Chapter 9
Mathematical Models for Resource Management and Allocation in CDNs

Tolga Bektaş and Iradj Ouveysi

9.1 Introduction

To achieve a cost-effective content delivery strategy that a CDN provider seeks, the resources of a CDN, consisting primarily of the network infrastructure, the content to be distributed in the network, and the caching servers (holding a set of objects) that are to be distributed throughout the network, need to be efficiently managed and allocated. Now that the customer preferences have begun to play a key role in provisioning CDN services, the provider should also take into account some specific Quality-of-Service (QoS) considerations in planning its content delivery activities.

Mathematical modeling is a powerful and an effective tool that can be used to efficiently solve the resource allocation and management problems in a CDN. The aim of this chapter is to demonstrate how a variety of problems of this domain can be formulated in terms of mathematical models, and how the resulting models can be solved efficiently using the available techniques. For this purpose, we review the recent literature in the next section; simultaneously describe the relevant work and present the associated mathematical models. Solution techniques that we believe to be appropriate for the resolution of these models are described in Sect. 9.3, where we will also illustrate how these techniques can be applied to some of the models presented in this chapter. Section 9.4 offers some new models for a number of CDN architectures, and Sect. 9.5 presents their performance results. We offer our thoughts for practitioners in Sect. 9.6, provide directions for further research in Sect. 9.7 and state our conclusions in Sect. 9.8.

Tolga Bektaş
School of Management, University of Southampton, Highfield, Southampton SO17 1BJ, UK,
e-mail: T.Bektas@soton.ac.uk

Iradj Ouveysi
Honorary research fellow, Electrical and Electronic Engineering Department, The University of Melbourne, Vic. 3010, Australia, e-mail: iradjouveysi@yahoo.co.uk

R. Buyya et al. (eds.), *Content Delivery Networks*,

9.2 Related Work

In this section, we review the relevant literature that offer mathematical models for resource management and allocation in CDNs, and at the same time present the related mathematical models. Before presenting the models, we define the terminology that will be used throughout the chapter. The term *content* refers to any kind of information that is available on the World Wide Web to public such as Web pages, multimedia files and text documents. *Object* refers to a specific item of the content, such as a sound file or a text document. The *content provider* issues content for the access of others, and a *CDN provider* (most often a commercial one) disseminates the content on behalf of the content provider. There may be a few exceptions where the content provider takes care of the content delivery itself, but in this chapter we shall assume that this task is outsourced to a CDN provider by the content provider. The term *client* refers to an individual (either person or corporate) who issue requests for content. The CDN providers hold either the whole or a subset of the content in *caching servers* that are deployed throughout the telecommunications network, through or by which client requests are served. For all the models that follow, we assume a given complete network $G = (V, E)$, where V is the set of nodes and $E = (\{i, j\} : i, j \in V)$ is the set of links. The node set V is further partitioned into three nonempty subsets I, J and S, where I is the set of clients, J is the set of nodes where caching servers[1] are (or can be) installed, and S is the set containing the origin servers ($S = \{0\}$ in case of a single origin server). All the models presented in this section use a common notation framework that is given in Table 9.1.

9.2.1 The Fundamental Problems

There are three fundamental problems that arise in designing a cost-effective delivery network on which most of the more complex mathematical models proposed within the CDN domain are based on. This section presents a brief overview of these three problems.

Caching server placement problem. Given an existing infrastructure, the *caching server placement problem* consists of optimally placing a given number of servers to a given number of sites, such that a cost function (overall flow of traffic, average delay the clients experience, and total delivery cost) is minimized [22]. Qiu et al. [26] offer two well-known mathematical models to the caching server placement problem, namely the uncapacitated p-median (e.g. see [2]) and facility location problems (e.g. see [11]). The problem of placing transparent caches is described by Krishnan et al. [19]. The objective function considered in this study is interesting in that it

[1] We have chosen to use the term *caching server* as opposed to *proxy server* to avoid confusion as the concept of proxy was originally used to perform filtering and request relay, etc., but this is not practical any more as web pages are changing fast and dynamically. It is therefore more appropriate to refer to the additional servers as caching servers, or simply caches.

Table 9.1 Summary of the notation used in the chapter

Sets	
I	Set of clients ($I \subset V$)
J	Set of nodes on which caching servers can be established ($J \subset V$)
S	Set of origin servers ($S \subset V$)
K	Set of objects
Parameters	
b_k	Size of object $k \in K$
λ_i	Aggregate request rate of client $i \in I$
h_{ij}	Fraction of the request originating from node $i \in I$ that can be satisfied by $j \in J$
c_{ij}	'Distance' between two nodes $i \in V$ and $j \in V$ (i.e. number of hops, cost)
f_{ij}	Amount of flow between a client $i \in I$ and a caching server $j \in J$
d_{ik}	Request rate of client $i \in I$ for object $k \in K$ per unit time
f_j	Cost of operating a caching server on node $j \in J$
ψ_j	Cost of placing an object on a caching server $j \in J$
β_j	Cost per unit of bandwidth required by caching server $j \in J$
δ_j	Cost per unit of processing power required by caching server $j \in J$
C_j	Units of processing power available at a caching server $j \in J$
s_j	Storage capacity of a caching server $j \in J$
l_k	Amount of bandwidth consumed by object $k \in K$
pw_k	Amount of processing power consumed by object $k \in K$
ρ_k	Revenue generated by providing object $k \in K$ to the clients
L_{ij}	Latency between two nodes $i \in V$ and $j \in V$
Δ_d	Upper bound on latency (may be defined in terms of a link or an object, or both
p_{jk}	Probability that object $k \in K$ exists at caching server $j \in K$
Variables	
$\vartheta_{jk} \in \{0,1\}$	1, if request for object $k \in K$ is directed to caching server $j \in J$; 0, otherwise
$x_{ij} \in \{0,1\}$	1, if client $i \in I$ is assigned to caching server $j \in J$; 0, otherwise
$x_{ijk} \in \{0,1\}$	1, if object $k \in K$ requested by client $i \in I$ is held at caching server $j \in J$; 0, otherwise
$y_j \in \{0,1\}$	1, if a caching server is active at node $j \in J$; 0, otherwise
$z_{jk} \in \{0,1\}$	1, if object $k \in K$ is placed on a caching server $j \in J$; 0, otherwise
$u_k \in \{0,1\}$	1, if object $k \in K$ is replicated (on any caching server $j \in J$); 0, otherwise
$r^k_{ji} \geq 0$	fraction of accesses for object $k \in K$ directed to server $j \in J$ requested by client $i \in I$

considers the case where the requested content is *not* found in a specific caching server. Thus, the cost of serving client i from server j is given by the following:

$$cost(i,j) = f_{ij}(h_{ij}c_{ij} + (1-h_{ij})(c_{ij} + c_{js})), \tag{9.1}$$

This cost function (9.1) is a good representation of how a CDN operates and has been used in formulating other problems (e.g. see [6, 17]).

Request routing. *Routing* in a computer network refers to sending data from one or more sources to one or more destinations so as to minimize the total traffic flowing on the network. For a detailed review on the problem as well as a survey of combinatorial optimization applications, we refer the reader to [24]. *Request routing*, on the other hand, is basically the process of guiding a client's request to a

suitable caching server that is able to serve the corresponding request. The problem is formally defined as, given a request for an object, selecting a server to address the request such that a cost function is minimized. For a mathematical formulation of the problem (albeit a simplified one) the reader may refer to [14].

Object placement. Previously mentioned studies assume that the content held in the origin server is entirely replicated onto the caching servers (in case of which the caching servers are usually referred to as *replica*s or *mirror*s). Unfortunately, this may not always be possible in situations where the objects are significantly large in size (i.e. multimedia files) and only a partial replication can be performed due to the limited storage capacity of the caching servers. In this case, any caching server can only hold a subset of the content. Determining which objects should be placed at each caching server under storage capacity restrictions is known as the *object placement problem*. The reader may see [18] for a mathematical model of this problem.

9.2.2 Integrated Problems

In this section, we present and discuss some of the more complex issues in CDNs in which several problems mentioned above jointly arise. We start by the static data placement problem defined on a network with no origin server which consists of placing objects so as to minimize the total access cost (the cost for a client $i \in I$ to access object $k \in K$ from node $j \in J$ is $b_k d_{ik} c_{ij}$). A mathematical model for this problem, as offered by Baev et al. [4], is given below.

$$\textbf{(M1)} \qquad \text{Minimize} \qquad \sum_{i\in I}\sum_{j\in I}\sum_{k\in K} b_k d_{ik} c_{ij} x_{ijk} \tag{9.2}$$

subject to

$$\sum_{j\in I} x_{ijk} = 1 \qquad \forall i \in I, k \in K \tag{9.3}$$

$$x_{ijk} \le z_{jk} \qquad \forall i, j \in I, k \in K \tag{9.4}$$

$$\sum_{k\in K} b_k z_{jk} \le s_j \qquad \forall i \in I \tag{9.5}$$

$$z_{jk} \in \{0,1\} \qquad \forall j \in I, k \in K \tag{9.6}$$

$$x_{ijk} \in \{0,1\} \qquad \forall i, j \in I, k \in K. \tag{9.7}$$

Model **M1** uses, a binary variable z_{jk} that equals 1 if object $k \in K$ is held at caching server $j \in J$, and 0 otherwise; as well as a three-index binary variable x_{ijk} that is equal to 1 if object $k \in K$ requested by client $i \in I$ is served by node $j \in J$ that holds a copy, and 0 otherwise. In this model, the objective function (9.2) expresses the total cost of serving requests for all nodes and objects. Note that $J = I$, i.e. each node acts both as a client and a potential caching server. Constraint (9.3) expresses

that each node's request should be forwarded to exactly one node. Constraint (9.4) indicates that an assignment to a node can only be made if that specific node is holding the requested object. Finally, constraint (9.5) relates to the limited capacity (s_j) of each node $j \in J$.

Laoutaris et al. [20] study the joint problem of object placement and node dimensioning, where the latter refers to determining the fraction of a given a total storage capacity to allocate to each node of the network. The overall aim is to minimize the average distance from all clients to all the requested objects. The authors assume that all the objects are unit-sized. Another study by the same authors [21] describes a model to solve the storage capacity allocation problem in CDNs, taking into account decisions pertaining to the location of the caching servers to be installed, the capacity that should be allocated to each caching server, and the objects that should be placed in each caching server. This model is defined on a tree network, and each node i has a set of ancestors denoted by $a(i)$, and a set of leaves denoted by $l(i)$. The model is rewritten based on the notation introduced earlier, as opposed to that used in [21].

$$\textbf{(M2)} \qquad \text{Maximize} \qquad \sum_{i\in I}\lambda_i \sum_{k\in K} d_{ik} \sum_{v\in a(i)} (c_{is} - c_{iv}) x_{ijk}$$

subject to

$$\sum_{v\in a(i)} x_{ijk} \leq 1 \qquad \forall i \in I, k \in K \tag{9.8}$$

$$\sum_{v\in l(i)} x_{ijk} \leq M z_{jk} \qquad \forall i \in I, k \in K \tag{9.9}$$

$$\sum_{j\in J}\sum_{k\in K} z_{jk} \leq D \tag{9.10}$$

$$x_{ijk} \in \{0,1\} \qquad \forall i \in I, j \in J, k \in K \tag{9.11}$$

$$z_{jk} \in \{0,1\} \qquad \forall j \in J, k \in K. \tag{9.12}$$

As one can see, this model is quite similar to **M1** presented by Baev et al. [4] but differs with respect to the objective function, which maximizes the savings that one can obtain by the placement of objects on the caching servers. Constraints (9.8) and (9.9) are related to the assignment of customers to the caching servers, where M is a sufficiently big number. Constraints (9.10) ensure that the node dimensioning is performed without exceeding the available resource of storage capacity, denoted by $D = \sum_{j\in J} s_j$. Since all objects are assumed to be unit-sized by Laoutaris et al. [21], the dimension of a node is therefore equivalent to the number of objects placed on that node.

Nguyen et al. [23] consider the problem of provisioning CDNs on shared infrastructures and propose a joint provisioning and object replication model so as to minimize the total cost of storage, request serving, and start-up. We will present their model here in terms of the already defined notations but also define the following additional parameters: An object can be placed on each caching server at

a unit cost ψ_j and a unit bandwidth cost β_j, while the unit processing power cost is denoted by δ_j and a total of C_j units of processing power is available at each caching server. Each object k consumes l_k units of bandwidth and c_k units of processing power. The service provider has revenue of ρ_k from each object k per unit time. Latency between two nodes i and j is denoted by L_{ij} which should be limited by an upper bound Δ_d. An additional binary decision variable u_k denotes if an object is replicated (in any caching server) or not, and variable r_{ji}^k denotes the fraction of accesses for object k requested by customer i that should be directed to server j.

$$\textbf{(M3)} \quad \text{Maximize} \quad \sum_{i\in I}\sum_{j\in J}\sum_{k\in K} r_{ji}^k \rho_k - \sum_{j\in J}\sum_{k\in K} \psi_j b_k z_{jk} - \sum_{i\in I}\sum_{j\in J}\sum_{k\in K} r_{ji}^k(\beta_j l_k + \delta_j c_k) - \sum_{j\in J} f_j y_j \tag{9.13}$$

subject to

$$\sum_{i\in I} r_{ji}^k pw_k \le C_j y_j \qquad \forall j \in J \tag{9.14}$$

$$\sum_{j\in J} r_{ji}^k pw_k = u_k d_{ik} \qquad \forall i \in I, k \in K \tag{9.15}$$

$$r_{ji}^k (L_{ij} - \Delta_d) \le 0 \qquad \forall i \in I, j \in J, k \in K \tag{9.16}$$

$$r_{ji}^k \le d_{ik} z_{jk} \qquad \forall i \in I, j \in J, k \in K \tag{9.17}$$

$$y_j \in \{0,1\} \qquad \forall j \in J \tag{9.18}$$

$$u_k \in \{0,1\} \qquad \forall k \in K \tag{9.19}$$

$$z_{jk} \in \{0,1\} \qquad \forall j \in J, k \in K. \tag{9.20}$$

Model **M3** has an objective function which maximizes the profit of the service provider, calculated by subtracting from the total revenue (first component of (9.13)) the total cost related to storage, bandwidth, CPU and site establishment. Constraints (9.14) enforce the capacity restrictions for each server whereas constraints (9.15) ensure that all requests are served. Constraints (9.16) guarantee that all requests are served within the allowable latency bound. Finally, constraints (9.17) dictate that a request can be served by a caching server only when the requested object is available therein.

The joint problem of server location, object placement and request routing is studied by Bektaş et al. [9], where a new model is proposed that extends the standard facility location model to CDNs by considering multiple objects and an incorporation of a suitable, albeit nonlinear, objective function similar to (9.1). The integer programming model, as proposed by Bektaş et al. [9], is given below:

$$\textbf{(M4)} \quad \text{Minimize} \sum_{j\in J} f_j y_j + \sum_{i\in I}\sum_{j\in J}\sum_{k\in K} (b_k d_{ik} c_{ij} z_{jk} x_{ij} + b_k d_{ik}(1 - z_{jk})(c_{jS} + c_{ij}) x_{ij}) \tag{9.21}$$

subject to

$$\sum_{j\in J} x_{ij} = 1 \qquad \forall i \in I \tag{9.22}$$

$$x_{ij} \le y_j \qquad \forall i \in I, j \in J \tag{9.23}$$

$$\sum_{k\in K} b_k z_{jk} \le s_j y_j \qquad \forall j \in J \tag{9.24}$$

$$y_j \in \{0,1\} \qquad \forall j \in J \tag{9.25}$$

$$x_{ij} \in \{0,1\} \qquad \forall i \in I, j \in J \tag{9.26}$$

$$z_{jk} \in \{0,1\} \qquad \forall j \in J, k \in K. \tag{9.27}$$

The objective function of model **M4** is a generalization of (9.1) to multiple clients, servers and objects. The first component denotes the total cost of caching server establishment. The second component has two parts, where the first part corresponds to the costs of serving the clients from the caching servers and the second part reflects the additional costs that occur in accessing the origin server when the requested object is not found in the corresponding caching server. Constraints (9.22) ensure that each client is assigned to a single caching server and constraints (9.23) dictate that this assignment is only possible when the server is active. Overcapacity usage in placing the objects onto each caching server is prohibited by constraints (9.24).

Bektaş et al. [8] have later considered the problem from an operational level by excluding the caching server deployment decisions, but at the same time, imposing a QoS constraint that imposes a limit on end-to-end object transfer delays, and propose the following model.

$$\textbf{(M5)} \qquad \text{Minimize} \sum_{i\in I}\sum_{j\in J}\sum_{k\in K} (b_k d_{ik} c_{ij} z_{jk} x_{ij} + b_k d_{ik} (1 - z_{jk})(c_{jS} + c_{ij}) x_{ij})$$

subject to

$$\sum_{j\in J} x_{ij} = 1 \qquad \forall i \in I \tag{9.28}$$

$$\sum_{j\in J} L_{ij} x_{ij} z_{jk} + \sum_{j\in J} (L_{ij} + L_{j0}) x_{ij} (1 - z_{jk}) \le \Delta_d \qquad \forall i \in I, j \in J, k \in K \tag{9.29}$$

$$\sum_{k\in K} b_k z_{jk} \le s_j y_j \qquad \forall j \in J \tag{9.30}$$

$$x_{ij} \in \{0,1\} \qquad \forall i \in I, j \in J \tag{9.31}$$

$$z_{jk} \in \{0,1\} \qquad \forall j \in J, k \in K. \tag{9.32}$$

Model **M5** has a similar structure to **M4**. However, it excludes the caching server deployment decisions, but incorporates QoS restrictions represented by constraints (9.29). It has been observed that one can write constraints (9.29) in a much simpler form as $x_{ij} \le z_{jk}$, $\forall i \in I, j \in J, k \in \mathscr{Q}_{ij}$, where $\mathscr{R}_{ij} = \{k \in K | t(b_k, d_{ij}) > \Delta_d\}$ and $\mathscr{Q}_{ij} = \{k \in K | L_{ij}) \le \Delta_d$ and $(L_{ij} + L_{j0}) > \Delta_d\}$ for each pair (i, j) [8]. The former relates to objects k for which the time required to transfer such objects k from

caching server j exceeds the allowable delay limit, and the latter consists of a subset of objects k for which the time required to transfer such objects from caching server j is within allowable delay limit, but does not allow for retrieval of this object from the caching server due to the QoS constraint.

All of the above models are based on the assumption that the CDN operates with a single origin server (i.e. $|S| = 1$). While this is most often the case in practice, there are situations where a content provider may deploy multiple origin servers (possibly on the same site) for a variety of reasons, such as increasing system reliability or the storage capacity. To take into account multiple origin servers, a model is proposed by Bektaş et al. [6] for the joint problem of caching server placement, request routing, and object placement. We present this model in the following,

$$\textbf{(M6)} \quad \text{Minimize} \sum_{j\in J} f_j y_j + \sum_{i\in I}\sum_{j\in J}\sum_{s\in S}\sum_{k\in K} (b_k d_{ik} c_{ij} z_{jk} x_{ij} + b_k d_{ik}(1-z_{jk})(c_{ij}+c_{js}t_{js})x_{ij}) \tag{9.33}$$

subject to

$$\sum_{j\in J} x_{ij} = 1 \qquad \forall i \in I \tag{9.34}$$

$$x_{ij} \le y_j \qquad \forall i \in I, j \in J \tag{9.35}$$

$$\sum_{k\in K} b_k z_{jk} \le s_j y_j \qquad \forall j \in J \tag{9.36}$$

$$\sum_{s\in S} t_{js} = 1 \qquad \forall j \in J \tag{9.37}$$

$$y_j \in \{0,1\} \qquad \forall j \in J \tag{9.38}$$

$$x_{ij} \in \{0,1\} \qquad \forall i \in I, j \in J \tag{9.39}$$

$$z_{jk} \in \{0,1\} \qquad \forall j \in J, k \in K \tag{9.40}$$

$$t_{js} \in \{0,1\} \qquad \forall j \in J, s \in S. \tag{9.41}$$

Model **M6** is an extension of **M4** to multiple origin servers and uses an additional binary variable t_{js} that is equal to 1 if caching server $j \in J$ is assigned to an origin server $s \in S$, and 0 otherwise. Note that the objective function has been augmented so as to consider all the available origin servers, and an extra constraint (9.37) has been added that dictates each caching server should be assigned to a single origin server to further forward the requests for objects which they do not hold, in the event that these objects are requested by their clients.

9.3 Solution Algorithms

There are two classes of algorithms for the solution of the above mentioned problems and the associated mathematical models. The first is the class of exact

algorithms which are able to yield optimal solutions at the expense of rather significant computational times, and the second is the class of heuristic algorithms which usually require relatively small amount of computational effort, but unfortunately unable to guarantee the identification of the optimal solution. Amongst a number of available exact solution methods, we will focus here on two methods that are based on decomposition, since they allow for a break-down of the original model into smaller sized and easier-to-solve subproblems.

9.3.1 Benders' Decomposition

Benders Decomposition [10] is a technique that allows a model to be split into two subproblems. More specifically, given a model of the following form,

$$(\mathscr{P}) \qquad \text{Minimize } \mathbf{cx}+\mathbf{fy} \text{ subject to } \mathbf{Ax}+\mathbf{By}=\mathbf{d}, \mathbf{x}\in\mathbf{X}, \mathbf{y}\in\mathbf{Y}, \tag{9.42}$$

where $\mathbf{x}$ and $\mathbf{y}$ are the column vectors of variables, $\mathbf{c}$ and $\mathbf{f}$ are the row vectors of cost coefficients, $\mathbf{A}$ and $\mathbf{B}$ are the constraint coefficient matrices, and $\mathbf{d}$ is the column vector of right hand side values, all with appropriate dimensions. $\mathbf{X}$ and $\mathbf{Y}$ are nonempty (we assume that the former to be continuous and the latter integer) sets in which variables $\mathbf{x}$ and $\mathbf{y}$ are defined, respectively.

To illustrate the application of Benders' decomposition on problem $\mathscr{P}$, we first rewrite problem $\mathscr{P}$ in the following form.

$$(\mathscr{P}_1) \qquad \underset{\tilde{\mathbf{y}}\in\mathbf{Y}}{min}\{\mathbf{f}\tilde{\mathbf{y}}+\underset{\mathbf{x}\in\mathbf{X}}{min}\{\mathbf{cx}:\mathbf{Ax}=\mathbf{d}-\mathbf{B}\tilde{\mathbf{y}}\}\}, \tag{9.43}$$

where $\mathbf{y}$ is preset as $\mathbf{y}=\tilde{\mathbf{y}}$. Since the inner minimization problem in $\mathscr{P}_1$ (denoted by $\mathscr{S}$) expressed in terms of the x variables only is linear and continuous, one can replace it by its dual using dual variables $\mathbf{w}$ to each of the constraints of $\mathscr{S}$:

$$\underset{\tilde{\mathbf{y}}\in\mathbf{Y}}{min}\{\mathbf{f}\tilde{\mathbf{y}}+\underset{\mathbf{w}}{max}\{\mathbf{w}(\mathbf{d}-\mathbf{B}\tilde{\mathbf{y}}):\mathbf{wA}\le\mathbf{c}\}\}, \tag{9.44}$$

Assuming that the feasible region of the dual of $\mathscr{S}$ is nonempty (as otherwise this would imply the primal problem being either infeasible or unbounded), the original problem $\mathscr{P}$ can be rewritten as,

$$\text{Minimize} \qquad z+\mathbf{fy} \tag{9.45}$$

subject to

$$z \ge \boldsymbol{\tau}^{\mathbf{r}}(\mathbf{d}-\mathbf{By}) \qquad \tau^r\in\Upsilon \tag{9.46}$$

$$\boldsymbol{\varsigma}^{\mathbf{u}}(\mathbf{d}-\mathbf{By})\le 0 \qquad \varsigma^u\in\Psi \tag{9.47}$$

$$\mathbf{y}\in\mathbf{Y}, \tag{9.48}$$

called the Master Problem. Sets Υ and Ψ denote extreme points and extreme rays of the feasible space of the dual problem, respectively. Constraints (9.46) are those defined for each extreme point of the feasible region of the dual of $\mathscr{S}$, and constraints (9.47) are those written for each extreme ray of the dual of $\mathscr{S}$, whenever it is infeasible.

The authors of [8] observe that model **M5** has a special structure which makes it suitable for the application Benders' decomposition. We illustrate this on a linearization of model **M5** using auxiliary linearization variables φ_{ijk}. These variables correspond to the product $x_{ij}z_{jk}$ in the objective function of **M5** (see [8] for linearization details):

$$\text{Minimize} \sum_{i\in I}\sum_{j\in J}\sum_{k\in K}\left(b_k\lambda_{ik}\left(c_{ij}+c_{j0}\right)x_{ij}-b_k\lambda_{ik}c_{j0}\varphi_{ijk}\right) \tag{9.49}$$

subject to

$$\sum_{j\in J}x_{ij}=1, \qquad \forall i\in I \tag{9.50}$$

$$\sum_{k\in K}b_k z_{jk}\le s_j \qquad \forall j\in J \tag{9.51}$$

$$\varphi_{ijk}-x_{ij}\le 0 \qquad \forall i\in I, j\in J, k\in K \tag{9.52}$$

$$\varphi_{ijk}-z_{jk}\le 0 \qquad \forall i\in I, j\in J, k\in K \tag{9.53}$$

$$x_{ij}-z_{jk}\le 0 \qquad \forall i\in I, j\in J, k\in \mathscr{Q}_{ij} \tag{9.54}$$

$$x_{ij}\ge 0 \qquad \forall i\in I, j\in J \tag{9.55}$$

$$z_{jk}\in\{0,1\} \qquad \forall j\in J, k\in K \tag{9.56}$$

$$\varphi_{ijk}\in[0,1] \qquad \forall i\in I, j\in J, k\in K. \tag{9.57}$$

Upon fixing the object location variables that appear in this linearization to some feasible configuration as $z_{jk}=\bar{z}_{jk}$, the resulting subproblem further decomposes into smaller problems for each client $i\in I$, shown as follows:

$$\text{Minimize} \sum_{j\in J}\sum_{k\in K}(b_k\lambda_{ik}(c_{ij}+c_{j0})x_{ij}-b_k\lambda_{ik}c_{j0}\varphi_{ijk}) \tag{9.58}$$

subject to

$$\sum_{j\in J}x_{ij}=1 \tag{9.59}$$

$$\varphi_{ijk}-x_{ij}\le 0 \qquad \forall j\in J, k\in K \tag{9.60}$$

$$\varphi_{ijk}\le z^*_{jk} \qquad \forall j\in J, k\notin \mathscr{Q}_{ij} \tag{9.61}$$

$$x_{ij}\le z^*_{jk} \qquad \forall j\in J, k\in \mathscr{Q}_{ij} \tag{9.62}$$

$$x_{ij}\ge 0 \qquad \forall j\in J.$$

Each subproblem, although still integer, is observed to bear the integrality property. This distinctive feature allows one to relax the integrality restrictions on the x_{ij}

variables and solve its dual. Let α_i, θ_{ijk}, ω_{ijk} and ζ_{ijk} be the dual variables corresponding to constraints (9.59), (9.60), (9.61) and (9.62), respectively. One can then construct the master problem as follows,

$$\text{Minimize} \sum_{i \in I} \xi_i \tag{9.63}$$

subject to

$$\xi_i + \sum_{j \in J} \sum_{k \in K} z_{jk}(\tilde{\omega}_{ijk} + \tilde{\zeta}_{ijk}) \geq \tilde{\alpha}_i \qquad (\alpha, \theta, \omega, \zeta) \in \mathscr{P}_i^{\mathscr{D}} \tag{9.64}$$

$$\tilde{\alpha}_i - \sum_{j \in J} \sum_{k \in K} z_{jk}(\tilde{\omega}_{ijk} + \tilde{\zeta}_{ijk}) \leq 0, \qquad (\alpha, \theta, \omega, \zeta) \in \mathscr{W}_i^{\mathscr{D}} \tag{9.65}$$

$$\sum_{k \in K} b_k z_{jk} \leq s_j \qquad \forall j \in J$$

$$z_{jk} \in \{0,1\} \qquad \forall j \in J, k \in K,$$

where (9.64) are the optimality constraints with the coefficients corresponding to an optimal solution to the dual problem and calculated as follows,

$$\tilde{\alpha}_i = \min_{j \in \mathscr{F}_i \cup \mathscr{H}_i} \left\{ \sum_{k \in K} b_k \lambda_{ik}(c_{ij} + c_{j0}) - \sum_{k \in K : z^*_{jk} = 1} b_k \lambda_{ik} c_{j0} \right\}$$

$$\tilde{\omega}_{ijk} = \begin{cases} b_k \lambda_{ik} c_{j0}, & \text{if } z^*_{jk} = 0 \\ 0, & \text{otherwise} \end{cases}$$

$$\sum_{k \in K : z^*_{jk} = 0} \tilde{\zeta}_{ijk} = \tilde{\alpha}_i + \sum_{k \in K} \tilde{\theta}_{ijk} - \sum_{k \in K} b_k \lambda_{ik}(c_{ij} + c_{j0}) \qquad \forall j \in J,$$

where $\mathscr{F}_i = \{j \in J | \mathscr{Q}_{ij} = \mathscr{R}_{ij} = \emptyset\}$ and $\mathscr{H}_i = \{j \in J | z^*_{jk} = 1, \forall k \in \mathscr{Q}_{ij}\}$. Constraints (9.65) are written for every extreme ray that correspond to an infeasible solution to the dual problem. Due to the number of optimality and infeasibility constraints that are present in the master problem, it is not practically possible to solve it as is. One therefore needs to resort to a strategy where one starts with a restricted master problem including only a limited number constraints, and additional constraints are iteratively added to this restricted problem until the optimal solution is reached. The reader is referred to [8] for details of this approach along with various refinements that are used to increase the efficiency of the algorithm, such as the use of Pareto-optimal cuts and cut elimination.

9.3.2 Lagrangean Relaxation and Decomposition

Lagrangean relaxation is an approach where some of the constraints in a model are dualized (or relaxed) in a Lagrangean fashion so as to obtain a problem that is easier

to solve. Consider problem $\mathscr{P}$ presented in the previous section and assume that constraints $\mathbf{Ax}+\mathbf{By}=\mathbf{d}$ are those that "complicate" the model. Using a vector of Lagrangean multipliers denoted by μ, these constraints can be dualized as shown in the following,

$$\begin{array}{lll} (\mathscr{P}_\mu) & \text{minimize} & \mathbf{cx}+\mathbf{fy}+\mu(\mathbf{Ax}+\mathbf{By}-\mathbf{d}) \\ & \text{subject to} & \mathbf{x}\in\mathbf{X},\mathbf{y}\in\mathbf{Y}, \end{array}$$

which yields an immediate decomposition of $\mathscr{P}_\mu$ into two subproblems, one being

$$\min_{x\in\mathbf{X}}(\mathbf{c}+\mu\mathbf{A})\mathbf{x}),$$

defined only in x variables, and the other being

$$\min_{y\in\mathbf{Y}}(\mathbf{f}+\mu\mathbf{B})\mathbf{y}),$$

that is defined only in y variables. The solution value of $\mathscr{P}_\mu$, for any given μ, is a lower bound on the optimal solution value of $\mathscr{P}_\mu$. To find the best possible lower bound, one has to solve the following piecewise linear concave optimization problem, $\max_\mu \mathscr{P}_\mu$, usually named as the Lagrangean dual problem and solved by means of nondifferentiable optimization techniques.

There are several applications of Lagrangean relaxation to tackle some of the problems mentioned earlier, including the one proposed by Qiu et al. [26] for the server placement problem, by Nguyen et al. [23] for the overlay distribution network provisioning problem, and by Bektaş et al. [8] for the joint problem of object placement and request routing in a CDN. In this section, we will illustrate the use of this technique on an integer linear programming model proposed in [8], but with a different type of relaxation. The model we present below is an alternative linearization of **M5** using an auxiliary linearization variable v_{jk}.

$$\text{Minimize} \qquad \sum_{i\in I}\sum_{j\in J}\sum_{k\in K} b_k\lambda_{ik}(c_{ij}+c_{j0})x_{ij}-\sum_{j\in J}\sum_{k\in K} v_{jk} \tag{9.66}$$

subject to

$$\sum_{j\in J} x_{ij}=1 \qquad \forall i\in I \tag{9.67}$$

$$\sum_{k\in K} b_k z_{jk}\le s_j \qquad \forall j\in J \tag{9.68}$$

$$v_{jk}-Mz_{jk}\le 0 \qquad \forall j\in J,k\in K \tag{9.69}$$

$$v_{jk}-\sum_{i\in I} b_k\lambda_{ik}c_{j0}x_{ij}\le 0 \qquad \forall j\in J,k\in K \tag{9.70}$$

$$x_{ij}-z_{jk}\le 0 \qquad \forall i\in I,j\in J,k\in\mathscr{Q}_{ij} \tag{9.71}$$

$$x_{ij} \geq 0 \qquad \forall i \in I, j \in J \tag{9.72}$$

$$z_{jk} \in \{0,1\} \qquad \forall j \in J, k \in K \tag{9.73}$$

$$v_{jk} \geq 0 \qquad \forall j \in J, k \in K. \tag{9.74}$$

By dualizing constraints (9.67), (9.69) and (9.70) using respectively σ_i, π_{jk} and η_{jk} as the Lagrangean multipliers, we obtain the following relaxed problem denoted by $\mathscr{R}$:

$$(\mathscr{R}) \qquad \text{Minimize} \qquad \sum_{i\in I}\sum_{j\in J}\left(\sum_{k\in K}(b_k d_{ik}(c_{ij}+c_{j0}(1-\eta_{jk})))-\sigma_i\right)x_{ij} + \sum_{j\in J}\sum_{k\in K}\left(\pi_{jk}+\eta_{jk}-1\right)v_{jk} - M\sum_{j\in J}\sum_{k\in K}\pi_{jk}z_{jk} - \sum_{i\in I}\sigma_i \tag{9.75}$$

subject to

$$\sum_{k\in K} b_k z_{jk} \leq s_j \qquad \forall j \in J \tag{9.76}$$

$$x_{ij} - z_{jk} \leq 0 \qquad \forall i \in I, j \in J, k \in \mathscr{Q}_{ij} \tag{9.77}$$

$$x_{ij} \geq 0 \qquad \forall i \in I, j \in J \tag{9.78}$$

$$z_{jk} \in \{0,1\} \qquad \forall j \in J, k \in K \tag{9.79}$$

$$v_{jk} \geq 0 \qquad \forall j \in J, k \in K. \tag{9.80}$$

Problem $\mathscr{R}$ decomposes into two subproblems, one in x and z variables, and the other in v variables. The latter is solvable through inspection by setting $v_{jk} = 0$ if $\pi_{jk} + \eta_{jk} - 1$ is nonnegative, and $v_{jk} = 1$ otherwise. As for the former subproblem, notice that the x variables only appear in constraints (5), and thus can be fixed to $x_{ij} = 0$ if $\sum_{k\in K}(b_k d_{ik}(c_{ij} + c_{j0}(1-\eta_{jk}))) - \sigma_i$ is nonnegative and $x_{ij} = \hat{z}_{jk}$ otherwise, where $\hat{z}_{jk}$ is the solution to the following problem,

$$\text{Maximize} \qquad \sum_{j\in J}\sum_{k\in K}\pi_{jk}z_{jk}$$

subject to

$$\sum_{k\in K} b_k z_{jk} \leq s_j \qquad \forall j \in J$$

$$z_{jk} \in \{0,1\} \qquad \forall j \in J, k \in K,$$

which further decomposes into a series of binary knapsack problems, one for each $j \in J$, each of which can be solved efficiently in $\mathscr{O}(|K|s_j)$ time using dynamic programming.

9.3.3 Heuristic Algorithms

Contrary to exact algorithms, heuristic algorithms are fast and scalable solutions methods for instances that are beyond the reach of exact algorithms for problems of a CDN provider. *Greedy heuristics* are heuristic algorithms that search for a solution to a given problem by choosing the best possible alternative at each iteration (i.e. the option that reduces the cost by the greatest amount) but neglects the effect decision on the overall search. These algorithms therefore yield locally optimal solutions most of the time. The advantage of such heuristics lie in their computational speed at tackling problems and scalability in being applicable to very large instances. This explains the popularity of greedy heuristics in the CDN literature and we refer the reader to, amongst many others, [18, 21, 26, 27, 31]. *Approximate algorithms*, on the other hand, are heuristics that guarantee to find a solution with a value that is within a constant factor of the optimal solution and for which applications within the CDN domain can be found [4, 20]. *Simulated annealing* and *Tabu search* belong to a class of more sophisticated heuristic techniques, named as metaheuristics, in that they make use of special mechanisms to prevent the search from being trapped in the local minima. We refer the reader to [7] for a two-level implementation of simulated annealing for the joint problem of object placement and request routing, and to [15] for an application of a tabu search algorithm on the same problem.

Table 9.2 A categorization of the existing models and solution approaches

SP	RR	OP	CD	Reference	Solution Approaches
x				[22]	Dynamic programming
x				[26]	Lagrangean relaxation, greedy heuristics
x				[27]	Greedy heuristics
x				[5]	Greedy heuristics
x				[17]	Dynamic programming
	x			[14]	Integer programming models
		x		[13]	Greedy heuristics
		x		[18]	Greedy heuristics
		x	x	[20]	Exact and approximate algorithms
		x	x	[21]	Greedy heuristics
x	x			[1]	Heuristic algorithms
x		x		[29]	Dynamic programming
x		x		[30]	Heuristic algorithms
	x	x		[4]	Approximate algorithm
	x	x		[3]	Heuristic algorithms
	x	x		[28]	Analytic and heuristic algorithms
	x	x		[7]	Simulated annealing
	x	x		[8]	Benders' decomposition, Lagrangean relaxation
	x	x		[15]	Tabu search
x	x	x		[23]	Lagrangean relaxation
x	x	x		[9]	Benders' decomposition, greedy heuristic

Table 9.2 presents a categorization of the existing models and solution approaches for a variety of resource allocation and management problems (i.e. Server Placement (SP), Request Routing (RR), Object Placement (OP), and Content Delivery (CD)) in CDNs, including additional references. It is not meant to be a complete list but rather a representation of the wide variety of tools that have been used up to now to solve these problems.

9.4 New Models for Alternative CDN Architectures

Most of the models described in the previous sections are based on various CDN architectures and have their limitations due to the restrictive assumptions made to facilitate the modeling. In this section, we propose new models for more general situations that -to the best of our knowledge- have not been considered before in terms of mathematical modeling. For the purposes of illustrations, we present the models using a sample small-scale instance. The instance has a network structure consisting of a single origin server ($|S| = 1$), three active caching servers ($J = \{1,2,3\}$) and ten clients ($I = \{1,2,\ldots,10\}$). The architecture of the sample network topology is depicted in Fig. 9.1. In this sample instance, we assume that there are five objects to be distributed ($K = \{1,2,\ldots,5\}$) with their sizes (in, for instance, GBs) $b_1 = 94, b_2 = 75, b_3 = 96, b_4 = 61$ and $b_5 = 82$. The capacities of the caching servers are given as $s_1 = 156, s_2 = 162$ and $s_3 = 85$ (again, in GBs), which range from 20% to 40% of the total size of the objects. The distances c_{ij} for all $i \in I, j \in J$ are randomly distributed between 1 and 5 (see Table 9.4 in the Appendix for the full matrix), whereas the distances between each caching server and the origin server are given as $c_{1,0} = 20$, $c_{2,0} = 15$ and $c_{3,0} = 18$. For simplicity, we assume that the each client has a uniform request rate for each object (i.e. $d_{ik} = 1$ for all $i \in I, k \in K$).

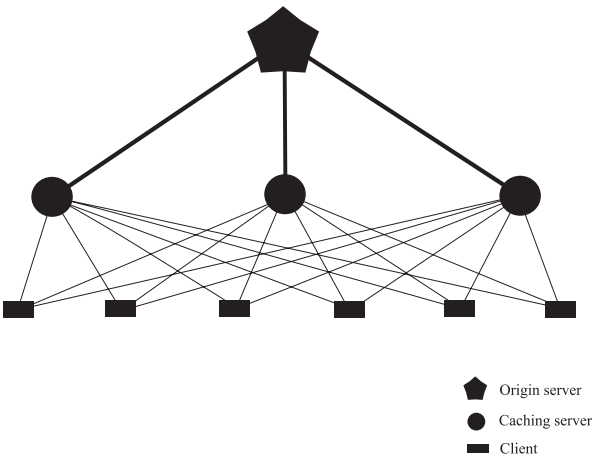

Fig. 9.1 Architecture of the sample network topology

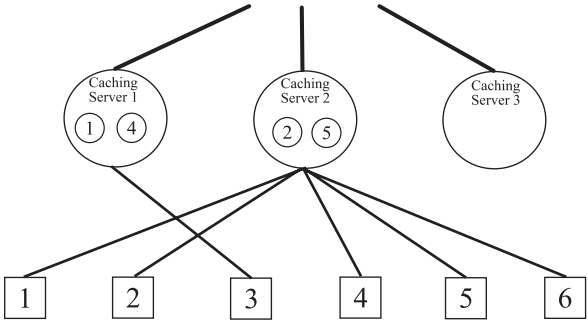

Fig. 9.2 Solution for the sample problem obtained by model **M4**

As an initial scenario to benchmark the ones that follow, we have used model **M4** to solve the sample problem. All models in this section have been solved using the state-of-the-art nonlinear integer programming solver `BONMIN`[2] through the online NEOS server.[3] The optimal solution of **M4** is depicted in Fig. 9.2, with the client-server assignments represented by the bold links and the object placements are shown within each caching server. The total cost of this solution is 31229.

9.4.1 Object Retrieval from Multiple Servers

The models presented in the previous sections generally assume a CDN architecture where a client $i \in I$ is assigned to a single caching server $j \in J$ (which will be henceforth referred to as the *primal server*) from which it retrieves the requested objects, and when the requested object is not available in server $j \in J$, the request is forwarded to the origin server by the primal server from where the object is fetched. While such a strategy may be appropriate where the number of caching servers is high and the administrative costs of requesting from other caching servers is significant, it may not always prove to be a viable option when there exists a high number of objects with similar request rates and when the storage capacities of caching servers are limited (meaning that there will be many requests forwarded to the origin server). In order to prevent this, an alternative strategy may be to direct a client's request for an object to the origin server only when the object is not available in *any* other caching server (as suggested by Datta et al. [14]). This means that each client would be allowed to retrieve objects from other caching servers. A second question arises here as to what happens when the requested object is not found in any of the caching servers. To address this, as a first step, we will restrict ourselves to the situation where a client's request would be forwarded to the origin server only via its

[2] Available at `https://projects.coin-or.org/Bonmin`

[3] Available at `neos.mcs.anl.gov/neos/solvers/minco:Bonmin/AMPL.html`

primal server, but we will also discuss when this assumption is relaxed. The model for the former case is as follows:

(**M7**) Minimize
$$\sum_{i\in I}\sum_{j\in J}\sum_{k\in K} b_k d_{ik} c_{ij} x_{ijk} + \sum_{i\in I}\sum_{k\in K}\left[\sum_{j\in J}\left(b_k d_{ik}(c_{ij}+c_{j0})x_{ij}(1-\sum_{t\in J} x_{itk})\right)\right] \quad (9.81)$$

subject to

$$\sum_{j\in J} x_{ij} = 1 \qquad \forall i \in I \quad (9.82)$$

$$\sum_{j\in J} x_{ijk} \leq 1 \qquad \forall i \in I, k \in K \quad (9.83)$$

$$\sum_{k\in K} b_k z_{jk} \leq s_j \qquad \forall j \in J \quad (9.84)$$

$$\sum_{j\in J} x_{ijk} \geq z_{jk} \qquad \forall i \in I, j \in J, k \in K \quad (9.85)$$

$$x_{ijk} \leq z_{jk} \qquad \forall i \in I, j \in J, k \in K \quad (9.86)$$

$$x_{ij} + z_{jk} - x_{ijk} \leq 1 \qquad \forall i \in I, j \in J, k \in K \quad (9.87)$$

$$x_{ij} \in \{0,1\} \qquad \forall i \in I, j \in J \quad (9.88)$$

$$x_{ijk} \in \{0,1\} \qquad \forall i \in I, j \in J, k \in K \quad (9.89)$$

$$z_{jk} \in \{0,1\} \qquad \forall j \in J, k \in K. \quad (9.90)$$

In model **M7**, the objective function is composed of two cost components. The first represents the cost of serving each client directly from (multiple) caching servers. The second component, on the other hand, models the situation where a request is forwarded to the origin server by the primal server only if the object is not located at any other caching server (i.e. when $z_{jk} = 0$ for all $j \in J$ implying $\left(1 - \sum_{t\in J} x_{itk}\right) = 1$). Constraints (9.82) represent the assignment of each client to its primal server. Constraints (9.83) dictate the condition that each client receives each object from at most one caching server and constraints (9.86) make sure that the request is served only from a single caching server that holds the requested object. Storage capacity limitations for each caching server are implied by constraints (9.87). For any request, constraints (9.87) give priority that the request be served by the primal server if it holds the object (i.e. $x_{ijk} = 1$ if $x_{ij} = z_{jk} = 1$). The solution of model **M7** outputs a solution that is depicted in Fig. 9.3 with an optimal solution value of 14001, which is a solution that is about 55% less costly than that obtained by model **M4**.

In Fig. 9.3, client assignments to primal servers are represented by bold links whereas requests that are routed to other servers are represented by the lighter links (i.e. clients 1, 3, 4 and 5 are assigned to caching server 2, but they retrieve objects 1 and 4 from caching server 1, since their primal server does not hold this object). Any request for object 2 in this case has to be further requested by the origin server,

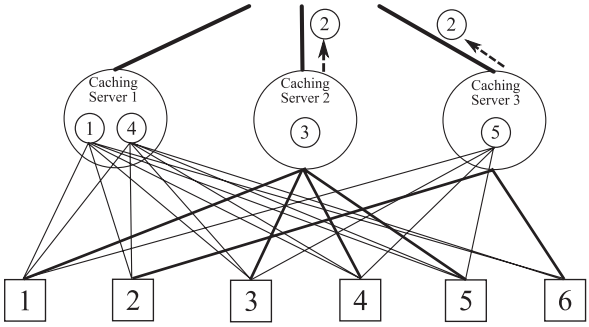

Fig. 9.3 Solution for the sample problem obtained by model **M7**

as none of the caching servers hold this specific object. In this case, clients 1, 3, 4 and 5 receive object 2 through their primal server (no. 2), and clients 2 and 6 receive it through their primal server (no. 3).

To incorporate more flexibility into the distribution strategy, we provide below another model which allows a client's request for an object to be forwarded (and thus served to the client) by any caching server in the network. For this model, we define a new binary variable v_{ijk} that equals 1 if object k is served to client i from the origin server *via* caching server j, and 0 otherwise.

$$\textbf{(M8)} \qquad \text{Minimize} \qquad \sum_{i\in I}\sum_{j\in J}\sum_{k\in K}\left(b_k d_{ik} c_{ij} x_{ijk} + b_k d_{ik}(c_{ij}+c_{j0})v_{ijk}\right)$$

subject to

$$\sum_{j\in J}(x_{ijk}+v_{ijk}) = 1 \qquad \forall i\in I, k\in K \tag{9.91}$$

$$\sum_{k\in K} b_k z_{jk} \le s_j \qquad \forall j\in J \tag{9.92}$$

$$x_{ijk} \le z_{jk} \qquad \forall i\in I, j\in J, k\in K \tag{9.93}$$

$$v_{ijk} \le 1 - z_{jk} \qquad \forall i\in I, j\in J, k\in K \tag{9.94}$$

$$x_{ijk}, v_{ijk} \in \{0,1\} \qquad \forall i\in I, j\in J, k\in K \tag{9.95}$$

$$z_{jk} \in \{0,1\} \qquad \forall j\in J, k\in K. \tag{9.96}$$

The objective function of model **M8** is composed of two components. The first represents the total cost of object transfer from the caching servers to the clients. The second represents the cost of fetching a requested object from the origin server. Constraints (9.91) ensure that a client receives any object either directly from or through one of the caching servers. Constraints (9.92) impose capacity restrictions on the caching servers, (9.93) state that a client can not be served by a caching server unless the requested object is held therein, (9.94) enforce the condition that an object can not be requested from the origin server if there exists at least one caching server $j \in J$ that holds it. The solution of model **M8** outputs a solution that is depicted in Fig. 9.4 with

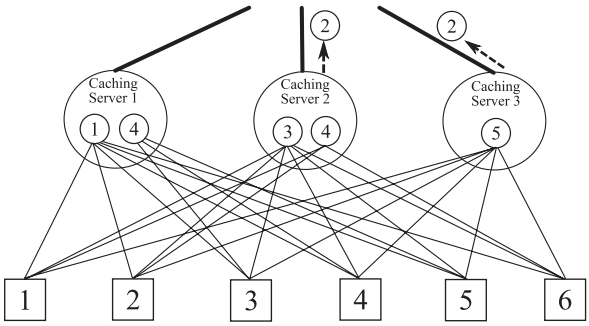

Fig. 9.4 Solution for the sample problem obtained by model **M8**

an optimal solution value of 13940, which is even less costly than that of **M7**. The solution shown in Fig. 9.4 indeed illustrates the flexibility afforded to the distribution process where any client can receive any object from (or through) any of the servers. As an example, we note that client 1 receives object 1 from caching server 1, objects 3 and 4 from caching server 2, object 5 from caching server 3, and object 2 through caching server 2 (which further forwards this request to the origin server).

9.4.2 Survivability in CDN Design

Survivability of a telecommunications network is defined as its ability to operate under a link or a server failure. As for the former case, there already exists a rather significant literature (e.g. see [25]) which can be adapted to CDN design by establishing back-up links between the clients and the servers that can be activated whenever the primal link fails. The latter case, however, is quite relevant as most of the previously stated models are based on the assumption that each client is connected to and served from or via a single caching server. In the event that its primal server should fail, the client need immediately be served by another caching server (even if the requested object is not located there since the caching server acts as a pathway to the origin server). Therefore, for a CDN to be 'survivable', one needs to design it such that each client should be assigned a *back-up* (or *stand-by*) server, to which its requests should be redirected in the event of a primal server failure. In this light, we offer here a model which extends **M7** to the survivable case. The model for this case is presented as follows:

(**M9**) Minimize $\sum_{j\in J} f_j y_j$

$$+\sum_{i\in I}\sum_{j\in J}\sum_{k\in K}(b_k d_{ik} c_{ij} z_{jk} x^p_{ij} + b_k d_{ik}(1-z_{jk})(c_{jS}+c_{ij})x^p_{ij})$$

$$\gamma\sum_{i\in I}\sum_{j\in J}\sum_{k\in K}(b_k d_{ik} c_{ij} z_{jk} x^p_{ij} + b_k d_{ik}(1-z_{jk})(c_{jS}+c_{ij})x^b_{ij}) \quad (9.97)$$

subject to

$$\sum_{j\in J} x_{ij}^{p} = 1 \qquad \forall i \in I \tag{9.98}$$

$$\sum_{j\in J} x_{ij}^{b} = 1 \qquad \forall i \in I \tag{9.99}$$

$$x_{ij}^{p} + x_{ij}^{b} \leq y_j \qquad \forall i \in I, j \in J \tag{9.100}$$

$$\sum_{k\in K} b_k z_{jk} \leq s_j y_j \qquad \forall j \in J \tag{9.101}$$

$$y_j \in \{0,1\} \qquad \forall j \in J \tag{9.102}$$

$$x_{ij} \in \{0,1\} \qquad \forall i \in I, j \in J \tag{9.103}$$

$$z_{jk} \in \{0,1\} \qquad \forall j \in J, k \in K. \tag{9.104}$$

In **M9**, x_{ij}^{p} is a binary variable that is equal to 1 if server j acts as a primal server for client i, and 0 otherwise; and x_{ij}^{b} is another binary variable that is equal to 1 if caching j acts as a back-up server for client i, and 0 otherwise. The first two components of the objective function (9.97) are similar to that of **M4**. The third component represents the cost of providing back-up service to the clients in the event of a break-down. Since the break-downs are not very likely to occur frequently, this cost will not arise very often. The parameter $0 \leq \gamma \leq 1$ is therefore provided to adjust the impact of the back-up service cost on the CDN design. Thus, when $\gamma = 0$, the CDN provider will not take into account the cost of providing back-up service to its clients, although the CDN itself will be designed in such a way. When $\gamma = 1$, then the total cost will include the additional cost of providing the back-up service, even though this service may never be used. In this model, constraints (9.98) are associated with the primal server assignments, whereas constraints (9.99) ensure that each client is also assigned to a back-up server. Constraints (9.101) impose capacity restrictions on the active caching servers. The output of the solution of model **M9** on the sample problem is depicted in Fig. 9.5. The optimal solution value in this case is 55561.7 for $\gamma = 0.7$ and 41657.3 for $\gamma = 0.3$.

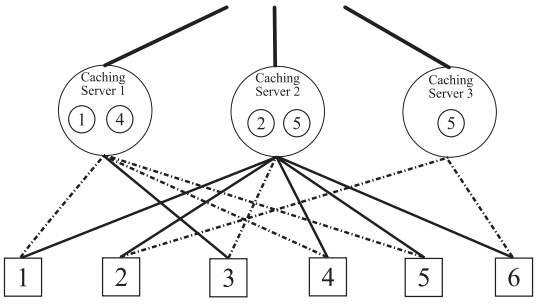

Fig. 9.5 Solution for the sample problem obtained by model **M9**

The solution given in Fig. 9.5 shows the primal server assignments by bold links and back-up assignments by dotted links (i.e. caching server 2 acts as a primal server for client 1, but in the event that it fails, client 1 is immediately routed to caching server 1).

9.5 Performance Results

To give the reader a flavour of the computational performance of the new models **M7**-**M9**, we present the results of a limited set of computational experiments carried out on a set of instances. These instances have been generated in the same way as described by Bektaş et al. [9]. The instances are based on a network with three caching servers and ten clients. The number of objects to be distributed ranges from 20 to 90, in increments of 10. We note that the request rates for the objects are not uniform in this case, but have been generated using a Zipf-like distribution (see [12, 32]) in the form $P_K(i) = \Omega i^{-\alpha}$ where $\Omega = \left(\sum_{j=1}^{K} j^{-\alpha}\right)^{-1}$ is a normalization constant and the distribution parameter is set as $\alpha = 0.733$.

The results of these experiments are given in Fig. 9.6, which shows the corresponding solution values obtained with models **M4**, **M7**, **M8** and **M9** (run twice with $\gamma = 0.3$ and $\gamma = 0.7$). As the figure shows, the performance of models **M7** and **M8** are quite similar and both provide better results than that of **M4**. On the other hand, **M9** results in solutions with substantially higher costs due to the addition of the survivability component. The time required for the solution of these models are given in Table 9.3. These values imply that, even with very small-scale problems as

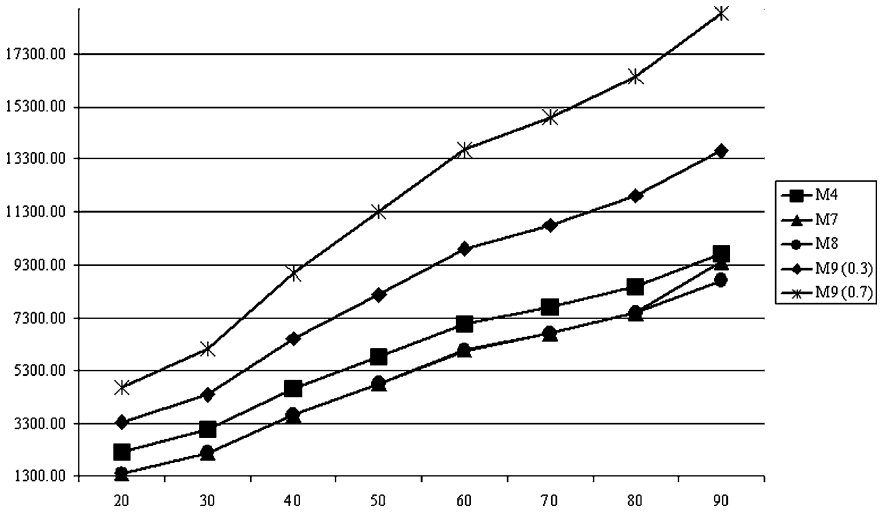

Fig. 9.6 Cost comparison of the models on the sample instances

Table 9.3 Computational solution times (in seconds) of the models on the sample instances

$\|I\|$	$\|J\|$	$\|K\|$	**M4**	**M7**	**M8**	**M9**(0.3)	**M9**(0.7)
3	10	20	0.02	230.48	161.29	0.05	0.23
3	10	30	0.05	256.54	679.54	0.09	0.42
3	10	40	0.31	118.17	88.29	1.19	0.50
3	10	50	0.34	161.58	418.15	0.29	0.21
3	10	60	0.16	1764.42	1867.74	1.01	0.77
3	10	70	0.13	384.59	395.48	0.13	0.79
3	10	80	0.27	3600.00	3600.00	1.80	1.94
3	10	90	0.45	3600.00	3600.00	1.23	3.90

the ones considered here, solving models **M7** and **M8** prove to be quite difficult. In fact, for instances with $|K| = 80$ and $|K| = 90$, the optimal solutions of these models could not be obtained within a time limit of one hour (the values shown in Fig. 9.6 for these instances are the best possible values attained within the time limit). The other models, however, are easily solved for these instances, although the values shown in Table 9.3 imply that the solution times will expectedly increase as the size of the instances grow larger.

9.6 Visionary Thoughts for Practitioners

It is clear that, for a dynamic and active environment such as a CDN, most applications call for the use of fast and scalable methods among which heuristics are the popular choice. For instance, greedy heuristics (e.g. see [19]), topology-informed heuristic methods [16] or hot-spot algorithms [26] are known to be widely used for caching server location problems. However, although one may show the superiority of one heuristic method to another, one has no indication the quality of the solutions obtained with such methods. Our intention through this chapter is to stress the importance of using mathematical models and especially exact solution approaches in solving CDN problems and to recognize that there are benefits to reap in using these approaches. Indeed, mathematical modeling can be used as benchmarks to assess a variety of heuristic methods in terms of solution quality. This would certainly aid in choosing the correct type of a heuristic method in practice. Such an approach, for instance, has been taken by Laoutaris et al. [21], where the authors propose and evaluate the performance of a greedy method (and its variations) by comparing it with an exact solution approach.

Mathematical modeling techniques can also be used to gain insight to a variety of CDN problems arising in practice and to determine what mitigating actions can be taken. For instance, Nguyen et al. [23] use a Lagrangean-based solution algorithm based on a mathematical model to evaluate the effect of object clustering on the total revenue of a CDN provider using this algorithm.

Finally, we believe that the flexibility of mathematical models in easily accommodating additional constraints or the change in the problem parameters would

facilitate the analysis of a variety of scenarios and help the decision maker choose the right alternative. For instance, a CDN provider may wish to assess a number of differing request routing or object placement strategies under certain parameter settings. While one may argue that this can also be performed using heuristic methods, we believe that these may not yield as precise solutions as those which may be obtained through the use of mathematical models, since the quality of the solutions found by the former is not always known.

9.7 Future Research Directions

We believe that further research on CDN modeling lies in two main directions: new model realization and algorithm development. As for the former, the new models proposed here show that there are indeed situations that have not been modeled before and even hint for the possibility of developing of other models for even more complex situations that are most likely to arise in practice. Some suggestions in this respect would be to incorporate survivability issues or caching server placement decisions into models **M7** or **M8**, or the addition of QoS restrictions (such as those proposed by Bektaş et al. [8]) models **M7**-**M9**. Such attempts will undoubtedly result in more complex models, which we expect mostly to be in the form of nonlinear integer programming formulations.

As demonstrated in this chapter through numerical experiments, obtaining solutions to models such as **M7** or **M8** can prove to be quite difficult even for very small-scale instances. This further necessitates devising new exact algorithms that are able to efficiently tackle these complex models. This chapter suggests that, in terms of exact solution methods, decomposition based methods coupled with linearization strategies for the nonlinear models are a promising direction. However, these exact methods will most likely be unable to cope with large-scale instances, which further indicates the need for fast and scalable heuristic methods that can address these problems. To our belief, the development of heuristic and exact solution techniques should go hand-to-hand, in that one approach should be used as a complementary to the other. Such strategies have proven to be of good use in developing even better methodologies for some problems (e.g. see [7, 9, 15]).

9.8 Conclusions

In this chapter, we have outlined the fundamental problems in managing and allocating resources (the network, caching servers, and objects) in a CDN faced by the CDN provider. We have presented the existing mathematical models proposed earlier for these problems in a common framework. Discussions and examples have been provided on how several exact and heuristic methods can be tailored in solving the problems and the associated models. This chapter also offers novel mathematical

models for a variety of situations that have not yet been investigated in depth, such as designing a survivable CDN.

This chapter shows that mathematical modeling is a powerful tool to address the problems faced by the CDN provider and obtain a deeper understanding into the nature of the problem. As mentioned in the previous section, mathematical models also facilitate the solution of problems they represent, by providing a generic framework on which efficient exact solution algorithms can be devised. This chapter suggests that, in terms of exact solution algorithms, those that are based on decomposition ideas are most likely to be successful for the solution of CDN problems. Exact algorithms are also crucial in assessing the quality of heuristic approaches, especially heuristics of a greedy nature, which are known to be widely used in solving many problems of a CDN.

Acknowledgements Some of the materials presented in this chapter appear in a preliminary form in Computers & Operations Research Journal [8, 9].

Appendix

The distance matrix (can be interpreted as the number of hops between each $i \in I, j \in J$) for the sample problem is given below.

Table 9.4 The distance matrix for the sample instance

c_{ij}	$j=1$	$j=2$	$j=3$
$i=1$	1	1	3
$i=2$	5	4	1
$i=3$	1	5	5
$i=4$	1	4	5
$i=5$	1	3	2
$i=6$	5	5	1

References

1. Almeida, J., Eager, D., Vernon, M., Wright, S.: Minimizing delivery cost in scalable streaming content distribution systems. IEEE Transactions on Multimedia **6**, 356–365 (2004)
2. Avella, P., Sassano, A., Vasil'ev, I.: Computational study of large-scale p-median problems. Mathematical Programming **109**, 89–114 (2007)
3. Backx, P., Lambrecht, T., Dhoedt, B., DeTurck, F., Demeester, P.: Optimizing content distribution through adaptive distributed caching. Computer Communications **28**, 640–653 (2005)
4. Baev, I., Rajaraman, R.: Approximation algorithms for data placement in arbitrary networks. In: Proceedings of the 12th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 661–670 (2001)
5. Bassali, H., Kamath, K., Hosamani, R., Gao, L.: Hierarchy-aware algorithms for CDN proxy placement in the Internet. Computer Communications **26**, 251–263 (2003)

6. Bektaş, T.: Discrete location models for content distribution. Unpublished PhD Dissertation, Bilkent University, Ankara, Turkey (2005)
7. Bektaş, T., Cordeau, J.F., Erkut, E., Laporte, G.: A two-level simulated annealing algorithm for efficient dissemination of electronic content. Journal of the Operational Research Society **35**, 3860–3884 (2008)
8. Bektaş, T., Cordeau, J.F., Erkut, E., Laporte, G.: Exact algorithms for the joint object placement and request routing problem in content distribution networks. Computers & Operations Research (2008). In press
9. Bektaş, T., Oğuz, O., Ouveysi, I.: Designing cost-effective content distribution networks. Computers & Operations Research **34**, 2436–2449 (2007)
10. Benders, J.: Partitioning procedures for solving mixed-variables programming problems. Numerische Mathematik **4**, 238–252 (1962)
11. Berman, O., Krass, D.: An improved IP formulation for the uncapacitated facility location problem: Capitalizing on objective function structure. Annals of Operations Research **136**, 21–34 (2005)
12. Breslau, L., Cao, P., Fan, L., Phillips, G., Shenker, S.: Web caching and Zipf-like distributions: evidence and implications. In: Proceedings of IEEE INFOCOM'99, Vol. 1, pp. 126–134. New York (1999)
13. Cidon, I., Kutten, S., Soffer, R.: Optimal allocation of electronic content. Computer Networks **40**, 205–218 (2002)
14. Datta, A., Dutta, K., Thomas, H., VanderMeer, D.: World Wide Wait: a study of Internet scalability and cache-based approaches to alleviate it. Management Science **49**, 1425–1444 (2003)
15. Dubuc, G., Bektaş, T., Cordeau, J.F., Laporte, G.: Une heuristique de recherche avec tabous pour la conception de réseaux de distribution de contenu électronique INFOR **45**, 175–185 (2007)
16. Jamin, S., Jin, C., Jin, Y., Raz, D., Shavitt, Y., Zhang, L.: On the placement of Internet instrumentation. In: Proceedings of IEEE INFOCOM'00, pp. 295–304 (2000)
17. Jia, X., Li, D., Hu, X., Wu, W., Du, D.: Placement of web-server proxies with consideration of read and update operations on the Internet. The Computer Journal **46**(4), 378–390 (2003)
18. Kangasharju, J., Roberts, J., Ross, K.: Object replication strategies in content distribution networks. Computer Communications **25**, 376–383 (2002)
19. Krishnan, P., Raz, D., Shavitt, Y.: The cache location problem. IEEE/ACM Transactions on Networking **8**, 568–582 (2000)
20. Laoutaris, N., Zissimopoulos, V., Stavrakakis, I.: Joint object placement and node dimensioning for Internet content distribution. Information Processing Letters **89**, 273–279 (2004)
21. Laoutaris, N., Zissimopoulos, V., Stavrakakis, I.: On the optimization of storage capacity allocation for content distribution. Computer Networks **47**, 409–428 (2005)
22. Li, B., Golin, M., Italiano, G., Deng, X., Sohraby, K.: On the optimal placement of web proxies in the Internet. In: Proceedings of IEEE INFOCOM'99, Vol. 3, pp. 1282–1290. New York (1999)
23. Nguyen, T., Safaei, F., Boustead, P., Chou, C.: Provisioning overlay distribution networks. Computer Networks **49**, 103–118 (2005)
24. Oliveira, C., Pardalos, P.: A survey of combinatorial optimization problems in multicast routing. Computers & Operations Research **32**, 1953–1981 (2005)
25. Ouveysi, I., Wirth, A., Yeh, A., Oguz, O.: Large scale linear programs and heuristics for the design of survivable telecommunication networks. Annals of Operations Research **124**, 285–293 (2003)
26. Qiu, L., Padmanabhan, V., Voelker, G.: On the placement of web server replicas. In: Proceedings of IEEE INFOCOM'01, Vol. 3, pp. 1587–1596 (2001)
27. Radoslavov, P., Govindan, R., Estrin, D.: Topology informed Internet replica placement. Computer Communications **25**, 384–392 (2002)
28. Wauters, T., Coppens, J., De Turck, F., Dhoedt, B., Demeester, P.: Replica placement in ring based content delivery networks. Computer Communications **29**, 3313–3326 (2006)

29. Xu, J., Li, B., Lee, D.: Placement problems for transparent data replication proxy services. IEEE Journal on Selected Areas in Communications **20**, 1383–1398 (2002)
30. Xuanping, Z., Weidong, W., Xiaopeng, T., Yonghu, Z.: Data Replication at Web Proxies in Content Distribution Network, *Lecture Notes in Computer Science*, Vol. 2642, pp. 560–569. Springer-Verlag, Xian (2003)
31. Yang, M., Fei, Z.: A model for replica placement in content distribution networks for multimedia applications. In: Proceedings of IEEE International Conference on Communications (ICC '03), Vol. 1, pp. 557 –561 (2003)
32. Zipf, G.: Human Behavior and the Principle of Least-Effort. Addison-Wesley, Cambridge, MA (1949)