavassiliou, *Senior Member, IEEE*

Abstract—Cloud-oriented content delivery networks (CCDNs) constitute a promising alternative to traditional content delivery networks. Exploiting the advantages and principles of the cloud, such as the pay as you go business model and geographical dispersion of resources, CCDN can provide a viable and cost-effective solution for realizing content delivery networks and services. In this paper, a hierarchical framework is proposed and evaluated toward an efficient and scalable solution of content distribution over a multiprovider networked cloud environment, where inter and intra cloud communication resources are simultaneously considered along with traditional cloud computing resources. To efficiently deal with the CCDN deployment problem in this emerging and challenging computing paradigm, the problem is decomposed to graph partitioning and replica placement problems while appropriate cost models are introduced/adapted. Novel approaches on the replica placement problem within the cloud are proposed while the limitations of the physical substrate are taken into consideration. The performance of the proposed hierarchical CCDN framework is assessed via modeling and simulation, while appropriate metrics are defined/adopted associated with and reflecting the interests of the different identified involved key players.

Index Terms—Content delivery network, cloud computing, networked cloud, replica placement

1 Introduction

Content delivery networks (CDNs) have been introduced to overcome the limitations of the Internet, related to how the users perceive quality of service (QoS) when they are accessing and obtaining content [1]. In a CDN, content is replicated over the resources at cache servers (also known as edge or surrogate servers) scattered over the globe. Client requests are redirected to the most suitable location based on network-related criteria such as traffic volume, quality, and proximity, as well as service-oriented criteria such as server load, response time, and availability.

Traditional CDNs are successful at supporting web traffic, but require significant investments and deployment efforts, and are to a large extent application specific. Commercial CDNs such as Akamai, LimeLight Networks, and Mirror Image place data centers and edge servers in numerous geographical locations to enhance responsiveness and content locality at the expense of an increase in service price.

However, in the emerging Future Internet paradigm the solution lies within the cloud [2]. A cloud-oriented CDN (CCDN) merely uses the cloud for extensible and fungible capacity or the cloud subsumes CDNs, with features like replication of content across geographically dispersed locations carried over onto the cloud. CCDNs can reduce the effort and cost of producing complete, dedicated solutions in an environment with shorter development and time-to-market cycles.

• *The authors are with the School of Electrical and Computer Engineering, National Technical University of Athens, Iroon Polytechniou 9, Zografou, Athens 15780, Greece. E-mail: chrisap@noc.ntua.gr, arisleiv@netmode.ntua.gr, papavass@mail.ntua.gr.*

Manuscript received 15 July 2012; revised 14 Nov. 2012; accepted 4 Feb. 2013; published online 13 Feb. 2013.
For information on obtaining reprints of this article, please send e-mail to: tdsc@computer.org, and reference IEEECS Log Number TDSCSI-2012-07-0161.
Digital Object Identifier no. 10.1109/TDSC.2013.12.

Cloud emerges as the de facto standard for computing purposes and applications, supporting a multiplicity of different models with respect to usage, business philosophy adopted, and respective research value. Adding intra- or intercloud communication resources to the resource mix leads to a networked cloud computing environment (NCE) for the delivery of infrastructure as a service (IaaS) [3]. Toward the solution of a content delivery over a NCE, the surrogate server simply becomes another resource in the cloud, while virtual servers are connected with intra- or intercloud communication paths used for content distribution.

Mapping of virtual (e.g., storage and communication) resources to substrate ones across multiple administrative domains is a key problem in a NCE for efficiently delivering cloud IaaS [4]. However, for the purpose of delivering a CCDN where the characteristics of the CDN are inherited to the cloud, virtual to physical resource mapping must be addressed in conjunction with the so-called surrogate placement problem; that is the placement of surrogates to locations closer to end users [1].

While the problem has been studied extensively in traditional CDNs, these results can not be directly applied to a cloud based CDN [5]. In a multiprovider networked cloud environment where multiple cloud service providers (CPs) coexist, transit network providers may be used for interconnecting cloud sites. Within this cloud "ecosystem," virtual surrogate servers are deployed on cloud sites. However, efficient content distribution paths must be also deployed, while the underlying physical network topology allows for higher degrees of freedom in their design. What is more, these paths may span multiple administrative domains. Moreover, the replication direction in the content distribution graph becomes of paramount importance regarding CCDN cost, due to the existing cost model in clouds, where uploading and downloading costs vary. Finally, appropriate cloud brokering service is required, so that the multicloud-based CDN exhibits cost and operational efficiency. Our proposed work aims specifically at tackling the aforementioned challenges.

1.1 Paper Contribution and Structure

The focus of this paper is placed on the introduction, design, and assessment of a complete model and framework aiming specifically at the deployment of a CCDN paradigm within a multiprovider Networked Cloud Environment. To efficiently deal with the CCDN problem in this emerging and challenging computing paradigm, the problem is decomposed to 1) graph partitioning that mainly addresses aspects related to the multiprovider nature of the problem, in a scalable and efficient manner and 2) the replica placement problem that emphasizes on the actual selection of the surrogate servers and their mapping on the corresponding substrate within a single domain.

With reference to Step 1, initially the CCDN deployment geographic area is broken down to smaller ones, called service clusters and as a result an intercluster content distribution graph is created. Assuming coexistence of several CPs over the same service area, cost-efficient partitioning of the intercluster content distribution graph is performed, and therefore, each service cluster is assigned to a CP, based on a metaheuristic technique—iterated local search (ILS). In previous work, we proposed the adoption of the ILS for efficient partitioning of a request on virtual resources, for the purpose of resource mapping over networked clouds. Due to its intrinsic simplicity, general applicability, and scalability, we take on the same technique for the graph partitioning problem, properly formulated, however, to facilitate the cost-efficient deployment of a CCDN.

Following, with reference to Step 2, each selected CP utilizes an appropriate surrogate placement algorithm to determine the number and location of surrogates hosted at a subset of the available cloud sites within the particular domain. In the proposed framework, we introduce two solutions. The first one—virtual surrogate placement (VSP)—is executed in two phases: 1) the surrogate server selection phase and 2) the content distribution path selection phase within the domain. The two-phase procedure has been inspired by resource mapping algorithms that we have presented in previous studies in the context of network virtualization (e.g., [4], [6]). However, in the particular work, we propose a novel MIP formulation of the surrogate placement problem over a networked cloud for the surrogate server selection phase. The second approach—SNA inspired greedy virtual surrogate placement (SNA-GVSP)—is a novel greedy surrogate placement heuristic inspired by social network analysis (SNA). In our previous work, the efficiency of introducing social metrics in the resource mapping process was demonstrated in the context of network virtualization [7]. In this paper, the goal of the proposed novel approach is to greedily assign end-users to virtual surrogate servers, maximizing the average shortest path betweeness centrality (SPBC) of the selected set of surrogates so that each end user is assigned to at least one cloud site, and the QoS requirements for end-user requests are satisfied.

The key contributions of this paper are as follows:

- Description of the emerging paradigm and model for CCDN capitalizing on advances in virtualization, and design of a conceptual architecture framework (e.g., hierarchical framework) for delivering CCDNs.
- Definition, adoption, and proper adaptation of models that can be used in a holistic and dynamic way to reflect costs and objectives for CCDN delivery in NCE (i.e., define inter- and intracloud link costs, main quantitative objectives, and so on).
- Setup and formulation of emerging optimization problems along with associated constraints, to identify an efficient and feasible virtual surrogate server placement for the CCDN.
- Adoption of appropriate methodologies for dealing with the joint replica placement/resource mapping problem in a NCE, taking into account efficiency and scalability.
- Definition and measurement of key metrics associated with and reflecting the interests of the different identified involved players (i.e., accounting, social, utilization metrics) that allow for proper assessment of proposed paradigm and corresponding approaches/solutions.
- Presentation of numerical results to demonstrate the entire life cycle of the assessment process and the emerging tradeoffs.

The rest of the paper is organized as follows: Section 2 provides the required background on the server placement problem in traditional CDNs and cloud-based CDNs. Section 3 provides a high-level description of the introduced CCDN framework, along with identification of involved key players and their roles. In Section 4, the formalism, problem formulation, and adopted solution for problem at hand are presented in detail. In Section 5, a thorough evaluation of the proposed framework over NCE through modeling and simulation is presented, including the definition and classification of related key metrics. Finally, Section 6 concludes the paper.

2 Related Work

The replica placement problem in traditional CDNs refers to finding the best set of servers to place content replicas. Replica placement belongs to the NP-complete class of problems [8]. The CDN performance can be affected by decisions such as:

1. the number of surrogates required,
2. their location,
3. the cost model adopted including storage cost, content retrieval cost, and content updating cost, and
4. QoS considerations.

Depending on the consideration of the above elements, the problem has been traditionally formulated in literature as the k-median problem [9], the facility location problem [9], the minimum k-center problem [10], the dynamic programming model [11], and so on. A thorough analysis on the mathematical models used for resource management in CDNs can be found in [12]. With respect to the adopted cost model, initial attempts on the surrogate placement problem included only content downloading cost (e.g., [9], [13]). In subsequent works, in addition to retrieval cost, content update cost (e.g., [11]) and storage cost (e.g., [14]) were added to the problem formulation as well. Therefore, Kalpakis et al. [15] considered all three costs. In further attempts, QoS considerations related to responsiveness requirements in content replication were added to the replica placement problem, usually by enforcing constraints on user to surrogate server proximity (e.g., [16], [17], [18], [19]).

All the aforementioned studies propose solutions that are static in nature in the sense that prior knowledge of request patterns is required to make replica placement decisions. We follow the same paradigm in this work, by looking into the complex problem of determining the location and number of surrogates needed to serve a set of users by taking into consideration QoS constraints. Moreover, in the adopted methodology, retrieval, update, and storage costs are considered. The problem, however, is transferred to the networked cloud environment; hence, constraints posed by the virtualization environment are also taken into consideration, as mentioned in the previous section. As a result, a different approach of the replica placement problem is required.

With the advent of cloud computing, few studies have been provided in literature that look into optimizing a cloud-based CDN. Jin et al. [20] propose a conceptual architecture for Content Delivery as a Service (CoDaaS). The virtual Content Delivery Service (vCDS) is built on top of a cloud for user generated content. However, authors do not look into the specifics of building the vCDS overlay. A different approach is followed by Srinivasan et al. [21] that allows content providers to dynamically deploy and instantiate CDN modules on core and edge routers with programming functionality over content distribution paths from the origin server to the users. The particular cloud-based CDN solution, called ActiveCDN, allows users to be redirected to the most appropriate node based on geolocation. CDN modules are deployed on an "as needed" basis leading to a dynamic network topology. Carlsson et al. [22] explore design tradeoffs in a system that leverages both cloud and peer resources as means to reduce the load on origin servers and the delivery cost for the content provider. Authors pinpoint the importance of optimized cloud replica placement in decreasing content delivery cost, by defining and evaluating baseline policies related to content replication/replica placement and request redirection. These, however, are in the context of the peer assisted cloud-based CDN while each cloud server is solely associated with a unique nonoverlapping locality region.

MetaCDN [2] system is a commercially available cloud-based CDN that provides an interface for standard cloud providers to be used for content delivery. The system, via an appropriate web portal, grants end users with a number of different options related to cost and QoS. Specifically, it enables a set of replica deployment options that consequently define the request redirection policies. These include:

1. maximizing coverage and performance, where as many replicas as possible are deployed to all available cloud locations,
2. deploy content in specific locations, where a user nominates regions and MetaCDN matches the requested regions,
3. cost optimized, or
4. QoS optimized deployment in the user nominated regions.

However, the details of the replica placement strategies are not provided. Chen et al. [5] are the first ones to investigate the problem of placing server replicas in storage cloud CDNs along with building content distribution paths among them. Their goal is to minimize the cost incurred on CDN providers while satisfying QoS requirements for end users.

In the proposed work, we go one step further by considering the optimized replica placement problem and distribution path construction for a multidomain networked cloud environment. The goal is to minimize the CCDN deployment cost while satisfying QoS requirements for end users and capacity constraints of physical resources (disk and bandwidth capacity resources). The networked cloud environment is spanning multiple administrative domains, because competing CPs with overlapping service areas are considered. Intradomain and interdomain content distribution paths are determined based on the cost for reserving resources along these shared physical paths.

3 CCDN Hierarchical Framework

In a traditional CDN, the basic actors involved are end users, the Content Provider, and the CDN provider. End users are the entities who access content from the content providers website. The content provider is the entity that owns or is licensed to sell content or content assets. The origin server of the content provider holds those assets. The CDN provider deploys, maintains, and provides the infrastructure facilities for timely and reliable content delivery; in an overlay CDN, it essentially consists of a set of edge servers (surrogates or replicas) that deliver copies of content to end users [1].

In the context of a NCE, content providers may deploy their own cloud-based CDN, leasing cloud IaaS from 1) CPs (e.g., Amazon EC2) and 2) transit network providers, which provide intercloud virtual network service (e.g., on demand QoS-guaranteed paths) [3], [4]. Transit network providers may use L2/L3 virtual networking technologies (e.g., VLANs) to establish network overlays among the multiple CPs. Thus, a content provider acts also as a CCDN provider. The lease can be negotiated either directly with CPs and transit network providers or via some appropriate brokerage service (denoted in the following as CCDN Broker).

In the proposed centralized brokerage service, the CCDN broker is a third-party service provider that acts as the interface between the CCDN Provider and CPs/transit network providers. The CCDN broker is responsible for creating the intercluster content distribution graph and partitioning this among the competing CPs, following the solution proposed in this paper. Finally, it is the responsibility of each CP to establish the partial CDN solution by executing an appropriate surrogate placement strategy. The interconnected partial CDNs are merged by the CCDN broker to deliver the final CCDN solution to the requester (CCDN provider). Alternatively, a distributed approach for the CCDN broker, located at each CP, may be followed where the CCDN provider directly contacts a CP and

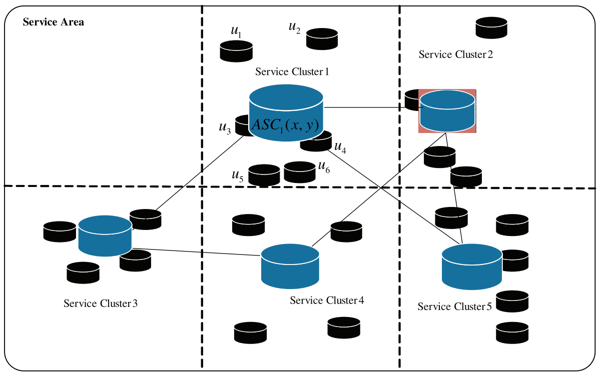

Fig. 1. Service area clustering and intercluster content distribution graph.

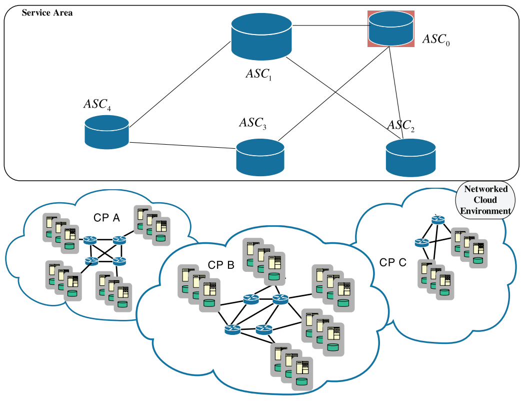

Fig. 2. Intercluster content distribution graph over a networked cloud environment.

bilateral agreements between CPs and transit network providers enable the CCDN deployment on a multi-provider NCE.

Each of the involved actors have their own priorities in terms of cost, efficiency, and QoS [23]. For example, end users have specific requirements in terms of QoS, CCDN providers need to minimize the CCDN deployment cost on top of the multiprovider NCE, while CPs need to efficiently utilize their resources to maximize their revenue. The CCDN broker is interested in 1) addressing the requirements posed by the CCDN provider and 2) minimizing the overall expenditure.

Nevertheless, providing a CDN over networked clouds requires the adoption of an appropriate edge server placement scheme, taking into consideration the multi-provider environment and the geolocation of potential users. As summarized in the introductory part, this can be accomplished by a three step procedure presented in the following.

CCDN predesign phase and assumptions. Surrogate server placement on interconnected clouds should follow the geographical dispersion of the recipients of content, that is end users. Therefore, the CCDN provider must identify potential clientele in terms of location in a geographically defined service area. The service area grid is then broken down by the CCDN broker into a number of smaller subregions, denoted as service clusters. In this study, we assume k service clusters without looking into the efficiency of the clustering technique. It is noted that various alternative clustering approaches may be considered, depending on the details of the offered service, as well as the provider's business model and customer basis. Such discussion, however, is considered outside the scope of this paper.

Each service cluster is abstracted to a node called abstracted service cluster ASC_k. The ASC_k is characterized by location coordinates as defined by the center of mass $ASC_k(x, y)$ of the service cluster, estimated by geolocating of CDN end users. Additional properties of the ASC_k are the CDN related characteristics of each service cluster such as: 1) end users within the cluster, 2) request rate and size per user, and 3) the size of replicated content. The aforementioned abstraction is depicted in Fig. 1, specifically for service cluster 1. For the set of $kASCs$, a partial mesh graph is established in the proposed framework. The highlighted ASC denotes the origin server, assuming that the CCDN provider has selected a priori the service cluster where the origin server will be located. The extracted content distribution graph will be subsequently used for assigning service clusters to CPs. A partial mesh graph is considered here as it provides redundant content distribution paths from the origin server's cluster toward the service clusters of the entire service area. Alternative topologies with different features and properties (i.e., tree-based topologies) may be also adopted.

Service cluster to CP assignment. The coexistence of several competing CPs is assumed, providing full overlapping over the same service area. The goal for the CCDN Broker in this step is to partition efficiently the intercluster content distribution graph between the various CPs. Each cluster can be assigned to only one CP while each CP may serve multiple clusters.

The CCDN Broker forwards the extracted intercluster content distribution graph to the CPs (Fig. 2). CPs are requested to provide the broker with information related to the number of cloud sites that are available per service cluster as well as unit provisioning costs related to every cloud site, allowing for differentiated pricing based on locality of service. Based on this information and the adopted graph partitioning scheme (e.g., [4], [24]), the CCDN broker concludes upon the most cost-effective service cluster to CP assignment and sends the corresponding parts of the intercluster content distribution graph to the to the selected CPs for further processing.

Virtual surrogate server placement per service cluster. Upon being assigned one or more partial intercluster content distribution graphs with information regarding the location of the users and their requirements within each service cluster, the CP executes an appropriate surrogate placement algorithm (Fig. 3). In the proposed framework, we introduce two novel algorithms: the VSP heuristic and SNA-GVSP heuristic. The objective of the problem at hand is to determine the number and location of replicas hosted at a subset of the available cloud sites within the CP's domain, in the particular service cluster, so that each end user is assigned to at least one site and the QoS requirements for end-user requests are satisfied. One of the surrogate servers within a service cluster may be selected to facilitate the communication between the rest of the surrogates and the origin server that belongs to different service cluster. This server is selected based on proximity to the ASC and

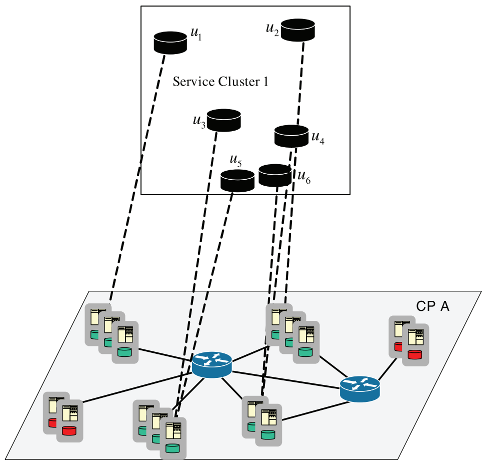

Fig. 3. Replica placement in a networked cloud.

capacity requirements and is denoted hereafter as transit server. The CCDN broker is responsible for informing the Content/CCDN provider on overall CCDN cost and deployment information.

Table 1 provides a summary of the basic concepts described above.

4 Problem Formulation

4.1 CCDN Framework Notation: Bottom-Up Approach

4.1.1 *Substrate*

The substrate network topology for every CP $i \in I$ (where I denotes the set of cloud providers) is modeled as an undirected graph $G_i^S = (N_i^S, E_i^S)$. Each substrate node $n_i^S \in N_{A,i}^S \subseteq N_i^S$, where $N_{A,i}^S$ is a physical resource set with common properties including the node's type, e.g., $A = \{nodetype, OS, virtEnv, etc.\}$. Moreover, each node $n_i^S \in N_{A,i}^S$ is characterized by its location $n_i^S(x,y)$ and an explicit set B of resources, e.g., $B = \{CPU, memory, storage\}$ for $A = \{server\}$. Each resource b has a capacity of $CAP_b(n_i^S), b \in B$. Moreover, every substrate link $uv_i^S \in E_i^S, u, v \in N_i^S$ is associated with a bandwidth capacity $BW(uv_i^S)$. Similarly, every path between substrate nodes $u, v \in N_i^S$ is denoted as $P_i^S(u,v)$.

4.1.2 *Service Cluster*

We assume the size of replicated content to be W GB at each virtual surrogate server. Regarding the content outsourcing practice we assume a noncooperative pull-based approach, where user requests are redirected with a ratio of ρ [11] to their closest surrogate server. If there is a cache miss, surrogate servers pull content from the transit server. Most popular CDN providers (e.g., Akamai, Mirror Image) use this approach [1].

We assume N end-users $U = \{u_0, u_1, \ldots, u_N\}$ in the service cluster under consideration. Each user is associated with 1) a specific location determined by the user's coordinates $u_n(x,y)$, 2) a request rate $r(u_n)$, and 3) the size of the request w_n expressed in bytes. A user has a request of w_n which could be less, equal or greater than W due to the use of a web cache or multiple repeated requests without caching [5].

Suppose that the hit ratio of the edge server is ρ, then $\rho r(u_n)$ of a user's requests may be served by the edge server while the rest of the requests $(1-\rho)r(u_n)$ will be served by the transit server, where $r(u_n)$ requests requires a bandwidth of $w_n r(u_n)$. Hence, the bandwidth requirements along the path from the transit to the edge server are $(1-\rho)w_n r(u_n)$.

4.1.3 *Content Distribution Graph*

The intercluster content distribution graph is modeled as an undirected graph $G^D = (N^D, E^D)$. The set of $ASCs$ constitute the nodes of G^D. Essentially, $n_0^D \equiv ASC_0$ contains the origin server as set by the Content/CCDN provider. Moreover, $\forall ASC_k, k \neq 0$ there can be multiple content distribution paths $P(ASC_0,\ ASC_k)$. Each vertex $n_k^D \in N^D$ is associated with the geographic boundaries of the service cluster ASC_k^{area}, the geographical coordinates of the $ASC_k(x,y)$, and the size of the replicated content W. Each link $n_k n_l{}^D \in E^D, \forall k, l, k \neq l$ defines intercluster content distribution virtual links between ASC_k and ASC_l.

4.1.4 *CCDN Provisioning Costs*

Within the administrative domain of a single CP i, the opening cost of site $n_i^S \in N_{A,i}^S, A = \{server\}$ is estimated as the sum of two costs:

1. The cost of reserving computational resources at the cloud site (i.e., the virtual surrogate server). In this particular paradigm, the cost of storing the replicated content is taken into consideration. It is calculated by the unit storage cost $\mathcal{C}_{n_i^S}^{b=sto}$ per GB at the specific site charged by the CP multiplied by the replica size in GB.

TABLE 1
Terminology Reference Table

CCDN	Cloud-oriented Content Delivery Network.
CP	Cloud service Provider.
Transit network provider	Interconnects clouds belonging to different CPs with virtual links.
CCDN provider	Content Provider that deploys his own CDN over leased networked cloud resources.
Service Area	A geographically defined area where the CCDN provider grants content delivery services to end-users.
CCDN broker	Provides CCDN brokerage services to CCDN providers, based on the CCDN provider's service area, potential clientele, available CPs and transit network providers.
Service Cluster	Base CCDN service area unit as defined by the CCDN broker.
ASC	The Abstracted Service Cluster is a point representation of a service cluster, estimated as the center of mass of end-users with unit mass residing in the particular service cluster.

The unit storage cost is defined as a piecewise linear and convex function of the utilization of the disk resources over a time window $U(CAP_{b=sto}(n_i^S))$ at the cloud site, as defined in Section 5.

2. The cost of transferring the replicated content at site n_i^S from site $m_i^S \in N_{A,i}^S, A = \{server\}$; since interconnected cloud sites are considered, the required capacity of virtual links will be estimated for transferring cache missing content to the site n_i^S from site m_i^S at a unit communication cost of $\mathcal{P}_{m_i^S n_i^S}$ per Gbps charged by the CP. These virtual links correspond to either direct physical links between the cloud sites or substrate paths.

Finally, it is noted that the access cost for the user u_n that is assigned to the site/server n_i^S is calculated by the unit downloading cost $\mathcal{D}_{n_i^S}$ charged by the CP multiplied by the user request size in GB.

4.2 Service Cluster to CP Assignment

The proposed framework aims to establish the CCDN paradigm on a multicloud environment facilitated by multiple competing CPs. That requires essentially the placement of surrogate servers across the service area and the allocation of users to edge servers. For that purpose, as described in the previous paragraph, the service area is broken down to service clusters, and a content distribution graph is created spanning these service clusters. The CCDN Broker in the proposed hierarchical framework is responsible for the cost-efficient partitioning of the intercluster content distribution graph between the various CPs.

4.2.1 Content Distribution Graph: Partitioning Objective

Every CP $i \in I$, upon receiving the intercluster content distribution graph, identifies for each service cluster k the set of substrate servers that match each requested n_k^D in terms of storage capacity requirements W. The set is denoted as follows:

$$R_i^{n_k^D} = \{n_i^S \in N_{A,i}^S \subseteq N_i^S : A = \{server\},$$
$$CAP_{b=sto}(n_i^S) > W, n_i^S \ in \ ASC_k^{area}\}, \forall i \in I.$$

Similarly, each CP identifies for every requested $n_k n_l{}^D \in E^D$, intercluster candidate substrate links/paths in E_i^S having as endpoints members of the sets $R_i^{n_k^D}$ and $R_i^{n_l^D}$, where $k \neq l$. The set is denoted as follows:

$$L_i^{n_k n_l{}^D} = \{P_i^S(u,v) \subseteq E_i^S : u \in R_i^{n_k^D}, v \in R_i^{n_l^D}\}, \forall i \in I.$$

Therefore, it provides the CCDN broker with: 1) the corresponding unit storage costs for every cloud site $n_i^S \in R_i^{n_k^D} \forall k$, 2) the unit communication cost for every $P_i^S(u,v) \in L_i^{n_k n_l{}^D} \forall k, l, k \neq l$. Based on the aforementioned costs, an aggregate corresponding partitioning cost per CP i is established that will enable the CCDN broker to assign $ASCs$ to CPs.

The aggregate unit storage costs for CP i per service cluster k is calculated as the average unit storage cost of all cloud sites $n_{i}^S \in R_i^{n_k^D}$ divided by the size of substrate resource set $R_i^{n_k^D}$:

$$\mathcal{C}(R_i^{n_k^D}) = \frac{\overline{\mathcal{C}}_{n_i^S}^{b=stor}}{|R_i^{n_k^D}|}.$$

In the same fashion, the aggregate unit communication cost for CP i between $ASC_k \equiv n_k^D$ and $ASC_l \equiv n_l^D$ is calculated as the average unit communication cost of all $P_i^S(u,v) \in L_i^{n_k n_l{}^D}$ divided by the size of substrate resource set $L_i^{n_k n_l{}^D}$:

$$\mathcal{P}(L_i^{n_k n_l{}^D}) = \frac{\overline{\mathcal{P}}_{u_i^S v_i^S}}{|L_i^{n_k n_l{}^D}|}.$$

According to [25], interdomain communication cost is a magnitude higher than intradomain communication. Adhering to the specifications set by Houidi et al. [24], the intercloud provisioning cost $\forall n_k n_l{}^D \in E^D, k \neq l$ between CPs i and j, $i \neq j \in I$ is estimated as follows:

$$\mathcal{P}(L_{ij}^{n_k n_l{}^D}) = max(\mathcal{P}(L_i^{n_k n_l{}^D}), \mathcal{P}(L_j^{n_k n_l{}^D})) + p.$$

The penalty p represents this extra cost of a content distribution link spanning multiple CPs. In this framework and in line with the above observations in [24] and [25], p grants the interdomain link cost an order of magnitude larger than the maximum intradomain link provisioning cost of the two endpoint CPs.

The objective function with respect to provisioning cost is defined as follows:

$$\begin{aligned} \min \quad & \sum_{\forall i \in I} \sum_{\forall n_k^D \in N^D} \mathcal{C}(R_i^{n_k^D}) z_{n_k^D}^i \\ & + \sum_{\forall i \in I} \sum_{\forall n_k n_l{}^D \in E^D} \mathcal{P}(L_i^{n_k n_l{}^D}) y_{n_k n_l{}^D}^i \\ & + \sum_{\forall i} \sum_{\forall j, j \neq i} \sum_{\forall n_k n_l{}^D \in E^D} \mathcal{P}(L_{ij}^{n_k n_l{}^D}) y_{n_k n_l{}^D}^{ij}, \end{aligned} \tag{1}$$

where the binary variable $z_{n_k^D}^i = 1$ when the requested node $ASC_k \equiv n_k^D$ is assigned to CP i. Similarly, the binary variable $y_{n_k n_l{}^D}^i$ is set to 1 when the requested link $n_k n_l{}^D$ has been exclusively assigned to CP i. Finally, $y_{n_k n_l{}^D}^{ij} = 1, i \neq j$ when the requested link $n_k n_l{}^D$ spans multiple CPs, while i, j are the two endpoint CPs.

The first summation term in objective function (1) defines the cost of assigning the various ASCs to the CPs. The second and third summation terms, in the same fashion, determine intra- and interdomain link communication costs between the various $ASCs$ assigned to the various CPs, respectively, in the content distribution path.

4.2.2 Partitioning Techniques

Graph partitioning or graph bisection is a common NP-complete problem with a large number of applications such as mapping, and so on. Graph partitioning entails partitioning the vertices of a graph into a given number of disjoint subsets, so that the number of nodes in each subset is less than a given threshold and the number of cut edges is minimum. In the case of unbalanced partitions, the k-cut problem and the multiway partitioning problem are fundamental partitioning problems. For the k-cut problem, the goal is to find a minimum weight set of edges whose removal separates the graph into k disconnected components. When considering link weights the problem for an arbitrary k is NP-complete [26].

Graph partitioning in the context of NCE has been considered as a variant of the min k-cut problem by Xin et al. [3] while the use of appropriate metaheuristic techniques have been proposed by Leivadeas et al. [4]. Specifically, in the latter the use of the ILS metaheuristic for splitting a networked cloud request across multiple CPs is proposed. In the context of network virtualization environment, Houidi et al. [24] split a request for a virtual network topology across multiple infrastructure providers using both max-flow min-cut algorithms (Ford-Fulkerson algorithm) and linear programming techniques. In the particular study, the approaches proposed by Leivadeas et al. [4] and Houidi et al. [24] have been adopted.

4.3 Virtual Surrogate Server Placement per Service Cluster

For the actual surrogate server placement within a service cluster, two heuristics are introduced in the following; that is the VSP and a greedy replica placement algorithm inspired from SNA referred to as SNA-GVSP. With regards to the VSP, a mixed integer programming (MIP) formulation to solve the (virtual) surrogate placement problem at hand is proposed. SNA-GVSP is inspired by recent trends in adopting social network related features in the design of algorithms, network topologies, and protocols, resulting into socio-aware networked computing systems [27]. In the following two sections a detailed presentation of these two approaches is provided.

4.3.1 Virtual Surrogate Placement

For the actual virtual surrogate server placement within a service cluster, the objective is to find a replication strategy that will minimize the total cost by provisioning virtual surrogate servers on a subset of the available cloud sites so that each end user is assigned to at least one site and the QoS requirements for end-user requests are satisfied [5]. An approach based on the principles set by Chowdhury et al. [28] and Papagianni et al. [6] was properly adapted to the replica placement problem, taking into the consideration the specific requirements and constraints posed by the virtualized environment.

Content delivery graph. Assuming that the ASC_k has been assigned to $CP\ i$, the substrate network node set in the service cluster ASC_k is denoted as $N^S_{i,k}$, where $n^S_i \in N^S_i$ and $n^S_i\ in\ ASC^{area}_k$. Similarly, the substrate network edge set in the service cluster ASC_k is denoted as $E^S_{i,k}$, where $uv^S_i \in E^S_i$ and $u, v \in N^S_{i,k}$. For each user $u_n\ in\ ASC^{area}_k$, a pseudonode $p_i = \psi(u_n)$ is created with coordinates $u_n(x, y)$. Node set $N^S_{i,k}$ is augmented with these pseudonodes. The new node set is denoted as $N^{S'}_i$:

$$N^{S'}_{i,k} = \{n^S_i \in N^S_{i,k}\} \cup \{\psi(u_n) | \forall u_n \in U, u_n\ in\ ASC^{area}_k\}.$$

To support QoS in the CCDN, an appropriate measure of QoS for content delivery must be selected. The maximum (routing) distance between the edge server and the user captures the communication quality between the two nodes and can be measured by either hop count or delay [5]. In the same study, the authors claim that in the case that hop count information is not available, geographic distance can be used as an indicator of delay. The same approach is followed in this study. A maximum user to edge server distance D requirement is posed by the CCDN requester to the selected CPs.

To match this requirements, for each pseudonode a cluster $\Omega(\psi(u_n))$ can be created in the augmented node set $N^{S'}_i$ with radius D, where

$$\Omega(\psi(u_n)) = \{n^S_i \in N^S_{i,k} | dis(\psi(u_n), n^S_i) \leq D\}.$$

The user will be assigned eventually to a cloud site (virtual edge server) that falls within the limits of its cluster.

Each pseudonode is connected to every substrate node within its cluster with infinite bandwidth creating a set of pseudo-edges. These pseudoedges are added to the edge set $E^S_{i,k}$:

$$\begin{aligned} E^{S'}_{i,k} = E^S_{i,k} \cup \{&(\psi(u_n), n^S_i) | \\ &\forall u_n \in U, u_n\ in\ ASC^{area}_k, \forall n^S_i \in \Omega(\psi(u_n))\}. \end{aligned}$$

Pseudonodes and pseudoedges are connected to the substrate graph creating the undirected content delivery graph $G^{S'}_{i,k} = (N^{S'}_{i,k}, E^{S'}_{i,k})$.

Flow allocation problems and solution. Since a user can be satisfied either 1) by the first virtual edge server on the path e transit server in the service cluster or 2) by the origin r, via the transit server in the service cluster, we consider a commodity in the content delivery graph originated at the transit node $n^S_{0,i} \in N^S_{A,i}, A = \{server\} \subseteq N^S_{i,k}$ and ending at the pseudonode $p_i = \psi(u_n), \forall p_i \in N^{S'}_{i,k} \setminus N^S_{i,k}$ denoted as $(n^S_{0,i}, p_i)$. The resource allocation problem in the content delivery graph is formulated as a Mixed Integer Programming (MIP) $|U|$-commodity flow problem, where the communication demands among the $N^{S'}_{i,k}$ nodes are specified by a $|N^{S'}_{i,k}| \times |N^{S'}_{i,k}|$ demand matrix.

Assuming that traffic bifurcation is allowed, the commodity toward the pseudonode p_i will be routed via the set of paths $\mathbf{P}^{S'}_{i,k}(n_0, p)$. The available bandwidth capacity of a path $P^{S'}_{i,k}(n_0, p) \in \mathbf{P}^{S'}_{i,k}(n_0, p)$ in the content delivery graph is restricted to the capacity of the most loaded substrate link in the path—because the pseudoedges have infinite capacity—and must exceed the bandwidth requirements of the commodity:

$$\begin{aligned} BW(n^S_{0,i}, p_i) &= BW(n^S_{0,i}, \psi(u_n)) = (1 - \rho) w_n r(u_n) \\ BW(n^S_{0,i}, p_i) &\leq \sum_{\forall P^{S'}_{i,k}(n_0,p) \in \mathbf{P}^{S'}_{i,k}(n_0,p)} BW(P^{S'}_{i,k}(n_0, p)) \end{aligned}$$

$$\begin{aligned} BW(P^S_{i,k}(n_0, p)) &= \min_{\forall (uv^S_i) \in P^{S'}_{i,k}(n_0,p)} \mathcal{BW}(uv^S_i) \\ \mathcal{BW}(uv^S_i) &= BW(uv^S_i) - \sum_{\forall (n^S_{0,i}, j_i), (uv^S_i) \in P^{S'}_{i,k}(n_0,k)} BW(n^S_{0,i}, j_i). \end{aligned}$$

Solving the flow allocation problem, taking into consideration the traffic flows from the transit server to the virtual surrogates for content update, results into pseudonode to substrate node assignment. However, because MIP problems are known to be NP-hard [28] and, hence, computationally intractable, the optimal fractional solution is computed for the problem's Linear Programming relaxation. The relaxed problem can be solved by any suitable linear programming method, in polynomial time. A rounding technique is applied [28] to obtain the integer solution of the MIP problem. Assigning pseudonodes to a substrate node will determine the assignment of users to edge servers and the subset of cloud sites that will be utilized for the CCDN deployment.

Once the assignment of end users to virtual surrogates has been successfully completed, appropriate content delivery paths from the transit server to the surrogates are selected, by solving explicitly the flow allocation problem allowing traffic bifurcation (multicommodity flow problem). Alternatively, a shortest path algorithm can be applied to restrict each flow to a single path.

MIP formulation. For the sake of simplicity, scripts S and i will be omitted in the following. Moreover because the problem formulation reflects user assignment within a single service cluster, script k is also omitted.

Variables. $x_{uv}^{n_0p}$: A binary variable set to 1 if the traffic flow (n_0, p) is routed via the link $(u, v) \in E^{S'}$.

$f_{uv}^{n_0p}$: The amount of traffic for the traffic flow (n_0, p) routed over the link $(u, v) \in E^{S'}$ from u to v.

h_j: A binary variable set to 1 if the substrate node $j \in N_A^S \subseteq N^S, A = \{server\}$ is selected to host a virtual surrogate server.

Objective

$$\begin{aligned}\min \quad & \sum_{\forall p \in N^{S'} \setminus N^S} \sum_{uv \in E^S} \alpha \mathcal{P}_{uv} f_{uv}^{n_0p} \\ & + \sum_{j \in N_A^S \subseteq N^S, A = \{server\}} \beta W \mathcal{C}_j^{b=sto} h_j \\ & + \sum_{\forall p \in N^{S'} \setminus N^S} \sum_{w \in N^{S'} \setminus N^S} \sum_{j \in N_A^S \subseteq N^S, A=\{server\}} \gamma w_p \mathcal{D}_j x_{wj}^{n_0p}.\end{aligned} \tag{2}$$

Subject to:

$$\begin{aligned} & \sum_{v \in N^{S'}} f_{uv}^{n_0p} - \sum_{v \in N^{S'}} f_{vu}^{n_0p} = 0, \\ & \qquad \forall p \in N^{S'} \setminus N^S, n_0 \in N^S, \forall u \in N^{S'} \setminus \{n_0, p\} \\ & \sum_{v \in N^{S'}} f_{n_0v}^{n_0p} - \sum_{v \in N^{S'}} f_{vn_0}^{n_0p} = BW(n_0, p), \\ & \qquad \forall p \in N^{S'} \setminus N^S, n_0 \in N^S \\ & \sum_{v \in N^{S'}} f_{pv}^{n_0p} - \sum_{v \in N^{S'}} f_{vp}^{n_0p} = -BW(n_0, p), \\ & \qquad \forall p \in N^{S'} \setminus N^S, n_0 \in N^S \end{aligned} \tag{3}$$

$$\begin{aligned} & f_{uv}^{n_0p} + f_{vu}^{n_0p} \leq \mathcal{BW}(uv) x_{uv}^{n_0p} \\ & \qquad \forall u, v \in N^{S'}, \forall p \in N^{S'} \setminus N^S, n_0 \in N^S \end{aligned} \tag{4}$$

$$\sum_{\forall p \in N^{S'} \setminus N^S} \left(f_{uv}^{n_0p} + f_{vu}^{n_0p} \right) \leq \mathcal{BW}(uv) \; \forall u, v \in N^{S'}, n_0 \in N^S \tag{5}$$

$$\begin{aligned} & W x_{wj}^{n_0p} \leq CAP_{b=sto}(j), \quad \forall w, p \in N^{S'} \setminus N^S, \\ & \qquad \forall j \in N_A^S \subseteq N^S, A = \{server\}, n_0 \in N^S \end{aligned} \tag{6}$$

$$\sum_{\forall p \in N^{S'} \setminus N^S} \sum_{j \in \Omega(w) \subseteq N^S} x_{wj}^{n_0p} \leq 1, \quad \forall w \in N^{S'} \setminus N^S \tag{7}$$

$$\begin{aligned} & \sum_{\forall p \in N^{S'} \setminus N^S} \sum_{j \in V_A^S \cap \Omega(w) \subseteq N^S} x_{wj}^{n_0p} = 1, \\ & \qquad \forall w \in N^{S'} \setminus N^S, A = \{server\} \end{aligned} \tag{8}$$

$$\begin{aligned} & h_j \leq x_{wj}^{n_0p}, \quad \forall j \in V_A^S \subseteq N^S, A = \{server\}, \\ & \qquad n_0 \in N^S, \forall w, p \in N^{S'} \setminus N^S \end{aligned} \tag{9}$$

$$x_{uv}^{n_0p} = x_{vu}^{n_0p}, \quad \forall u, v \in N^{S'}, \forall p \in N^{S'} \setminus N^S, n_0 \in N^S \tag{10}$$

$$\begin{aligned} & x_{uv}^{n_0p} \leq \lceil f_{uv}^{n_0p} + f_{vu}^{n_0p} \rceil, \quad \forall u, v \in N^{S'}, \\ & \qquad \forall p \in N^{S'} \setminus N^S, n_0 \in N^S \end{aligned} \tag{11}$$

$$f_{uv}^{n_0p} \geq 0, \;\; \forall u, v \in N^{S'}, \forall p \in N^{S'} \setminus N^S, n_0 \in N^S \tag{12}$$

$$x_{uv}^{n_0p} \in \{0, 1\}, \;\; \forall u, v \in N^{S'}, \forall p \in N^{S'} \setminus N^S, n_0 \in N^S \tag{13}$$

$$h_j \in \{0, 1\}, \quad \forall j \in N_A^S \subseteq N^S, A = \{server\}. \tag{14}$$

- The first two summation terms in the objective function (relation 2) correspond to the overall cost of the substrate communication and storage resources allocated to the CCDN request. Specifically, the first term reflects the total cost of bandwidth allocated on substrate links for uploading replicated content, while the second term corresponds to the total cost of computational resources allocated to the set of selected virtual edge servers. The last summation corresponds to the access (downloading) cost of user (pseudonode p) that is assigned to virtual surrogate server at cloud site j. Every term of the summation is multiplied with an appropriate parameter α, β, and γ, respectively. These parameters are weights illustrating the importance of each summation term in the optimization process.
- Flow conservation is guaranteed via constraints set (3).
- Constraint sets (4) and (5) ensure that the sum of all flows that are routed through the substrate link uv does not exceed its available bandwidth capacity.
- Constraint set (6) ensures that the requested capacity $i \in I$ for a virtual edge server that is mapped to substrate node $w \in N_A^S \subseteq N^S, A = \{server\}$ does not exceed the substrate node's available capacity.
- Constraint sets (7) and (8) are adopted to ensure that a pseudonode (user) is assigned to only one substrate node of type $A = \{server\}$ that is selected to host a virtual edge server. Constraints sets (9) ensures that every time a pseudonode is assigned to a substrate node, this node is activated as a surrogate server.
- Constraint sets (10) and (11) ensure that the binary variable $x_{uv}^{n_0p}$ is set whenever there is traffic from the flow (n_0, p) routed over the substrate link (u, v) regardless of the direction.
- Constraints (12), (13), and (14) provide the domain constraints of the variables.

4.3.2 SNA Inspired Greedy Virtual Surrogate Placement

Cronin et al. [10] proposed the *transit node* heuristic for mirror placement on the Internet, where mirrors are placed on candidate hosts in descending order of out-degree. The heuristic was based on the assumption that nodes with the highest out-degrees can reach more nodes with lower latencies. We have adopted this assumption utilizing a popularity metric instead, according to SNA [27].

Popularity of nodes/users in networks has been successfully captured by the notion of centrality and respective metrics [29]. Centrality in graph theory and network analysis is a quantification of the relative importance of a vertex within the graph. It is implicitly related to the number of connections a node has toward its neighbors. Two basic categories can be identified: 1) centrality metrics related to the degree information of the node (e.g., degree centrality), and 2) centrality metrics based on the geodesic instances of nodes (e.g., SPBC). In this case, SPBC is selected for every n_i^S that is the ratio of the number of shortest paths traversing node n_i^S over the total number of shortest paths in the service cluster:

$$SPBC_{n_i^S} = \frac{SP_{u_i^S v_i^S}(n_i^S)}{SP_{u_i^S v_i^S}}, n_i^S, u_i^S, v_i^S \in R_i^{n_k^D}. \tag{15}$$

The SNA-GVSP heuristic is presented in Algorithm 1, adopting the notation described in the previous sections. The goal is to greedily assign end users to virtual surrogate servers, maximizing the average SPBC of the selected set of surrogates, so that each end user is assigned to at least one cloud site, and the QoS requirements for end user requests are satisfied. Subsequently, to identify the content delivery paths from the transit to surrogates servers within the service cluster, shortest paths are selected.

Algorithm 1. SNA-GVSP
```
for all $u_n$ where $u_n \; in \; ASC_k^{area}$ do
    $\Omega(u_n) = \{n_i^S \in R_i^{n_k^D} | dis(u_n, n_i^S) \le D\}$
    $MAX_{SPBC} = 0$
    for all $n_i^S \in \Omega(\psi(u_n))$ do
        if $SPBC_{n_i^S} \ge MAX_{SPBC}$ then
            $MAX_{SPBC} = SPBC_{n_i^S}$
            Assign $u_n$ to cloud site $n_i^S$.
        end if
    end for
    if site $n_i^S$ is CLOSED then
        Open $n_i^S$
    end if
end for
```

4.4 CCDN Deployment Cost

Solving the replica placement problem within each service cluster will define as described in Objective (2): 1) the overall cost for storing the replicas at the selected cloud sites, 2) the access cost of the users within the service cluster, and 3) the cost of updating the replicas from the transit server. Thus, the CCDN deployment cost denoted as C_{SC} is calculated by the CCDN cost per service cluster summed over the entire service area:

$$\begin{aligned} C_{SC} = &\sum_{\forall k} \sum_{\forall p \in N_{i,k}^{S'} \setminus N_{i,k}^S} \sum_{uv \in E_{i,k}^S} \mathcal{P}_{uv} f_{uv}^{n_{0,i}^S p} \\ &+ \sum_{j \in N_A^S \subseteq N^S, A=\{server\}} WC_j^{b=sto} h_j \\ &+ \sum_{\forall k} \sum_{\forall p \in N_{i,k}^{S'} \setminus N_{i,k}^S} \sum_{w \in N_{i,k}^{S'} \setminus N_{i,k}^S} \sum_{j \in N_{A,i}^S \subseteq N_{i,k}^S, A=\{server\}} w_p \mathcal{D}_j x_{wj}^{n_{0,i}^S p}. \end{aligned}$$

The aforementioned cost define the partial cost of deploying CCDNs in service clusters. The overall cost of the CCDN solution is complemented by the intercluster and possibly intercloud content distribution paths from the origin server at cluster $k = 0$ to transit servers denoted as C_{ISC}:

$$\begin{aligned} C_{ISC} = &\sum_{\forall k \neq 0} \sum_{\forall n_{0,j}^S n_{0,i}^S \in P(ASC_0, ASC_k)} \mathcal{P}_{n_{0,j}^S n_{0,i}^S} \\ &\sum_{\forall p \in N_{i,k}^{S'} \setminus N_{i,k}^S, n_{0,i}^S \in N_{i,k}^S} BW(n_{0,i}^S p). \end{aligned}$$

5 Performance Evaluation

In this section, the efficiency of the proposed approach is evaluated via modeling and simulation. A discrete event Java-based simulator (simulator for controlling virtual infrastructures—CVISim) was used for that purpose [6]. CVISim is based on the JUNG software library [30] that provides a common and extendible language for the manipulation, analysis, and visualization of data that can be represented as a graph. The CPLEX Callable Library for Java (Academic Research Edition v12.2) has been used to solve the relaxed MIP problem [31]. All the simulations were executed on a Personal Computer Quad-Core AMD Opteron @ 2.5 GHz with 6.00 GB of RAM.

To allow for a thorough performance evaluation and impact assessment of the various components of the introduced paradigm, in the following we compare different approaches and methodologies with respect to: 1) the content distribution graph partitioning technique for service cluster to CP assignment and 2) the adopted scheme for virtual surrogate server placement per service cluster.

In particular, the adopted ILS metaheuristic [4] is compared against the MaxFlow heuristic presented in [24] to obtain a suboptimal but feasible service cluster to CP assignment. Moreover, for benchmarking purposes both algorithms are compared against an exact solution, where the partitioning problem is formulated as a quadratic programming problem and through appropriate transformation a linear integer program is extracted [24]. The linear integer program is solved using the branch and bound algorithm. For solving the virtual surrogate server placement problem, the approaches proposed in Section 4.3 are utilized and assessed, that is VSP heuristic and SNA-GVSP along with greedy user (GU) and greedy site (GS) heuristics presented in [5]. The GU is based on the assignment of end users to cloud sites with the lowest cost while considering to assign first end users with the least number of potential sites. On the other hand, GS is based on finding a cloud site with a maximum utility (based on cost and total requested

TABLE 2
Evaluated Approaches

Approach	Graph Partitioning	Surrogate Placement
Exact:SNA-GVSP	Exact	SNA-GVSP
ILS:SNA-GVSP	ILS	SNA-GVSP
MaxFlow:SNA-GVSP	MaxFlow	SNA-GVSP
Exact:VSP	Exact	VSP
Exact:GU	Exact	GU
Exact:GS	Exact	GS

traffic volume) and the total assignment of all its potential users to it.

The sets of approaches considered and evaluated are presented in Table 2.

5.1 Experimentation Setup

Two separate experiments were performed: *Experiment 1*, with the goal to evaluate the performance of the proposed content distribution graph partitioning approach in terms of the quality of the solution obtained and time complexity; and *Experiment 2*, with the goal of evaluating the overall CCDN solution and the impact of the adopted scheme for virtual surrogate server placement per service cluster in conjunction with the partitioning approach. The two experiments were arranged as follows:

- *Experiment 1*. To evaluate the performance of the three partitioning approaches, ILS, MaxFlow, and exact partitioning algorithm are applied, while all utilizing the same replica placement algorithm (i.e., SNA-GVSP). Regarding available clouds (substrate resources), three *scenarios* were examined in detail; in the first scenario, (*scenario 1*) five CPs are known to the CCDN Broker while in the second one (*scenario 2*) 10 CPs are assumed. An incremental set of end users was used for CCDN requests (500-1,000 users with a step of 100 users) in *scenarios 1* and 2. In *scenario 3*, the number of end users is set to 500 while an incremental set of CPs (5-20 with a step of five CPs) was known to the CCDN Broker. For every end-user set, 10 simulations were executed while each simulation includes 10 incoming CCDN requests. These three scenarios allow for the evaluation of the performance of the proposed framework with respect to several key metrics (e.g., cost, scalability, computational time) under different numbers of CPs and varying sizes of end users.
- *Experiment 2*. To assess the performance of the virtual surrogate server placement scheme, the four heuristics described before are applied and compared (i.e., VSP, SNA-GVSP, GU, and GS) while adopting an exact solution for the partitioning phase. Ten simulations were executed for each one of them while each simulation includes 10 incoming CCDN requests. The number of users for each CCDN request is uniformly distributed in the interval [300-500]. Regarding the available cloud providers, five CPs are assumed to be known the CCDN Broker.

Regarding both experiments, the specifics of the available substrate physical topologies, CCDN request characteristics, and CCDN costs are provided in the following.

Substrate and service cluster creation. Substrate network topologies per CP were randomly generated as partial mesh topologies with 50 nodes in a (3×3) service area grid [28]. Two types of nodes have been used (servers and routers) to provide a more realistic networked cloud environment. Each node is characterized from its properties such as type (server, router), operating system (Windows, JUNOS, and so on) and virtualization environment (Xen, VMware, JUNOS specific, and so on) as well as geographical coordinates. Moreover, the nodes have different resources based on their type, e.g., CPU computational capacity, memory, disk space for servers. The probability of generating a specific type of node is 90 percent for servers and 10 percent for routers. Routers are only used for forwarding purposes. The storage (GB) and bandwidth (GBps) capacities of the substrate nodes and links were real numbers uniformly distributed between 50 and 100. Each subgrid is considered a service cluster.

CCDN requests. User requests arrive according to a Poisson process with an average rate of four request per 100 time units, while they are assumed to have exponentially distributed lifetime with an average of 1,000 time units. Each user has an average rate of 25 Kbps with an average request size of 25 KB. The replica size is considered to be on average 150 MB.

CCDN costs. Typical cloud costs for 2010 are presented in [5]. As we need to use more than four CPs for the different service clusters, we randomly generate cloud costs for the different cost types within the range of prices presented above. The unit storage cost in this study is considered a convex piecewise linear function of the disk resources utilization for the particular cloud site, and it is associated to a flat cost per unit denoted as $\mathcal{C}^{b=sto}_{n_i^S,flat}$, which is uniformly distributed in the interval [0.15-0.25]:

$$\mathcal{C}^{b=sto}_{n_i^S} = \begin{cases} \mathcal{C}^{b=sto}_{n_i^S,flat} & \text{for} \quad U\big(CAP_{b=sto}(n_i^S)\big) \leq 25\% \\ 1.2\mathcal{C}^{b=sto}_{n_i^S,flat} & \text{for} \quad 25\% \leq U\big(CAP_{b=sto}(n_i^S)\big) \leq 50\% \\ 1.5\mathcal{C}^{b=sto}_{n_i^S,flat} & \text{for} \quad 50\% \leq U\big(CAP_{b=sto}(n_i^S)\big) \leq 75\% \\ 2\mathcal{C}^{b=sto}_{n_i^S,flat} & \text{for} \quad 75\% \leq U\big(CAP_{b=sto}(n_i^S)\big) \leq 100\%. \end{cases}$$

The unit cost for communication between two cloud sites is uniformly distributed in the interval [0.135-0.18]. In the same way the unit downloading cost is set at 0.17, in line with the study in [5].

Finally, given the aforementioned experimentation setup and corresponding costs, appropriate weights α, β and γ in Objective 2 are selected based on extensive experimentation so that the different terms in the objective are of equivalent importance, without any of them dominating the overall optimization.

5.2 Comparative Results

Metrics. To quantify the performance of these approaches, we use the metrics presented in Table 3. *Mapping Cost* is a commonly used metric for evaluating the mapping solution on behalf of the CP [28], [4]. The mapping cost reflects the cost for allocating substrate resources to CCDN

TABLE 3
Evaluation Metrics

Mapping Cost	$C_{map} = \sum_{\forall k} \sum_{\forall p \in N_{i,k}^{S'} \setminus N_{i,k}^{S}} \sum_{uv \in E_{i,k}^{S}} f_{uv}^{n_{0,i}^{S} p} + \sum_{\forall k} \sum_{j \in N_{A,i}^{S} \subseteq N_{i,k}^{S}, A=\{server\}} Wh_j + \sum_{\forall k} \sum_{\forall p \in N_{i,k}^{S'} \setminus N_{i,k}^{S}} w_p$
Partitioning Cost	$\sum_{\forall i \in I} \sum_{\forall n_k^D \in N^D} \mathcal{C}(R_i^{n_k^D}) z_{n_k^D}^i + \sum_{\forall i \in I} \sum_{\forall n_k n_l{}^D \in E^D} \mathcal{P}(L_i^{n_k n_l{}^D}) y_{n_k n_l{}^D}^i + \sum_{\forall i} \sum_{\forall j, j \neq i} \sum_{\forall n_k n_l{}^D \in E^D} \mathcal{P}(L_{ij}^{n_k n_l{}^D}) y_{n_k n_l{}^D}^{ij}$
Computation Time	Computation time required to partition a content distribution graph among the number of available CPs.
CCDN Deployment Cost	$C_{SC} + C_{ISC}$ (as defined in subsection 4.4)
Number of Surrogate Servers	The number of cloud sites that are selected by the replica placement algorithm to host a virtual surrogate server.
$SPBC_{CCDN}$	Shortest Path Betweeness Centrality for a particular CCDN solution is defined as the average of individual SPBCs (15) of the selected cloud sites comprising the final CCDN solution.
Path Length	The average number hops between end-users and the transit server in a service cluster, averaged over all service clusters.

requests. With regards to the CCDN broker, *Partitioning Cost* reflects the efficiency of the partitioning technique adopted to assign service clusters to CPs. Computational efficiency of each partitioning scheme is quantified by measuring *Computation Time*. With regards to CCDN deployment and content replication for the Content/CCDN provider, the overall *CCDN Deployment Cost* along with the *Number of Surrogate Servers* selected and the *Path Length* within a service cluster indicate the efficiency of the adopted replica placement approach. Finally, the *SPBC* for the CCDN solution measures the impact of using social characteristics of the infrastructure in the particular server placement problem.

Experiment 1 Results. The partitioning solution is evaluated on the basis of the relative average partitioning cost and computation time as described in Table 3. The relative partitioning cost is defined as the ratio of the partitioning cost for ILS:SNA-GVSP and MaxFlow:SNA-GVSP over the Exact:SNA-GVSP.

In *scenario 1*, the content distribution graph is split among five CPs. The relative partitioning cost (Fig. 4) for ILS:SNA-GVSP ranges from approximately 1.5 to 3.5 percent and for MaxFlow:SNA-GVSP from 6 to 8 percent. In terms of computation time, both heuristics outperform the Exact:SNA-GVSP as shown in Table 4.

In *scenario 2*, the content distribution graph is split among 10 CPs. The relative average partitioning cost (Fig. 5) for ILS:SNA-GVSP ranges from approximately 3.5 to 6.5 percent and for MaxFlow:SNA-GVSP from 11 to 15 percent. Regarding computation time, in Table 5, we notice that for the Exact:SNA-GVSP the median is approximately 660 msec. Following a similar trend for MaxFlow:SNA-GVSP is 4.8 msec, and for ILS:SNA-GVSP is approximately 2.6 msec. The increase in time exhibits the dependency of the particular metric on the number of CPs. This is evident also in Fig. 7, where the computation time is plotted for an increasing number of CPs.

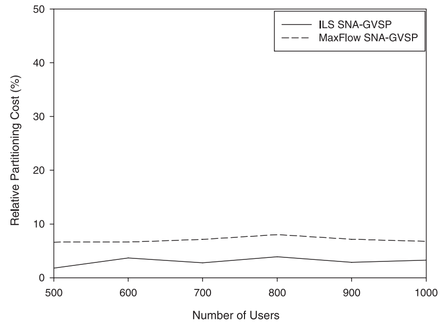

Fig. 4. *Scenario 1*: Relative partitioning cost for five CPs.

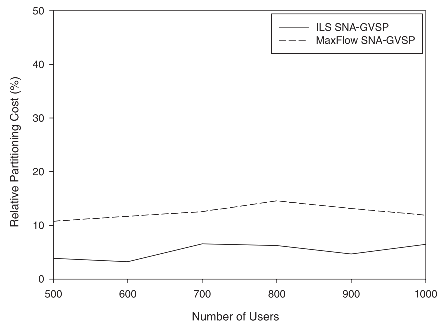

Fig. 5. *Scenario 2*: Relative partitioning cost for 10 CPs.

TABLE 4
Computational Time (msec)—Five CPs

Approach	Min	Max	Median
ILS:SNA-GVSP	0.96	1.31	1.12
MaxFlow:SNA-GVSP	1.08	2.10	1.39
Exact:SNA-GVSP	172.95	245.87	205.47

TABLE 5
Computational Time (msec)—10 CPs

Approach	Max	Min	Median
ILS:SNA-GVSP	1.86	3.2	2.60
MaxFlow:SNA-GVSP	3.60	5.92	4.82
Exact:SNA-GVSP	646.76	670.14	657.765

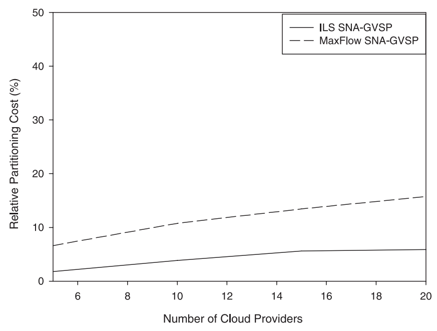

Fig. 6. *Scenario 3*: Relative partitioning cost.

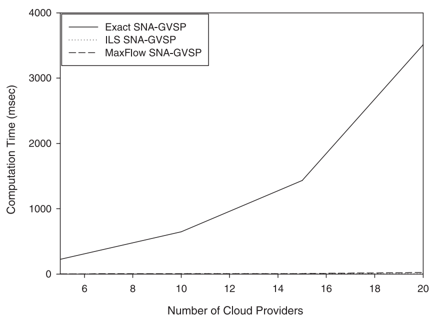

Fig. 7. *Scenario 3*: Computational time (msec).

The aforementioned observations are reenforced by the results of *scenario 3*. According to Fig. 6, the relative cost increases for both heuristics, with an increasing number of CPs. However, the efficiency of the ILS metaheuristic is illustrated by a smoother slope in the cost graph. Moreover, an exponential increase in computation time is noted for the exact partitioning solution, while for both heuristics, time increases linearly with the number of CPs and kept at very low levels—signifying the scalability of the adopted approaches (Fig. 7).

By observing Figs. 4 and 5, we notice that the partitioning scheme is not dependent on the number of end users. The user set is reduced to the set of $ASCs$; hence, the resulting content distribution graph that is split among CPs depends on the definition of the service cluster and the number of competing CPs over the same service area, and not on the number of actual end users.

Experiment 2 Results. As explained before, the objective of this experiment is to evaluate the performance of the replica placement approaches adopted (assuming the same selected partitioning scheme). Therefore, for demonstration purposes, Exact:VSP, Exact:SNA-GVSP, Exact:GU, and Exact:GS are selected and utilized as defined in Table 2. The metrics adopted are the ones related to the CP and CCDN deployment, along with $SPBC_{CCDN}$ metric, as defined in Table 3.

According to Table 6, VSP outperforms all greedy heuristics in terms of all individual metrics—apart from $SPBC_{CCDN}$ that is a specific objective of SNA-GVSP. Results reveal the efficiency of using a more sophisticated surrogate placement algorithm and the proposed formulation, as described in Section 4.3.1. Naturally, the reduced number of selected surrogate servers and path length reduce the deployment cost for the Content/CCDN provider and the cost of allocating substrate resources to incoming CCDN requests for the CP. On the other hand, one must note that adopting SNA features in a surrogate placement scheme contribute to the enhancement of the inherent characteristics of the CCDN solution; a fact evident from the comparison of the SNA-GVSP and GU and GS algorithms. SNA-related criteria drive the algorithm to select a smaller number of surrogate servers, reusing "popular" substrate nodes leading to smaller mapping and CCDN deployment costs. We also notice that by selecting substrate nodes that exhibit larger $SPBC$ leads to smaller paths in comparison to the GU and GS heuristics. According to the $SPBC$ definition, they have the largest number of shortest paths traversing them on the service cluster; therefore, it is more likely that they will be closer to the transit node. GU and GS have similar performance with GS, however, to be slightly better than GU. The Exact:VSP outperforms Exact:SNA-GVSP with regards to path length, because the VSP is directly formulated to minimize the paths of transit to surrogate servers.

6 Conclusions

In this work, a hierarchical framework is presented for deploying CCDNs. The problem is studied in a multi-provider networked cloud environment and is essentially broken down to appropriately defined graph partitioning and replica placement subproblems. Within the proposed framework, the identified potential clientele of a CCDN requester is projected to an abstract content distribution graph. Graph partitioning heuristics are adapted to provide a cost efficient splitting of the content distribution graph among existing cloud providers. Furthermore, to treat the replica placement problem within the emerging paradigm of network cloud environment, novel replica placement algorithms designed for the virtualized environment and/or inspired by SNA are introduced, leading to a concrete regional CCDN over networked cloud sites. The complete CCDN solution is complemented by intercloud links identified by the outcome of the content distribution graph partitioning.

Numerical results obtained through modeling and simulation allow for the detailed assessment of the

TABLE 6
Algorithm Comparison

Algorithm	Mapping Cost	CCDN Deployment Cost	Number of Surrogate Servers	$SPBC_{CCDN}$	Path Length
Exact:VSP	5.03	1.022	28.91	0.010	1.53
Exact:SNA-GVSP	5.47	1.110	31.19	0.054	2.47
Exact:GU	5.60	1.135	31.92	0.011	2.59
Exact:GS	5.59	1.130	31.90	0.011	2.58

proposed paradigm and demonstrate the effectiveness of the proposed approaches. Specifically, regarding the partitioning methodology, detailed evaluations reveal the scalability of adopted heuristics and the efficiency of the proposed partitioning cost model, witnessed by the acquired CCDN solutions. Following graph partitioning, a sophisticated replica placement algorithm for the networked cloud environment is proposed and compared against a greedy heuristic. Corresponding results demonstrate the effectiveness of the proposed approach, while they provide valuable insight on the positive potential effect of adopting social network features for designing CCDNs.

ACKNOWLEDGMENTS

This work was partially supported by the European Community's Seventh Framework Programme (FP7), OpenLab project under grant agreement number INFSO-ICT-287581.

REFERENCES

[1] A-M.K. Pathan, R. Buyya, and A. Vakali, "Content Delivery Networks: State of the Art, Insights, and Imperatives," *Content Delivery Networks,* vol. 9, no. 1, pp. 3-32, 2008.
[2] J. Broberg, R. Buyya, and Z. Tari, "MetaCDN: Harnessing 'Storage Clouds' for High Performance Content Delivery," *J. Network and Computer Applications,* vol. 32, no. 5, pp. 1012-1022, Sept. 2009.
[3] Y. Xin, I. Baldine, A. Mandal, C. Heermann, J. Chase, and A. Yumerefendi, "Embedding Virtual Topologies in Networked Clouds," *Proc. Sixth Int'l Conf. Future Internet Technologies (CFI '11),* pp. 26-29, June 2011.
[4] A. Leivadeas, C. Pappagianni, and S. Papavassiliou, "Efficient Resource Mapping Framework over Networked Clouds via Iterated Local Search Based Request Partitioning," to be published in *IEEE Trans. Parallel and Distributed Systems,* 2012.
[5] F. Chen, K. Guo, J. Lin, and T.F.L. Porta, "Intra-Cloud Lightning: Building CDNs in the Cloud," *Proc. IEEE INFOCOM,* pp. 433-441, Mar. 2012.
[6] C. Papagianni, A. Leivadeas, S. Papavassiliou, V. Maglaris, C. Cervello-Pastor, and A. Monje, "On the Optimal Allocation of Virtual Resources in Cloud Computing Networks," to be published in *IEEE Trans. Computers.*
[7] A. Leivadeas, C. Pappagianni, and S. Papavassiliou, "Socio-Aware Virtual Network Embedding," *IEEE Network Magazine,* vol. 26, no. 5, pp. 35-43, Sep./Oct. 2012.
[8] T.A. Neves, L.M.A. Drummond, L.S. Ochi, C. Albuquerque, and E. Uchoa, "Solving Replica Placement and Request Distribution in Content Distribution Networks," *Electronic Notes in Discrete Math.,* vol. 36, no. 1, pp. 89-96, Aug. 2010.
[9] L. Qiu, V.N.V. Padmanabhan, and G.M.G. Voelker, "On the Placement of Web Server Replicas," *Proc. IEEE INFOCOM,* pp. 1587-1596, Apr. 2001.
[10] E. Cronin, S. Jamin, C. Jin, A.R. Kurc, D. Rax, and Y. Shavitt, "Constraint Mirror Placement on the Internet," *IEEE J. Selected Areas in Comm.,* vol. 20, no. 7, pp. 1369-1382, Sept. 2002.
[11] X. Jia, D. Li, X. Hu, W. Wu, and D. Du, "Placement of Web-Server Proxies with Consideration of Read and Update Operations on the Internet," *Computer J.,* vol. 46, no. 4, pp. 378-390, 2003.
[12] T. Bektas, I. Ouveysi, R. Buyya, M. Pathan, and A. Vakali, "Mathematical Models for Resource Management and Allocation in CDNs," *Content Delivery Networks,* vol. 9, no. 2, pp. 225-250, 2008.
[13] P. Radoslavov, R. Govindan, and D. Estrin, "Topology-Informed Internet Replica Placement," *Computer Comm.,* vol. 25, no. 4, pp. 384-392, Mar. 2002.
[14] I. Cidon, S. Kutten, and R. Soffer, "Optimal Allocation of Electronic Content," *Computer Networks,* vol. 40, no. 2, pp. 205-218, Oct. 2002.
[15] K. Kalpakis, K. Dasgupta, and O. Wolfson, "Optimal Placement of Replicas in Trees with Read, Write, and Storage Costs," *IEEE Trans. Parallel and Distributed Systems,* vol. 12, no. 6, pp. 628-637, June 2001.
[16] X. Tang and J. Xu, "QoS-Aware Replica Placement for Content Distribution," *IEEE Trans. Parallel and Distributed Systems,* vol. 16, no. 10, pp. 921-932, Oct. 2005.
[17] H. Wang, P. Liu, and J. Wu, "A QoS-Aware Heuristic Algorithm for Replica Placement," *Proc. IEEE/ACM Seventh Int'l Conf. Grid Computing,* pp. 96-103, Sept. 2006.
[18] G. Rodolakis, S. Siachalou, and L. Georgiadis, "Replicated Server Placement with QoS Constraints," *IEEE Trans. Parallel and Distributed Systems,* vol. 17, no. 10, pp. 1151-1162, Oct. 2006.
[19] B. Li, M.J. Golin, G.F. Italiano, and X. Deng, "On the Optimal Placement of Web Proxies in the Internet," *Proc. IEEE INFOCOM,* vol. 3, pp. 1282-1290, Mar. 1999.
[20] Y. Jin, Y. Wen, G. Shi, G. Wang, and A. Vasilakos, "CoDaaS: An Experiment Cloud-Centric Content Delivery Platform for User-Generated Contents," *Proc. Int'l Conf. Computing, Networking and Comm. (ICNC),* pp. 934-938, Feb. 2012.
[21] S. Srinivasan, J.W.J. Lee, D. Batni, and H. Schulzrinne, "Active-CDN: Cloud Computing Meets Content Delivery Networks," technical report, Columbia Univ., 2012.
[22] N. Carlsson, G. Dan, D. Eager, and A. Mahanti, "Tradeoffs in Cloud and Peer-Assisted Content Delivery Systems," *Proc. IEEE Int'l Conf. Peer-to-Peer Computing,* pp. 249-260, Sept. 2012.
[23] "Operator Opportunities in Cloud Service Delivery," white paper, Ericsson, Feb. 2012.
[24] I. Houidi, W. Louati, W.B. Ameur, and D. Zeghlache, "Virtual Network Provisioning across Multiple Substrate Network," *Computer Networks,* vol. 55, no. 2, pp. 1011-1023, 2011.
[25] F. Zaheer, J. Xiao, and R. Boutaba, "Multi-Provider Service Negotiation and Contracting in Network Virtualization," *Proc. IEEE Network Operations and Management Symp. (NOMS),* pp. 471-478, June 2010.
[26] O. Goldschmidt, "Polynomial Algorithm for the k-Cut Problem," *Proc. 29th Ann. Symp. Foundations of Computer Science,* pp. 444-451, Oct. 1988.
[27] D. Katsaros, N. Dimokas, and L. Tassiulas, "Social Network Analysis Concepts in the Design of Wireless Ad Hoc Network Protocols," *IEEE Network,* vol. 24, no. 6, pp. 23-29, Nov./Dec. 2010.
[28] M. Chowdhury, M.R. Rahman, and R. Boutaba, "ViNEYard: Virtual Network Embedding Algorithms with Coordinated Node and Link Mapping," *IEEE/ACM Trans. Networking,* vol. 20, no. 1, pp. 206-219, Feb. 2012.
[29] M.E.J. Newman, *Networks: An Introduction.* Oxford Univ. Press, 2010.
[30] JUNG 2.0.1, http://jung.sourceforge.net, 2013.
[31] IBM ILOG CPLEX Optimizer, http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/, 2013.

Chrysa Papagianni received the diploma and the PhD degree in electrical and computer engineering from the National Technical University of Athens, Greece, in 2003 and 2009, respectively. She has been a senior R&D associate with the Network Management and Optimal Design Laboratory, National Technical University of Athens, since 2010. Her research interests lie in the areas of cloud computing, virtualization, network optimization and management, and service provisioning.

Aris Leivadeas received the diploma degree in electrical and computer engineering from the University of Patras, Greece, in 2008 and the MSc degree in mobile and personal communications from King's College London, United Kingdom, in 2009. He has been a member of the Network Management and Optimal Design Laboratory, National Technical University of Athens, since 2010, where he is currently working toward the PhD degree. His research interests include cloud computing, virtualization, and optimization.

Symeon Papavassiliou is currently an associate professor in Electrical and Computer Engineering Department at the National Technical University of Athens. From 1995 to 1999, he was a senior technical staff member at AT&T Laboratories, New Jersey. In August 1999 he joined the Electrical and Computer Engineering Department at the New Jersey Institute of Technology, where he was an associate professor until 2004. He has an established record of publications in his field of expertise, with more than 200 technical journal and conference published papers. He received the Best Paper Award in IEEE INFOCOM'94, the AT&T Division Recognition and Achievement Award in 1997, the US National Science Foundation Career Award in 2003, the Best Paper Award in IEEE WCNC 2012, and the Excellence in Research Award in Greece in 2012. His main research interests lie in the area of analysis and optimization of mobile and distributed systems. He is an associate editor for *IEEE Transactions on Parallel and Distributed Systems* and a technical editor for *IEEE Wireless Communications Magazine*. He is a senior member of the IEEE.