New Replica Server Placement Strategies using Clustering Algorithms and SOM Neural Network in CDNs

Ghazaleh Eslami[1], Abolfazl Haghighat[1], and Saeed Farokhi[2]
[1]Department of Computer Engineering, Islamic Azad University, Iran
[2]Department of Mechanical Engineering, Tehran University, Iran

Abstract: *Many service providers distribute various kinds of content over the internet. Content Distribution Networks (CDNs) use replication of either entire website or most used objects to bring content close to the users and improve communication delay. In order to deliver web contents, CDNs should decide where to place replica servers and how many replicas are needed. In this paper, a linear programming formulation for web server replica placement has been provided. We also present new algorithms using K-means, Fuzzy c-means clustering and Self-Organizing Maps (SOM) Neural network to place web server replicas. Our objective is to find best replica server sites, which minimize distance between replicas and clients- to keep replicas. We compare our algorithms with Greedy algorithm. We have considerable enhancement in terms of load balancing and Runtime.*

Keywords: *Distributed systems, server placement, clustering algorithms, CDN.*

Received January 6, 2014; accepted June 10, 2015

1. Introduction

The internet was originally conceived as “internet of hosts” but today, the key elements of growing World Wide Web are data and services (or content) [2]. Content Distribution Networks (CDNs) as advanced client/ server networks replicate content from the origin server to surrogate servers-some edge servers that act on behalf of origin server to improve accessibility, reliability, transparency and Quality of Service (QoS) perceived by end clients [10, 11, 14, 24]. CDN providers are either commercial (i.e., Akamai, Limelight, SAVVIS) or academic/ free (i.e., Coral, CoDeeN, Globule) and sign contract with CDN providers [2].

An important and critical aspect in CDN success is the way that replica servers are placed geographically to optimize content delivery [25]. Several algorithms have been proposed to address the replica placement problem [2]. Placement strategies are important because appropriate placement of server replicas benefits content providers by reducing latency for their clients, and benefits Internet Service Providers (ISPs) by reducing bandwidth consumption.
and Hot spot Algorithms. Greedy algorithm places one new facility at each step where in conjunction with the site already exists, yields the lowest cost but in Hotspot replicas are placed near the clients generating the greatest load.

Radoslavov *et al.* [27] consider the replica placement problem for CDNs and ignored the position of clients. Szymaniak *et al.* [33] proposed HotZone,

Replication can be coarse-grained (replication of an entire site or server) or fine-grained (replication of actually required objects) [15]. The major goal of this paper is to introduce three new coarse-grained replica server placement algorithms using K-means, Fuzzy c-means clustering and Self-Organizing Maps (SOM) neural network algorithms to minimize distance between replica servers and their corresponding clients, hence minimize runtime and latency and balance load between replicas.

2. Related Works

Replication is commonly employed by distributed systems to improve the communication delay experienced by their clients [15]. There have been several studies that have addressed the problem of replica placement on the network. Li *et al.* [20] proposed an optimal placement policy of web proxies for a target web server in the internet. Benoit *et al.* [7] also addressed the problem of placing replicas in tree networks. Qiu *et al.* [26] formulate web server replica placement as a minimum K-median graph theoretic problem and proposed Greedy
that provides nearly optimal results by considering overlapping neighbourhoods. Bartolini *et al.* [6] formulate the dynamic replica server placement as Semi Markov decision.

Asahara *et al.* [5] introduces a strategy for dynamically selecting replica server spots. Xu [38] defines the Fault Tolerant Facility Allocation (FTFA) problem for the placement of replica servers and formulate QOS-aware content replication for parallel access. Yang *et al*. [40] study the budgeted server placement problem in wireless and unstable networks. In [39] two degree-based replica placements are proposed which gives minimize access cost in P2P data grids. Alshayeji *et al*. [4] proposed a context-aware replica placement algorithm in P2P networks. For a review on recent replica placement algorithms on P2P networks see [12].

In [16, 28] authors formulate replicated server placement with QoS constraints. Xiong *et al.* [37] proposed a dynamic programming based replica placement algorithm that finds the optimal nodes for replicas. Subramanyam *et al.* [32] proposed a priori replica placement strategy in order to improve grid performance. Takeshita *et al.* [34] proposed a fast calculation method that used parallel processing based on exhaustive search for the replica placement problem.

In this paper, we first provide a Linear programming formulation for web server replica placement and then propose three new algorithms for web server replica placement. We use K-means, Fuzzy c-means clustering and SOM Neural network algorithms. The major objective of this paper is to solve replica placement to minimize runtime and latency and also balance load between replicas which is critical specially when requests are dynamically changing

3. Linear Programming Formulation

Here, we provide a linear programming formulation for web server replica placement. Let us have N_C clients and N_S sites. We are interested in choosing N_R Replicas among these sites ($N_S>N_R$) in order to minimize Euclidean distance between replicas and their corresponding clients. We define it as follows:

$$Totalsum = \min(\sum_{i=1}^{N_C}\sum_{j=1}^{N_S}\alpha_{ij}d_{ij}) \tag{1}$$

$$d_{ij}=\sqrt{(x_{ci}-x_{si})^2+(y_{ci}-y_{sj})^2} \qquad \sum_{j=1}^{N_S}\alpha_{ij}=1 \quad \forall i\in N_C \tag{2}$$

$$\alpha_{ij}>0,\ \forall i\in N_C, j\in N_S \tag{3}$$

Where the 0-1 matrix α_{ij}=1 if client *i* gets its content from server *j* otherwise zero.

Request Latency (RL_j) is another factor which is very crucial. So, we define RL_j as the average time needed to find the location of replicas and get content to clients.

$$RT_j=\frac{TotalRuntime}{N_R} \tag{4}$$

$$RL_j=RT_j+\mathrm{Re}sponse_j \tag{5}$$

Where RT_j is the average time needed to find the location of replicas and $Response_j$ is the time needed to deliver content from server j to its clients. So we can formulate our problem as follows:

$$\min\{\sum_{i=1}^{N_C}\sum_{j=1}^{N_S}\alpha_{ij}(d_{ij}+RL_j)\} \tag{6}$$

Load balancing is concerned with the balance use of replica servers among clients. Let each server responds to maximum N_L clients. So if each replica server responds to much more clients that N_L, leads to poor load balancing, more overall cost, more bandwidth consumption and more latency for clients to receive their request. We define N_L as follows:

$$N_L=\left\lceil\frac{N_C}{N_R}\right\rceil \tag{7}$$

We define Load Variance (*LV*) as follows:

$$LV=\sum_{j=1}^{N_R}\left|N_{S_j}-N_L\right| \tag{8}$$

$$LV_j=\left|N_{S_j}-N_L\right| \tag{9}$$

Where *LV* is the Total Load Variance between clients and replicas and LV_j is load variance of replica server *j* and N_{S_j} is the number of clients which direct their request to replica server *j*. Our objective is to minimize the following function:

$$\min\{\sum_{i=1}^{N_C}\sum_{j=1}^{N_S}\alpha_{ij}(d_{ij}+RL_j)\times\frac{LV_j}{N_L}\} \tag{10}$$

$$\forall j\in N_S \qquad \sum_{i=1}^{N_C}\sum_{j=1}^{N_S}\alpha_{ij}=N_{S_j} \tag{11}$$

$$\alpha_{ij}\in\{0,1\}$$

4. K-means, Fuzzy c-means and SOM Algorithms for Web Server Replica Placement

Clustering is an unsupervised classification. The main goal of clustering is to group similar objects together so each group becomes a cluster [1, 3]. In the following, we propose a version of K-means, Fuzzy c-means clustering and SOM algorithms for web server replica placement. We first group our data (clients) into random clusters and then we find the nearest nodes (selected server site) to centers of these clusters to place our replicas.

4.1. K-means Clustering for Replica Server Placement

The K-means Clustering is probably the most well-known data clustering algorithm [1]. The Algorithm starts with *k* initial seeds of clustering. All the n objects are then compared with each seed by means of Euclidean distance and assigned to the closest cluster seed. The procedure is then repeated over and over again. The algorithm stops when the changes in the cluster seeds from one stage to the next are close to zero or smaller than a pre-specified [22].

In the problem of web server replica placement, we want to select *M* replicas among *N* sites. In other words, we partition *C* clients into *k* disjoint subsets and then we select the *k* centers (winners) from *N* which are closest to *k* (*k*=*M*).

The K-means algorithm for web server replica placement is shown in Algorithm 1.

Algorithm 1: The K-means algorithm for web server replica placement

```
1. Input:
   a. k:number of centers
   b. M:number of replicas
2. Initialization (k first random centers: $c_{1,\dots,} c_k$)
3. choose centers
Repeat
    For each client
        a. Calculate the Euclidean distances between client and different centers
        b. Assign the client to the nearest center ($c_1,\dots,c_k$)
For each cluster
        a. Calculate the new center
        b. Replace new centers with old ones($c_1=c_{new1},\dots,c_k=c_{newk}$)
  End for
Until no changes between old centers and new centers
4. for each center i (i=1,...,k) {center of clusters)
        a. Find a point from N which is closest to the center i
        b. Assign new point as the center of the cluster {replica server placement}
End for
```

The Basic K-means algorithm [21] includes three steps. We add one step to basic K-means to solve web server replica placement. In the fourth step we find M replicas which are nearest to K-means seed. The time complexity of the K-means algorithm for replica server placement is *O*(*CMl*), where *l* is the total number of iterations, *M* is the total number of servers (cluster seeds), and *C* is the total number of objects. Normally, *M*<<*N* and *l*<<*C* [29]. The space complexity of K-means algorithm for web server replica placement is *O*(*M*+*C*). The reason behind choosing K-means for web server replica placement is its simplicity but K-means is NP-complete.

The criterion minimized by K-means method is the sum of within cluster distances to centers [9]. The criterion for K-means web server replica placement can be written as:

$$Totalsum = \min(\sum_{i=1}^{N_c}\sum_{j=1}^{c_k}\alpha_{ij}d_{ij}) \tag{12}$$

$$\sum_{j=1}^{c_k}\alpha_{ij} = 1 \qquad \forall i \in N_C \tag{13}$$

4.2. Fuzzy c-Means Algorithm

The Fuzzy c-means clustering algorithm was proposed by Dunn in 1972 and generalized by Bezdek [8, 18]. We abbreviate Fuzzy c-means as FCM. Assuming that c clusters are to be generated from *n* (here *n*=*N*$_C$) data point x_i {*i*=1,…,*n*}. FCM clustering achieved by an iterative optimization process that minimize the objective function [18, 31]:

$$J = \sum_{i=1}^{n}\sum_{j=1}^{c}(u_{ji})^m \left\|x_i - v_j\right\|^2 \tag{14}$$

Subject to:

$$\sum_{i=1}^{c} u_{ji} = 1 \tag{15}$$

Where u_{ji} is the probabilistic membership of pattern x_i to centroid v_j and $1 \le m < \infty$ is the fuzzifier and d_{ij} represents the distance from a pattern x_i to the cluster center v_j. $\| \;\|$ denotes any inner product norm metric.

The process starts by randomly choosing *c* centroids and calculating v_j and u_{ji} for each object using following equations and calculating new centroids until the centroids stabilize.

$$v_j = \frac{\sum_{i=1}^{n}(u_{ji})^m x_i}{\sum_{i=1}^{n}(u_{ji})^m} \tag{16}$$

$$u_{ji} = (\sum_{k=1}^{c}(\frac{d_{ji}}{d_{ki}})^{\frac{2}{m-1}})^{-1} \tag{17}$$

Where $d_{ji} = \left\|x_i - v_j\right\|$

The Fuzzy c-means for web server replica placement is shown in Algorithm 2.

Algorithm 2: The Fuzzy c-means for web server replica placement

```
1. Random initialization of C centroids
2. Repeat
Updating C centroids by calculating $v_j$ and $u_{ji}$ until centroid stabilization {$u_{ji(new)}-u_{ji(old)}<\varepsilon$}
3. For each center j (j=1,...,c ) {center of clusters)
   a. Find a point from N which is closest to the center vj
   b. Assign new point as the center of the cluster {replica server placement}
End for
```

The time complexity of FCM is O(CdM^{2}i) [17] and space complexity is O(Cd+CM) Where *C*= number of data points (clients), *M*=number of cluster, *d*= dimension, *i*=number of iteration. We consider *d*=2. We should notice that each data point belongs to exactly one cluster.

4.3. Self Organizing Maps

T. Kohonen began to explore SOM in 1982. The SOM is applied to cluster and visualize data [30]. In a sense, SOM can be thought of as spatially constrained form of K-means clustering [36]. The SOM is trained iteratively and weight vectors are updated properly so that the nodes move to form clusters. The SOM has two steps: winner selection and weight adaption. In the first step neurons compete each other and one neuron becomes the winner in each step. In the weight

adaption, neurons are related by a neighborhood function dictating the structure of the map and neighbors of the winner update their weights [23]. The SOM algorithm is applicable to large data sets. The computational complexity scales linearly with the number of data samples and does not require huge amounts of memory [13].
The SOM for web server replica placement is shown in Algorithm 3.

Algorithm 3: *The SOM for web server replica placement*

```
1. Initialize codebook vector mi {i=1,...,num_replica}
2. Repeat
For all k=1 to num-clients
a. Winners selection: select the best matching unit (winner) $m_c$
{c=1,...,num_replica } $\|x(t)-m_c(t)\| = \min\{\|x(t)-m_c(t)\|\}$
b. Weight adaption: Update winner neuron and its topological
neighbors ($N_c$)
$m_i(t+1) = m_i(t) + \alpha(t)h_c(t)[x(t)-m_i(t)]$
$\forall i \in N_C(t)$
Until $\alpha <= \varepsilon$ or $\alpha = 0$
3. for each winner
a. Find a point from N which is closest to the winner i
b. Assign this point as the center of the cluster {replica server
placement phase}
End for
```

Where α is a scalar parameter that during the course of the process decreases monotonically ($0<\alpha<1$) [19, 35] and $h_c(t)$is the neighborhood function. A variety of neighborhood function can be used.

5. Computational Time

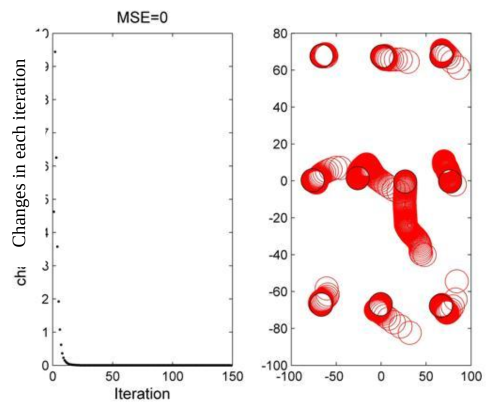
Table 1 lists the computational time of Web server replica placement algorithms. It can be seen that Fuzzy c-means has heavy computational time compared to SOM and K-means. SOM complexity scales linearly with the number of data samples and is significantly lower than other algorithms.

Table 1. Computational time of replica server placement algorithms.

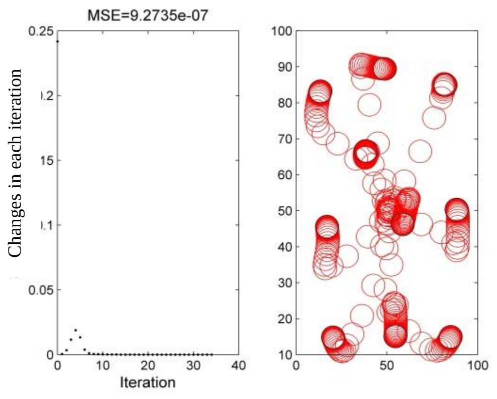
Replica Server Placement Algorithms	Computational Time
Greedy [26]	$O(N_S^2 N_R)$
Tree-based [20]	$O(N_S^{3} N_R^{2})$
Hot Spot [26]	$O(N_S^{2}+\min(N_S \log N_S + N_S N_R)$
K-means for web server replica placement	$O(N_C N_R i)$
Fuzzy c-means for web server replica placement	$O(N_C d N_R^{2} i)$
SOM for web server replica placement	$O(N_S)$

6. Simulation and Result

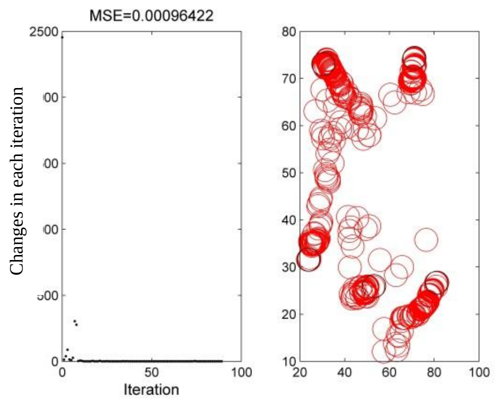
We have compared our algorithms with Greedy algorithm proposed by Qiu *et al*. [26] because it outperformed other techniques. In our simulation we assume that each client only uses a single replica. We run our algorithms on 100000 clients and 300 servers and vary the number of replicas from 5 to 50. K-means, Fuzzy c-means and SOM and Greedy algorithm was implemented by Matlab 2013 and we used random numbers. SOM learning rate was initiated as 0.9 and convergence criterion was set to 0.001 ($\alpha<0.001$).



a) K-means data clustering algorithm for web server replica placement.



b) Fuzzy C-means clustering algorithm for web server replica placement.



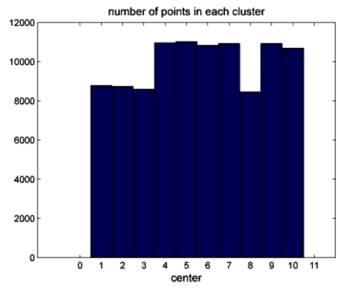
c) SOM algorithm for web server replica placement.

Figure 1.Changes of replica places in each iteration.

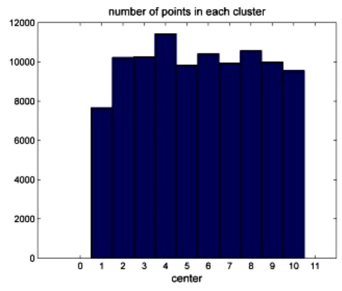
Neighbourhood function is defined to be decreasing around winner neuron.
Fuzzy c-means was implemented using a degree of fuzziness *m*=2. Changes of replica places in each iteration for K-means, Fuzzy c-means and SOM algorithms are shown in Figure 1 for 10 replicas .We see that SOM for web server replica placement needs less iteration. Figure 2 shows the number of points (clients) in each cluster (replica server). By comparing Figure 2.a, 2.b and 2.c, we observe that Fuzzy c-means performs best of all the others and hence with minimum LVs, balance load between replicas.

Table 2. Total Sum of K-means, fuzzy c-means, SOM and greedy algorithms.

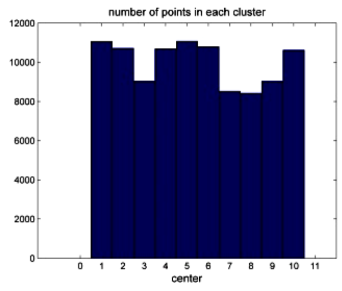
K-means Total sum	Fuzzy c-means Total sum	SOM Total Sum	Greedy Sum
3.5040e+006	3.5040e+006	3.5089e+006	2.9040e+006
3.2193e+006	3.2320e+006	3.2194e+006	2.7230e+006
2.9779e+006	2.9944e+006	2.9685e+006	2.5110e+006
2.7485e+006	2.7460e+006	2.7484e+006	2.3609e+006
2.6058e+006	2.6042e+006	2.5974e+006	2.2218e+006
2.4571e+006	2.4665e+006	2.4584e+006	2.1727e+006
2.3493e+006	2.3477e+006	2.3485e+006	2.0353e+006
2.2508e+006	2.2584e+006	2.2449e+006	1.9911e+006
2.1604e+006	2.1746e+006	2.1641e+006	1.9058e+006
2.0958e+006	2.1133e+006	2.0777e+006	1.8252e+006
2.0090e+006	2.0071e+006	2.0077e+006	1.8049e+006



a) K-means data clustering algorithm for web server replica placement.



b) Fuzzy C-means clustering algorithm for web server replica placement.



c) SOM algorithm for web server replica placement.

Figure 2. Number of points in each cluster.

Table 3. Runtime of K-means, Fuzzy C-means, SOM and greedy algorithms.

K-means Run time	Fuzzy c-means Runtime	SOM Run time	Greedy Run time
8.5552	17.4219	149.2705	219.1249
13.8859	23.5904	217.0253	256.5619
18.5917	33.1249	205.5674	288.8870
46.1003	30.9806	227.5833	290.0059
44.2977	34.3103	285.4735	299.7797
33.4744	36.7862	229.1504	334.5411
29.7780	58.0581	249.0660	334.7417
54.1601	45.5202	177.3717	394.3413
41.7402	85.3081	217.7661	394.6655
45.5037	53.9890	327.9885	417.0590
84.1300	87.7375	192.2722	431.3808

Tables 2, 3 and 4 show Total sum, Run time and LV obtained from our algorithms and Greedy algorithm proposed by Qiu *et al*. [26]. We can observe that although Greedy algorithm achieves minimum total sum but there is a significant difference between Greedy algorithm runtime and load variance proposed by Qiu *et al*. [26] and our algorithms. As can be observed, the runtime needed by K-means algorithm to compute its placements is 10 orders of magnitude lower than Greedy algorithm. Fuzzy c-means Runtime was on average 90% less than Greedy algorithm and SOM runtime was on average about 60% less than Greedy algorithm. As shown in Table 4, Fuzzy C-means clustering for web server replica placement performs the best and with minimum LV, balance the load between replica servers. Load Variance in Greedy algorithm is significantly larger than our algorithms which may led to poorly balanced replica servers and tends to become an unnecessary bottleneck leading to longer delays and more bandwidth consumption.

Table 4. Load Variance of K-means, Fuzzy C-means, SOM and Greedy algorithms.

K-means Load Variance	Fuzzy c-means Load Variance	SOM Load Variance	Greedy Load Variance
9040	6956	8990	32106
10844	3140	10466	21466
10807	6833	10289	21052
9082	9820	8326	9406
1863	2795	1631	14534
10398	7136	10202	18276
8332	7454	7198	13781
7718	5630	8338	19500
7646	6402	8504	13125
8750	8556	7740	14200
5312	5818	5766	17597

7. Conclusions

In this paper we have presented three new algorithms to solve web server replica placement problem using K-means, Fuzzy C-means and SOM by adding a new step in these algorithms. Although Greedy algorithm proposed by Qiu *et al*. [26] gives better results but data clustering algorithms for web server replica placement are much better in terms of load balancing and runtime which are important in delay and bandwidth consumption in CDN networks. Also, the Computational time of SOM, K-means and Fuzzy C-means algorithms for web server replica placement are significantly lower than Greedy algorithm proposed by Qiu *et al*. [26]. We also apply Linear Programming to formulate web server replica placement. We believe that our work provides insights to CDN providers on how to design CDNs to provide load balancing among replica servers.

References

[1] Abu Abbas O., "Comparision Between Data Clustering Algorithms," *The International Arab*

Journal of information Technology, vol. 5, no. 3, pp. 320-325, 2008.

[2] Almeida F. and Calistru C., "The Role of Content Distribution Networks in the Future Media Networks," *International Journal of Emerging Trends and Technology in Computer Science*, vol. 1, no. 1, pp. 77-85, 2012.

[3] Alnaji K. and Ashour W., "A Novel Clustering Algorithm using K-means," *International Journal of Computer Applications*, vol. 25, no. 1, pp. 25-30, 2011.

[4] Alshayeji M., Samrajesh M., Al-Behairy S., and Al-Enazi K., "Context-Aware Replica Placement in Peer-to-Peer Overlay Networks," *International Journal of Computer Theory and Engineering*, vol. 5, no. 2, pp. 383-389, 2013.

[5] Asahara M., Shimada A., Yamada H., and Konko K., "Strategy for Selecting Replica Server Spots on the Basis of Demand Fluctuations," *The Information Processing Society of Japan Online transactions*, vol. 1, no. 1, pp. 28-41, 2008.

[6] Bartolini N., Presti F., and petrioli C., "Optimal Dynamic Replica Placement in Content Delivery Networks," *in Proceeding of The 11th IEEE International Conference on Networks*, Sydney, pp. 125-130, 2004.

[7] Benoit A., Renaud P., and Robert Y., "Power-Aware Replica Placement and Update Strategies in Tree Networks," *in Proceeding of International Parallel and Distributed Processing Symposium IEEE*, Anchorage, pp. 2-13, 2011.

[8] Cannon R., Dave J., and Bezdek A., "Efficient Implementation of the Fuzzy C-Means Clustering Algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 2, pp. 248-255, 1986.

[9] Cordeiro R. and Mirkin B., "Minkowski Metric, Feature Weighting and Anomalous Cluster Initializing in K-Means Clustering," *Elsevier Pattern Recognition*, vol. 45, no. 3, pp. 1061-1075, 2012.

[10] Giura p. and Reyes G., "The Security Cost of Content Distribution Network Architecture," *in Proceeding of IEEE Annual Computer Software and Applications Conference Workshops*, Munich, pp. 128-135, 2011.

[11] Idowu S., "A Contrastive Study of Content Distribution Networks Models," *in Proceeding of Information Technology for People-Centred Development*, Abuja, pp.1-6, 2011.

[12] karun K. and Jayasudha J., "Recent Replica Placement Algorithms in P2P Networks-A Review," *International Journal of Computer Network and Information Security*, vol. 5, no. 5, pp. 55-63, 2013.

[13] Kangas J., Kohonen T., and Laaksonen J., "Variants of Self-organizing Maps," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 93-99, 1990.

[14] Khan S., Maciejewski A., and Siegel H., "Robust CDN Replica Placement Techniques," *in Proceeding of IEEE International Symposium on Parallel and Distributed Processing*, Rome, pp. 1-8, 2009.

[15] Khan S. and Ahmad I., "Comparison and Analysis of Ten Static Heuristics-Based Internet Data Replication Techniques," *Journal of Parallel and Distributed Computing*, vol. 68, no. 2, pp. 113-136, 2008.

[16] Kia H. and Ullah Khan S., "Server Replication in Multicast Networks," *in Proceeding of 10th International Conference on Frontiers of Information Technology*, Islamabad, pp. 337-34, 2012 .

[17] Kumar P. and Sirohi D., "Comparative Analysis of FCM and HCM Algorithm on Iris Data Set," *International Journal of Computer Applications*, vol. 5, no. 2, pp. 33-37, 2010.

[18] Kwok T., Smith K., Lozano S., and Taniar D., "Parallel Fuzzy c-Means Clustering for Large Data Sets," *in Proceeding of 8th International Euro-Par Conference*, Paderborn, pp. 365-374, 2002.

[19] Lawrence R., Almasi G., and Rushmeier H., "A Scalable Parallel Algorithm for Self-Organizing Maps with Applications to Sparse Data Mining Problems," *Data Mining and Knowledge Discovery*, vol. 3, no. 2, pp. 171-195, 1995.

[20] Li B., Golin M., Ialiano G., and Deng X., "On the Optimal Placement of Web Proxies in the Internet," *in Proceeding of 18th Annual Joint Conference of the IEEE Computer and Communications Societies*, new York, pp. 1282-1290 ,1999 .

[21] Meskine F. and Bahloul S., "Privacy preserving K-means clustering: A Survey," *The International Arab Journal of Information Technology*, vol. 9, no. 2, pp. 194-200, 2012.

[22] Mingoti S. and Lima J., "Comparing SOM Neural Network with Fuzzy C-Means, K-Means and Traditional Hierarchical Clustering algorithms," *European Journal of Operation Research*, vol. 174, no. 3, pp. 1742-1759, 2006.

[23] Moschou V., Ververidis D., and Kotropoulos C., "Assessment of Self-Organizing Map Variants for Clustering with Application to Redistribution of Emotional Speech Patterns," *Elsevier*, vol. 71, no. 1-3, pp. 147-156, 2007.

[24] Neves T., Drummond L., Ochi L., Albuquerque C., and Uchoa E., "Solving Replica Placement and Request Distribution in Content Distribution Networks," *Electronic Notes in Discrete Mathematics*, vol. 36, pp. 89-96, 2010.

[25] Passarella A., "A Survey on Content-Centeric Technologies for the Current Internet :CDN and

P2P Solutions," *Computer Communications Elsevier*, vol. 35, no. 1, pp. 1-35, 2012.

[26] Qiu L., Panmanabhan V., and Voelker G., "On the Placement of Web Server Replicas," *in Proceeding of 20th Annual Joint Conference of the IEEE Computer and Communications Society*, Anchorage, pp. 1587-1596, 2001.

[27] Radoslavov P., Govindan R., and Estrin D., "Topology-Informed Internet Replica Placement," *Computer Communications*, vol. 25, no. 4, pp. 384-392, 2001.

[28] Rodolakis G., Siachalou S., and Georgiadis L., "Replicated Server Placement with QoS Constraints," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 10, pp. 1151-1162, 2006.

[29] Singh S. and Chauhan N., "K-means v/s K-Medoids: A Comparative Study," *in Proceeding of National Conference on Recent Trends in Engineering and Technology*, Gujarat, pp. 7-10, 2011.

[30] Stefanovic P. and Kurasova O., "Visual Analysis of Self-Organizing Maps," *Nonlinear Analysis: Modeling and Control*, vol. 16, no. 4, pp. 488-504, 2011.

[31] Su M., Chou C., and Hsieh C., "Fuzzy C-means Algorithm with a Point Symmetry Distance," *International Journal of Fuzzy Systems*, vol. 7, no. 4, pp. 1-8, 2005.

[32] Subramanyam G., Lokesh G., and Kumari B., "A Priori Data Replica Placement Strategy in Grid Computing," *International Journal of Scientific and Engineering Research*, vol. 4, no. 7, pp. 1070-1076, 2013.

[33] Szymaniak M., Pierre G., and Steen M., "Latency-Driven Replica Placement," *in Proceeding of Symposium on Applications and the Internet,* Trento, pp. 399-405, 2005.

[34] Takeshita H., Shimizu S., Ishikawa H., Watanabe A., Arakawa Y., Yamanaka N., and Shiba K., "Fast Optimal Replica Placement with Exhaustive Search Using Dynamically Reconfigurable Processor," *Journal of Computer Networks and Communication* , vol. 2011, pp. 1-11, 2011.

[35] Vesanto J. and Alhoniemi E., "Clustering of the Self-Organizing Map," *IEEE Transactions on Neural Networks*, vol. 11, no. 3, pp. 586-600, 2000.

[36] Wehrens R. and Buydens L., "Self- and Super-Organizing Maps in R: The Kohonen Package," *Journal of Statistical Software*, vol. 21, no. 5, pp.1-15 2007.

[37] Xiong F., Ruchuan W., and Song D., "Dynamic Programming Based Replica placement Algorithm in Data Grid," *Chinese Journal of Electronics*, vol. 19, no. 4, pp. 699-704, 2010.

[38] Xu S., "Replica Placement Algorithms for Efficient Internet Content Delivery," *School of computer science*, Australia, pp. 699-704, 2009.

[39] Xunyi R., Ruchuan W., Qiang K., and Danwei C., "Degree-Based Replica Placement Algorithms for P2P Data Grids," *Chinese Journal of Electronics*, vol. 19, no. 3, pp. 486-490, 2010.

[40] Yang D., Fang X., and Xue G., "ESPN: Efficient Server Placement in Probabilistic Networks with Budget Constraint," *in Proceeding of Joint Conference of the IEEE Computer and Communications Societies*, Shanghai, pp. 1269-1277, 2011.

Ghazaleh Eslami received her BSc and MSc degree in Computer engineering from Tehran South Branch, Islamic Azad University in 2003 and 2006. She is a PHD student at Qazvin Branch, Islamic Azad University, Qazvin, Iran. Her research interests include Content Delivery Networks, Cloud Computing and DNA computing.

Abolfazl Haghihat received his BSc and MSc degree in electrical engineering from Sharif University, Tehran, Iran, in 1992 and 1995 and PhD degree in electrical engineering from Amir Kabir University of Technology, Tehran, Iran, in 2002. His research interests include Distributed Systems and Sensor Networks.

Saeed Farokhi received his BSc and MSc degrees in Mechanical engineering from Tehran University, Tehran, Iran, in 2008 and 2011. His research interests include Decision Support Systems, Quantitative Analysis, Forecasting and Optimization.