Fuzzy Design of A Video–on–Demand Network

Arash Abadpour[1,2], Attahiru Sule Alfa[1,2], and Jeff Diamond[2]
[1] Dept. of Elec. and Comp. Eng., Univ. of Manitoba, {abadpour,alfa}@ee.umanitoba.ca
[2] TRLabs, Winnipeg, Canada, diamond@win.trlabs.ca

***Abstract*— Designing a Video–on–Demand (VoD) system is in essence an optimization task aimed at minimizing the cost of communication and storage in the corresponding network. The decision variables of this problem are the locations of the nodes plus the content which should be cached in each node. Furthermore, an assignment strategy is needed to determine, for each customer, which node should be contacted for each video file. 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, inspired by the success of fuzzy optimization for similar problems in coding, a fuzzy objective function is derived which is heuristically shown to minimize the communication cost in a VoD network, while controlling the storage cost as well. Then, an iterative algorithm is proposed to find an optimum solution to the proposed problem. After addressing the mathematical details of the proposed method, a sample problem is presented followed by the solution produced for it by the proposed method. This solution is then extensively analyzed.**

I. Introduction

A Video–on–Demand (VoD) system is a dominantly one–way network, connecting a VoD service–provider to customers [1]. According to the tight bandwidth and latency constraints of a VoD system, rigorous analysis is necessary before its deployment [2]. Looking at the available literature, the common trend is to consider a tree structure for VoD networks [3], partly because the flow in the VoD network is one–directional. Also, the contents of the network, namely video files, are added very gradually and are not ever modified. The tree–structured network enables the designer to focus on portions of it and ignore the rest, at different stages, also known as aggregation [4].

In this paper, we formulate the VoD design problem as determining the location of a known number of nodes which should serve a given population with a given library. In addition to the nodes' locations, their content needs to be determined. Furthermore, for each customer and each video file, the solution should determine which node should be contacted. These three form the layers of the decision variables of an optimization problem in which the aggregate cost of communication and storage is minimized. A part of this problem is similar to location problems.

Many location problems are very well modeled using the Weber problem [5]. The one–node Weber Problem looks for a point which has minimum weighted sum of distances to a set of given points. The multiple–node version considers more than one node and incorporates an assignment [5]. Weber problems are known to be hard to solve problems [6] which have multiple solutions [7]. Rather than the limited attempts to give an exact solution to the Weber problem [8], the general approach for solving them is through a heuristic algorithm called p–median [9]. The idea of p–median is to select groups of customers and then to find the best location of a node which can serve each set. Then, for each node, its territory is calculated, and this iteration is repeated until the calculation converges. In data clustering and coding, this algorithm is called Hard C–Means (HCM) [10] and Lloyd–Max.

It is observed in different frameworks that the zero–one assignment in the classical Weber problem makes it very prone to falling into local minimum [11]. Hence, after the introduction of Fuzzy Sets, many researchers tried to apply this theory to different versions of the Weber Problem. Therefore, HCM was generalized into Fuzzy C–means (FCM) after the proposal of the fuzziness concept [12].

The VoD problem is a multi–layered Weber problem. Thus, in this paper, we use the fuzzy clustering style of writing the cost function for a Weber problem. Here, we also generalize the Weiszfeld method [13] to be able to work with powers of the Euclidean distance.

II. Proposed Method

Here, we work on a system which serves N customers, given as the set $\mathbf{X}$. Assuming that the system is in a steady state, each customer is assigned a weight, $\mu_{\vec{\mathbf{x}}} > 0$, which shows the relative utilization of that user [14]. Also, we assume that the library includes l video files. Several research projects have shown that the distribution of demands for different video files is well–approximated by the Zipf model [15], where the demand frequency for the j–th video file is given by,

$$d_j = cj^{-\alpha}. \qquad (1)$$

Here, c is a normalization constant and a reasonable approximation for α is given to be 0.729 [15].

The contents of the library will be cached in n nodes, with $\vec{\mathbf{n}}_i$ indicating the location of the i–th node. As the VoD service highly demands a one–by–one connection [1], [2], using the definition of $\mu_{\vec{\mathbf{x}}}$, we denote the total resources of the i–th node as l_i. If for example l_i is five, it means that the respective node can serve a set, $\tilde{\mathbf{X}}$, of customers which sum to five ($\sum_{\vec{\mathbf{x}}\in\tilde{\mathbf{X}}} \mu_{\vec{\mathbf{x}}} = 5$), simultaneously. Demands made from the i–th node, can potentially be for any of the l objects. Here, we assume that for any object j, the i–th node allocates a fixed amount of its resources to it, which we denote by L_{ij}, for which,

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

This portion can be zero if that object is not cached in that particular node. Note that neither l_is nor L_{ij}s are known prior to solving the problem.

While a node should manage its bandwidth, it should also deliberately decide which objects it will be caching, namely, for the i–th node, how many L_{ij}s are non–zero. 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 (for example see [2]). That approach leads to an objective function which includes two terms, making further derivations harder. In this paper, we propose another method.

Imposing no limitation on the number of non–zero 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}. For this goal, we define the value of L, as the minimum demand for an object at a node which justifies caching it there. Doing so, we add concern over optimality of storage to the optimization problem, through implicit comparison of $d_j L_{ij}$ with L to decide if the j–th object should be cached in the i–th node or not.

Assume that the customer $\vec{\mathbf{x}}$ demands the j–th object which is then supplied from the i–th node. The net cost of this transaction is modeled as $C_{\vec{\mathbf{x}}i}$, and is assumed to be independent of the object which is transmitted. Among other models, we consider the case in which,

$$C_{\vec{\mathbf{x}}i} = C \left\| \vec{\mathbf{n}}_i - \vec{\mathbf{x}} \right\|^{m_d}, m_d \geq 1. \tag{3}$$

Here C is a constant which 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. Here, we define $p_{\vec{\mathbf{x}}ij} \in \{0, 1\}$, where, $p_{\vec{\mathbf{x}}ij}$ is one iff $\vec{\mathbf{x}}$ asks node i for the j–th object. Thus,

$$\sum_{i=1}^{n} p_{\vec{\mathbf{x}}ij} = 1, \forall \vec{\mathbf{x}}, j, \tag{4}$$

and the expected cost of providing one object, any object, to this customer equals,

$$\Delta_{\vec{\mathbf{x}}} = \sum_{j=1}^{l} d_j \sum_{i=1}^{n} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij}. \tag{5}$$

Including weights of the customers, the expected cost of serving one demand from any customer equals,

$$\Delta = \frac{1}{\mu} \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}} \sum_{j=1}^{l} d_j \sum_{i=1}^{n} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij}, \tag{6}$$

where,

$$\mu = \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}}. \tag{7}$$

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

$$D_{ij} = \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}} p_{\vec{\mathbf{x}}ij}, \tag{8}$$

which, as in a stable network allocation and demand are identical, yields,

$$L_{ij} = D_{ij}, \forall i, j, . \tag{9}$$

and,

$$\sum_{i=1}^{n} L_{ij} = \sum_{i=1}^{n} D_{ij} = \mu, \forall j. \tag{10}$$

At this stage, the optimization problem is defined as minimizing (6), given (4),

$$L_{ij} = \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}} p_{\vec{\mathbf{x}}ij}, \tag{11}$$

and

$$(L_{ij} = 0) \vee (d_j L_{ij} \geq L), \forall i, j. \tag{12}$$

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 in total cached in the entire system. To measure this phenomenon, we define the value of c as the number of L_{ij}s which are nonzero. As 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,

$$\rho = \frac{c}{nl}. \tag{13}$$

Thus, while we mainly focus on minimizing Δ, we are also simultaneously looking for smaller values of ρ.

The optimization problem given here is analytically challenging, because it includes the binary decision variables of $p_{\vec{\mathbf{x}}ij}$s. The two available choices for the L_{ij}s also add to the difficulty of analytically tackling this problem. Here, we give an approximate form of this optimization problem, which resembles fuzzy clustering problems, and is analytically tractable. First, we relax (12) by adding L_{ij}s as controlling weights to the objective function which will push toward’s satisfaction of an approximate form of (9) defined as,

$$L_{ij} \simeq D_{ij}, \forall i, j. \tag{14}$$

Also, to control allocation for objects, we add (10) as a constraint. Furthermore, we translate the objective function into a fuzzy representation.

Looking at (6), we write the cost of serving the j–th object to all customers as,

$$\Delta_j = \frac{1}{\mu} \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}} \sum_{i=1}^{n} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij}. \tag{15}$$

Assuming $m_d = 2$ and $\mu_{\vec{\mathbf{x}}} \equiv 1$, this is a Weber problem, also known as HCM. It is shown that the fuzzy version of this problem, FCM, acts more robustly. Thus, we rewrite Δ_j using the weighted version of FCM [16]. To do so, $p_{\vec{\mathbf{x}}ij} \in \{0,1\}$ is replaced with $p_{\vec{\mathbf{x}}ij} \in [0,1]$ and Δ_j is rewritten as,

$$\hat{\Delta}_j = \frac{1}{\mu} \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}} \sum_{i=1}^{n} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij}^{m}, \tag{16}$$

for m close to one. Thus, we rewrite (6) as,

$$\hat{\Delta} = \frac{1}{\mu} \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}} \sum_{j=1}^{l} d_j \sum_{i=1}^{n} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij}^{m}. \tag{17}$$

where (4) is still in place and $p_{\vec{\mathbf{x}}ij} \in [0,1]$. Here, $m > 1$ is the fuzziness [12]. It is known that, generally, as m tends to 1^+, the fuzziness of $p_{\vec{\mathbf{x}}ij}$s reduces [17]. In this way, we will have

$$p_{\vec{\mathbf{x}}ij} \in [0,\delta] \cup [1-\delta, 1], \tag{18}$$

for a small δ.

As implied by the condition in (12), $L^{-1} d_j L_{ij}$ is expected to be either zero or bigger than one. In the first case ($L_{ij} = 0$), no customer is expected to rely on the i–th node for the j–th object. This is a two–choice constraint which makes the objective function discontinuous and hence hard to work with. Also, it adds an extra constraint to the optimization problem and makes the application of FCM–style optimization hard. To get around these problems, we define the term,

$$\varphi_{ij} = 1 + \left(\frac{d_j L_{ij}}{L} \right)^{-k}. \tag{19}$$

Here, k is a fixed power, with default values over 10. Now, as L_{ij} tends to zero, φ_{ij} approaches infinity. On the other hand, for $d_j L_{ij} \geq L$, φ_{ij} tends to one. We use this term to add a force to the objective function to carry out three goals. Namely, it helps satisfy (14), it forces small $d_j L_{ij}$s to become zero, and it becomes transparent when the solution converges.

Using φ_{ij}s, we rewrite (17) as,

$$\hat{\Delta} = \frac{1}{\mu} \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}} \sum_{j=1}^{l} d_j \sum_{i=1}^{n} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij}^{m} \varphi_{ij}. \tag{20}$$

Note that φ_{ij} is a decreasing function of L_{ij}. Thus, if L_{ij} becomes small, φ_{ij} tends to a very large value. Then, as $\hat{\Delta}$ is minimized, no $p_{\vec{\mathbf{x}}ij}$, for known i and j, will be other than zero, because otherwise they will have a big contribution to $\hat{\Delta}$. Thus, the introduction of φ_{ij} into (20) 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 which pushes (14) toward's satisfaction. 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 zero, it will result in other $L_{i'j}$s benefiting from (10) and growing larger. This will result in smaller $\varphi_{i'j}$s, eventually leading to a smaller $\hat{\Delta}$. Thus, the φ_{ij} weights are also beneficiary in pushing small L_{ij}s toward's zero. Finally, for active L_{ij}s, meaning those which are not zero, φ_{ij} is close to one. 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 (20) will eventually result in minimizing (6). To summarize, the VoD network design problem is approximated as minimizing,

$$\hat{\Delta} = \frac{1}{\mu} \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}} \sum_{j=1}^{l} d_j \sum_{i=1}^{n} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij}^{m} \varphi_{ij}, \tag{21}$$

given,

$$\sum_{i=1}^{n} p_{\vec{\mathbf{x}}ij} = 1, \forall \vec{\mathbf{x}}, j, \tag{22}$$

and,

$$\sum_{i=1}^{n} L_{ij} = \mu, \forall j. \tag{23}$$

Remembering the FCM methodology, we use Lagrange Multipliers and derive,

$$p_{\vec{\mathbf{x}}ij} = \frac{\left(C_{\vec{\mathbf{x}}i} \varphi_{ij}\right)^{-\frac{1}{m-1}}}{\sum_{i'=1}^{n} \left(C_{\vec{\mathbf{x}}i'} \varphi_{i'j}\right)^{-\frac{1}{m-1}}}. \tag{24}$$

$$L_{ij} = \mu \frac{\omega_{ij}^{\frac{1}{k+1}}}{\sum_{i'=1}^{n} \omega_{i'j}^{\frac{1}{k+1}}}, \tag{25}$$

where,

$$\omega_{ij} = \sum_{\vec{\mathbf{x}} \in \mathbf{X}} \mu_{\vec{\mathbf{x}}} C_{\vec{\mathbf{x}}i} p_{\vec{\mathbf{x}}ij}^{m}. \tag{26}$$

Furthermore,

$$\vec{\mathbf{n}}_i = \frac{\sum_{\vec{\mathbf{x}} \in \mathbf{X}} \psi_{\vec{\mathbf{x}}i} \left\| \vec{\mathbf{n}}_i - \vec{\mathbf{x}} \right\|^{m_d - 2} \vec{\mathbf{x}}}{\sum_{\vec{\mathbf{x}} \in \mathbf{X}} \psi_{\vec{\mathbf{x}}i} \left\| \vec{\mathbf{n}}_i - \vec{\mathbf{x}} \right\|^{m_d - 2}}. \tag{27}$$

Here,

$$\psi_{\vec{\mathbf{x}}i} = \mu_{\vec{\mathbf{x}}} \sum_{j=1}^{l} d_j \varphi_{ij} p_{\vec{\mathbf{x}}ij}^{m}. \tag{28}$$

While in the case of $m_d = 2$, (27) gives an explicit way for calculating $\vec{\mathbf{n}}_i$ as,

$$\vec{\mathbf{n}}_i = \frac{\sum_{\vec{\mathbf{x}} \in \mathbf{X}} \psi_{\vec{\mathbf{x}}i} \vec{\mathbf{x}}}{\sum_{\vec{\mathbf{x}} \in \mathbf{X}} \psi_{\vec{\mathbf{x}}i}}, \tag{29}$$

when m_d is not two, we need to utilize the generalized version of the Weiszfeld method [18]. Starting from the initial vectors given by (29) we use the iteration defined as (27) until the vectors converge. The details of these derivations are given in [19].

Here, we have not provided any convergence proof for the iteration depicted in (27). However, we do know that for $m_d = 2$, the iteration changes into a one–step weighted average calculation, shown in (29). Also, it is proved that for $m_d = 1$ the algorithm converges to a unique solution [13], in a linear fashion [18]. In numerous experiments it is observed that the algorithm does converges for other values of $m_d \in (1, 2]$. Using these results we propose the algorithm Fuzzy Video on Demand Network Design (fVoD).

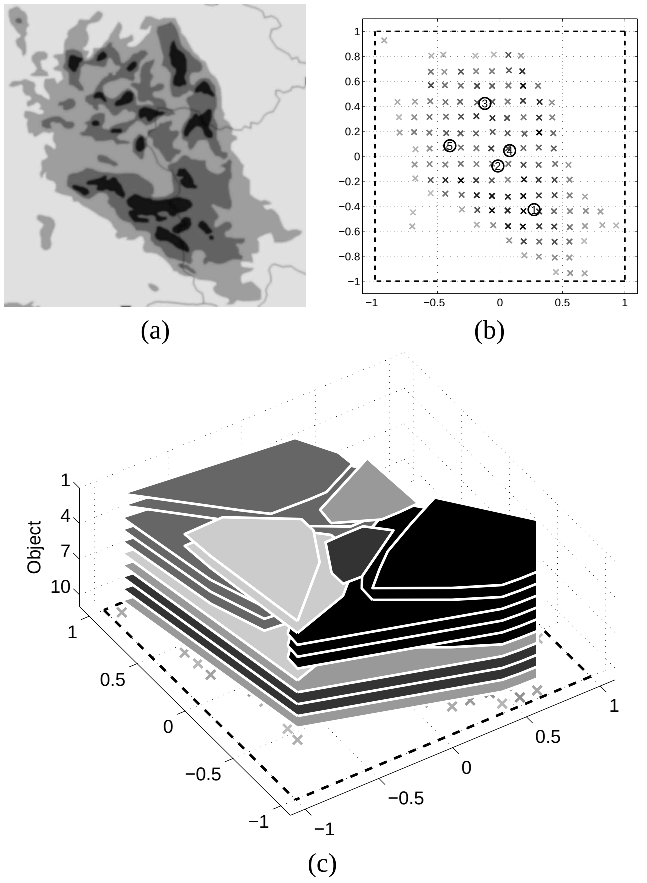
As we do not have any proof for the convexity, or concavity, of $\hat{\Delta}$ [20], we cannot make any argument about the possibility of *fVoD* not being trapped in a local minimum. In fact, in [7], the authors show the example of a similar problem which has 50 customers and 61 local minimums. In order to escape from a local minimum, we choose to run the algorithm for T independent times and then to pick the solution which corresponds to the least value of Δ. Before doing so, we filter out potential solutions for which ρ is bigger than a specified threshold ρ_0. We call this method the *fVoD*^{m} algorithm.

III. Experimental Results

The proposed algorithm is developed in MATLAB 7.0.4 on a PIV 3.00GHZ with 1GB of RAM. To produce the aggregated population, the gray–scale image shown in Figure 1–a is used as the population density map. From this image, $N = 133$ aggregated customers, as shown by the crosses in Figure 1–b, are extracted. Using these customers, the problem is defined as locating $n = 5$ nodes to cache $l = 10$ objects. Here, we select $m_d = 1.3$, $m = 1.1$, $k = 15$, $T = 1000$, d_j ∼Zipf(0.729), and $L = \frac{1}{2} d_l \mu$. Using these parameters, the *fVoD*^{m} algorithm takes about ten minutes to find a solution. From the set of 1000 potential solutions, which exhibit $\Delta \in [0.262, 0.521]$ and $\rho \in [0.34, 0.60]$, 151 solutions are filtered out by selecting $\rho_0 = 0.40$. From the rest, one is retrieved for which $\Delta = 0.303$ and $\rho = 0.40$.

The numbered circles in Figure 1–b show the location of the nodes in this solution. Looking at the structure of the network, shown in Figure 1–c, we observe that the first objects are cached in many nodes, 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_js, L_{ij} should be big, resulting in less number of L_{ij}s being non–zero (see (10)).

Using L_{ij}s, Figure 2–a shows the aggregate allocation at each node (l_i). Here, different colors denote different objects. Similarly, Figures 2–b shows aggregate allocation for different objects in the entire network. Here, different colors indicate different nodes. Looking at some internal variables, Figure 3–1 shows the histogram of $p_{\vec{\mathbf{x}}ij}$s. As seen here, 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. Figure 3–b shows the histogram of the active $\varphi_{ij}s$, showing their closeness to one. In fact, active $\varphi_{ij}s$ vary between 1.14 and 1, the ideal value. Furthermore, the solid line in Figure 3–c shows the number of objects cached in each node. Here, the dashed line shows the maximum value, 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 Figure 3–d, we see the number of nodes in which each object is cached. Here, the dashed line shows the maximum possible values, $\min\{L^{-1} d_j \mu, n\}$. This figure shows that while the first object is cached in every node, the four last objects are only cached in one node, each.



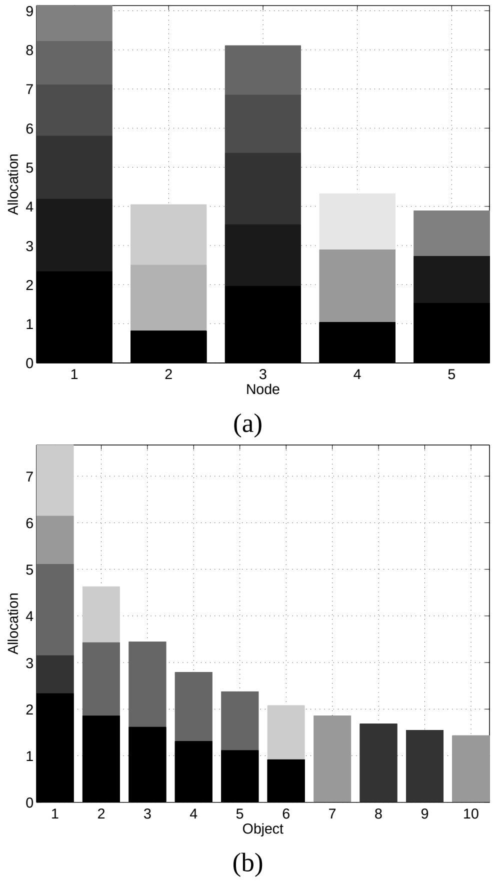
(a) (b)

(c)

Fig. 1. (a) Input population map. (b) Aggregated customers (crosses) and the location of the nodes after the solution is found by the proposed method (circles). (c) 3–D representation of the network as calculated by the proposed method. Different colors indicate content cached in different nodes.

IV. Conclusions

In this paper, we focused on the Video–on–Demand (VoD) network design problem. Using concepts and tools developed in signal coding, an optimization problem was shaped up which was heuristically shown to minimize the communication cost in a VoD network. Also, weights were added to the cost function to implicitly control storage cost. According to the zero–one 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 fuzzy domain. Then, a method was proposed to produce an optimal solution to the proposed objective function. It is worth to mention that the convergence



(a)

(b)

Fig. 2. Investigating the solution produced by the proposed algorithm. (a) Allocation in each node. (b) Allocation for each object. In (a) and (b) different colors show different objects and different nodes, respectively.

of the proposed algorithm was empirically observed. However, a more general proof for the generalized Weiszfeld method, proposed here, is still needed.

Acknowledgment

The research of *Attahiru Sule Alfa* is partially supported by a grant from NSERC. The authors appreciate comments by *Jeff Rohne* on the practical scenarios under which the proposed algorithm can be used. The first author wishes to thank Ms. *Azadeh Yadollahi* for her encouragement and valuable discussions.

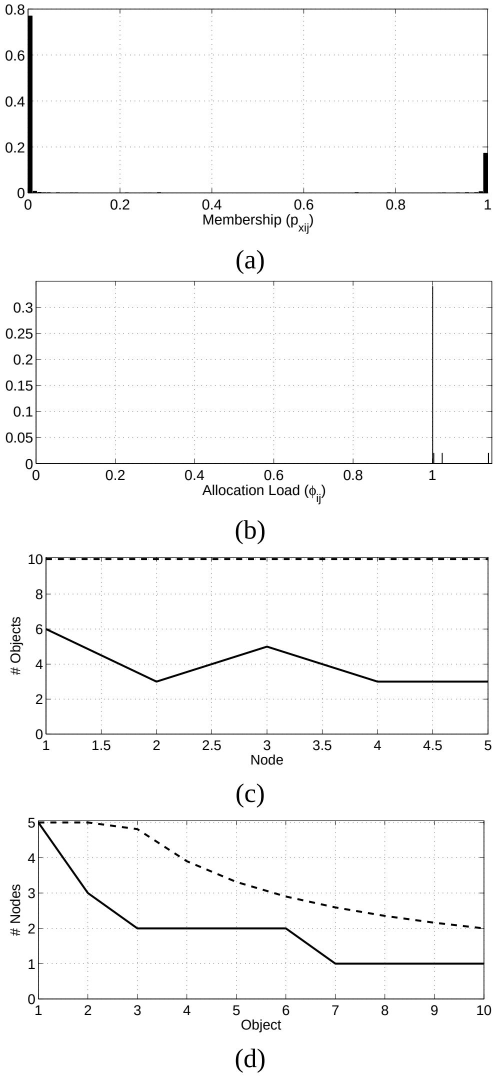
References

[1] H. Ma and K. G. Shin, "Multicast video–on–demand services," *ACM SIGCOMM Computer Communication Review*, vol. 32 (1), pp. 31–43, 2002.

[2] J.-P. Nussbaumer, B. V. Patel, F. Schaffa, and J. P. G. Sterbenz, "Networking requirements for interactive video on demand," *IEEE Journal on Selected Areas in Communications*, vol. 13 (5), pp. 779–787, 1995.

[3] C. Vassilakis, M. Paterakis, and P. Triantafillou, "Video placement and configuration of distributed video servers on cable TV networks," *Multimedia Networks*, vol. 8, pp. 92–104, 2000.

[4] 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: Springer–Verlag, 2002, pp. 179–205.



(a)

(b)

(c)

(d)

Fig. 3. Further investigation of the solution produced by the proposed algorithm. (a) Histogram of $p_{\bar{x}ij}$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.

[5] 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: Springer–Verlag, 2002, pp. 179–205.

[6] N. Megiddo and K. J. Supowit, "On the complexity of some common geometric location problems," *SIAM Journal on Computing*, vol. 13 (1), pp. 182–196, 1984.

[7] D. Eilon, D. T. Watson-Gandy, and N. Christofides, *Distribution Management*. New York: Hanfer, 1971.

[8] R. E. Kuenne and R. M. Soland, "Exact and approximate solutions to the multisource Weber problem," *Mathematical Programming*, vol. 3, pp. 193–209, 1972.

[9] L. Cooper, "Heuristic methods for location–allocation problems," *SIAM Rev.*, vol. SIAM Review, pp. 37–53, 1964.

[10] R. Duda and P. Hart, *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[11] A. Jain and R. Dubes, *Algorithms for Clustering*. NJ: Prentice–Hall, 1998.

[12] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. New York: Plenum Press, 1981.

[13] F. Rado, "The Euclidean multifactory location problem," *Operations*

Research, vol. 36 (3), pp. 485–492, 1988.

[14] J. R. Boucher, *Voice Teletraffic Systems Engineering*. Artech House Publishers, 1988.

[15] A. Dan, D. Sitaram, and P. Shahabuddin, “Scheduling polices for an on–demand video server with batching,” in *Proceedings of Multimedia 94*, San Francisco, CA, USA, 1994, pp. 15–23.

[16] 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 (5), pp. 1215–1220, 2005.

[17] J. M. Leski, “Generalized weighted conditional fuzzy clustering,” *IEEE Transactions on Fuzzy Systems*, vol. 11 (6), pp. 709–715, 2003.

[18] H. W. Kuhn, “A note on Fermat’s problem,” *Mathematical Programming*, vol. 4, pp. 98–107, 1973.

[19] A. Abadpour, A. S. Alfa, and J. Diamond, “Video–on–demand network design and maintenance using fuzzy optimization,” *Accepted for publication in IEEE Transactions on Systems Man and Cybernetics*, 2007.

[20] L. Cooper, “Solutions of generalized locational equilibrium models,” *Journal of Regional Science*, vol. 7, pp. 1–18, 1967.