Available online at www.sciencedirect.com
ScienceDirect
Computers & Operations Research 34 (2007) 2436–2449
computers & operations research
www.elsevier.com/locate/cor

Designing cost-effective content distribution networks

Tolga Bektas[a,*], Osman Oguz[a], Iradj Ouveysi[b]

[a]*Department of Industrial Engineering, Bilkent University 06800 Ankara, Turkey*
[b]*The University of Melbourne, VIC 3010, Australia*

Available online 13 October 2005

Abstract

In this paper, we present a novel technique for the problem of designing a Content Distribution Network (CDN), which is a technology used to efficiently distribute electronic content throughout an existing IP network. Our design proposal consists of jointly deciding on (i) the number and placement of proxy servers on a given set of potential nodes, (ii) replicating content on the proxy servers, and (iii) routing the requests for the content to a suitable proxy server such that the total cost of distribution is minimized. We model the problem using a nonlinear integer programming formulation. The novelty of the proposed formulation lies in simultaneously addressing three interdependent problems in this context as well as explicitly representing the distribution structure of a CDN through the objective function. We offer a linearization for the model, develop an exact solution procedure based on Benders' decomposition and also utilize a variant of this procedure to accelerate the algorithm. In addition, we provide a fast and efficient heuristic that can be used to obtain near-optimal solutions to the problem. Finally, the paper concludes with computational results showing the performance of the decomposition procedure and the heuristic algorithm on randomly generated Internet topologies.

Keywords: Content distribution network; Benders' decomposition; Integer programming; Exact solution

1. Introduction

The World Wide Web has experienced an explosive growth in the past decade. As the size of the content delivered and the number of users has increased tremendously in recent years, the clients have started to experience unacceptable delays. Other consequences of this high rate of usage are increased loads on the server(s) and network congestion. As Saroiu et al. [1] demonstrate in a recent study, the average size of the delivered content has changed from about 2 KB to 4 MB, which is an increase in the range of thousand-folds. According to Verma [2], if there is a serious degradation of performance in a network, this is most probably due to interfering cross-traffic, caused by the delivered content.

The delays experienced by the end users also have consequences from the perspective of content providers, particularly economic. As Zona Research reports [3], "the amount of time taken for web pages to load is one of the most critical factors in determining the success of a site and the satisfaction of its users". A widely observed standard is that a typical client will abandon a web site which fails to download in less than 8 s. According to Zona Research, about $4.35 billion may be lost in online sales in 1999 due to unacceptably slow response times. Moreover, the potential loss

* Corresponding author. Tel.: +90 312 2664054; fax: +90 312 2901544.
E-mail addresses: tolgab@bilkent.edu.tr (T. Bektas), ooguz@bilkent.edu.tr (O. Oguz), iradjouveysi@yahoo.co.uk (I. Ouveysi).

0305-0548/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2005.09.013

in 2001 is reported to be over $25 billion [4]. Hence, distributing content effectively to public has become a major problem of today.

Unless additional technological schemes are employed, these problems will continue to negatively affect the success of web sites and their potential sales. One immediate solution seems to be adding new infrastructure, although Datta et al. [5] argue that this approach is expensive and might only shift bottlenecks to other parts of the network rather than eliminating them. Thus, any solution should concentrate on efficient usage of the existing infrastructure.

1.1. Caching strategies

We start with introducing some terminology that is used in this paper. 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. We will use the term *object* to refer to a specific item of the content. The term *content provider* refers to a unit, which holds the content for the access of others. We will denote by the term *client* the network user who issues requests for content.

A widely adopted technique to reduce the overall traffic in the network is *caching*. Caching can be described as holding an accessed content in storage centers called *cache*s, to where future accesses to this specific content are made.

In general, there are three kinds of caching: client-based caching, server-based caching and proxy-based caching. The first type implements caching at the client side (either in the browser located on the client's computer or at a gateway) and serves only requests made from this specific location. Since these caches are usually of limited sizes, only a restricted amount of content can be stored. Thus, when a new content needs to be stored, it must replace an existing object. This brings the need for a replacement policy to determine which objects should be replaced by the new ones. Client-based caching is therefore an approach of limited use, since it can only serve a relatively small population of clients. But the real problem with such an approach is that the content provider has limited control over the content once it has been downloaded from the origin server and placed into caches [6]. The second approach, server-based caching, is performed by installing additional caches at various places within the server. Although this type of caching helps to share the load on the server to the side components, it has a small effect on reducing the incoming traffic to this server.

The third type is usually performed using a web proxy located somewhere between the client site (such as a company or a university proxy) and the origin server. When the client issues a request, the proxy will intercept the request and serve the client if the requested content is located in the cache. Otherwise, the request will be further sent to the server and the content will be accessed from the server itself. Furthermore, a copy will be stored at the proxy to serve further requests. These proxies are located at different points on the network, so they can serve a large number of clients and are very effective in reducing the network traffic.

This type of idea has given way to the new emerging technology referred to as *content distribution* (or *delivery*) *networks* (CDNs). The goal of CDNs is to replicate the content from the origin server(s) to geographically distributed surrogate sites, from which the clients receive the requested content on behalf of the origin server. CDNs therefore aim at moving the content as close as possible to the clients. A CDN can significantly improve the performance of a network and reduce the total cost associated with distributing content, since the clients no longer have to be served by the origin server but instead they receive the content from a proxy server located nearby.

The content that is to be distributed is primarily held in the origin server(s) owned by the content provider. The place where the replicated content is held is referred to as the *proxy* (*or surrogate*) *server*, installed by the CDN. Two types of replications are possible for content distribution. The first type, *full replication*, can be performed when proxies have large storage capacities and the content consists of small-sized objects (e.g. web pages, text files). The surrogate servers are then said to serve as *mirror*s of the origin server(s), holding the whole content. A second type of replication is where only a selective subset of content is replicated in the proxies. This is referred to as *partial replication* and is generally performed when the size of the content is significantly large (such as multimedia files) and the proxy has a limited storage capacity. We refer the reader to the papers by Datta et al. [5,7] for further details.

Commercially, a number of companies have started to offer hosting services for content distribution such as Akamai [8], Speedera [9], Digital Island [10] and Mirror Image [11]. As Vakali and Pallis [12] report, about 2500 companies are reported to be using CDNs as of December 2003. According to the same study, Akamai [8], for instance, has over 12,000 servers in 62 countries and hosts popular customers such as Apple, CNN, MSNBC, Reuters and Yahoo. Another large scale CDN, Digital Island [10], has about 2500 surrogate servers spanning 35 countries and hosts popular

pages such as AOL, Microsoft and Hewlett Packard. Medium sized CDNs, on the other hand, have smaller number of surrogate servers. Mirror Image [11], for instance, is a CDN with about 22 surrogate servers spanning North America, Europe and Asia. Being another CDN, Inktomi [13] has 10 surrogate servers deployed throughout China.

In this paper, we consider the problem of designing a CDN. Our design proposal consists of jointly deciding on (i) the number and placement of proxy servers on a given set of potential nodes (known as the *proxy server location problem*), (ii) replicating content on the proxy servers (known as the *object replication problem*), and (iii) routing the requests for the content to a suitable proxy server (known as the *request routing problem*) such that the total cost of distribution is minimized.

The rest of the paper is as follows. Section 2 provides a review on related work. Section 3 includes a formal description of the design problem and presents a novel integer programming model. An exact solution algorithm along with a variation is presented in Section 4. Section 5 reports on some computational experiments. The paper concludes with some future research directions in Section 6.

2. Previous research

Although CDN is a relatively new subject for research, significant amount of papers have been published in this area. This section is by no means an exhaustive review on the problem, but rather is an attempt to provide a brief background on the related literature.

The proxy server location problem has been studied well in the literature, and among many published papers we mention [14–16].

The object replication problem is also well studied with respect to CDNs, if not extensively. In [17], a distributed algorithm is offered to allocate electronic content over a network with a tree structure to minimize the total storage and communication cost. The optimal object location problem in CDNs with storage constraints has recently been investigated in [18,19]. The former formulates the problem as an integer program to minimize the average travel time of the objects and the latter emphasizes on the distribution of objects in multimedia applications.

The object replication problem is in some ways similar to the *database location problem* in computer communications networks. This problem in general consists of placing copies of a database throughout a computer network with regard to the trade-off between the cost of accessing the various copies of the database in the network and the cost of storing and updating the additional copies. Fisher and Hochbaum [20] presented a mixed-integer model for this problem. To solve a variant of this problem, Pirkul [21] has offered a Lagrangian based solution algorithm along with a heuristic procedure. The reader may also refer to [22–25] for additional studies on the database location problem.

There are a number of papers where different problem instances in CDNs are studied simultaneously. For example, Ryoo and Panwar [26] study the problem of distributing multimedia files in networks involving the determination of the communication link capacities, sizing the multimedia servers and distributing different types of content to each server. In another study by Xu et al. [27], given a maximum number of potential proxies and a tree-like topology, the authors investigate the problem of determining the optimal number and location of proxies along with placement of the replicas of a single object on the installed proxies. Xuanping et al. [28] discuss the joint problem of proxy server placement and object replication in a CDN, subject to a budget constraint. The authors assume that each client is assigned to its closest proxy. In a similar context, Laoutaris et al. [29] consider the joint problem of optimal location of the objects together with the capacity dimensioning of the proxies. Another work by the same authors [30] study the storage capacity allocation problem for CDNs, which takes into account the optimal location of the proxies, the capacity of each proxy and the objects that should be placed in each proxy. The assignment of clients is not considered as a decision problem, in that they assume a given hierarchical topology where the assignment of clients is predetermined. Nguyen et al. [31] have considered a problem similar to the one considered in this paper and proposed an integer linear programming formulation with a solution approach based on the Lagrangean relaxation techniques. However, Nguyen et al. [31] assume a different problem setting than the one considered in this paper, the details of which will be explained shortly. Finally, we mention the study by Almeida et al. [32] where they consider the problem of jointly routing requests and placing proxy servers, provide a corresponding optimization model and propose a number of heuristics in an attempt to minimize the total server and network delivery cost.

3. A location-based model for distributing electronic content

We consider a fully meshed network $G = (V, E)$, where V is the set of nodes and $E = (\{i, j\} : i, j \in V)$ is the set of logical links. In practice, we generally have a network that is not necessarily complete (the *physical layer*). However, we assume that there are logical direct links (one-hop paths) that connect every pair of nodes in the *link layer*. A logical link may pass through one or more physical links. The correspondence between the logical and physical links on the network can easily be established via an indicator function defined below:

$$\delta_{ij}^{l} = \begin{cases} 1 & \text{if the logical link } \{i, j\} \in E \text{ uses link } l \text{ in the physical network,} \\ 0 & \text{otherwise.} \end{cases}$$

Having defined such a correspondence, we can concentrate to work with the logical links. These links constitute the backbone of the network and are used for the distribution of the content. The unit cost of transferring an object over the link $\{i, j\} \in E$ is denoted by c_{ij} (similarly c_{jS} denotes the unit cost from proxy j to the origin server). This cost may represent unit bandwidth cost, number of physical hops, etc. In this paper, we assume that the link capacities are sufficiently large to allow the transfer of the available content. This assumption is valid when we are talking about nation-wide networks or networks of small size such as Intranets (similar assumptions have also been made in several previous studies, e.g. see [17,33]).

We assume here that a suitable routing protocol is used to ensure a guaranteed end-to-end bandwidth reservation to achieve some Quality of Service (QoS) capability in the network. The QoS functionalities can then be implemented in the physical layer of the backbone network parallel to the optimization task of content distribution that is performed in the link layer. However, QoS related concepts are out of the scope of this paper and we focus only on the content distribution optimization task here.

The node set V is further partitioned into three nonempty, mutually exclusive and exhaustive subsets as $V = I \cup J \cup S$, where I is the set of clients, J is the set of potential nodes on which proxy servers can be installed and S is the set of origin servers. We may either have single or multiple origin servers. We limit our study here to a single origin server (i.e. $|S| = 1$) but the model can easily be extended to the multiple server case ($|S| > 1$). Further, we assume without loss of generality that no client can directly access the origin server (e.g. for security reasons).

Each client is assumed to be served by exactly (or at least) one proxy server. In any case, we assume that the client retrieves the requested object as a whole from only a single server. This consideration is based on a well-stated result given by Kangasharju et al. [34], who demonstrated through simulation that retrieving an object as a whole from a single proxy results in a better performance compared to the situation where the client receives different parts of the object from different proxies. In the case that a content requested by a client is not found in the assigned proxy server, then the client is able to access it from the origin server via the path from the corresponding proxy server to the origin server, but at the expense of an additional transfer cost.

We assume that the capacity of the potential server at site j is s_j with the instantiation cost of f_j. We define K as the set of objects located in the origin server and assume that the size of each object $k \in K$ is b_k. Also we consider that the probability of requesting object $k \in K$ by client $i \in I$ is denoted by d_{ik}.

We emphasize that we consider an existing network in our study and that we ignore the option of network expansion here.

The CDN architecture just described is depicted in Fig. 1, with the origin server (in the box), three proxy servers and nine clients, where each client is connected to a single proxy server.

The problem then consists of simultaneously deciding on the following issues:

- the assignment of each client to a single (or multiple) proxy server;
- the number and the location of the proxy servers to be installed in the CDN on a given set of potential sites;
- the objects to be located in each proxy server.

The objective is to design a CDN such that the total cost is minimized. We refer to this problem as the *single server content distribution network problem* and denote it by SCDNP. It is important to note here that we consider this problem from the perspective of the CDN provider, who would like to reduce its expenses by minimizing the total cost. There may be several other schemes that can be employed in a CDN, such as dynamically selecting the best proxy for a client that is offering the lowest response time. We ignore such a scheme as we do not consider real-time decisions and

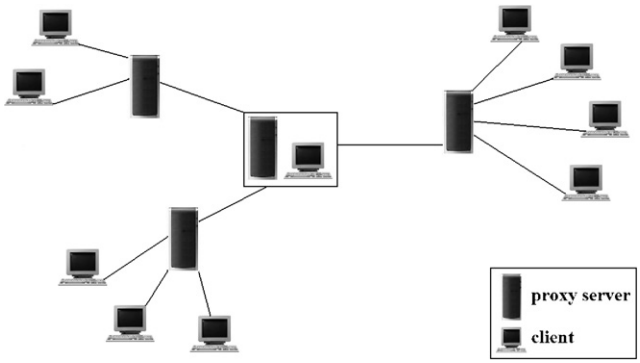

Fig. 1. A typical CDN with three proxy servers and nine clients.

our design consists of making all the decisions a priori. Although this may seem like a very static approach for such an application, replication for content distribution based on steady state demand rates are shown to have significant benefits in [35].

We define the following binary decision variables:

$$y_j = \begin{cases} 1 & \text{if node } j \in J \text{ is selected as a proxy server ,} \\ 0 & \text{otherwise,} \end{cases}$$

$$x_{ij} = \begin{cases} 1 & \text{if client } i \in I \text{ is assigned to proxy server } j \in J, \\ 0 & \text{otherwise,} \end{cases}$$

$$z_{jk} = \begin{cases} 1 & \text{if proxy server } j \in J \text{ holds object } k \in K, \\ 0 & \text{otherwise.} \end{cases}$$

We now present a novel integer programming formulation for the SCDNP as follows:

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

$$\text{s.t.} \quad \sum_{j\in J} x_{ij} = 1, \quad \forall i \in I, \tag{2}$$

$$x_{ij} \leqslant y_j, \quad \forall i \in I, \quad j \in J, \tag{3}$$

$$\sum_{k\in K} b_k z_{jk} \leqslant s_j y_j, \quad \forall j \in J, \tag{4}$$

$$y_j \in \{0, 1\}, \quad \forall j \in J, \tag{5}$$

$$x_{ij} \in \{0, 1\}, \quad \forall i \in I, \quad j \in J, \tag{6}$$

$$z_{jk} \in \{0, 1\}, \quad \forall j \in J, \quad k \in K. \tag{7}$$

The formulation presented above is a nonlinear integer programming model, which is known to be hard to solve. The objective function is given in (1). Here, the first summation represents the total cost of installing proxy servers. The second summation denotes the total cost of transferring the content. The first part of this summation is for the case when client i receives content k that is located in proxy j (reflected by the cost $b_k d_{ik} c_{ij} z_{jk} x_{ij}$ summed over all the proxies, clients and objects). In the case when the requested object is not located in a proxy server, an additional cost incurs to further request the object from the origin server. This is reflected in the second part of the summation by $(b_k d_{ik}(1 - z_{jk})(c_{jS} + c_{ij})x_{ij})$, over all proxies, clients and objects. Similar cost functions have also been suggested in [36,19,5].

Constraint (2) is the assignment constraint indicating that each client must be assigned to exactly one proxy server. Note that the case where each client can receive objects from different proxies may easily be accommodated by using

the following constraint instead of (2):

$$\sum_{j \in J} x_{ij} \geqslant 1, \quad \forall i \in I. \tag{8}$$

Constraint (3) implies that a client can be assigned to a node only if a proxy server is installed on that node. Constraint (4) implies that the total size of objects held in each proxy server is constrained by the available capacity. Finally, constraints (5)–(7) denote the integrality of the decision variables.

The model just presented for the SCDNP has $|J|+|I||J|+|J||K|$ binary variables and $|I|+|I||J|+|J|$ constraints. Next, we show the complexity status of the problem.

Proposition 1. *The SCDNP is $\mathcal{NP}$-hard.*

Proof. We prove the proposition by restriction (see [37]). Let us consider the following instance of the SCDNP: $K=\{1\}$ (i.e. there is only a single object), $b_1 = b$, $d_{i1} = d_i$ and $s_j \geqslant b, \forall j$ (i.e. all the proxies have a sufficiently large capacity). Since there are no capacity constraints in this case, (4) becomes redundant and $z_{j1} = 1, \forall j$. In this case, the problem becomes the uncapacitated facility location problem (FLP), which is known to be $\mathcal{NP}$-hard (see e.g. [38]). □

In the following, we give some observations regarding the connections of SCDNP with other well-known problems:

(1) If each proxy server has an infinite capacity, i.e. $s_j \gg M, \forall j \in J$, then the capacity restriction on the storage amount becomes redundant. In this case, since the whole content is replicated in each proxy server, the clients receive all their demands from the proxy servers. This problem is the *multicommodity uncapacitated FLP* (see [39,40]). When there is only a single object, the SCDNP becomes the well-known *uncapacitated FLP*.
(2) If there is no fixed cost of installing a proxy server and each proxy has an infinite storage capacity, then the problem becomes the well-known *p-median* problem.

Unlike the work of Nguyen et al. [31], we do not assume that the total capacity of established proxies is greater than the total demand. In fact, our model is a better representation of a real life operation in a CDN, where a requested object not available in the proxy server is cached from the origin server. This is exactly the reason for the nonlinearity of the objective function in the SCDNP model. In addition, Nguyen et al. [31] allow the client request to be fractionally served by proxies holding the requested content, where we do not allow for such a situation (as explained previously).

There are several ways to solve the nonlinear integer programming model presented above. One strategy would be to use techniques specifically devised for quadratic optimization problems. In this study, we consider a solution approach based on a special linearization of the model as well as heuristic procedures. These are presented in the following sections.

4. Model linearization

It is clearly seen that the objective function (1) contains a quadratic term due to the multiplication of the x_{ij} and z_{jk} variables. One way for linearization is to introduce a new binary variable into the formulation along with three sets of constraints (as shown in [41,42]). We hereby propose a more efficient linearization, using a continuous variable and only two sets of constraints. Although there are other linearization techniques available for such quadratic functions, as we will show later in this section, our linearization brings forth a special structure that will enable us to use the exact solution approach to our model. The following is the basis of our linearization.

Proposition 2. *The following constraints are sufficient to linearize the objective function* (1) *of the SCDNP model*,

$$\varphi_{ijk} \leqslant x_{ij}, \quad \forall i \in I, \ \ j \in J, \ \ k \in K, \tag{9}$$

$$\varphi_{ijk} \leqslant z_{jk}, \quad \forall i \in I, \ \ j \in J, \ \ k \in K, \tag{10}$$

where $\varphi_{ijk} = z_{jk} x_{ij}$ and is a continuous variable in [0, 1].

Proof. By simplifying the second summation of the objective function as $\sum_{i\in I}\sum_{j\in J}\sum_{k\in K}(b_k d_{ik}(c_{ij}+c_{jS})x_{ij} - b_k d_{ik} c_{jS}\varphi_{ijk})$, where $\varphi_{ijk} = z_{jk}x_{ij}$, the proof relies on the observation that the coefficient of φ_{ijk} in the objective function is $-b_k d_{ik} c_{jS}$, which is always negative. By definition, φ_{ijk} should be 1 if and only if $z_{jk} = 1$ and $x_{ij} = 1$, and 0 for all other cases. Now, assume that $z_{jk} = 1$ and $x_{ij} = 1$ for a specific (i, j, k) triplet. Then, according to constraints (9) and (10), φ_{ijk} is only constrained by the upper bound 1 and the minimizing objective function implies $\varphi_{ijk} = 1$. In all other cases (i.e. $x_{ij} = 1$, $z_{jk} = 0$; or $x_{ij} = 0$, $z_{jk} = 1$; or $x_{ij} = 0$, $z_{jk} = 0$) constraints (9) and (10) together imply $\varphi_{ijk} = 0$. □

Note that the linearizing variable φ_{ijk} is actually an indicator of whether client i is connected to the proxy server j and the proxy server holds the requested object k or not. Under the proposed linearization, the objective function (1) reduces to the following:

$$\sum_{j\in J} f_j y_j + \sum_{i\in I}\sum_{j\in J}\sum_{k\in K}(b_k d_{ik}(c_{ij}x_{ij} + c_{jS}(x_{ij} - \varphi_{ijk}))). \tag{11}$$

We can now construct the integer linear programming formulation of the SCDNP, denoted by M(SCDNP), as Minimize (11): s.t. (2)–(4), (9), (10), (5)–(7), $\varphi_{ijk} \in [0, 1]$.

5. A preliminary computational analysis of the SCDNP model

In order to assess the computational performance of the proposed model for the SCDNP, we have performed some preliminary computational experiments. These experiments are based on randomly generated Internet topologies, using an Internet topology generator, available at the web address http://topology.eecs.umich.edu/inet/, which mimics the characteristics of the real Internet topology. The instances used for comparison purposes have different number of potential proxy locations, clients and objects (denoted by $|J|$, $|I|$, and $|K|$, respectively). Based on the generated topology, we have created a hierarchical network as follows: The topology was modified such that the average cost between the origin server and the clients is two times the average cost between the origin server and the potential proxy locations, i.e., it is two times more expensive to send content to a client than that of sending content to a proxy server. The server is selected to be the first node of the generated network. The unit transfer cost on a link is generated in accordance with the hierarchical network. The size of each object is chosen from a uniform random variable between 0 and 1. The fixed cost of installing a proxy server is also a uniform random variable between 200 and 1000. The capacity of each proxy server is calculated as a random number between 1% and 50% of the total size of all objects. The demand distribution for the objects has been modelled using a Zipf-like distribution. This distribution is known to obey the characteristics of web requests (see [43]). The Zipf-like distribution assumes that the probability of a request for an object is inversely proportional to its popularity. More specifically, let a number of objects be ranked in order of their popularity where object i in this order is the ith most popular object. Then, given an arrival for a request, the conditional probability that the request is for object i is given by the following:

$$P_K(i) = \frac{\Omega}{i^{\alpha}},$$

where

$$\Omega = \left(\sum_{i=1}^{K}\frac{1}{i^{\alpha}}\right)^{-1}$$

is a normalization constant and α is an exponent. When $\alpha = 1$, we have the true Zipf-distribution. In [43], it is shown that α varies from 0 to 1 for different access patterns and is usually between 0.64 and 0.83 for web objects. This value was recorded to be $\alpha = 0.733$ for multimedia files in [19]. We have used this specific value in our implementation.

The metric used to assess each solution is a *normalized cost* metric, as used in other studies (e.g. [27]), which is defined as follows:

$$\text{normalized cost} = \frac{\text{cost of the network output by the procedure}}{\text{cost of the network without any replicated proxies}}.$$

Table 1
Computational analysis of M (SCDNP)

$\|J\|$	$\|I\|$	$\|K\|$	v_{IP}	Time	v_{LP}	Gap
2	5	10	0.196743	0	0.176784	10.15
2	5	20	0.220278	2.2	0.160009	26.79
2	5	30	0.243757	3.4	0.203921	22.16
2	5	40	0.203196	14.4	0.164390	23.61
2	5	50	0.159261	120.2	0.117373	27.75
2	10	10	0.259229	1.2	0.175000	34.56
2	10	20	0.232459	33	0.173245	28.54
2	10	30	0.195147	83.2	0.143140	26.21
2	10	40	0.185774	562.6	0.129690	34.28
2	10	50	0.153416	3266.4	0.103495	32.28
3	5	10	0.118412	2.2	0.053974	59.11
3	5	20	0.106240	13.6	0.050549	52.48
3	5	30	0.087297	18.2	0.027413	72.85
3	5	40	0.079468	29.8	0.018525	77.12
3	5	50	0.097295	27.8	0.052646	49.92
3	10	10	0.136549	3	0.059750	56.60
3	10	20	0.109765	578.2	0.032691	72.67
3	10	30	0.114386	1007.44	0.042822	63.59
3	10	40	0.129712	1265.6	0.057250	59.16
3	10	50	0.081450	3668.6	0.026444	67.54

Here, the cost of the network without any replicated proxies is the scheme where all the clients are assumed to retrieve the requested content from the origin server. Note that the smaller the normalized cost, the better the solution found by the procedure.

Table 1 presents the computational efficiency of the proposed model on randomly generated instances with varying configurations as shown in the first three columns of the table. The values presented in each line are the averages of five randomly generated instances with the same configuration. Columns v_{IP}, Time and v_{LP}, respectively, present the optimal solution value of the instance found by CPLEX 9.0, the required solution time and the LP relaxation provided by the model. Finally, the last column presents the gap between the LP-bound and the integer optimal solution and is calculated by $((v_{IP} - v_{LP})/v_{IP})100$.

As the table indicates, it becomes harder to solve the instances to optimality as the size of the instance increases. Furthermore, as also shown in the last column of the table, the gaps can be quite high. This is due to the type of linearization used in the previous section and refrains us from using any approach based on the LP-relaxations (such as a cutting plane algorithm). Although there are alternative tighter linearizations (e.g. see [44]), we will continue to use the proposed linearization in the rest of the paper, since it brings forth a special structure of the model that enables us to develop an exact solution algorithm based on decomposition. This is explained further in the next section.

6. An exact solution algorithm: Benders' decomposition

To solve M(SCDNP), we use the decomposition approach of Benders [45]. We also note that the solution approach presented here is a demonstration of an exact solution procedure for similar integer models with a quadratic objective function. The approach is simply based on fixing some of the variables to some predetermined values, yielding a *subproblem*, from which a *master problem* is derived and the approach iterates back and forth between two problems. Consider, now, fixing variables z_{jk} and y_j to either 0 or 1 (while maintaining feasibility) and denote these fixed values by $\overline{z}_{jk}$ and $\overline{y}_j$, respectively. Consequently, we can omit the capacity constraint (4) and the integrality constraints (5)

and (7). The resulting subproblem is as follows:

$$\begin{aligned}
\text{minimize} \quad & \sum_{j\in J} f_j \overline{y}_j + \sum_{i\in I}\sum_{j\in J}\sum_{k\in K}(b_k d_{ik}(c_{ij}x_{ij} + c_{jS}(x_{ij} - \varphi_{ijk}))) \\
\text{s.t.} \quad & \sum_{j\in J} x_{ij} = 1, \quad \forall i \in I, && (12)\\
& -x_{ij} \geqslant -\overline{y}_j, \quad \forall i \in I, \ j \in J, && (13)\\
& -\varphi_{ijk} + x_{ij} \geqslant 0, \quad \forall i \in I, \ j \in J, \ k \in K, && (14)\\
& -\varphi_{ijk} \geqslant -\overline{z}_{jk}, \quad \forall i \in I, \ j \in J, \ k \in K, && (15)\\
& x_{ij} \in \{0,1\}, \quad \forall i \in I, \ j \in J, && (16)\\
& \varphi_{ijk} \in [0,1], \quad \forall i \in I, \ j \in J, \ k \in K. && (17)
\end{aligned}$$

It is obvious that the subproblem still includes $|I||J|$ binary variables. In what follows, we show that we can relax the binary variables in the interval [0, 1] and still obtain an integral solution.

Proposition 3. *The constraint matrix A of the subproblem with the constraints* (12)–(15) *is totally unimodular* (TU).

Proof. Rearranging the constraints of the subproblem, we first observe that the constraint matrix is of the form (A, I), where I corresponds to constraints (13) and (15), and A corresponds to the remaining constraints. Therefore, it suffices to show that A is TU (see [46, p. 39]). It is easy to see that each entry of A is either $+1$, -1 or 0 and each column contains at most two nonzero coefficients. Now, consider a partition $M_1 = M$ and $M_2 = \emptyset$ of the set M of rows. In this partition, each column j containing two nonzero coefficients satisfies $\sum_{i\in M_1} a_{ij} - \sum_{i\in M_2} a_{ij} = 0$. Thus, A is TU and the LP relaxation of the subproblem provides an integral solution. □

Based on the result of the preceding proposition, we relax constraint (16) such that $0 \leqslant x_{ij} \leqslant 1$. We can then omit constraints (16) and (17), since these are already implied by the remaining constraints.

We can now proceed with the dual of the subproblem, as shown below:

$$\begin{aligned}
\text{maximize} \quad & \sum_{j\in J} f_j \overline{y}_j + \sum_{i\in I} u_i - \sum_{i\in I}\sum_{j\in J} \overline{y}_j w_{ij} - \sum_{i\in I}\sum_{j\in J}\sum_{k\in K} \overline{z}_{jk} n_{ijk} \\
\text{s.t.} \quad & u_i - w_{ij} + \sum_{k\in K} m_{ijk} \leqslant \sum_{k\in K} b_k d_{ik}(c_{ij} + c_{jS}), \quad \forall i \in I, \ j \in J, \\
& -m_{ijk} - n_{ijk} \leqslant -b_k d_{ik} c_{jS}, \quad \forall i \in I, \ j \in J, \ k \in K, \\
& u_i : \text{free}, \quad \forall i \in I, \\
& w_{ij}, m_{ijk}, n_{ijk} \geqslant 0, \quad \forall i \in I, \ j \in J, \ k \in K,
\end{aligned}$$

where u_i, w_{ij}, m_{ijk} and n_{ijk} are dual variables associated with constraints (12), (13), (14) and (15), respectively. In the objective function, $\sum_j f_j \overline{y}_j$ is a constant since the y_j's are fixed. We show in the following that the dual problem is bounded and has a feasible solution.

Proposition 4. *The dual problem is always bounded and feasible for a given set of feasible y_j and z_{jk} variables.*

Proof. Since the dual problem is generated from the primal subproblem, it suffices to show that the latter is always bounded and feasible. Notice that the set of $\overline{z}_{jk}$ and $\overline{y}_j$'s are always chosen to be feasible (satisfying constraints (4), (5) and (7)) to construct the primal subproblem. Thus, it is always possible to find a feasible set of x_{ij}'s (e.g. assigning each client to the closest proxy server). Since each of the variables can take values in the interval [0,1], the subproblem and its dual are bounded and feasible. □

The objective function of the dual subproblem provides a cut for the master problem, which is shown in the following:

$$\begin{aligned}
&\text{minimize} \quad z_0 \\
&\text{s.t.} z_0 \geqslant \sum_{i\in I} u_i^* - \sum_{i\in I}\sum_{j\in J} w_{ij}^* y_j - \sum_{i\in I}\sum_{j\in J}\sum_{k\in K} n_{ijk}^* z_{jk} + \sum_{j\in J} f_j y_j, \qquad (18)\\
&\sum_{k\in K} b_k z_{jk} \leqslant s_j y_j, \quad \forall j \in J,\\
&z_0 \geqslant 0,\\
&y_j \in \{0,1\}, \quad \forall j \in J,\\
&z_{jk} \in \{0,1\}, \quad \forall j \in J, \ k \in K.
\end{aligned}$$

In the master problem, constraint (18) is Benders' cut and $u_i^*, w_{ij}^*, m_{ijk}^*$ and n_{ijk}^* are the optimal values of the respective variables in the dual subproblem. The master problem, in general, includes an exponential number of Benders' cuts (18), and thus is not computationally tractable. However, this problem may be overcome by relaxing these cuts and dynamically generating only a subset of them at each iteration. This is accomplished by using the values of the dual variables to compute an extreme point and generate the corresponding optimality cut. We provide below a formal definition of the algorithm:

Benders' decomposition algorithm:

(1) Let $\text{LB} = -\infty$, $\text{UB} = +\infty$. Fix z_{jk} and $y_j, \forall j \in J, k \in K$, to some feasible configuration.
(2) Repeat the following until $(\text{UB} - \text{LB})/\text{UB} \leqslant \varepsilon$, where ε is a convergence parameter indicating the maximum allowable gap between the upper and lower bounds.

   (a) Solve the subproblem and set $\text{UB} = \min\{z^*, \text{UB}\}$, where z^* is the optimal solution value of the subproblem.
   (b) Add the corresponding Benders' cut to the master problem. Let the optimal solution value of the master problem be z_0^*. Set $\text{LB} = z_0^*$, update $\overline{z}_{jk}, \overline{y}_j$.
(3) Stop and conclude with the optimal solution.

In any iteration of Benders' algorithm, the optimal solution value of the master problem provides a lower bound on the optimal solution value of the main problem, which monotonically increases at every iteration. The solution value of each subproblem, on the other hand, is an upper bound, not necessarily decreasing at each iteration. This is why the upper bound is chosen as $\text{UB} = \min\{z^*, \text{UB}\}$ in Step (2) of the algorithm, where z^* is as defined therein.

Note that the master problem is a mixed integer linear program with $|J| + |J||K|$ binary variables and a single continuous variable. Although the master problem has a number of binary variables much smaller than the original problem, our preliminary computational experiments show that it is still difficult to solve directly. Since a master problem needs to be solved at each iteration of the algorithm, this may turn out to be quite costly. To overcome this drawback, we make use of a modification that will help to accelerate the procedure.

6.1. Modified algorithm

The modification, as proposed in [47], partitions the procedure into two stages. In the first stage, only the LP-relaxations of the master problems are solved, where the relaxations are obtained by substituting the integrality conditions on y_j and z_{jk} by $0 \leqslant y_j \leqslant 1$ and $0 \leqslant z_{jk} \leqslant 1$, respectively. This procedure is continued (a) until no further iterations are possible (which indicates that we have found the optimal solution value of the LP-relaxation of the original problem), (b) for a prespecified number of iterations, or (c) when the gap between the upper and lower bounds is less than a prespecified amount. One can then switch to the second stage, in which the mixed-integer master problem is solved. Note that this modification results in the optimal solution of the model provided that the "integer" master problem is used in the algorithm or otherwise, the modified Benders' algorithm only ends up with the optimal LP-relaxation value of the original model. We refer to this modification as algorithm MB1. It should also be noted that the complexity of algorithm MB1 is proportional to the complexity of the mixed integer master problem (i.e. number of integer variables) that is solved in each iteration of the second stage.

7. Heuristic algorithms

Considering the size of the instances that may arise in designing a CDN, we also present a fast and a simple heuristic algorithm for the solution of the SCDNP. Given any ordering $\{1, 2, \ldots, |J|\}$ of the set of potential proxy locations, the heuristic begins with opening the first proxy in the first iteration and continues on opening the ith proxy of the ordering in the ith iteration, until all the potential proxies are opened. The ordering chosen here can be such that the proxies are sorted in the nondecreasing order of f_j's or the nonincreasing order of s_j's. At every iteration, the request routing and object replication problems are solved. The former is done through assigning each client to the proxy with minimum total cost. The latter is based on what we refer to here as the *saving* of an object $k \in K$, calculated as follows:

$$\pi_{jk} = b_k c_{jS} \sum_{N(j)} d_{ik}, \tag{19}$$

where $N(j)$ denotes the set of clients connected to proxy j. A similar measure has also been used by Xuanping et al. [28]. Verbally, the saving of an object is the amount of cost reduced by placing object k on a proxy j. Then, the objects are sorted in the non-increasing order of their savings. Let $\{O_{j1}, O_{j2}, \ldots\}$ denote this order. The objects are placed in the available proxy server(s) using this order without violating the capacity constraints. We also use $b(L)$ to denote $\sum_{k\in L} b_k$. The outline of the procedure is given in the following:

procedure GH
Let the best solution be $\overline{c} = +\infty$ and $l = 1$.
Repeat the following while $l \leqslant |J|$.
  Select the first l sites to be opened, denoted by $J \supseteq \overline{J} = \{1, \ldots, l\}$.
  for each client $i \in I$
    let $x_{ij^*} := 1$ such that $\sum_{k\in K} b_k d_{ik} c_{ij^*} = \min_{j\in\overline{J}} \left(\sum_{k\in K} b_k d_{ik} c_{ij}\right)$
  for each server $j \in \overline{J}$
    Sort the objects and let the ordering be $\{O_{j1}, O_{j2}, \ldots, O_{j|K|}\}$.
    $L := \emptyset$.
    $t := 1$
    while $b(L) \leqslant s_j$
    begin
      $k := O_{jt}$
      if $b(L \cup \{k\}) \leqslant s_j$
        $z_{jk} := 1$
        $L := L \cup \{k\}$
      $t \leftarrow t + 1$
    end
  Calculate the cost of the current solution c^l.
  If $c^l < \overline{c}$, set $\overline{c} = c^l$.
  $l \longleftarrow l + 1$.

Here, set L given in the algorithm is used to record the set of objects located on a proxy, for each $j \in J$. We name this procedure as the *greedy heuristic* and denote it by *GH*.

8. Computational results for the proposed algorithms

In this section, we report our computational experience with the decomposition algorithm and the heuristic procedure presented in the previous sections. All the algorithms were implemented in C on a Sun UltraSPARC 12 × 400 MHz with 3 GB RAM. State-of-the-art optimization software CPLEX 9.0 was used to solve the linear and integer programs at each step of MB1. For this algorithm, we have set the convergence parameter to $\varepsilon = 0.01\%$ (i.e. the algorithm stops as soon as the gap between the upper and lower bounds is less than 0.01%). The algorithm was specified to switch from the first stage to the second stage (i.e. start to solve the master problem as a mixed-integer problem) when the gap

Table 2
Comparison results of MB1, CPLEX and GH

$\|J\|$	$\|I\|$	$\|K\|$	*Gap*	*Iter*	*Int*	t_{MB1}	t_C	$d_{C/MB1}$	$d_{C/GH}$	$d_{MB1/GH}$
2	50	10	0.00	50.8	33.6	12.6	53.2	0.00	1.18	1.18
2	100	10	0.00	77.4	54	51.4	2334.4[a]	0.07	3.56	3.63
2	150	10	0.00	46.2	39	23.6	2228.4[a]	0.00	1.23	1.23
2	200	10	2.53	143	122	213.0	2944.6	−0.03	3.51	3.48
2	50	20	15.77	217.2	174.8	2775.2	5806.8[a]	−0.23	4.78	4.53
2	100	20	5.83	166	139.2	1281.4	2969.4[a]	0.09	1.42	1.50
2	150	20	17.01	223.2	200	2650.8	10803[a]	0.34	2.32	2.66
2	200	20	8.11	126.6	107.4	865.0	6504.2[a]	0.05	0.79	0.81
3	50	10	9.90	304.2	173.2	972.8	353.4	0.00	3.77	3.77
3	100	10	13.83	245.4	200	539.6	2144	0.00	1.58	1.57
3	150	10	6.57	181	125.6	312.6	2346	−0.07	1.97	1.90
3	200	10	17.55	103.8	71.2	478.2	2311.8[a]	0.28	0.24	0.53

[a]Indicates that the set contains problems for which the solution time of CPLEX exceeded 3 h (10 800 s) without finding the optimal solution.

between the upper and lower bounds falls under 5%. We have also imposed an upper limit on the number of integer master problems that can be solved in MB1, which is 200.

One alternative way to solve the proposed model is to use a commercial software. We have chosen to use the state-of-the-art optimization software CPLEX 9.0 as a benchmark to assess the performance of MB1. For comparison purposes, a time limit of 10 800 s (3 h) was imposed on CPLEX.

The performance of the algorithm was tested on random instances, which are generated using the parameters explained in Section 5. For the experiments, a batch of 60 problems were generated with the following specifications: The number of potential proxy locations ($|J|$) ranges between 2 and 3. That of clients ($|I|$) ranges between 50 and 200 and that of objects ($|K|$) ranges between 10 and 20. There are 12 different combinations of these dimension parameters as may be seen in the first 3 columns of Table 2. The numerical values in each row are calculated over five problems for each configuration.

The average final gap, the average number of iterations, the average number of integer master problems solved and the average time required by algorithm MB1 to solve the instances are given under the columns *Gap*, *Iter*, *Int*, and t_{MB1}, respectively. The next column, t_C, corresponds to the average time required for CPLEX 9.0 to solve the instances. In the case that either algorithm MB1 or CPLEX exceeds the predefined limit, we consider the value of the best feasible solution obtained so far as the output of the algorithm.

As far as the results given in Table 2 are concerned, we see that algorithm MB1 is able to solve to optimality problems with $|I| = 2$, $|K| = 10$ and $|J| \leqslant 150$ considerably faster than CPLEX. We observe that as the instances increase in size, the final gap of algorithm MB1 increases as well. To this purpose, we additionally provide column $d_{C/MB1}$, which shows the percent difference between the final solutions found by MB1 and CPLEX for each instance. This value is calculated by $((v_C - v_{MB1})/v_{MB1})100$, where v_C and v_{MB1} are the values of the solutions found by CPLEX and MB1, respectively. The differences presented in this column indicate that the solutions found by both algorithms do not significantly differ. However, algorithm MB1 is able to obtain such solutions in substantially shorter computation times as compared to those of CPLEX. The reason for the increasing gap is due to the linearization used here, which is shown to output very large integer gaps.

Table 2 also includes the results obtained by the GH, given in the last two columns. The columns $d_{C/GH}$ and $d_{C/MB1}$, respectively, present the percent deviations of the GH from that of CPLEX and MB1. These are calculated by $((v_{GH} - v_C)/v_C)100$ and $((v_{GH} - v_{MB1})/v_{MB1})100$, respectively, where v_{GH} is the value of the solutions found by *GH*. As indicated in these two columns, the heuristic is able to find solutions that are within at most 5% of the solutions found by the exact solution algorithms. Since this heuristic runs very fast (within a few seconds for all the instances given in the table), we have not recorded the solution times.

9. Conclusions and further issues

Content distribution network is a new technology aimed at increasing the effectiveness of the distributing content. In this paper, we have developed an integer programming model to optimally locate proxy servers, allocate each object to a proxy server and assign clients to the proxies so as to minimize the total transfer cost of the content. Our approach differs from the related studies considering a similar problem with respect to the objective function, which allows one to consider the case when the requested object is not found in any proxy server.

Since the proposed model includes a quadratic objective function, we have made use of a linearization in devising a decomposition-based exact algorithm, along with a variant to accelerate it. Computational experiments based on randomly generated Internet topologies suggest that the proposed algorithm is superior to CPLEX and is very efficient especially when the number of clients is huge and the number of objects is comparably small (as generally is the case in multimedia environments). The algorithm is also a demonstration of an exact solution technique for similar integer models with quadratic objective functions.

The level and the distribution of traffic tend to vary in a telecommunications network, which is partly due to the stochastic nature of customer demand and partly due to the variability in usage pattern over the course of a day. In such a situation, the algorithm proposed here can be executed repeatedly in periodic time units to take into consideration such a multi-hour nature traffic pattern that will help in optimizing the content distribution.

There are many open optimization problems for content delivery and caching as indicated by Datta et al. [5]. Within the scope of this paper, several extensions may consist of issues such as proxy-sizing (the problem of determining the optimal capacities of the proxies) and pricing (the problem of finding the price for the CDN to charge to its customers), as well as considering CDNs with multiple origin servers. In this study, we have not taken into account any bandwidth limitations as we assumed to be working on networks with such a characteristic (such as *national networks* or *Intranets*). However, large networks spanning multiple countries usually have a limited bandwidth capacity. Introducing a bandwidth constraint into the proposed model would surely make the solution much more difficult and hence is offered as a further research avenue. Finally, while the exact solution approach developed here may be of use for smaller structures (such as virtual private networks or multimedia networks), fast heuristic algorithms would be required for very large networks, such as the Internet itself. The greedy algorithm offered herein seems to be a viable alternative in this respect, as it is shown to be capable of providing near-optimal solutions.

Acknowledgements

We are indebted to Erhan Erkut and Kursad Asdemir for their constructive comments during the earlier stages of this research, and to Paul Boustead for providing valuable suggestions and comments later on. Thanks are also due to two anonymous referees for their comments that helped to improve the quality of the paper.

References

[1] Saroiu S, Gummadi KP, Dunn RJ, Gribble SD, Levy HM. An analysis of Internet content delivery systems. In: Proceedings of 5th symposium on operating systems design and implementation (OSDI), Boston, MA; December 2002.
[2] Verma DC. Content distribution networks: an engineering approach. 1st ed., New York: Wiley; 2001.
[3] The economic impacts of unacceptable web-site download speeds. Technical Report, Zona Research, April 1999. Available at <http://also.co.uk/e-hosting.html>[accessed 24.06.2004].
[4] The need for Speed II. Zona Market Bulletin, Zona Research, April 2001. Available at <http://www.websiteoptimization.com/speed/1/> [accessed 24.06.2004].
[5] 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 2003;49(10):1425–44.
[6] Kangasharju J. Internet content distribution. PhD thesis, L'Universite de Nice -Sophia Antipolis; April 2002.
[7] 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 electronic companion pages. 2003 [available at <http://mansci.pubs.informs.org/>].
[8] Akamai <http://www.akamai.com>
[9] Speedera <http://www.speedera.com>
[10] Digital Island <http://www.sandpiper.net>
[11] Mirror Image <http://www.mirror-image.com>

[12] Vakali A, Pallis G. Content delivery networks: status and trends. IEEE Internet Computing 2003;7(6):68–74.
[13] Inktomi <http://www.inktomi.com>
[14] Li B, Golin MJ, Italiano GF, Deng X, Sohraby K. On the optimal placement of web proxies in the Internet. In: Proceedings of IEEE INFOCOM'99, vol. 3, New York, March 1999. p. 1282–90.
[15] Qiu L, Padmanabhan VN, Voelker GM. On the placement of web server replicas. In: Proceedings of IEEE INFOCOM'01, vol. 3, AK, USA: Anchorage; April 2001. p. 1587–96.
[16] Woeginger GJ. Monge strikes again: optimal placement of web proxies in the internet. Operations Research Letters 2000;27(3):93–6.
[17] Cidon I, Kutten S, Soffer R. Optimal allocation of electronic content. Computer Networks 2002;40(2):205–18.
[18] Kangasharju J, Roberts J, Ross KW. Object replication strategies in content distribution networks. Computer Communications 2002;25(4): 376–83.
[19] 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, 2003. p. 557–61.
[20] Fisher ML, Hochbaum DS. Database location in computer networks. Journal of the Association for Computing Machinery 1980;27(4): 718–35.
[21] Pirkul H. An integer programming model for the allocation of databases in a distributed computer system. European Journal of Operational Research 1986;26:401–11.
[22] Gavish G. Optimization models for configuring distributed computer systems. IEEE Transactions on Computers 1987;C-36(7):773–93.
[23] Gavish B, Suh MW. Configuration of fully replicated distributed database system over wide area networks. Annals of Operations Research 1992;36:167–92.
[24] Chari K. Resource allocation and capacity assignment in distributed systems. Computers and Operations Research 1996;23(11):1025–41.
[25] Hakimi SL, Schmeichel EF. Locating replicas of a database on a network. Networks 1997;30(1):31–6.
[26] Ryoo J, Panwar SS. File distribution in networks with multimedia storage servers. Networks 2001;38(3):140–9.
[27] Xu J, Li B, Lee DL. Placement problems for transparent data replication proxy services. IEEE Journal on Selected Areas in Communications 2002;20(7):1383–98.
[28] Xuanping Z, Weidong W, Xiaopeng T, Yonghu Z. Data replication at web proxies in content distribution network. Lecture notes in computer science, vol. 2642. Berlin: Springer; 2003. p. 560–9.
[29] Laoutaris N, Zissimopoulos V, Stavrakakis I. Joint object placement and node dimensioning for Internet content distribution. Information Processing Letters 2004;89(6):273–9.
[30] Laoutaris N, Zissimopoulos V, Stavrakakis I. On the optimization of storage capacity allocation for content distribution. Computer Networks 2005;47(3):409–28.
[31] Nguyen T, Chou CT, Boustead P. Resource optimization for content distribution networks in shared infrastructure environment, In: Proceedings of the Australian telecommunications networks applications conference (ATNAC 2003), 2003. Available at <http://atnac2003.atcrc.com/ORALS/NGUYEN-resource.pdf>.
[32] Almeida JM, Eager DL, Vernon MK, Wright SJ. Minimizing delivery cost in scalable streaming content distribution systems. IEEE Transactions on Multimedia 2004;6(2):356–65.
[33] Backx P, Lambrecht T, Dhoedt B, DeTurck F, Demeester P. Optimizing content distribution through adaptive distributed caching. Computer Communications 2005;28(6):640–53.
[34] Kangasharju J, Ross KW, Roberts J. Performance evaluation of redirection schemes in content distribution networks. Computer Communications 2001;24(2):207–14.
[35] Venkataramani A, Yalagandula P, Kokku R, Sharif S, Dahlin M. The potential costs and benefits of long-term prefetching for content distribution. Computer Communications 2002;25(4):367–75.
[36] Krishnan P, Raz D, Shavitt Y. The cache location problem. IEEE/ACM Transactions on Networking 2000;8(5):568–82.
[37] Garey MR, Johnson DS. Computers and intractability: a guide to the theory of NP-completeness. San Franciso, CA: W.H. Freeman and Company; 1979.
[38] Cornuejols G, Nemhauser GL, Wolsey LA. The uncapacitated facility location problem. In: Mirchandani PB, Francis RL, editors. Discrete location theory. New York: Wiley; 1990. p. 119–71 [chapter 3].
[39] Gao L-L, Robinson Jr. EP. A dual-based optimization procedure for the two-echelon uncapacitated facility location problem. Naval Research Logistics 1992;39:191–212.
[40] Klincewicz JG, Luss H. A dual-based algorithm for multiproduct uncapacitated facility location. Transportation Science 1987;21:198–206.
[41] Padberg M. The Boolean quadric polytope: some characteristics, facets, and relatives. Mathematical Programming 1989;45(1):139–72.
[42] Plastria F. Formulating logical implications in combinatorial optimisation. European Journal of Operational Research 2002;140(2):338–53.
[43] 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, New York; March 1999. p. 126–34.
[44] Adams WP, Sherali HD. A tight linearization and an algorithm for zero-one quadratic programming problems. Management Science 1986;32(10):1274–90.
[45] Benders JF. Partitioning procedures for solving mixed-variables programming problems. Numerische Mathematik 1962;4:238–52.
[46] Wolsey LA. Integer programming. New York: Wiley; 1998.
[47] McDaniel D, Devine M. A modified Benders' partitioning algorithm for mixed integer programming. Management Science 1977;24(3):312–9.