Video-on-Demand Network Design and Maintenance Using Fuzzy Optimization

Arash Abadpour, Attahiru Sule Alfa, *Member, IEEE*, and Jeff Diamond

***Abstract*—Video-on-demand (VoD) is the entertainment source that, in the future, will likely overtake regular television in many aspects. Although many companies have deployed working VoD services, some aspects of the VoD should still undergo further improvement in order for it to reach to the foreseen potentials. An important aspect of a VoD system is the underlying network in which it operates. According to the huge number of customers in this network, it should be carefully designed to fulfill certain performance criteria. This process should be capable of finding optimal locations for the nodes of the network as well as determining the content that should be cached in each one. While this problem is categorized in the general group of network optimization problems, its specific characteristics demand a new solution to be sought for it. In this paper, which is inspired by the successful use of fuzzy optimization in similar problems in other fields, a fuzzy objective function that is heuristically shown to minimize the communication cost in a VoD network is derived while also controlling the storage cost. Then, an iterative algorithm is proposed to find a locally optimal solution to the proposed objective function. Capitalizing on the unrepeatable tendency of the proposed algorithm, a heuristic method for picking a good solution from a bundle of solutions produced by the proposed algorithm is also suggested. This paper includes a formal statement of the problem and its mathematical analysis. In addition, different scenarios in which the proposed algorithm can be utilized are discussed.**

***Index Terms*—Fuzzy optimization, network design, video-on-demand (VoD).**

I. Introduction

A VIDEO-ON-DEMAND (VoD) system is dominantly a one-way network that transmits large video files from a service provider to customers [1]. When a customer demands a video file, one of the nodes of the system will provide a temporary copy that can be watched for a certain period of time [2]. According to the tight bandwidth and latency constraints of a VoD system, rigorous analysis is necessary before its deployment [3], [4].

Looking at the available literature, the common trend is to consider a tree structure for the VoD network (e.g., see [5], [6], and the references therein). One reason for this consideration is that the flow in the network is one directional. Moreover, the contents of the network are very gradually added and are not ever modified. The tree-structured network enables the designer to focus on portions of it while ignoring the rest at different stages. This is known in the literature as aggregation [7]. Fig. 1 shows a sample problem overlaid by a potential solution. Here, the degree of darkness of the points shows their demand volume (i.e., the darker it is, the higher is the volume). The assumption made here is that, before designing the network, we have a practically acceptable estimation for the distribution of customers over different geographical locations. Here, a customer may be a household, a block, or a neighborhood, at different stages.

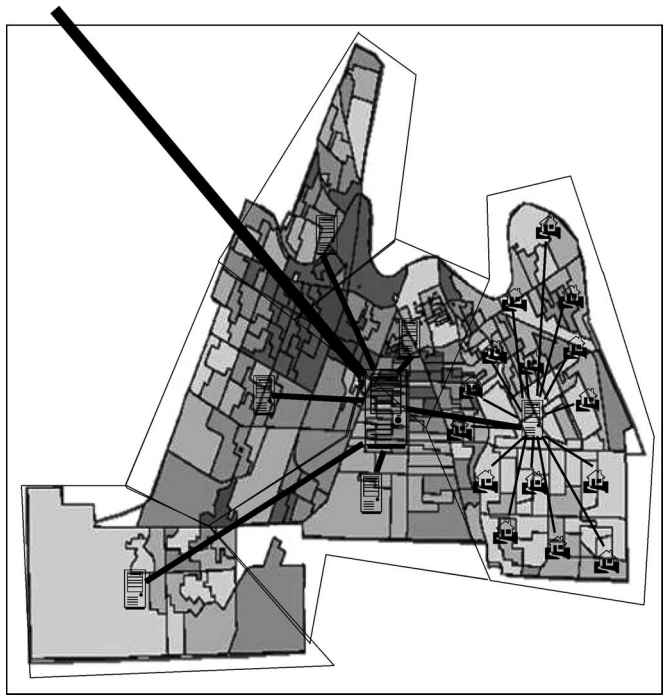


Fig. 1. Imaginary tree-structured VoD network.

In this paper, we look at one subregion related to one node in Fig. 1. In this way, the VoD design problem is formulated as locating a known number of nodes to serve a given population with a given library. In addition, the contents at the nodes are also unknown. Furthermore, the assignment in the network should be determined; namely for each customer and each video file, the solution should indicate which node should be contacted. These three layers of unknowns are the decision variables of the optimization problem in which the cost in the VoD network is minimized. The definition of cost used in this paper includes the cost of communication in the network plus the cost of storing the files in the nodes.

The VoD system design is a task that has to be guided by a human supervisor. Thus, we do not look for a fully automated algorithm. This is because the problem depends on

Manuscript received July 20, 2007. The work of A. S. Alfa was supported in part by the Natural Sciences and Engineering Research Council. This paper was recommended by Associate Editor N. R. Pal.

A. Abadpour and A. S. Alfa are with the Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, MB R3T 5V6, Canada, and also with the TRLabs, Winnipeg, MB R3T 6A8, Canada.

J. Diamond is with the TRLabs, Winnipeg, MB R3T 6A8, Canada.

Digital Object Identifier 10.1109/TSMCB.2007.912744

many different parameters that may not be easily incorporated into it at the same time. In addition, VoD design is not a task done thousands of times in a fraction of a second, unlike an image segmentation task for example. So, we are looking for a system that can help the designer enhance the design and also have the chance to intervene at any moment. Furthermore, while the definition given in the above assumes that the problem includes designing the whole network, practically, more demand is for partial solutions such as staged growth of a network or optimization of caching in it. The first problem refers to the case where due to the growth of population in a region, a new node is to be added, and the algorithm needs to locate that node, while the others are fixed. The later one addresses the issue of change in the population spread, where nodes are fixed, and we look for a more optimal caching and assignment strategy. Thus, while we formulate the problem as designing a network from scratch, we are always aware that the algorithm must be capable of addressing partial demands as well. To reach this level of flexibility, we use a fuzzy clustering-style optimization framework.

The rest of this paper is organized as follows. First, Section II briefly reviews the related literature. Then, Section III introduces the proposed method. The paper continues with Section IV, which discusses the experimental results. Finally, Section V concludes this paper.

II. Literature Review

The problem of VoD network design appears to be similar to many other problems at first glance. However, a closer look reveals that it is in fact essentially different from all of them in terms of system structure, and an independent solution should be sought for it. Here, we briefly review some of those problems.

A. Discrete Domain Problems

A category of problems, including Minimum Spanning Tree, looks for exact solutions to problems that resemble the VoD problem from a mathematical perspective (e.g., see [8]). These problems are studied in the discrete domain and generally are not applicable in the unclear circumstances where we need engineering-level approximations, such as in VoD. Other samples of these problems are network bisection [9], [10] and tree clustering [11]. See [12] for a full coverage of these problems and [13] for a fuzzy approach.

B. General Network Optimization

The general category of network optimization problems focuses on optimizing the flow of commodities in a network [14]. A well-known example of these problems, i.e., maximum flow, tries to maximize the flow from one node to another one in a graph given the constraints of maximum link capacity. This problem has been extensively analyzed, and efficient algorithms for solving it are available in many textbooks (e.g., see [15]). A more complex version of this is the minimum-cost commodity flow problem. A sample commodity flow problem assumes that a car manufacturer has factories around a geographical area, each one capable of building a number of models that are then sent to a number of retailers. The problem is to determine which factory should allocate resources to building which car and how much of the resources should be allocated. Then, we need to determine where the cars should be shipped [16]. This problem can be solved using linear programming. A further expansion on this problem is the minimum-cost multicommodity flow, where different objects share the same network, and each one has its own set of constraints on link capacities [17].

The key difference between this group of problems and the VoD problem is that in the VoD problem, we have further information about the popularity of the objects transmitted in the network. Moreover, the VoD problem is essentially concerned with the nodes' capability to handle requests, while maximum flow focuses on the links' capacities. Furthermore, in the VoD problem, every customer demands all of the commodities with different known probabilities. So, the VoD problem can be considered as a generalized version of the multicommodity problem. See [18]–[20] for details about flow optimization problems in telecommunications. We emphasize that many of these algorithms reduce to linear optimization tasks (e.g., see the numerous examples given in [20]).

C. Replication Problem

The problem of VoD network design seems to be similar to the replication problem which arises in content delivery networks (CDNs) [21]–[23]. However, the CDN problem looks for the best ways of spreading and caching content in an available network, such as the Internet. However, in contrast with VoD networks, which communicate a few giant static objects, a typical CDN contains numerous fairly small web pages and media files that are modified from time to time. Furthermore, unlike VoD networks [24], CDN does not follow a known access probability model [25].

D. Assignment

An aspect of the VoD problem is similar to the assignment problem, where the goal is to devise a bijective assignment between two same-sized sets of the given constraints [26]. Different versions of that problem use either linear [27] or quadratic [28], [29] cost functions. The assignment theory is not directly applicable to the VoD problem because there, the main goal is the assignment of the nodes to each other. While in the VoD problem, a few nodes serve many customers. However, a portion of the VoD problem is to devise an optimal assignment between nodes and customers.

E. Districting

Districting, which is the problem of partitioning a plane into contiguous regions [30], is known to be an NP-difficult problem for which very regularly different solutions exist, which satisfy the constraints to similar extents [31]. Hence, generally, the solution to districting is nondominated [32]. The general approach to solving districting problems is evolutionary tactics [33], [34]. Simulated annealing is another method frequently used [35], [36]. Our interest in districting ends here, because

that problem produces a top-level clustering and not a detailed connection pattern that is needed in the VoD problem.

F. Location Theory and Weber Problems

A characteristic property of the VoD problem is that we are looking not only for the efficient flow of information in the network but also for how the actual network should be built. This is in contrast to the other problems discussed in Sections II-B and C, which assume that the network is given.

The two-layered structure of many location problems is very well modeled using the Weber problem [37]. The one-node Weber problem looks for a point that has the minimum weighted sum of distances to a set of given points. The multiple-node version considers more than one node, while their number is known, and incorporates an assignment as well. There are numerous good surveys of the Weber problem and its different variants (e.g., see [38]). For a comprehensive discussion of the history of this problem and its roots, see [37], [39], and [40]. It is interesting to know that a book [41] published in 1750 is among the references cited for this problem [40]. An important aspect of the Weber problem is that it is called by different names in different fields, and even some researchers are unaware of the parallelism of the concepts and approaches [37]. For example, the essence of many vector quantization [42], data clustering [43], and location allocation [44] problems is the Weber problem. Drezner *et al.* [37] argue that any location problem can be traced back to a Weber problem.

The Weber problems are known to be hard-to-solve problems [45] that have multiple solutions [46]. Rather than the limited attempts to give an exact solution to the Weber problem (e.g., see [47]), the general approach for solving them is through a heuristic algorithm called p-median [44], [48], [49]. The idea of p-median is to select groups of customers and then to find the best location of a node that can independently serve each set. Then, for each node, its territory is calculated, which results in a subset of customers. This iteration is repeated until the results converge. See [50] for more details. In data clustering, this algorithm is called Hard C-Means (HCM) [51], [52]. In coding, it is called Lloyd–Max [53] or Linde-Buzo-Gray (LBG) algorithm [42] and also K-means clustering [54]. It is observed in many different frameworks that the zero–one assignment used in the classical version of the Weber problem makes it very prone to falling into the local minimum [55]. Hence, after the introduction of fuzzy sets, many researchers tried to apply this more natural theory to different versions of the Weber problem [56]. In this way, HCM was generalized into a minimum-variance fuzzy clustering technique called ISODATA [57]. Then, it was generalized by Bezdek [58], [59] when he defined the fuzziness concept and proposed the Fuzzy C-Means (FCM) algorithm. Subsequently, extensions of the FCM to different cluster shapes [60]–[62] and criteria [63] plus faster algorithms [64] were developed. A good survey of the main characteristics of the different fuzzy clustering algorithms is given in [61] and [65].

Looking back at the VoD problem, it is a multilayered Weber problem because of the existence of more than one object in the library. So, in this paper, we use the fuzzy clustering style of writing the cost function for a Weber problem. It is worth to mention that fuzzy clustering produces fuzzy assignments, while we need binary results for the VoD problem. However, there exist relabeling methods for producing binary results [66], [67]. Moreover, it is known that by controlling the fuzziness parameter, the results can be made less fuzzy [68]. Here, we select the second approach. In addition, we use another tool from Weber problems' theory to be able to work on powers of the Euclidean distance. While in coding and clustering, the squared Euclidean distance is an obvious choice (e.g., see [69]), we need a more general distance function for the VoD problem. The Weiszfeld method [70] (also known as the Miehle Algorithm [71], [72] and a few other names [44], [73]) is a practical method to deal with the Euclidean distance (and not its square). Here, we also generalize the Weiszfeld method to powers of the Euclidean distance.

G. Soft Computing Approaches

In [74], Chen briefly reviews the nonfuzzy literature of location theory and uses fuzzy set tools to tackle the problem. The approach devised in that work uses heuristically driven fuzzy rules to find optimal locations for distribution centers. See [75]–[78] and the references therein for other more extensively analyzed models. Moreover, see [79] for a model that incorporates fuzzy sets and neural networks, and see [80, Ch.14] for a comprehensive survey of these approaches. Furthermore, similar to any other large-scale hard-to-tackle problem, genetic algorithms have also been tried for giving a solution to the location problem [81]. In addition, see [82] and [83] for the application of evolutionary algorithms, heuristics, simulated annealing, and other soft computing methods in network design problems. Because fuzzy clustering has been demonstrated to work well for similar problems, we choose to accept it instead of other soft computing approaches.

III. Proposed Method

This section introduces the proposed method. The discussion begins with formally defining the problem in Section III-A. Then, the implementation of the problem as a linear programming task is briefly looked into in Section III-B. This will be used as a benchmark with which we will compare the results of the proposed algorithm. The main contributions of this paper are discussed in Section III-C, where a depiction of the VoD network design task as a fuzzy optimization problem is given. Then, Section III-D proposes an algorithm to give a locally optimal solution to the proposed problem, and Section III-E discusses a heuristic method to select an optimal solution from a bundle of locally optimal solutions. Finally, Section III-F talks about different scenarios in which the proposed algorithm can function. Table I shows the nomenclature used in this paper.

A. Problem Statement

Given the set $\mathbf{X}$, assume that there are N customers. In the model developed in this paper, we look at the problem in a large scale. Thus, the system is assumed to be in a steady state, and each customer is assigned a weight, where $\mu_{\vec{x}} > 0$ shows the relative utilization of this user. It is clear that setting all $\mu_{\vec{x}}$

TABLE I
NOMENCLATURE

N	Number of customers.
$\mathbf{X}$	Set of all customers.
$\vec{\mathbf{x}}$	Location of one customer.
$\mu_{\vec{\mathbf{x}}}$	Weight of customer $\vec{\mathbf{x}}$.
l	Library size.
d_j	Popularity of object j.
n	Number of nodes.
$\vec{\mathbf{n}}_i$	Location of node i.
l_i	Total resources of node i.
L_{ij}	Allocation for object j in node i.
$C_{\vec{\mathbf{x}}i}$	communication cost between customer $\vec{\mathbf{x}}$ and node i.
$p_{\vec{\mathbf{x}}ij}$	Assignment of customer $\vec{\mathbf{x}}$ to node i for object j.
μ	Total weight of the customers.
L	Minimum allocatable resource.
D_{ij}	Demand in node i for object j.
c	Total number of cached files.
ρ	Utilization of the caching strategy.
Δ	Communication cost.
$\hat{\Delta}$	Fuzzy objective function.
$l_j^\star$	Allocation for object j in the entire network.
φ_{ij}	Mediator variable defined in (23).
ω_{ij}	Mediator variable defined in (31).
$\psi_{\vec{\mathbf{x}}i}$	Mediator variable defined in (34).

equal to a constant gives a problem in which all customers are equally important, whereas we use this more general model because we can deal with aggregation more efficiently in this way. This idea is adopted from measuring the traffic intensity in terms of erlangs [84] (see [85] for an example in another field). Knowing that $\mu_{\vec{\mathbf{x}}} = 1$ indicates a continuously active customer, we also tolerate $\mu_{\vec{\mathbf{x}}} > 1$ as a customer that demands more than one video file at a time, for example, an apartment building.

The central problem dealt with here is designing the network that gives all the customers access to a library of video files, which we assume includes l items. Several research projects have shown that the distribution of demands for different video files is well approximated by the Zipf model [24]. In this model, the demand frequency for the jth video file is given by $d_j = cj^{-\alpha}$, where c is a normalization constant. A reasonable approximation for α is 0.729 [24]. See [86] and [87] for more comprehensive reviews and details.

For practical reasons, including computational efficiency, we are not interested in looking at independent video files. So, we assume that the library includes ul video files bundled into l chunks of u video files. Hence, we have a library of l same-sized objects where

$$d_j = \frac{\sum_{j'=u(j-1)+1}^{uj} j'^{-\alpha}}{\sum_{j'=1}^{ul} j' - \alpha}, \qquad 1 \le j \le l \tag{1}$$

is the demand frequency of the jth object. However, note that the algorithm proposed in this paper is independent of the model for d_j's.

The contents of the library will be cached in n nodes. The geometric location of these nodes (with the ith node denoted as $\vec{\mathbf{n}}_i$) and their contents are other unknowns of the problem. As the VoD service highly demands a one-by-one connection [1], [3], using the definition of $\mu_{\vec{\mathbf{x}}}$, we denote the total resources of the ith node as l_i. We assume that this value is based upon a solution to the VoD network design given by a hypothetical algorithm. If, for example, l_i is 5, it means that the respective node can serve a set $\tilde{\mathbf{X}}$ of customers that simultaneously sum to 5 ($\sum_{\vec{\mathbf{x}}\in\tilde{\mathbf{X}}} \mu_{\vec{\mathbf{x}}} = 5$). Demands made from the ith node can potentially be for any of the l objects. Here, we do a steady-state analysis by assuming that for any object j, the ith node allocates a fixed amount of its resources to it, which we denote by L_{ij}. This portion can be zero if that object is not cached in that particular node. In this way, we have

$$l_i = \sum_{j=1}^{l} d_j L_{ij} \qquad \forall i. \tag{2}$$

From this point on, L_{ij}'s and l_i's are only used in visualizations. We emphasize that neither l_i's nor L_{ij}'s are known prior to solving the problem. In fact, l_i's are calculated based on L_{ij}'s, which are themselves among the decision variables.

While a node should manage its bandwidth, it should also deliberately decide which objects it will be caching, namely, for the ith node, how many L_{ij}'s are nonzero. The traditional way of embedding this aspect in the optimization problem is to add different terms for communication cost and storage cost into the objective function (e.g., see [3]). That approach leads to an objective function that includes two terms, which makes further derivations harder. In this paper, we propose a different method.

Imposing no limitation on the number of nonzero L_{ij}'s, the optimal network will tend to cache every object in every node. This is identical to having $l = 1$ and contradicts the goal behind slicing the library. Returning to the definition of L_{ij}, our demand is to have either $L_{ij} = 0$ or a "big" L_{ij}. This means that if the jth object is cached in a node, then there should certainly be a reasonable demand for it. For this goal, we define the value of L as the minimum demand for an object at a node that justifies caching it there. Doing so, we add implicit concern over the optimality of storage to the optimization problem. This is carried out through implicit comparison of $d_j L_{ij}$ with L to decide if the jth object should be cached in the ith node or not.

Assume that customer $\vec{\mathbf{x}}$ demands the jth object that is then supplied from the ith node. The net cost of this transaction is modeled as $C_{\vec{\mathbf{x}}i}$ and is assumed to be independent of the object that is transmitted. The simplest assumption is that $C_{\vec{\mathbf{x}}i}$ is given as a lookup table for one set of nodes. In this way, the algorithm proposed here cannot optimize the location of the nodes, while the rest of the algorithm will still work. As a better formulation, we consider the case where $C_{\vec{\mathbf{x}}i}$ is a function of the Euclidean distance between $\vec{\mathbf{x}}$ and $\vec{\mathbf{n}}_i$. Here, we focus on the cost model defined as

$$C_{\vec{\mathbf{x}}i} = C\|\vec{\mathbf{n}}_i - \vec{\mathbf{x}}\|^{m_d}, \qquad m_d \ge 1 \tag{3}$$

where C is a constant that is ignored in the rest of this analysis.

Focusing on a particular customer $\vec{\mathbf{x}}$, this customer will demand access to all objects with different probabilities. Assuming that this customer demands the jth object for $j = 1, \ldots, l$

from the k_jth node, where $1 \leq k_j \leq n$, the expected cost of providing one object, any object, to this customer is equal to

$$\Delta_{\vec{\mathbf{x}}} = \sum_{j=1}^{l} d_j C_{\vec{\mathbf{x}}k_j}. \tag{4}$$

To simplify (4), we define $p_{\vec{\mathbf{x}}ij} \in \{0, 1\}$, where $p_{\vec{\mathbf{x}}ij}$ is 1 iff $\vec{\mathbf{x}}$ asks the node i for the jth object. So

$$\sum_{i=1}^{n} p_{\vec{\mathbf{x}}ij} = 1 \qquad \forall \vec{\mathbf{x}}, j. \tag{5}$$

Now, we can rewrite (4) as

$$\Delta_{\vec{\mathbf{x}}} = \sum_{j=1}^{l} \left(d_j \sum_{i=1}^{n} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij} \right). \tag{6}$$

Including the weights of the customers, the expected cost of serving one demand from any customer is equal to

$$\Delta = \frac{1}{\mu} \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \left(\mu_{\vec{\mathbf{x}}} \sum_{j=1}^{l} \left(d_j \sum_{i=1}^{n} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij} \right) \right). \tag{7}$$

Here

$$\mu = \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}}. \tag{8}$$

Following the definition of L_{ij} as the allocation at node i for the jth object, we define D_{ij} as the demand in the ith node for the jth object. Thus, we have

$$D_{ij} = \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}} p_{\vec{\mathbf{x}}ij}. \tag{9}$$

Note that in a stable network, allocation and demand are identical. Thus, we should have

$$L_{ij} = D_{ij} \qquad \forall i, j. \tag{10}$$

Satisfaction of this equation depends not only on L_{ij}'s but also on $p_{\vec{\mathbf{x}}ij}$'s through D_{ij}'s. While (9) yields

$$\sum_{i=1}^{n} D_{ij} = \mu \qquad \forall j \tag{11}$$

accompanied by (10), it also gives

$$\sum_{i=1}^{n} L_{ij} = \mu \qquad \forall j. \tag{12}$$

At this stage, the optimization problem is defined as minimizing

$$\Delta = \frac{1}{\mu} \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \left(\mu_{\vec{\mathbf{x}}} \sum_{j=1}^{l} \left(d_j \sum_{i=1}^{n} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij} \right) \right) \tag{13}$$

subject to

$$\sum_{i=1}^{n} p_{\vec{\mathbf{x}}ij} = 1 \qquad \forall \vec{\mathbf{x}}, j \tag{14}$$

$$L_{ij} = \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}} p_{\vec{\mathbf{x}}ij} \qquad \forall i, j \tag{15}$$

$$(L_{ij} = 0) \text{ or } (d_j L_{ij} \geq L) \qquad \forall i, j. \tag{16}$$

Here, $p_{\vec{\mathbf{x}}ij}$'s, L_{ij}'s, and $\vec{\mathbf{n}}_i$'s are the decision variables. While this optimization problem is defined as minimizing Δ, it is still important to know how many objects are totally cached. To measure this phenomenon, we define the value of c as the number of L_{ij}'s that are nonzero. As two extreme solutions, consider the case in which every object is cached in every node, and the case in which no node is caching anything except for one node, which caches every object. The associated c values for these solutions are nl and l, respectively. Note that at least the second case is a locally optimal solution. Moreover, that case can happen with a set of nodes caching all the objects, with each object being cached only by one node and all nodes residing on the same exact physical location. This solution is also locally optimal and gives $c = l$. As the analysis shows that $l \leq c \leq nl$, and to be independent of n and l, we define the utilization of the caching strategy as $\rho = c/nl$. Accordingly, while we mainly focus on minimizing Δ, we also want to keep the value of ρ small. This is partly done by the minimum bound on nonzero L_{ij}'s. We return to the value of c and ρ in Section III-C.

In Section III-B, we rewrite the problem defined in this section as a mixed integer linear programming (MILP) problem, which will be translated into the General Algebraic Modeling System (GAMS) and solved using the NEOS server [88]–[90]. Then, in Section III-C, we reformulate the problem using an approximation to relax one of the constraints and then transfer it into the fuzzy domain.

B. MILP Formulation

To write the optimization problem given at the end of Section III-A as an MILP problem, we use (9) and (10), and rewrite (16) as

$$d_j \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}} p_{\vec{\mathbf{x}}ij} \leq \mu q_{ij} \qquad \forall i, j \tag{17}$$

$$d_j \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}} p_{\vec{\mathbf{x}}ij} + \mu(1 - q_{ij}) \geq L \qquad \forall i, j. \tag{18}$$

Here, q_{ij}'s are unknown binary decision variables. These two constraints [accompanied by (14)] and the objective function given in (13) constitute the MILP optimization problem. The decision variables of this optimization problem are $p_{\vec{\mathbf{x}}ij}$'s, q_{ij}'s, and $\vec{\mathbf{n}}_i$'s. Note that $q_{ij} = 1$ iff the jth object is cached in the ith node. Similarly, substituting $q_{ij} = 0$ in (17) yields $p_{\vec{\mathbf{x}}ij} = 0$ for all $\vec{\mathbf{x}}$'s and the given i and j. After setting up this problem as a GAMS model, we use the BDMLP solver. The BDMLP is a simplex-based solver that is designed for small- and medium-sized problems [91]. Using this general-purpose solver, we find a typical solution that a blind optimizer will give to this

problem. This solution will then be used to assess the efficiency of the proposed algorithm.

C. Proposed Formulation

The optimization problem given at the end of Section III-A is not analytically attractive because it includes the binary decision variables of $p_{\vec{\mathbf{x}}ij}$'s. The two available choices for L_{ij}'s also add to the difficulty of analytically tackling this problem. In the following parts of this section, we give an approximate form of this optimization problem, which resembles fuzzy clustering problems, and is analytically manageable. First, we relax (16) by adding L_{ij}'s as controlling weights to the objective function. The weights are designed in a way that we can exclude (16) from the constraints but still satisfy an approximate form of (10) defined as

$$L_{ij} \simeq D_{ij} \qquad \forall i, j. \tag{19}$$

In addition, to control the allocation of objects, we add (12) as a constraint. We then translate the objective function into a fuzzy representation.

Looking at (7), we write the cost of serving the jth object to all customers as

$$\Delta_j = \frac{1}{\mu} \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \left(\mu_{\vec{\mathbf{x}}} \sum_{i=1}^{n} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij} \right). \tag{20}$$

This objective function, which is accompanied by the constraint given in (5), is very well known, as described below. Assuming $m_d = 2$ and $\mu_{\vec{\mathbf{x}}} \equiv 1$, this is a simple Weber problem. As briefly reviewed in Section II-F, this problem is called HCM in clustering theory. The name HCM is selected because we are doing a binary (hard) clustering of a given set of points into spherical clusters, which are identified by their means (C-means). This problem has been known for a long time in the literature, and several generalizations of it exist. A straightforward generalization is the introduction of weights into the problem (e.g., see [92] and [93]). This means that the different data points are differently important. As also mentioned in Section II-F, fuzzy clustering is a heuristic extension to this problem, in which $p_{\vec{\mathbf{x}}ij} \in \{0, 1\}$ is replaced with $p_{\vec{\mathbf{x}}ij} \in [0, 1]$, and (20) is rewritten as

$$\hat{\Delta}_j = \frac{1}{\mu} \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \left(\mu_{\vec{\mathbf{x}}} \sum_{i=1}^{n} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij}^m \right) \tag{21}$$

for m close to 1. As mentioned in Section II-F, this problem is called FCM, and experimental analysis during the last two decades has shown that this extension is beneficial in different problems in coding and pattern recognition [94]. Here, we choose the same approach and give a fuzzy version of (7) by rewriting it as

$$\hat{\Delta} = \frac{1}{\mu} \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \left(\mu_{\vec{\mathbf{x}}} \sum_{j=1}^{l} \left(d_j \sum_{i=1}^{n} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij}^m \right) \right) \tag{22}$$

where (5) is still in place, and $p_{\vec{\mathbf{x}}ij} \in [0, 1]$. Here, $m > 1$ is the fuzziness [58]. It is known that as m generally leans to 1^+, the fuzziness of $p_{\vec{\mathbf{x}}ij}$'s reduces [68]. In this way, we will have $p_{\vec{\mathbf{x}}ij} \in [0, \delta] \cup [1 - \delta, 1]$ for a small δ. While this is the classical approach also used in FCM [59], we further use relabeling [66] to convert the minimally fuzzy $p_{\vec{\mathbf{x}}ij}$'s into members of $\{0, 1\}$ by picking the most likely connection for each customer and each object. Note that we still need to incorporate storage cost into the objective function.

As mentioned earlier, $L^{-1} d_j L_{ij}$ is expected to be either 0 or bigger than 1. In the first case, no customer is expected to rely on the ith node for the jth object. This is a two-choice constraint that makes the objective function discontinuous and, hence, hard to work with. Moreover, it adds an extra constraint to the optimization problem and makes the application of FCM-style optimization hard. To solve these problems, we define the term φ_{ij} as

$$\varphi_{ij} = 1 + \left(\frac{d_j L_{ij}}{L} \right)^{-k}. \tag{23}$$

Here, k is a fixed power, with default values of over 10. Now, as L_{ij} leans toward 0, φ_{ij} approaches infinity. On the other hand, φ_{ij} leans toward 1 for $d_j L_{ij} \geq L$. We use this term to add a force to the objective function to carry out three goals. Namely, it helps satisfy (19), it forces small $d_j L_{ij}$'s to become 0, and it becomes transparent when the solution converges. We will discuss these points in detail in the next paragraph.

Using φ_{ij}'s, we rewrite (22) as

$$\hat{\Delta} = \frac{1}{\mu} \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \left(\mu_{\vec{\mathbf{x}}} \sum_{j=1}^{l} \left(d_j \sum_{i=1}^{n} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij}^m \varphi_{ij} \right) \right). \tag{24}$$

Note that φ_{ij} is a decreasing function of L_{ij}. Thus, if L_{ij} becomes small, φ_{ij} will lead to a very large value. Then, as $\hat{\Delta}$ is minimized, the nonzero $p_{\vec{\mathbf{x}}ij}$ for the given i and j will lead to a big contribution to $\hat{\Delta}$ and thus will not happen in the maximizer. Thus, the introduction of φ_{ij} into (24) has the effect of refraining customers from requesting objects from nodes in which they are not cached or where little resources are allocated to them. In a similar way, for large L_{ij}'s, φ_{ij} becomes small, and thus more $p_{\vec{\mathbf{x}}ij}$'s will be attracted to the respective node. This will result in increasing D_{ij}. In a similar way, smaller L_{ij}'s are likely to result in smaller D_{ij}'s. This is a balancing force that allows (19) to be satisfied. On the other hand, when L_{ij} becomes small, and hence no $p_{\vec{\mathbf{x}}ij}$ is big, then if this L_{ij} becomes 0, it will result in other $L_{i'j}$'s benefiting from (12) and growing larger. This will result in smaller $\varphi_{i'j}$'s, which eventually leads to a smaller $\hat{\Delta}$. Thus, the φ_{ij} weights are also beneficial in pushing small L_{ij}'s toward 0. Finally, for active L_{ij}'s, meaning those which are not 0, φ_{ij} is close to 1. If we can also have minimally fuzzy $p_{\vec{\mathbf{x}}ij}$'s, for which $p_{\vec{\mathbf{x}}ij}^m \simeq p_{\vec{\mathbf{x}}ij}$, then $\hat{\Delta}$ will be approximately equal to Δ. Thus, φ_{ij}'s will be transparent, and minimizing (24) will eventually result in minimizing (7). In Section IV, we show how these tendencies act in a real problem. We also show how to move from (19) to (10).

To summarize, the VoD network design problem is approximated as minimizing

$$\hat{\Delta} = \frac{1}{\mu} \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \left(\mu_{\vec{\mathbf{x}}} \sum_{j=1}^{l} \left(d_j \sum_{i=1}^{n} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij}^m \varphi_{ij} \right) \right) \tag{25}$$

subject to

$$\sum_{i=1}^{n} p_{\vec{\mathbf{x}}ij} = 1 \qquad \forall \vec{\mathbf{x}}, j \tag{26}$$

$$\sum_{i=1}^{n} L_{ij} = \mu \qquad \forall j. \tag{27}$$

This formulation is one of the main contributions of this paper. From this point on, we will work on giving a solution to this optimization problem. To do so, in the next section, we propose an algorithm for finding a local minimizer for this problem.

D. Solving the Problem: Single Trial

Remembering the FCM methodology, we use Lagrange multipliers to rewrite the objective function as [95]

$$\Phi = \hat{\Delta} + \sum_{j=1}^{l} \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \lambda_{\vec{\mathbf{x}}j} \left(\sum_{i=1}^{n} p_{\vec{\mathbf{x}}ij} - 1 \right) + \sum_{j=1}^{l} \gamma_j \left(\sum_{i=1}^{n} L_{ij} - \mu \right). \tag{28}$$

Setting $\partial \Phi / \partial p_{\vec{\mathbf{x}}ij} = 0$ and using (5), we have

$$p_{\vec{\mathbf{x}}ij} = \frac{\left(C_{\vec{\mathbf{x}}i} \varphi_{ij} \right)^{-\frac{1}{m-1}}}{\sum_{i'=1}^{n} \left(C_{\vec{\mathbf{x}}i'} \varphi_{i'j} \right)^{-\frac{1}{m-1}}}. \tag{29}$$

Moreover, deriving $\partial \Phi / \partial L_{ij}$ and equating it to 0, and then using (12), we have

$$L_{ij} = \mu \frac{\omega_{ij}^{\frac{1}{k+1}}}{\sum_{i'=1}^{n} \omega_{i'j}^{\frac{1}{k+1}}}. \tag{30}$$

Here

$$\omega_{ij} = \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij}^m. \tag{31}$$

To optimize Φ in terms of $\vec{\mathbf{n}}_i$'s, we will look at the case when m_d is 2 and when it is not independent. When $m_d = 2$, using

$$\frac{\partial C_{\vec{\mathbf{x}}i}}{\partial \vec{\mathbf{n}}_i} = 2(\vec{\mathbf{n}}_i - \vec{\mathbf{x}}) \tag{32}$$

we have

$$\vec{\mathbf{n}}_i = \frac{\sum_{\vec{\mathbf{x}} \in \mathbf{X}} \psi_{\vec{\mathbf{x}}i} \vec{\mathbf{x}}}{\sum_{\vec{\mathbf{x}} \in \mathbf{X}} \psi_{\vec{\mathbf{x}}i}}. \tag{33}$$

Here

$$\psi_{\vec{\mathbf{x}}i} = \mu_{\vec{\mathbf{x}}} \sum_{j=1}^{l} d_j \varphi_{ij} p_{\vec{\mathbf{x}}ij}^m. \tag{34}$$

When m_d is not 2, we cannot use the formulation given in (33) for finding $\vec{\mathbf{n}}_i$ [96]. Thus, we utilize a method similar to the Weiszfeld approach [70], except for the fact that the original Weiszfeld method only works for $m_d = 1$ [40], [82]. Using (3), we have [97]

$$\frac{\partial C_{\vec{\mathbf{x}}i}}{\partial \vec{\mathbf{n}}_i} = m_d \|\vec{\mathbf{n}}_i - \vec{\mathbf{x}}\|^{m_d - 2} (\vec{\mathbf{n}}_i - \vec{\mathbf{x}}). \tag{35}$$

Hence, equating $\partial \Phi / \partial \vec{\mathbf{n}}_i$ with 0, we have

$$\vec{\mathbf{n}}_i = \frac{\sum_{\vec{\mathbf{x}} \in \mathbf{X}} \psi_{\vec{\mathbf{x}}i} \|\vec{\mathbf{n}}_i - \vec{\mathbf{x}}\|^{m_d - 2} \vec{\mathbf{x}}}{\sum_{\vec{\mathbf{x}} \in \mathbf{X}} \psi_{\vec{\mathbf{x}}i} \|\vec{\mathbf{n}}_i - \vec{\mathbf{x}}\|^{m_d - 2}}. \tag{36}$$

Now, we use the fixed-point method. To do this, we use the initialization vector of

$$\vec{\mathbf{n}}_i^0 = \frac{\sum_{\vec{\mathbf{x}} \in \mathbf{X}} \psi_{\vec{\mathbf{x}}i} \vec{\mathbf{x}}}{\sum_{\vec{\mathbf{x}} \in \mathbf{X}} \psi_{\vec{\mathbf{x}}i}}. \tag{37}$$

This idea is adopted from the similar initialization in the Weiszfeld method [48]. Now we define

$$\vec{\mathbf{n}}_i^{t+1} = \frac{\sum_{\vec{\mathbf{x}} \in \mathbf{X}} \psi_{\vec{\mathbf{x}}i} \left\| \vec{\mathbf{n}}_i^t - \vec{\mathbf{x}} \right\|^{m_d - 2} \vec{\mathbf{x}}}{\sum_{\vec{\mathbf{x}} \in \mathbf{X}} \psi_{\vec{\mathbf{x}}i} \left\| \vec{\mathbf{n}}_i^t - \vec{\mathbf{x}} \right\|^{m_d - 2}}. \tag{38}$$

The computation of (38) is performed for each value of i until $\|\vec{\mathbf{n}}_i^{t+1} - \vec{\mathbf{n}}_i^t\|$ becomes less than a specified threshold. Note that while here we have used the power model shown in (3), a similar approach is applicable to many other formulations, at least to many of those that satisfy

$$C_{\vec{\mathbf{x}}i} = C\left(\|\vec{\mathbf{n}}_i - \vec{\mathbf{x}}\|^2 \right) \tag{39}$$

for a differentiable function C (see Appendix I).

Here, we have not provided any convergence proof for the iteration depicted in (38). However, we do know that for $m_d = 2$, the iteration changes into a one-step weighted average calculation, as shown in (33). In addition, it is proved that for $m_d = 1$, the algorithm converges to a unique point [98] in a linear fashion [40]. Numerous experiments show that for the case of $m_d \geq 1$, the algorithm converges. Based on this observation, we conjecture that for $m_d \geq 1$, the iteration given in (38) converges. Providing a proof for this conjecture is still an open problem. However, in Appendix II, we give a theorem that shows why $m_d \geq 1$ is important.

The approach adopted here is based on the basic Weiszfeld method, whereas there are methods for accelerated Weiszfeld that increase the convergence speed (e.g., see [37] and [99]). We do not address the acceleration methods in this paper.

Using these results in an FCM-style iterative algorithm, we can produce a locally optimal solution. However, note that this

***fVoD* Algorithm**
Inputs: n: Number of nodes. l: Number of objects. $\mathbf{X}$: Set of all customers. $\mu_{\vec{\mathbf{x}}}$: Weights of the customers. d_j: Objects' demands. m_d: Power in (3). m: Fuzziness. k: Power in (23).
Outputs: L_{ij}: Allocation pattern. $\vec{\mathbf{n}}_i$: Nodes' locations. $p_{\vec{\mathbf{x}}ij}$: Membership values. Δ: Value of the objective function.
1- $\hat{\Delta} = \infty$ **2-** Randomly produce $\vec{\mathbf{n}}_i$s. **3-** Randomly produce L_{ij}s which comply with (12). **4-** Use (29) to produce $p_{\vec{\mathbf{x}}ij}$s. **5-** Use (30) to produce L_{ij}s. **6-** if $m_d = 2$, use (33) to produce $\vec{\mathbf{n}}_i$s. **7-** if $m_d \neq 2$, use (37) and (38) to produce $\vec{\mathbf{n}}_i$s. **8-** $\hat{\Delta}^o = \hat{\Delta}$ **9-** Use (24) to produce $\hat{\Delta}$. **10-** If $\hat{\Delta}^{-1}\lvert\hat{\Delta} - \hat{\Delta}^o\rvert > \varepsilon$, go to 4. **11-** Calculate D_{ij}s using (9). **12-** $L_{ij} = D_{ij}$.

Fig. 2. Details of the algorithm fVoD. Everywhere, $j = 1, \ldots, l$, $i = 1, \ldots, n$, and $\vec{\mathbf{x}} \in \mathbf{X}$. Here, ε is the precision that we select to be 10^{-4} for our experiments.

solution is only likely to satisfy (19), and not (10). Here, we propose a method to alter the solution to produce one that does satisfy (10). Having calculated the optimal $p_{\vec{\mathbf{x}}ij}$'s, we can calculate D_{ij}'s using (9). According to (11), if we replace all L_{ij}'s with the respective D_{ij}'s, (27) will be intact. However, this manipulation might result in an increase in c because of a previous unbalance in allocation demand that has resulted in a spurious small utilization. Thus, we argue that the new value of c shows the actual number of cached objects. Nevertheless, this manipulation does not affect Δ.

Utilizing the results derived in this section, we propose the algorithm depicted in Fig. 2 to find a potential locally optimal network design. We call this algorithm fuzzy VoD network design or fVoD.

As we do not have any proof for the convexity or concavity of $\hat{\Delta}$ [100], we cannot make any argument about the possibility of fVoD not being trapped in a local minimum. In fact, in [46], the authors show the example of a similar problem that has 50 customers and 61 local minimums (also see [101]). However, we do know that fVoD does converge, because at any iteration the produced solution is better than what was worked out in the previous one (here we are assuming that we have proof for the convergence of (38) for $m_d \in (1, \infty) - \{2\}$). So, we can safely say that fVoD does find a locally optimal solution, whereas the outcome is dominantly affected by the initial choice of $\vec{\mathbf{n}}_i$'s and L_{ij}'s [102]. We use this unrepeatability to propose the fVoD^m algorithm in the next section. Before that, an analysis of the computational cost of the proposed algorithm and a note about the allocation-related variables are given below.

Define the variables w and u as the numbers of iterations of the Weiszfeld-style calculation given in (38) and fVoD, before they converge, respectively. According to the experimental results, the maximum typical values for these parameters are $w = 10$ and $u = 25$ when $N = 50$, $n = 5$, and $l = 10$. Now, the approximate cost of fVoD is $15Nnlu$ flops and $15Nnlu + 9Nnwu$ flops for the cases of $m_d = 2$ and $m_d \neq 2$, respectively.

While L_{ij}'s are the decision variables in the proposed optimization problem, they can also be used in deriving other measures such as allocation in any node or for any object. In this way, in addition to allocation at each node, which is defined in (2), we define the allocation for the jth object in the entire network as

$$l_j^\star = d_j \sum_{i=1}^{n} L_{ij}. \tag{40}$$

Accompanied by l_i, these measures will be used in visualizing a potential solution.

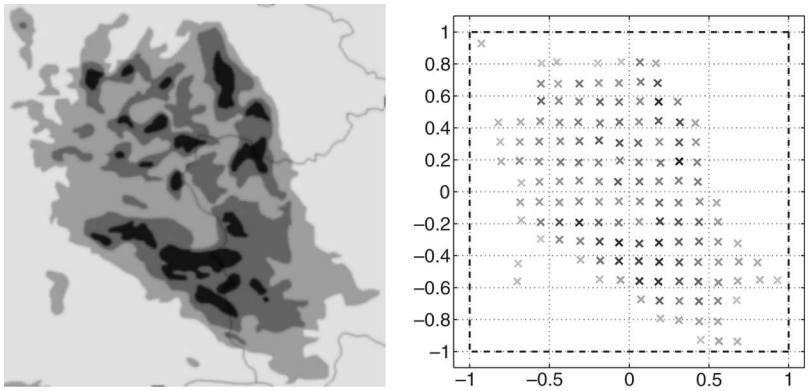
E. Solving the Problem: Searching for a Better Solution

As mentioned in Section III-D, the solution produced by fVoD may only be locally optimal. However, we do know that running multiple instances of fVoD produces potentially different solutions. Thus, we propose to run fVoD for T independent times. Let us call the ith solution S_i, which is represented by the pair (Δ_i, ρ_i). A practical way for dealing with the existence of two optimality criteria, i.e., Δ_i and ρ_i, is to devise a function of them as the total optimality criterion and then to pick the best solution based on that. Note that, here, Δ_i and ρ_i directly represent the communication cost and the storage cost, respectively. While seeming simple to implement, this approach demands a cost model that can address the cost of communication and storage at the same time. For this purpose, a reasonable idea is to calculate a linear combination of Δ_i and ρ_i, with the weights estimated from the actual cost of the equipments. However, we refrain from following this approach because we do not have these figures. To deal with this problem, we filter out the solutions that demand too much storage space. Therefore, we pick a value of ρ_0, and then from the set of solutions that satisfy $\rho_i \leq \rho_0$, we pick the one that corresponds to the least Δ. We call this algorithm fVoD^m. Note that the more computational resources are given to the fVoD^m algorithm, the more optimal results it will produce. This scalability is important for procedures designed for real applications.

F. Application Scenarios

In Section III-D, the general structure of the fVoD algorithm in presented. In addition, in Section III-E, a heuristic technique is devised to find more optimal solutions. In this section, we refer to the practical framework in which the proposed algorithms can be utilized. First, the minimal and maximal scenarios are discussed.

The minimal scenario in which fVoD can be implemented is as a part in a bigger human-directed design process. In this way, as a designer works on the VoD network, the relationships given in Section III-C can be used to optimize one aspect of the design. For example, knowing the placement of the nodes and



(a) (b)

Fig. 3. Population density in the first scenario. (a) Input population map. (b) Aggregated customers.

the caching pattern, we can use (29) to calculate the optimal assignment. Moreover, in knowing the nodes' locations and assignment, one can use (30) to optimize caching. Similarly, in knowing caching and assignment, the right combination of (33) or (37) and (38) will give the optimal location for the nodes.

The maximal scenario is the assumption that fVoD is responsible for designing the whole network, including locating the nodes. This hypothetical scenario can be carried out by fVoD. However, it should be emphasized that this scenario is mostly of theoretical interest because the problem that VoD service providers are more concerned with is the growth and maintenance of available networks. These concerns are addressed by fVoD through a slight change in the algorithm depicted in Fig. 2. As discussed in Section III-D, the essence of fVoD is the local improvement of a design. Thus, by jumping over the recalculation of L_{ij}'s, $\vec{\mathbf{n}}_i$'s, or $p_{\vec{\mathbf{x}}ij}$'s, whichever and wherever necessary, these variables will be made fixed in the entire design. In this way, we can optimize only a part of the design.

Here, as an example, we discuss the two cases of staged network growth and caching optimization. In both scenarios, we define the values $\theta_1, \ldots, \theta_n \in \{0, 1\}$, where $\theta_i = 1$ means the location of the ith node cannot change. The consecutive changes in the algorithm depicted in Fig. 2 would be rewriting Lines 6 and 7 as, "... to produce $\vec{\mathbf{n}}_i$ for all $\theta_i = 0$." Moreover, in Line 2, only those $\vec{\mathbf{n}}_i$'s for which θ_i is 0 will be randomized. In the new version of fVoD, adding n_1 nodes to a network with n_2 available nodes will be carried out through using $\theta_1, \ldots, \theta_{n_1+n_2}$, where

$$\theta_i = \begin{cases} 0, & i \leq n_1 \\ 1, & \text{otherwise.} \end{cases} \tag{41}$$

Here, the initial values for $\vec{\mathbf{n}}_i$, $i > n_1$, come from the available nodes. Similarly, recalculating the optimal caching and assignment strategy for an available network will be carried out through using an all-one θ sequence, where no $\vec{\mathbf{n}}_i$ is randomized. In Section IV, these scenarios will be looked into using sample problems.

IV. Experimental Results

The proposed algorithm is developed in MATLAB 7.0.4 on a Pentium 4 3.00 GHz with 1 GB of RAM. In this section, we analyze three scenarios and show the outcome of the algorithm in each one of them. First, we assume that a population is given and that the algorithm should design the whole network. This scenario is discussed in Section IV-A. Then, in Section IV-B, we assume that the underlying geographical area has expanded, and thus a new node should be added to the network. Finally, Section IV-C assumes that the pattern of population has changed and hence the caching policy should be reoptimized. In each scenario, the contributions of the proposed algorithm are discussed.

To produce the aggregated population, in each case, a gray-scale image is used as the population density map. This image is sliced into blocks, and the average color and the weighted central point for each block are calculated. Rejecting those blocks that exhibit an average color of less than a minimum threshold, the central points are stored in a set, each accompanied by the respective weight, i.e., the average color.

A. First Scenario: Complete Design

The image used in this scenario is the one shown in Fig. 3(a), from which $N = 133$ aggregated customers, as shown in Fig. 3(b), are extracted. Using these customers, the problem is defined as locating $n = 5$ nodes to cache $l = 10$ objects. Here, we select the power in the cost model to be $m_d = 1.3$. Moreover, the objects' demand frequencies are calculated based on a Zipf(0.729) distribution.

Using the BDMLP solver, this problem is solved in 6 min on a PIV 2.53 GHz with 512 MB of RAM. The result is a solution for which $\Delta = 0.463$ and $\rho = 0.2$. Fig. 4 shows this solution. As seen in Fig. 4(a), the first and second nodes are located on the same physical point. These two nodes make a hypernode that caches each object once and only once.

To give a visual representation of the resulting network, we draw a 3-D graph with l layers, in which each layer shows the connection pattern for one object. To give an intelligible visualization of the connections, in each layer, the convex hull of all the customers that access the same object through the same node is drawn. This is performed in Fig. 4(b) for the solution generated by BDMLP and in other figures used for visualizing other solutions hereafter. In this figure, each shade indicates one node. Here, clearly, every connection originates from the hypernode.

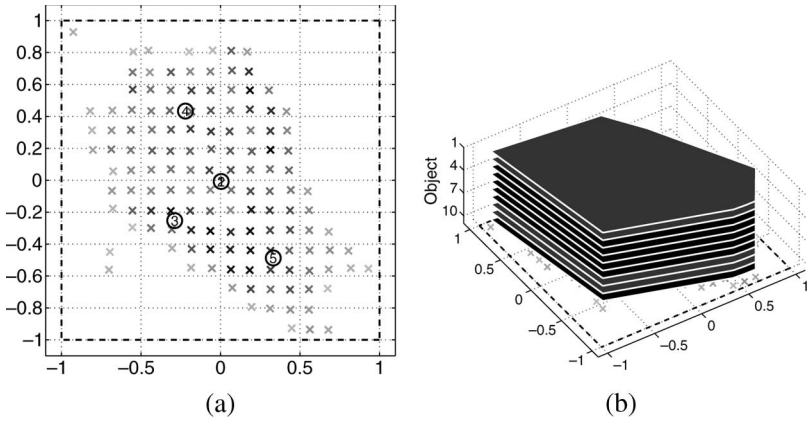


Fig. 4. VoD network designed by BDMLP in the first scenario. (a) Location of the nodes. (b) Three-dimensional representation of the network.

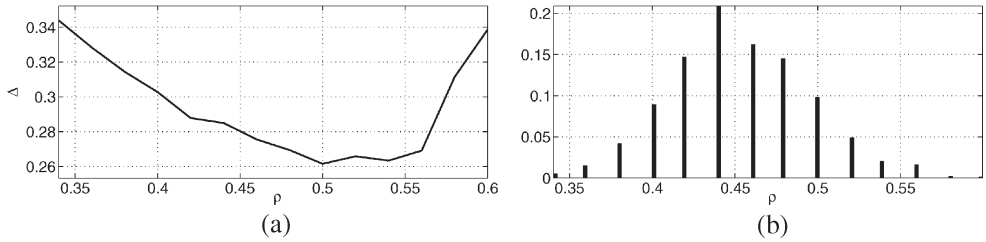


Fig. 5. Analysis of the solutions produced by the proposed algorithm for the first scenario. (a) Least Δ for each value of ρ. (b) Histogram of values of ρ.

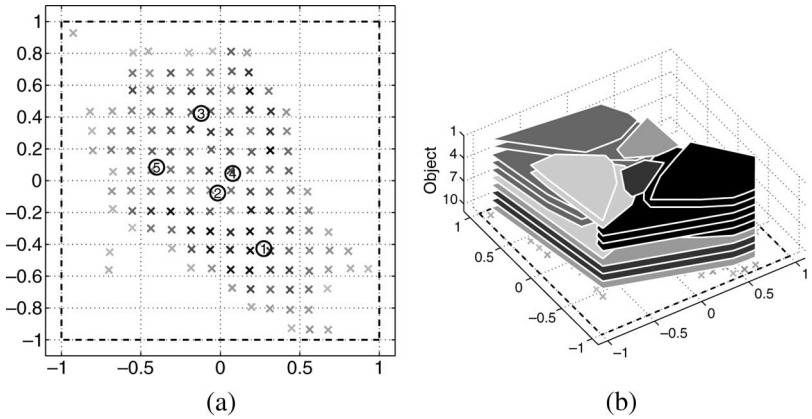


Fig. 6. VoD network designed by the proposed algorithm in the first scenario. (a) Location of the nodes. (b) Three-dimensional representation of the network.

Attempting to optimize this solution using subsets of variables, it appears to be a locally optimal solution to the approximate formulation. However, note that this solution neglects the main reason behind slicing the library into objects. We will also see that the proposed algorithm is able to produce a solution in which the communication cost is considerably lower.

To apply the proposed method, the values of the parameters are selected as $m = 1.1$, $k = 15$, and $T = 1000$. Further, to produce a reasonable value of L, $(1/2)d_l\mu$ is used. Here, $d_l\mu$ represents the maximum possible value of $d_j L_{ij}$ for $j = l$ and thus the maximum bound for L. Using these parameters, the fVoD^m algorithm takes about 10 min to run. Note that except for the set of customers and n, other scenarios use the same values of parameters, where L is independently calculated in each case.

In this scenario, the fVoD^m algorithm produces 1000 potential solutions, for which we have $\Delta_i \in [0.262, 0.521]$ and $\rho_i \in [0.34, 0.60]$. Fig. 5(a) shows the least Δ for each value of ρ in these solutions. As seen here, as ρ approaches half, the cost decreases. The increase of Δ for $\rho > 0.5$ relates to locally optimal solutions that cache very inefficiently. To describe the unexpectable rise in the cost for these values of ρ, we refer to the histogram of ρ, as shown in Fig. 5(b). As seen here, there are very few solutions for which $\rho > 0.53$. The increased least available cost for this range of ρ is then because less local solutions in this range have been examined.

Here, we select the utilization threshold to be $\rho_0 = 0.40$. After the filtering stage, 849 solutions remain, from which fVoD^m retrieves one for which $\Delta = 0.303$ and $\rho = 0.4$. This solution is comprehensively analyzed in the next parts of this section.

Fig. 6(a) shows the location of the nodes in this solution. Looking at the structure of the network, as shown in Fig. 6(b), we observe that the first objects are cached in many nodes,

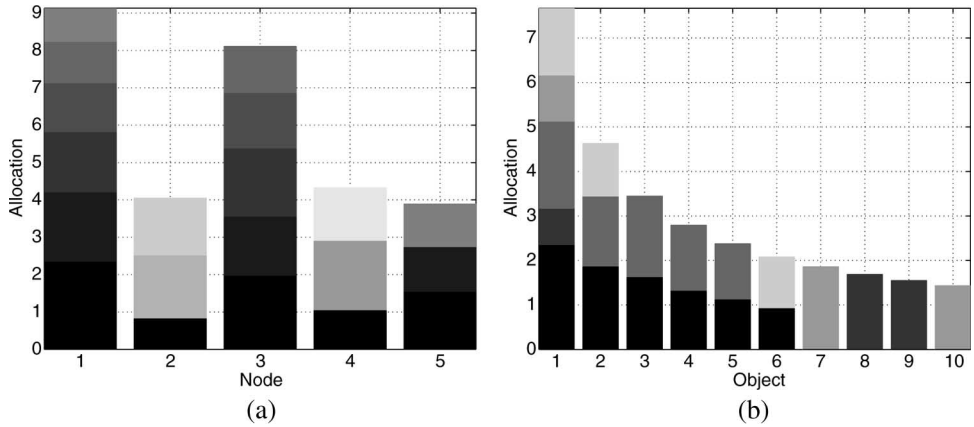


Fig. 7. Aggregate allocation for nodes and objects in the solution to the first scenario. (a) Allocation in each node. Different colors show different objects. (b) Allocation for each object. Different colors show different nodes.

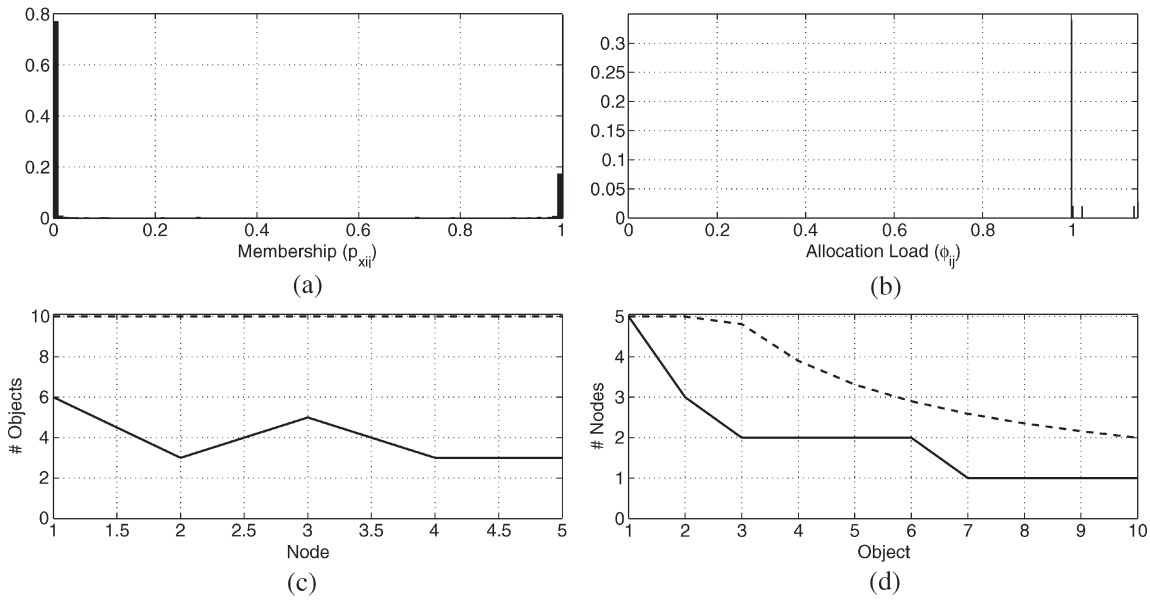


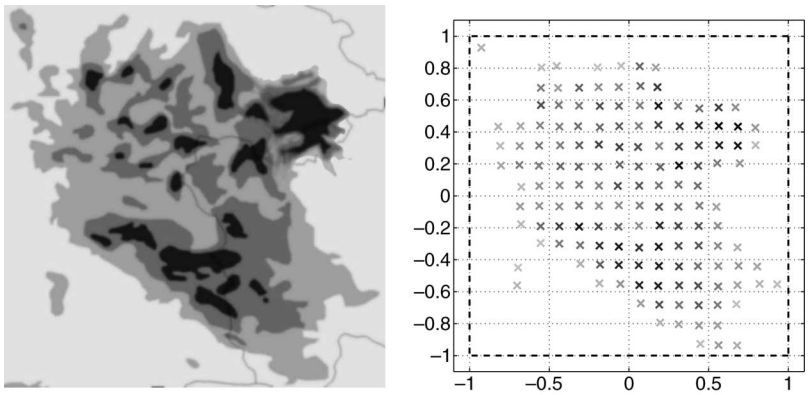
Fig. 8. Investigating the solution to the first scenario. (a) Histogram of $p_{\vec{\mathbf{x}}ij}$'s. (b) Histogram of active φ_{ij}'s. (c) Number of objects cached in each node. (d) Number of nodes in which each object is cached. In (c) and (d), the solid lines show the actual values while the dashed lines show the maximum possible ones.

while the last ones are each only stored in one node. This was expected because the criterion for caching an object is the value of $d_j L_{ij}$. Thus, for the last objects, which have smaller d_j's, L_{ij} should be big, which results in less number of L_{ij}'s being nonzero [see (12)].

Using L_{ij}s, Fig. 7(a) shows the aggregate allocation at each node (l_i). Here, different colors denote different objects. Similarly, Fig. 7(b) shows the aggregate allocation for different objects in the entire network $(l_j^\star)$. Here, different colors indicate different nodes. We can use these charts to show how the caching strategy changes in the network under different circumstances.

Finally, we look at some internal variables (see Fig. 8). Fig. 8(a) shows the histogram of $p_{\vec{\mathbf{x}}ij}$'s. As seen here, the assignment is minimally fuzzy. In fact, 96% of $p_{\vec{\mathbf{x}}ij}$'s are either less than 0.03 or more than 0.97. Fig. 8(b) shows the histogram of the active φ_{ij}'s, which shows their closeness to 1. In fact, active φ_{ij}'s vary between 1.14 and 1, i.e., the ideal value. Furthermore, the solid line in Fig. 8(c) shows the number of objects cached in each node. Here, the dashed line shows the maximum value, i.e., l. Thus, in the network designed here, in average, each node has cached 40% of the library $(\rho = 0.4)$. Looking at the solid line in Fig. 8(d), we see the number of nodes in which each object is cached. Here, the dashed line shows the maximum possible values, i.e., $\min\{L^{-1} d_j \mu, n\}$. This figure shows that while the first object is cached in every node, the four last objects are only cached in one node each.

Comparing the solution rendered by the proposed method and the one produced by BDMLP, we observe that with a similar computational complexity, the proposed algorithm cuts the cost at about 35%. Thus, for the next scenarios, we only analyze the results of the proposed algorithm. Note that the intrinsic level of flexibility that is present in fVoD is not achievable in a general optimization problem unless the problem is reorganized to separate known parameters from decision variables, in every case, independently.



(a) (b)

Fig. 9. Population density in the second scenario. (a) Input population map. (b) Aggregated customers.

The charts shown in Figs. 7 and 8 show the solution from different perspectives. These visualizations are helpful in looking into the details of a solution and in finding probable shortcomings. To save space, we do not give detailed figures for the two next scenarios, and only the structure of the network and the values of Δ and ρ are discussed.

B. Second Scenario: Adding One Node

As the underlying population pattern changes, the efficiency of the VoD system designed based on that pattern may decrease. This is one of the main challenges that the VoD service providers are facing. In the second scenario, we assume that it is necessary for a VoD network, here the network designed in Section IV-A, to add a new node to the system because of a newly added region to its coverage area. Subsequently, this change will result in the recalculation of the caching strategy for all nodes plus a new assignment.

The population density map for this scenario is shown in Fig. 9(a), with the new region added in the right. Using this map, $N = 144$ aggregated customers, as shown in Fig. 9(b), are extracted. To have the numerical figures to describe the situation before optimization is carried out, we use a minimal version of fVoD to recalculate the optimal assignment using (29). This also results in a new caching strategy, as shown in Fig. 10. Comparing this figure with Fig. 7(a) shows the change in allocation caused by fVoD to fit the available network to the new circumstances. Here, we have $\Delta = 0.337$ and $\rho = 0.4$. This means that the application of the previous design for the new population increases the cost by 11%. Now, we use fVoD^m to locate the new node and also to recalculate the caching strategy and the assignment.

To do so, we use the θ_i sequence defined as (0, 1, 1, 1, 1, 1), which means that one extra node is needed while the five available ones cannot be displaced. The output of fVoD^m after 10 min of calculation is a solution for which $\Delta = 0.308$ and $\rho = 0.4$. These figures show that the application of fVoD^m has caused about 10% decrease in communication cost. Comparing this figure with the cost in the first scenario, the addition of the new node has compensated for the newly added region, and the cost is only less than 2% more in the new solution. Fig. 11 shows this solution. As requested, five available nodes are not displaced, while their cached content is recalculated. Moreover,

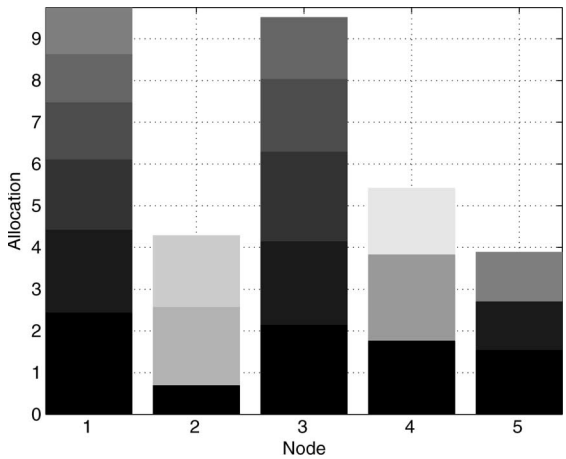


Fig. 10. Aggregate allocation for nodes in the solution to the first scenario after it is optimized by the proposed algorithm to fit the second scenario. Different colors show different objects.

as anticipated, the new node is located close to the newly added population. These results should now be investigated by a supervisor to decide whether the costs of erecting a new node justify the reduction in costs.

C. Third Scenario: Caching Optimization

Section IV-B discussed the case in which the change in population was to be dealt with by adding a new node. Here, we analyze a less severe situation, where the change in population does not justify the addition of a new node, according to a hypothetical expert's idea. Thus, the algorithm only needs to recalculate the optimal caching and assignment strategies. Fig. 12(a) shows the new population map, from which 150 customers are extracted, as shown in Fig. 12(b).

To find the current costs, we use a minimal fVoD to recalculate the optimal assignment using the known nodes and their respective content. This situation results in $\Delta = 0.323$ and $\rho = 0.37$, which shows about 5% increase in cost.

Now, using the θ_i sequence defined as (1, 1, 1, 1, 1, 1), the fVoD^m algorithm is used to optimize both the caching strategy and the assignment. Taking about 10 min of calculation, a solution is rendered in which $\Delta = 0.319$ and $\rho = 0.4$, which shows a 1% decrease in cost. Again, we see that fVoD^m is able

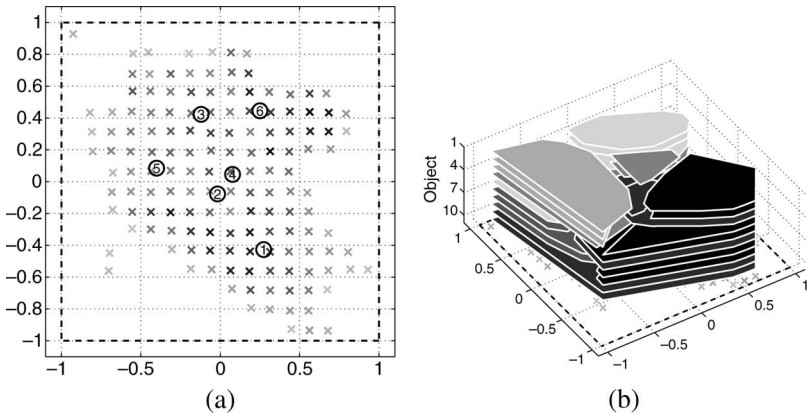


Fig. 11. VoD network designed by the proposed algorithm in the second scenario. (a) Location of the nodes. (b) Three-dimensional representation of the network.

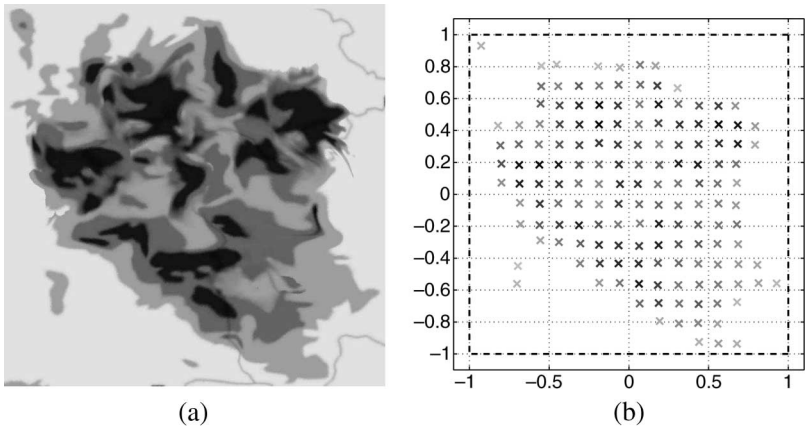


Fig. 12. Population density in the third scenario. (a) Input population map. (b) Aggregated customers.

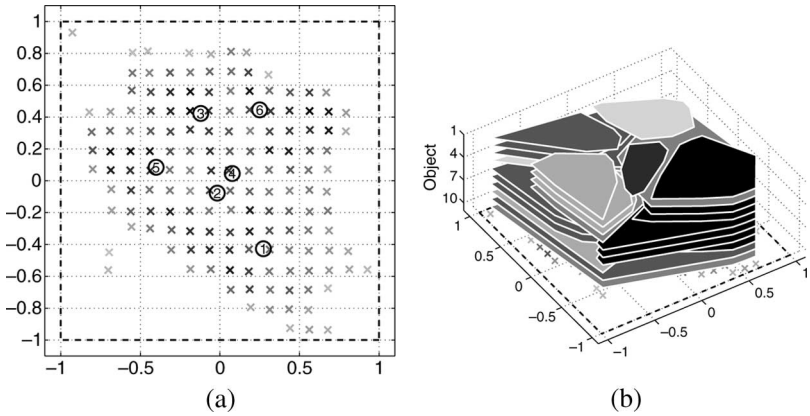


Fig. 13. VoD network designed by the proposed algorithm in the third scenario. (a) Location of the nodes. (b) Three-dimensional representation of the network.

to produce a solution in which cost is reduced. The new solution is shown in Fig. 13.

Thus, it was shown that the proposed algorithm and its minimal implementations are able to address a vast group of optimization requests in a VoD network. These demands vary from recalculating the optimal assignment or caching to designing the whole network. A major point about the proposed algorithm is that here we are using the same algorithm for different optimization tasks. In addition, the proposed algorithm is designed in a way that the more computational resources are given to it, the more optimal its output will get. Therefore, the algorithm can be used as a quick effort for finding a slightly better solution or a time-consuming more global search. Note that while we did not give a mathematical proof for the convergence of the generalized Weiszfeld, it is utilized in the experimental results discussed in this paper for about half a million times, and it has converged in every single utilization. Empirically, based on the experiments reported on here, it appears to be a reasonable conjecture that the convergence of the proposed method could be guaranteed, although we have no proof for it.

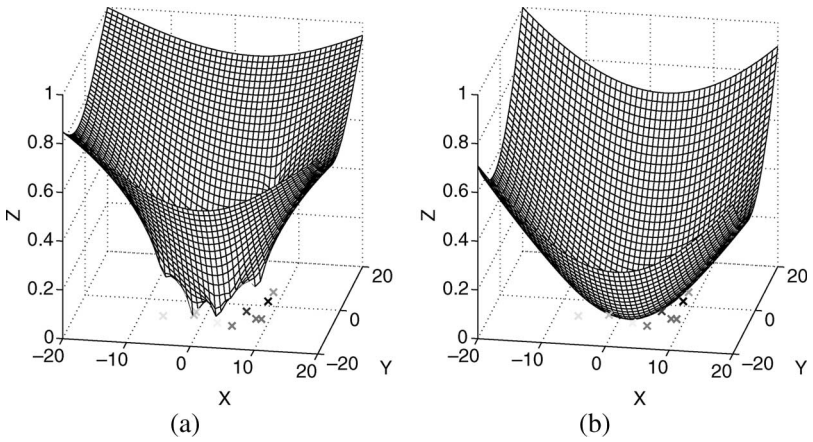


Fig. 14. Samples of the function $f(\vec{\mathbf{x}})$ given in (44) for two different values of m. Here, $n = 10$, and ω_i's and $\vec{\mathbf{a}}_i$'s are the same for the two cases. (a) $m = 0.2$. (b) $m = 1.2$.

V. Conclusion

In this paper, we have focused on the VoD network design problem. Using the concepts and tools available in signal coding, an optimization problem has been developed, which was shown to minimize the communication cost in a VoD network. Moreover, weights were added to the cost function to implicitly control the storage cost. According to the 0–1 property of the original problem, the objective function included binary variables, which made it mathematically hard to work with. So, looking back at fuzzy clustering, the problem was transferred into the fuzzy domain. The transformation was carried out in a way that the fuzziness of the solution was guaranteed to be acceptably low. Then, a method was proposed to produce a locally optimal solution to the proposed objective function using an iterative three-stage algorithm. Benefiting from the fact that the proposed algorithm is unrepeatable, another algorithm was proposed to produce a set of potential solutions and then to heuristically pick a proper one of them. Then, defining three main scenarios, the application of the proposed algorithm was discussed. The first scenario investigated the hypothetical application of the proposed method in designing the whole network. The two other scenarios discussed adding a new node to the network and recalculating caching and assignment, both because of changes in the population density map. In all cases, the contributions of the proposed algorithm were discussed using both numerical measures and also visual representations. In addition, the result of the proposed algorithm in the first scenario was compared with that of MILP. It was shown that MILP traps in a local minimum, in which only one hypernode serves the whole network. It is worth to mention that the convergence of the proposed algorithm was empirically observed. However, a more general proof for the generalized Weiszfeld method proposed here is still needed.

Appendix I
More General Definition of the Distance-Based Cost Function

Assuming that the relationship between the cost of communication and the distance is as given in (39), an analysis similar to what is given in Section III-C shows that

$$\vec{\mathbf{n}}_i = \frac{\sum\limits_{\vec{\mathbf{x}}\in\mathbf{X}} \psi_{\vec{\mathbf{x}}i} C'\left(\|\vec{\mathbf{n}}_i - \vec{\mathbf{x}}\|^2\right)\vec{\mathbf{x}}}{\sum\limits_{\vec{\mathbf{x}}\in\mathbf{X}} \psi_{\vec{\mathbf{x}}i} C'\left(\|\vec{\mathbf{n}}_i - \vec{\mathbf{x}}\|^2\right)}. \tag{42}$$

Using the fixed-point method and the initialization given in (37), we conjecture that for some functions, the iteration

$$\vec{\mathbf{n}}_i^{\,t+1} = \frac{\sum\limits_{\vec{\mathbf{x}}\in\mathbf{X}} \psi_{\vec{\mathbf{x}}i} C'\left(\|\vec{\mathbf{n}}_i^{\,t} - \vec{\mathbf{x}}\|^2\right)\vec{\mathbf{x}}}{\sum\limits_{\vec{\mathbf{x}}\in\mathbf{X}} \psi_{\vec{\mathbf{x}}i} C'\left(\|\vec{\mathbf{n}}_i^{\,t} - \vec{\mathbf{x}}\|^2\right)} \tag{43}$$

converges. Until now, there is proof for the cases of $C(x) = x$ and $C(x) = \sqrt{x}$, and we have empirical proof for $C(x) = x^{m_d/2}$, $m_d \geq 1$. The general case is an open problem.

Appendix II
On the Importance of $m_d \geq 1$

Theorem: Assume that the integer $n \geq 1$ and the positive values of $\omega_1, \ldots, \omega_n$ are given. Moreover, assume that n vectors $\vec{\mathbf{a}}_1, \ldots, \vec{\mathbf{a}}_n$ in $\mathbb{R}^2$ are given (here we restrict the discussion to $\mathbb{R}^2$, but a similar argument is valid for $\mathbb{R}^k$, $k \geq 1$). The function f is defined as

$$f(\vec{\mathbf{x}}) = \sum_{i=1}^{n} \omega_i \|\vec{\mathbf{x}} - \vec{\mathbf{a}}_i\|^m, \quad \vec{\mathbf{x}} \in \mathbb{R}^2. \tag{44}$$

If $m \geq 1$, then f has one and only one local minimum, which is also its global minimum. For the case of $m < 1$, we can provide examples in which f has many local minimums.

Proof: We first prove that moving along any line in $\mathbb{R}^2$, the values of the function $h(\vec{\mathbf{x}}) = \|\vec{\mathbf{x}}\|^m$, for $m > 1$, constitute a convex function. To prove this claim, assume that we are moving along the line $\ell : \{\vec{\mathbf{a}} + \lambda\vec{\mathbf{v}}\}, \vec{\mathbf{a}} \perp \vec{\mathbf{v}}, \|\vec{\mathbf{v}}\| = 1$. Defining

$g(\lambda) = h(\vec{\mathbf{a}} + \lambda\vec{\mathbf{v}})$, we have $g(\lambda) = (\|\vec{\mathbf{a}}\|^2 + \lambda^2)^{m/2}$, which yields

$$g'(\lambda) = m\left(\|\vec{\mathbf{a}}\|^2 + \lambda^2\right)^{\frac{m-1}{2}} \sqrt{1 - \frac{\|\vec{\mathbf{a}}\|^2}{\|\vec{\mathbf{a}}\|^2 + \lambda^2}}\,\mathrm{sgn}(\lambda) \quad (45)$$

which is an increasing function. Here, $\mathrm{sgn}(\lambda)$ is the sign function. Hence, g is convex, and so will be the function $h(\vec{\mathbf{x}}) = \|\vec{\mathbf{x}} - \vec{\mathbf{a}}\|^m$ for $m > 1$ and constant $\vec{\mathbf{a}} \in \mathbb{R}^2$.

As ω_i's are positive, the intersections of the function f with straight lines also give convex functions. Note that the condition $m > 1$ is vital in (45). Moreover, according to (45), the derivative accepts both negative and positive values. Hence, there exists a point in which it gets zero.

Fig. 14 shows two samples of the function $f(\vec{\mathbf{x}})$ for two different values of m. Here, $n = 10$, and ω_i's and $\vec{\mathbf{a}}_i$'s are the same for the two cases. Fig. 14(a) shows the case when $m = 0.2$. Here, we can see numerous local minimums. In contrary, Fig. 14(b) shows that when $m = 1.2$, there exists one and only one global minimum.

ACKNOWLEDGMENT

The authors would like to thank J. Rohne of TRLabs for comments on the practical scenarios under which the proposed algorithm can be used; the NEOS team, especially J. Sarich; A. Yadollahi for her encouragement and valuable discussions; and the respected anonymous referees for their constructive suggestions.

REFERENCES

[1] H. Ma and K. G. Shin, "Multicast video-on-demand services," *Comput. Commun. Rev.*, vol. 32, no. 1, pp. 31–43, Jan. 2002.
[2] T. D. Little and D. Venekatesh, "Prospects for interactive video-on-demand," *IEEE Multimedia*, vol. 1, no. 3, pp. 14–24, 1994.
[3] J.-P. Nussbaumer, B. V. Patel, F. Schaffa, and J. P. G. Sterbenz, "Networking requirements for interactive video on demand," *IEEE J. Sel. Areas Commun.*, vol. 13, no. 5, pp. 779–787, Jun. 1995.
[4] T. S. Perry, "The trials and travails of interactive TV," *IEEE Spectr.*, vol. 33, no. 4, pp. 22–28, Apr. 1996.
[5] C. Vassilakis, M. Paterakis, and P. Triantafillou, "Video placement and configuration of distributed video servers on cable TV networks," *Multimedia Syst.*, vol. 8, no. 2, pp. 92–104, Mar. 2000.
[6] J. Segarra and V. Cholvi, "Distribution of video-on-demand in residential networks," in *Proc. 8th Int. Workshop Interactive Distrib. Multimedia Syst.*, 2001, pp. 50–61.
[7] R. L. Francis, T. J. Lowe, and A. Tamir, "Demand point aggregation for location models," in *Facility Location, Applications and Theory*, Z. Drezner and H. W. Hamacher, Eds. Berlin, Germany: Springer-Verlag, 2002, pp. 179–205.
[8] K. Park, K. Lee, S. Park, and H. Lee, "Telecommunication node clustering with node compatibility and network survivability requirements," *Manage. Sci.*, vol. 46, no. 3, pp. 363–374, Mar. 2000.
[9] K. H. Muralidhar and M. K. Sundareshan, "On the decomposition of large communication networks for hierarchical control implementation," *IEEE Trans. Commun.*, vol. COM-34, no. 10, pp. 985–987, Oct. 1986.
[10] Y. G. Saab, "A fast and robust network bisection algorithm," *IEEE Trans. Comput.*, vol. 44, no. 7, pp. 903–913, Jul. 1995.
[11] M. Maravalle, B. Simeone, and R. Naldini, "Clustering on trees," *Comput. Stat. Data Anal.*, vol. 24, no. 2, pp. 217–234, Apr. 1997.
[12] V. K. Balakrishnan and C. Moire, *Network Optimization*, 5th ed. ser. Mathematics. London, U.K.: Chapman & Hall, 1995.
[13] J. A. M. Prez, J. M. M. Vega, and J. L. Verdegay, "Fuzzy location problems on networks," *Fuzzy Sets Syst.*, vol. 142, no. 3, pp. 393–405, Mar. 2004.
[14] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, "Network optimization," in *Handbook of Applied Optimization*, P. M. Paradalos and M. G. C. Resende, Eds. New York: Oxford Univ. Press, 2002, pp. 352–363.
[15] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1993.
[16] F. Glover and D. Klingman, "Network applications in industry and government," *AIEE Trans.*, vol. 9, pp. 363–376, 1976.
[17] P. Chardaire and A. Lisser, "Minimum-cost multicommodity flow," in *Handbook of Applied Optimization*, P. M. Paradalos and M. G. C. Resende, Eds. New York: Oxford Univ. Press, 2002, pp. 404–422.
[18] M. Gerla and L. Kleinrock, "On the topological design of distributed computer networks," *IEEE Trans. Commun.*, vol. COM-25, no. 1, pp. 48–60, Jan. 1977.
[19] G. Anandalingam, "Optimization of telecommunications networks," in *Handbook of Applied Optimization*, P. M. Paradalos and M. G. C. Resende, Eds. New York: Oxford Univ. Press, 2002.
[20] E. Gourdin, M. Labbe, and H. Yaman, "Telecommunication and location," in *Facility Location, Applications and Theory*, Z. Drezner and H. W. Hamacher, Eds. Berlin, Germany: Springer-Verlag, 2002, pp. 275–305.
[21] Y. Chen, L. Qiu, W. Chen, L. Nguyen, and R. Katz, "Efficient and adaptive Web replication using content clustering," *IEEE J. Sel. Areas Commun.*, vol. 21, no. 6, pp. 979–994, Aug. 2003.
[22] M. Yang and Z. Fei, "A model for replica placement in content distribution networks for multimedia applications," in *Proc. IEEE Int. Conf. Commun.*, Anchorage, AK, 2003, pp. 557–561.
[23] J. Dilley, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman, and B. Weihl, "Globally distributed content delivery," *IEEE Internet Comput.*, vol. 6, no. 5, pp. 50–58, Sep./Oct. 2002.
[24] A. Dan, D. Sitaram, and P. Shahabuddin, "Scheduling polices for an on-demand video server with batching," in *Proc. Multimedia*, San Francisco, CA, 1994, pp. 15–23.
[25] A. Vakali and G. Pallis, "Content delivery networks: Status and trends," *IEEE Internet Comput.*, vol. 7, no. 6, pp. 68–74, Nov./Dec. 2003.
[26] E. Cela, "Assignment problems," in *Handbook of Applied Optimization*, P. M. Paradalos and M. G. C. Resende, Eds. New York: Oxford Univ. Press, 2002.
[27] R. K. Ahuja, T. L. Magnanti, J. B. Orlin, and M. R. Reddy, "Applications of network optimization," in *Network Models—Handbooks of Operations Research*, vol. 7, M. O. Ball, T. L. Magnanti, C. L. Manoma, and G. L. Nemhauser, Eds. Amsterdam, The Netherlands: Elsevier, 1995, pp. 1–83.
[28] R. E. Burkard, E. Cela, P. Pardalos, and L. Pitsoulis, "The quadratic assignment problem," in *Handbook of Combinatorial Optimization*, vol. 3, P. Pardalos and D.-Z. Du, Eds. Norwell, MA: Kluwer, 1998, pp. 241–338.
[29] F. Randl, "The quadratic assignment problem," in *Facility Location, Applications and Theory*, Z. Drezner and H. W. Hamacher, Eds. Berlin, Germany: Springer-Verlag, 2002, pp. 275–305.
[30] B. Fleischmann and J. Paraschis, "Solving a large scale districting problem: A case report," *Comput. Oper. Res.*, vol. 15, no. 6, pp. 521–533, Nov. 1988.
[31] F. Pereira, J. Figueira, V. Mousseau, and B. Roy, "Multiple criteria districting problems: The public transportation network pricing system of the Paris region," *Ann. Oper. Res.*, vol. 154, no. 1, pp. 69–92, Oct. 2007.
[32] R. Johnston, "The 2000 Annual Political Geography Lecture, Manipulating maps and winning elections: Measuring the impact of malapportionment and gerrymandering," *Polit. Geogr.*, vol. 21, no. 1, pp. 1–31, Jan. 2002.
[33] F. Bacao, V. Lobo, and M. Painho, "Applying genetic algorithms to zone design," *Soft Comput.*, vol. 9, no. 5, pp. 341–348, May 2005.
[34] B. A. Norman, A. E. Smith, E. Yildirim, and W. Tharmmaphornphilas, "An evolutionary approach to incorporating intradepartmental flow into facilities design," *Adv. Eng. Softw.*, vol. 32, no. 6, pp. 443–453, Jun. 2001.
[35] T. Yanga, M. Rajasekharanb, and B. A. Petersc, "Semiconductor fabrication facility design using a hybrid search methodology," *Comput. Ind. Eng.*, vol. 36, no. 3, pp. 565–583, Jul. 1999.
[36] S. D'Amico, S. J. Wang, R. Batta, and C. Rump, "A simulated annealing approach to police district design," *Comput. Oper. Res.*, vol. 29, no. 6, pp. 667–684, May 2002.
[37] Z. Drezner, K. Klamroth, A. Schobel, and G. O. Wesolowsky, "The Weber problem," in *Facility Location, Applications and Theory*, Z. Drezner

and H. W. Hamacher, Eds. Berlin, Germany: Springer-Verlag, 2002, pp. 179–205.
[38] R. L. Francis, F. Leon, and J. A. White, *Facility Layout and Location, an Analytical Approach*. Englewood Cliffs, NJ: Prentice-Hall, 1992.
[39] H. W. Kuhn, "On a pair of dual nonlinear programs," in *Methods of Nonlinear Programming*, J. Abadie, Ed. Amsterdam, The Netherlands: North-Holland, 1967, pp. 38–54.
[40] H. W. Kuhn, "A note on Fermat's problem," *Math. Program.*, vol. 4, no. 1, pp. 98–107, Dec. 1973.
[41] T. Simpson, *The Doctrine and Application of Fluxions*. London, U.K.: John Nourse, 1750.
[42] A. Gersho and R. Gray, *Vector Quantization and Signal Compression*. Boston, MA: Kluwer, 1992.
[43] A. Jain, M. Murty, and P. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, Sep. 1999.
[44] L. Cooper, "Location-allocation problems," *Oper. Res.*, vol. 11, pp. 331–343, 1963.
[45] N. Megiddo and K. J. Supowit, "On the complexity of some common geometric location problems," *SIAM J. Comput.*, vol. 13, no. 1, pp. 182–196, Feb. 1984.
[46] D. Eilon, D. T. Watson-Gandy, and N. Christofides, *Distribution Management*. New York: Hanfer, 1971.
[47] R. E. Kuenne and R. M. Soland, "Exact and approximate solutions to the multisource Weber problem," *Math. Program.*, vol. 3, no. 1, pp. 193–209, Dec. 1972.
[48] L. Cooper, "Heuristic methods for location-allocation problems," *SIAM Rev.*, vol. 6, no. 1, pp. 37–53, Jan. 1964.
[49] R. F. Love and J. G. Morris, "A computation procedure for the exact solution of location-allocation problems with rectangular distances," *Nav. Res. Logist. Q.*, vol. 22, no. 3, pp. 441–453, Sep. 1975.
[50] Z. Rezner, "Location," in *Handbook of Applied Optimization*, P. M. Paradalos and M. G. C. Resende, Eds. New York: Oxford Univ. Press, 2002, pp. 632–640.
[51] E. Ruspini, "A new approach to clustering," *Inf. Control*, vol. 15, no. 1, pp. 22–32, Jul. 1969.
[52] R. Duda and P. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.
[53] A. K. Jain, *Fundamentals of Digital Image Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1989.
[54] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probability*, Berkeley, CA, 1967, pp. 281–297.
[55] A. Jain and R. Dubes, *Algorithms for Clustering*. Englewood Cliffs, NJ: Prentice-Hall, 1998.
[56] H. Sofyan, "Fuzzy clustering," in *Statistical Case Studies*, W. Hardle, Y. Mori, and P. Vieu, Eds., 2004, pp. 131–142.
[57] J. Dunn, "A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters," *J. Cybern.*, vol. 3, no. 3, pp. 32–57, 1973.
[58] J. C. Bezdek, *Pattern Recognition With Fuzzy Objective Function Algorithms*. New York: Plenum, 1981.
[59] M. Trivedi and J. Bezdek, "Low-level segmentation of aerial images with fuzzy clustering," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-16, no. 4, pp. 589–598, Jul. 1986.
[60] D. E. Gustafson and W. C. Kessel, "Fuzzy clustering with a fuzzy covariance matrix," in *Proc. IEEE CDC*, San Diego, CA, 1979, vol. 2, pp. 761–766.
[61] I. Gath and A. Geva, "Unsupervised optimal fuzzy clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 7, pp. 773–781, Jul. 1989.
[62] J. M. Leski, "Fuzzy c-varieties/elliptotypes clustering in reproducing kernel Hilbert space," *Fuzzy Sets Syst.*, vol. 141, no. 2, pp. 259–280, Jan. 2004.
[63] R. Krishnapuram and J. Kim, "Clustering algorithms based on volume criteria," *IEEE Trans. Fuzzy Syst.*, vol. 8, no. 2, pp. 228–236, Apr. 1995.
[64] T. Cheng, D. Goldgof, and L. Hall, "Fast fuzzy clustering," *Fuzzy Sets Syst.*, vol. 93, no. 1, pp. 49–56, Jan. 1993.
[65] N. Iyer and A. Kandel, "Feature-based fuzzy classification for interpretation of mammograms," *Fuzzy Sets Syst.*, vol. 114, no. 2, pp. 271–280, Sep. 2000.
[66] E. Tsao, J. Bezdek, and N. Pal, "Fuzzy Kohonen clustering networks," *Pattern Recognit.*, vol. 27, no. 5, pp. 757–764, May 1994.
[67] A. M. Massone, F. Masulli, and A. Petrosini, "Fuzzy clustering algorithms on Landsat images for detection of waste areas: A comparison," in *Advances in Fuzzy Systems and Intelligent Technologies*. Maastricht, The Netherlands: Shaker, 2000, pp. 165–175.
[68] J. M. Leski, "Generalized weighted conditional fuzzy clustering," *IEEE Trans. Fuzzy Syst.*, vol. 11, no. 6, pp. 709–715, Dec. 2003.
[69] R. Krishnapuram and J. Keller, "A possibilistic approach to clustering," *IEEE Trans. Fuzzy Syst.*, vol. 1, no. 2, pp. 98–110, May 1993.
[70] E. Weiszfeld, "Sur le point pour lequel la somme des distances de n points donnes est minimum," *Tohoku Math J.*, vol. 43, pp. 355–386, 1936.
[71] W. Miehle, "Link-length minimization in networks," *Oper. Res.*, vol. 6, no. 2, pp. 232–243, Mar./Apr. 1958.
[72] J. B. Rosen and G. L. Xue, "On the convergence of Miehle's algorithm for the Euclidean multifacility location problem," *Oper. Res.*, vol. 40, no. 1, pp. 188–191, Jan./Feb. 1992.
[73] H. W. Juhn and R. E. Kuenne, "An efficient algorithm for the numerical solution of the generalized Weber problem in spatial economics," *J. Reg. Sci.*, vol. 4, no. 2, pp. 21–33, Dec. 1962.
[74] C.-T. Chen, "A fuzzy approach to select the location of the distribution center," *Fuzzy Sets Syst.*, vol. 118, no. 1, pp. 65–73, Feb. 2001.
[75] U. Bhattacharya, J. R. Rao, and R. N. Tiwari, "Fuzzy multi-criteria facility location problem," *Fuzzy Sets Syst.*, vol. 51, no. 3, pp. 277–287, Nov. 1992.
[76] R. I. John and S. C. Bennett, "The use of fuzzy sets for resource allocation in an advance request vehicle brokerage system—A case study," *J. Oper. Res. Soc.*, vol. 48, no. 2, pp. 117–123, Feb. 1997.
[77] C. Kahraman, D. Ruan, and I. DoImagean, "Fuzzy group decision-making for facility location selection," *Inf. Sci.*, vol. 157, no. 1/2, pp. 135–153, Dec. 2003.
[78] C. Araz, H. Selim, and I. Ozkarahan, "A fuzzy multi-objective covering-based vehicle location model for emergency services," *Comput. Oper. Res.*, vol. 34, no. 3, pp. 705–726, 2007.
[79] R. J. Kuo, S. C. Chi, and S. S. Kao, "A decision support system for selecting convenience store location through integration of fuzzy ahp and artificial neural network," *Comput. Ind.*, vol. 47, no. 2, pp. 199–214, Feb. 2002.
[80] H.-J. Zimmerman, *Fuzzy Set Theory and Its Applications*, 4th ed. Berlin, Germany: Springer-Verlag, 2001.
[81] B. Bozkaya, J. Zhang, and E. Erkut, "An efficient genetic algorithm for the p-median problem," in *Facility Location, Applications and Theory*, Z. Drezner and H. W. Hamacher, Eds. Berlin, Germany: Springer-Verlag, 2002, pp. 179–205.
[82] J. Brimberg, P. Hansen, N. Mladenovic, and E. D. Taillard, "Improvement and comparison of heuristics for solving the uncapacitated multisource Weber problem," *Oper. Res.*, vol. 48, no. 3, pp. 444–460, May 2000.
[83] F. Altipatmak, B. Dengiz, and A. E. Smith, "Optimal design of reliable computer networks: A comparison of metaheuristics," *J. Heuristics*, vol. 9, no. 6, pp. 471–487, Dec. 2003.
[84] J. R. Boucher, *Voice Teletraffic Systems Engineering*. Norwood, MA: Artech House, 1988.
[85] K. Tutschku and P. Tran-Gia, "Spatial traffic estimation and characterization for mobile communication network design," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 5, pp. 804–811, Jun. 1998.
[86] K. Almeroth, A. Dan, D. Sitaram, and W. Tetzlaff, "Long term resource allocation in video delivery systems," in *Proc. 16th Annu. Joint Conf. IEEE Comput. Commun. Soc.*, 1997, pp. 1333–1340.
[87] C. Griwodz, M. Bär, and L. C. Wolf, "Long-term movie popularity models in video-on-demand systems: Or the life of an on-demand movie," in *Proc. 5th ACM Int. Conf. Multimedia*, Seattle, WA, 1997, pp. 349–357.
[88] J. Czyzyk, M. Mesnier, and J. Mor, "The NEOS server," *IEEE Comput. Sci. Eng.*, vol. 5, no. 3, pp. 68–75, Jul.–Sep. 1998.
[89] W. Gropp and J. Mor, "Optimization environments and the NEOS server," in *Approximation Theory and Optimization*, M. D. Buhmann and A. Iserles, Eds. Cambridge, U.K.: Cambridge Univ. Press, 1997, pp. 167–182.
[90] E. Dolan, "The NEOS Server 4.0 Administrative Guide," Math. and Comput. Sci. Division, Argonne Nat. Lab., Argonne, IL, Technical Memorandum ANL/MCS-TM-250, 2001.
[91] BDMLP. [Online]. Available: http://www.gams.com/solvers/bdmlp/main.htm
[92] T. Cundari, C. Sarbu, and H. F. Pop, "Robust fuzzy principal component analysis (FPCA). A comparative study concerning interaction of carbon–hydrogen bonds with molybdenum–oxo bonds," *J. Chem. Inf. Comput. Sci.*, vol. 42, no. 6, pp. 1363–1369, 2002.
[93] C. Sarbu and H. Pop, "Principal component analysis versus fuzzy principal component analysis. A case study: The quality of Danube water (1985–1996)," *Talanta*, vol. 65, no. 5, pp. 1215–1220, Mar. 2005.

[94] J. Bezdek, J. Keller, R. Krisnapuram, and N. R. Pal, *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*. Boston, MA: Kluwer, 1999.
[95] J. V. Greenman, "Introduction to optimisation theory and the calculus of variations," in *Mathematical Topics in Telecommunications*, vol. 1, K. W. Cattermole and J. O'Reilly, Eds. New York: Wiley, 1984, pp. 1–36.
[96] F. Plastria, "Continuous covering location problems," in *Facility Location, Applications and Theory*, Z. Drezner and H. W. Hamacher, Eds. Berlin, Germany: Springer-Verlag, 2002, pp. 275–305.
[97] A. Hyvarinen, J. Karhunen, and E. Oja, *Independent Component Analysis*. Hoboken, NJ: Wiley, 2001.
[98] F. Rado, "The Euclidean multifactory location problem," *Oper. Res.*, vol. 36, no. 3, pp. 485–492, 1988.
[99] Z. Drezner, "A note on accelerating the Weiszfeld procedure," *Location Sci.*, vol. 3, no. 4, pp. 275–279, Dec. 1995.
[100] L. Cooper, "Solutions of generalized locational equilibrium models," *J. Reg. Sci.*, vol. 7, no. 1, pp. 1–18, Jun. 1967.
[101] S. Krau, "Extensions du probleme de Weber," Ph.D. dissertation, Ecole Polytechnique de Montréal, Montréal, QC, Canada, 1997.
[102] D.-W. Kim, K. H. Lee, and D. Lee, "A novel initialization scheme for the fuzzy c-means algorithm for color clustering," *Pattern Recognit. Lett.*, vol. 25, no. 2, pp. 227–237, Jan. 2004.

Arash Abadpour was born in Tehran, Iran, in 1979. He received the B.Sc. degree and the M.Sc. degree in scientific computation and computer sciences from the Sharif University of Technology, Tehran, in 2003 and 2005, respectively. He is currently working toward the Ph.D. degree in the Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, MB, Canada.

He currently works on the QoS-constrained information-theoretic capacity of CDMA systems in the University of Manitoba. He is also a Research Assistant with Telecommunications Research Labs (TRLabs), Winnipeg. His research interests are in optimization, with emphasis on stochastic systems and models and pattern recognition.

Attahiru Sule Alfa (M'01–A'02–M'05) received the B.Eng. degree from the Ahmadu Bello University, Zaria, Nigeria, the M.Sc. degree from the University of Manitoba, Winnipeg, MB, Canada, and the Ph.D. degree from the University of New South Wales, Sydney, Australia.

He is currently the Natural Sciences and Engineering Research Council of Canada (NSERC) Industrial Research Chair of Telecommunications and a Professor with the Department of Electrical and Computer Engineering, University of Manitoba. He has published in several journals, including Stochastic Models, Queueing Systems: Theory and Applications, Naval Research Logistics, Performance Evaluation, Journal of Applied Probability, Advances in Applied Probability, Mathematics of Computations, and Numerische Mathematik. He carries out research in queuing and network theories, with applications mostly to telecommunication systems. He has also applied these theories to manufacturing and transportation and traffic systems in the past. His current research interests include wireless communication networks, mobility, Internet traffic, stochastic models, performance analysis, network restoration, and teletraffic forecasting models. He has contributed significantly in matrix-analytic methods for stochastic models.

Dr. Alfa is a member of the Association of Professional Engineers and Geoscientists of the Province of Manitoba and the Institute for Operations Research and the Management Sciences. His papers were also published in the *IEEE Journal of Selected Areas in Communications*, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE TRANSACTIONS ON WIRELESS, IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE TRANSACTIONS ON COMMUNICATIONS, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS.

Jeff Diamond received the B.Sc. and M.Sc. degrees in physics, the M.Sc. degree in management science, and the Ph.D. degree in operations research from the University of Manitoba, Winnipeg, MB, Canada.

Since 1996, he has been a Research Scientist and a Research and Development Manager with Telecommunication Research Laboratories, Winnipeg. His research interests are in the areas of communication network modeling and optimization and in secure multiparty computation.