Available online at www.sciencedirect.com
ScienceDirect
Computational Statistics & Data Analysis 51 (2007) 6285–6294
COMPUTATIONAL STATISTICS & DATA ANALYSIS
www.elsevier.com/locate/csda

Genetic clustering of social networks using random walks

Aykut Firat*, Sangit Chatterjee, Mustafa Yilmaz

College of Business Administration, Northeastern University, Boston, MA 02115, USA

Received 9 December 2005; received in revised form 11 January 2007; accepted 13 January 2007
Available online 1 February 2007

Abstract

In the era of globalization, traditional theories and models of social systems are shifting their focus from isolation and independence to networks and connectedness. Analyzing these new complex social models is a growing, and computationally demanding area of research. In this study, we investigate the integration of genetic algorithms (GAs) with a random-walk-based distance measure to find subgroups in social networks. We test our approach by synthetically generating realistic social network data sets. Our clustering experiments using random-walk-based distances reveal exceptionally accurate results compared with the experiments using Euclidean distances.

Keywords: Genetic algorithms; Clustering with medoids; Synthetic data generation

1. Introduction

Popularized by the theory of "six degrees of separation" (Milgram, 1967), social networks are increasingly being used to model, organize, and study the relationships between social actors shifting the focus from isolated individuals to connected networks. In recent years there has been a proliferation of social network web sites such as Friendster, LinkedIn, and Orkut, which create maps of the relationships between individuals based on their social familiarity. Many other social phenomena such as disease transmission, corporate communication, supplier–customer relationships, and terrorist activities can also be seen as examples of social networks.

While a mathematical theory of social complexity remains a pipe dream, the emergence of huge amounts of social network data is prompting researchers to apply heuristic techniques to find patterns in social networks (Barabási, 2005). Clustering, for example, is one of the techniques used by mathematical sociologists to unravel subgroups in a social network based on pre-defined measures of cohesion (Jain et al., 1999).

Clustering a social network, however, is an NP-hard problem, thus using brute force techniques rapidly becomes impractical as the network size increases. Even clustering 333 nodes into two clusters using brute force needs to check about a googol (10^{100}) possibilities. Thus, heuristic approaches are needed to obtain approximate solutions.

In this study, we use genetic algorithms (GAs) as a heuristic search technique to cluster social networks. In order to accurately identify the clusters, we adopt a distance measure based on random walks and use a *medoids-based* genetic representation. Our experiments with synthetically created social networks reveal that random-walk-based distance

* Corresponding author. Tel.: +1 617 3734801; fax: +1 617 3734804.
E-mail address: a.firat@neu.edu (A. Firat).

0167-9473/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.csda.2007.01.010

measure is clearly superior to Euclidean distances, and genetic clustering competes well with established clustering techniques such as hierarchical clustering.

The rest of this paper is organized as follows. In Section 2, we provide the necessary background on graph theoretic concepts, define the network clustering problem, and overview the clustering work based on random walks. Section 3, provides the details of deriving the random-walk-based distance measure, and Section 4 describes the details of genetic representation. Our experiments are discussed in Section 5.

2. Background

A graph is a set of objects called nodes connected by links called edges, which can be directed or undirected. A directed graph with weighted edges is called a network. A weighted graph $\mathbf{G}$ can be denoted by $(\mathbf{V}, \mathbf{E}, w)$, where $\mathbf{V}$ is the set of nodes and $\mathbf{E}$ is the set of edges, and $w : \mathbf{E} \to \mathbb{R}^+$, is a function that measures the degree of similarity between the nodes (higher values indicating higher similarity). We use w_{ij}, which are symmetric; i.e., $w_{ij} = w_{ji}$, to express the similarity between nodes i and j. Also, $w_{ij} = 0$ if and only if there is no edge connecting nodes i and j.

The objective of network clustering is to find a decomposition of the node set into subsets that are highly intra-connected and sparsely inter-connected. More precisely, the k-clustering of a graph aims to find k non-overlapping subsets $\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_k$ of $\mathbf{V}$ such that their corresponding sub-graphs are highly connected according to some distance measure.

The graph clustering paradigm postulates that "natural" groups in networks have the following general property (Van Dongen, 2002): "A random walk in G that visits a highly intra-connected cluster will likely not leave the cluster until many of its nodes have been visited." This idea was first used by Hagen and Kahng (1992) in circuit clustering via stochastic simulation of a random walk; i.e., by generating random numbers and taking random steps accordingly. Simulation of a random walk is, however, quite expensive in terms of running time, so other deterministic approaches were sought. In Harel and Koren (2001), for example, separating edges of a graph were identified based on low "escape probabilities" between the edges of different clusters, and in Van Dongen (2002) these edges are found using a flow simulation analysis. Both of these methods are similar in their operation, in that the boundaries of a cluster are found out either by iteratively sharpening "escape probabilities" or by increasing flow where the current is strong and demoting flow where the current is weak. After some iteration, the edges of natural groups are revealed as probability of escape decreases beyond a threshold or the current across borders between different groups withers away.

In approaches that use (GAs) for network clustering, we see Freeman utilizing a simple genetic algorithm to find two subgroups in a small network that models the social activities of 18 American women (Freeman, 1993, 2003). The fitness function used in his work classifies a node based on the sum of edge weights incident on that element leading to nodes in the same set versus the sum of incident edge weights leading to nodes in different sets. Later on, Borgatti and Everett (1997) expanded on Freeman's algorithm.

3. The random-walk-based distance measure

A random walk is a simple stochastic process that formalizes the intuitive idea of taking successive steps, each in a random direction. In a network, a random walk starts from an initial node, and based on a probability distribution, progresses to neighboring nodes. Using Markov chains, we can model this process and measure closeness between nodes. In this study, for example, we use the average number of steps a random walk takes starting from node i to reach node j and come back to node i as the distance between nodes i and j. This quantity, known as *average commute time* (ACT), is defined in terms of *average first passage times*. The average first passage time $m(k \mid i)$ is defined as the average number of steps in a random walk that starts from node i and reaches node k for the first time. This quantity is recursively defined as follows (Norris, 1997):

$$m(k \mid i) = \begin{cases} m(k \mid k) = 0, \\ m(k \mid i) = 1 + \sum_{j=1, j \neq k}^{N} p_{ij} m(k \mid j) \quad \text{for } i \neq k, \end{cases} \tag{1}$$

where p_{ij} is the probability of transitioning from node i to j. The base condition, $m(k \mid k) = 0$, states that transitioning of a node to itself is not permissible.

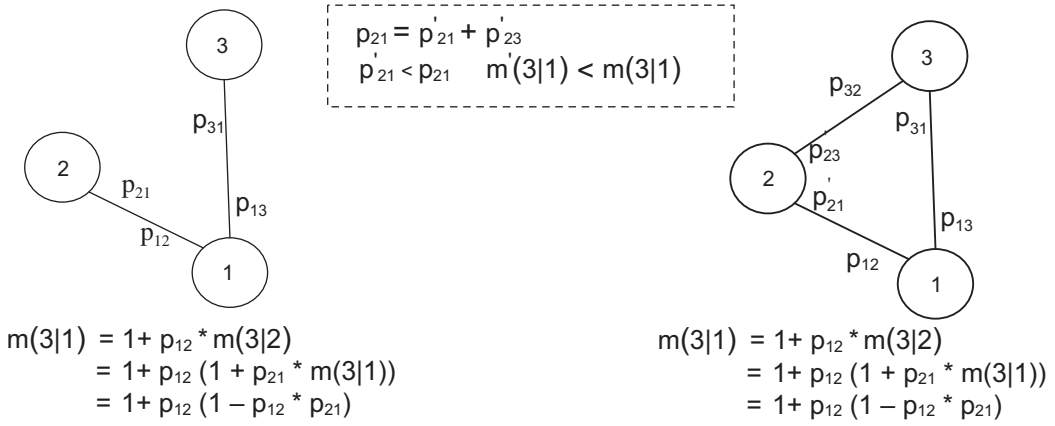


Fig. 1. Distance between nodes gets shorter as more paths are added to the system.

ACT, $n(i, j)$, is the symmetrized version of the average first passage time; i.e., $n(i, j) = m(j|i) + m(i \mid j)$, and is shown to be a distance measure between any two nodes that can be computed using the following formula (Yen et al., 2005):

$$n(i, j) = V_{\mathrm{G}}(\mathbf{u}_i - \mathbf{u}_j)\mathbf{L}^{+}(\mathbf{u}_i - \mathbf{u}_j)^{\mathrm{T}}, \tag{2}$$

where $\mathbf{u}_i$ is a basis vector with ith dimension being 1 and the rest being 0 (e.g., $\mathbf{u}_2 = [0\ 1\ 0\ 0 \cdots 0]$), V_{G} is the volume of the graph ($V_{\mathrm{G}} = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij}$), and $\mathbf{L}^{+}$ is the Moore–Penrose pseudo-inverse of the Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{A}$. Here, $\mathbf{A}$ is the similarity matrix (with individual elements w_{ij}), and $\mathbf{D}$ is the degree matrix, whose diagonal elements set to $D_{ii} = \sum_{j=1}^{n} w_{ij}$ and other elements to zero. It is shown by (Fouss et al., 2005) that $\mathbf{L}^{+}$ can be obtained using $\mathbf{L}$ as follows:

$$\mathbf{L}^{+} = \left(\mathbf{L} - \frac{\mathbf{e}\mathbf{e}^{\mathrm{T}}}{n}\right)^{-1} + \frac{\mathbf{e}\mathbf{e}^{\mathrm{T}}}{n}, \tag{3}$$

where n is the number of nodes in the network and $\mathbf{e}$ is a column vector made of 1s ($[111 \cdots 1]^{T}$) of size n.

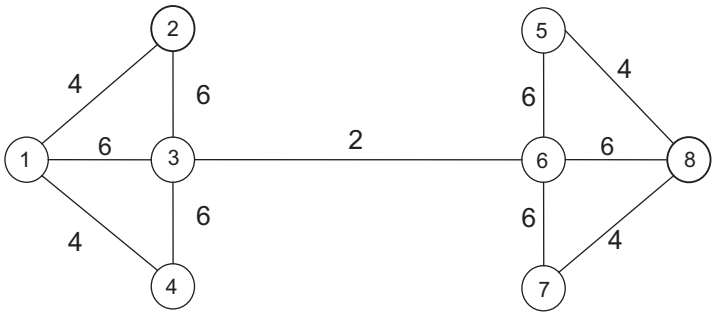
With this random-walk-based measure, distance between nodes gets shorter as more paths are added to the system (Yen et al., 2005). As a simple illustration, consider the case shown in Fig. 1, where the addition of a path between nodes 2 and 3 reduces the average first passage time m(3|1).

This measure is appealing for social networks, because people who belong to the same subgroup will be connected by a comparatively large number of short paths, whereas there will be fewer paths connecting people who are in different subgroups. For the same reason, random-walk-based clustering will not have a special tendency towards spherically shaped clusters or ones of similar sizes (Harel and Koren, 2001), thus the quality of the resulting clusters is expected to be better compared to Euclidean distance approaches.

4. Genetic representation

A GA is a computer simulation of the biological evolutionary processes and is often used as a search technique to find approximately optimum solutions to combinatorial optimization problems (Chatterjee et al., 1996). Network clustering is one such combinatorial problem that can be approached with GAs (Falkenauer, 1998). Problem representation in GAs may be done in several ways with vastly different efficiency outcomes. A representation, for example, may eliminate the barren parts of a search space, thus increasing the efficiency of the algorithm, or may introduce redundancy by searching the same points repeatedly, thereby decreasing the efficiency. Representation in GAs, like in many other domains, is a mixture of art and science.

One possible representation in the network clustering problem is defining an array of size n (n = the number of nodes in the network), and restraining the values of each array element to be between 1 to k, where k is the number of



$$\mathbf{N}=\begin{array}{c} \\ \\ 3 \\ \\ \\ \\ 7 \\ \\ \end{array}\begin{array}{c}\begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array}\\ \begin{bmatrix} 0 & 14.4 & 10 & 14.4 & 76.4 & 64 & 76.4 & 74 \\ 14.4 & 0 & 12.4 & 21.6 & 78.8 & 66.4 & 78.8 & 76.4 \\ 10 & 12.4 & 0 & 12.4 & 66.4 & 54 & 66.4 & 64 \\ 14.4 & 21.6 & 12.4 & 0 & 78.8 & 66.4 & 78.8 & 76.4 \\ 76.4 & 78.8 & 66.4 & 78.8 & 0 & 12.4 & 21.6 & 14.4 \\ 64 & 66.4 & 54 & 66.4 & 12.4 & 0 & 12.4 & 10 \\ 76.4 & 78.8 & 66.4 & 78.8 & 21.6 & 12.4 & 0 & 14.4 \\ 74 & 76.4 & 64 & 76.4 & 14.4 & 10 & 14.4 & 0 \end{bmatrix}\end{array}$$

Fig. 2. An example network and its random-walk-base distance matrix.

clusters. This representation does not eliminate any part of the search space, thus all possible solutions can potentially be searched. At the same time, it introduces redundancy and has been criticized for crippling the effect of the crossover operation because semantically identical parents may produce off-springs with little resemblance to their genetic encoding (Pattarin et al., 2004). A group-centric representation may alleviate these problems by creating k bins, and assigning a subset of the n nodes to each bin. With this approach, however, the crossover operation is problematic because genetic information can be lost during crossover or the same node may be assigned to two different clusters at the same time.

Another possible representation is *k-medoids* in which each cluster center is represented with one of the network nodes. These medoid nodes capture the nearest nodes into the clusters they represent. For this representation, an array of size k is defined and the value of each array element is restrained to be between 1 and n. For example, a possible gene corresponding to the network in Fig. 2 will be [3 7] for $k = 2$. Then each node is assigned to the nearest cluster based on its distance matrix **N** producing the allocation: [{1, 2, 3, 4}, {5, 6, 7, 8}].

The k-medoids representation incorporates a strong heuristic that the nodes in a cluster are likely to be in close proximity to a central node, which eliminates parts of the search space in which cluster allocations do not satisfy that heuristic. Because the genome length is very small, the search is faster compared with the other aforementioned cases. On the other hand, the unexplored part of the search space may in some cases potentially include a promising point, which is not taken into account. For this reason, we also considered an extension of this approach as explained below.

4.1. Medoid-based representation with exception bins

We consider an extension to the medoid-based representation by using *exception-bins* that contain nodes, which do not obey the allocation suggested by the medoids. These exception bins are appended to the end of the genome. For example, while [3 7] suggests an allocation of [{1, 2, 3, 4}, {5, 6, 7, 8}] in the purely medoids approach, [3 7{5, 6}{2}]

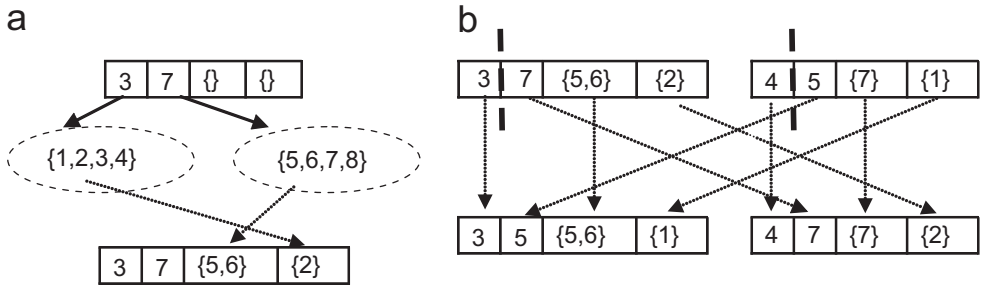


Fig. 3. Mutation and crossover in medoid-based representation with exception bins: (a) mutation; (b) crossover.

will correspond to an allocation of [{1, 3, 4, 5, 6}, {2, 7, 8}] in our modified representation because of the exceptions indicated by the exception bins. (The same representation can also be implemented with a gene of length $k + n$, where the first k nodes represent the medoids, and the remaining n nodes represent the exceptions.) As seen in the example, and illustrated in Fig. 3, these exception bins can have more than one element in it.

This new representation can be seen as a hybrid of the group-centric approach and the medoid-based one. At one extreme, when the exception bins are empty the representation is equivalent to that of a pure medoid based one, and at the other extreme when exception bins determine the complete allocation of nodes into clusters the representation is equivalent to that of a group-centric one.

With this representation, *crossover* operation is defined by interchanging genes across a randomly determined point as usual. Furthermore, exception bins move with the medoids they are attached to. For example, when genes [3 7 {5, 6}{2}] and [4 5 {7}{1}] crossover, they will produce two children: [3 5 {5, 6}{1}] and [4 7 {7}{2}]. If there is a conflict between the exceptions, a tie breaker is applied based on which exception applies first. For example, a gene like [3 7 {2}{2}] corresponds to [{1, 3, 4}, {2, 5, 6, 7, 8}] because exception one is not applicable (2 is already part of the first cluster), and the second exception is applied.

Mutation is the mode of exception creation to the regular allocation of the nodes to clusters based on proximity to the medoids. Before an allocation is made, a random number is generated and compared against the mutation rate. If mutation applies, then that allocation is not made to its cluster indicated by its distance, but to another cluster randomly determined. This exception is recorded by adding that point to the appropriate exception bin. Furthermore, mutation may also be used to randomly perturb the medoids, empty the exception bins, or any one of the bin members.

Various functions may be used as the *fitness function*, such as the inverse of the sum of the distances to the medoids, or the inverse of the sum of all pair-wise distances within a group. In our experiments, we minimized the sum of all pair-wise distances between nodes, primarily because this measure is invariant to the cluster medoid and only depends on the allocation of nodes to clusters. Identical allocations always return the same fitness value regardless of the medoid locations.

5. Experiments

In this section, we discuss our experiments to evaluate the performance of genetic clustering with a random-walk-based distance measure. All the algorithms described in this study have been implemented in Matlab, using its GAs and Statistics Toolboxes.

To evaluate our approach we primarily considered the following questions. First, how accurate are the clustering results, and how do they compare with clustering social networks using Euclidean distances? Second, how efficient is the approach, and what is its algorithmic complexity? To answer the first question, we needed to know the correct cluster allocations for a given social network data set. For the data sets we know of, however, these allocations are not known a priori, so testing our approach with those data sets does not allow us to answer the first question. For that reason, we chose to create our own data sets in a way that will allow us to know the cluster assignments.

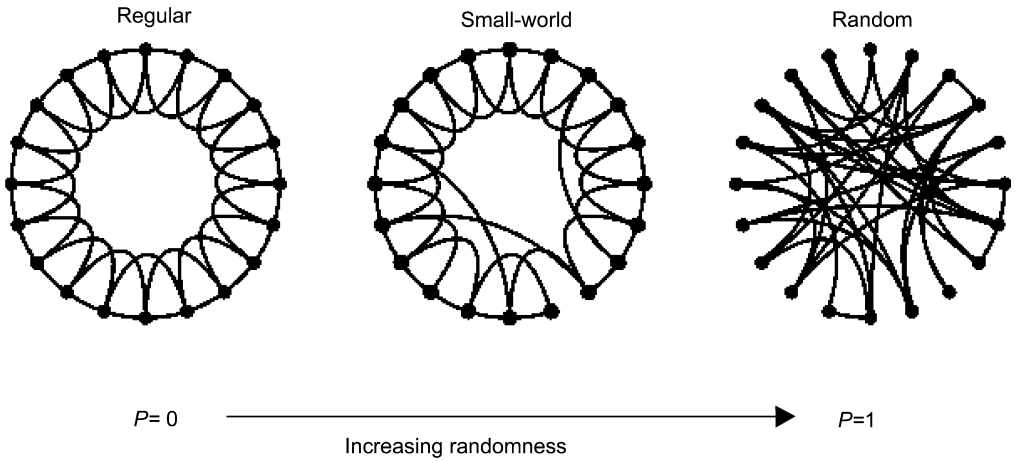


Fig. 4. Random rewiring procedure for interpolating between a regular ring lattice and a random network, without altering the number of vertices or edges in the graph. (Adapted by permission from Macmillan Publisher Ltd: Nature (Watts and Strogatz, 1998), copyright (1998)).

5.1. Synthetic data creation

We considered two approaches for creating synthetic social network data sets. In the first approach outlined by Watts and Strogatz (1998), small-world networks (popularly known as the networks with six degrees of separation) are seen to lie somewhere in between regular and completely random networks as illustrated in Fig. 4. To create synthetic small-world networks, an interpolation between regular and random networks is applied. The interpolation starts from a ring lattice with n vertices and m edges per vertex, and rewires the edges of a vertex with some probability p.

Similar to this idea, Tsvetovat and Carley (2005) proposes an algorithm to create social networks that mimics the scale-free small-world networks such as the ones observed in terrorist organizations. We have chosen to use this method because it has a clear annotation of the generated clusters, which is crucial for the task we want to accomplish. Using this method, k clusters with varying sizes are created for a given set of n nodes, and each node is randomly assigned to one of the clusters. The edges within each cluster are established with a probability p_i. Next, leader nodes are randomly selected for each cluster, and edges are established with probability p_e. Finally, a connectivity check is performed to make sure that all nodes are connected to each other.

5.2. Network clustering experiments

For our network clustering experiments, we first created data sets with varying sizes and clusters. One example of such a data set of 50 nodes and 6 clusters is shown in Fig. 5 after color annotation and layout optimization using the NV2D graph visualization and layout software (Nv2D, 2005).

From these networks, we obtained Euclidean distances by taking the inverse of the similarities between each node, and calculating the all-pairs shortest path matrix. We generated the random-walk-based distances for these data sets as explained in Section 3. Finally, we ran clustering experiments on both data sets using

(a) Genetic clustering using *k-medoids*.
(b) Genetic clustering using *k-medoids with exception bins* ($k\text{-}medoids^{+}$).
(c) Hierarchical clustering using *inner squared distance* (*h-ward).*
(d) Hierchical clustering using *nearest neighbor linkage (h-single)* .

In hierarchical clustering experiments we used a *cut-off at k-cluster level* in order to make meaningful comparisons.

Although, the experiments produce cluster allocations that can be compared with the known cluster allocations, counting the number of accurate allocations is not a trivial process. Labels of the clusters are arbitrary and associating output clusters with the actual clusters may not always be clear. For that reason, we devised a label independent counting

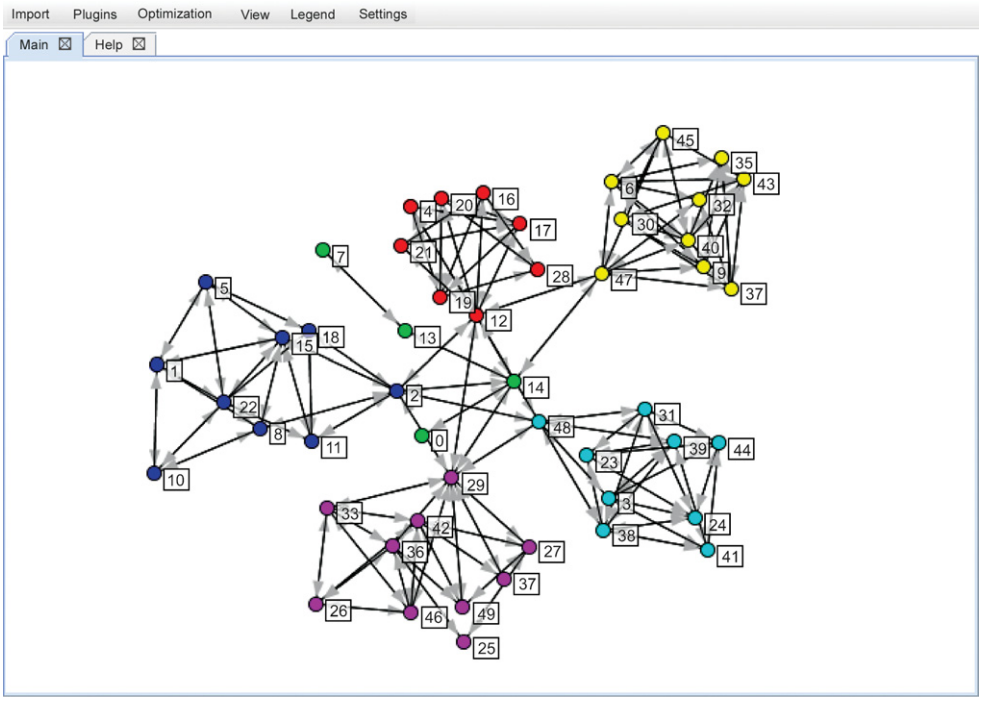


Fig. 5. Visualization of a synthetic social network data set of size 50.

process that works as follows: we compare the k output and actual clusters, and produce a matrix of size $k \times k$ that holds the number of intersections between output cluster i and actual cluster j. Then we choose cluster associations that maximize the total number of intersections.

In Table 1, we show the six number summary of clustering experiments done on 50 different generated data sets under five different settings. These results clearly show that random-walk-based distance measure, ACT, produces superior accuracy compared to Euclidean distances in social network clustering. Both genetic representations consistently performed better than the hierarchical clustering algorithms. Although we expected $k\text{-}medoids^{+}$ to perform better than the *k-medoids*, there was not any substantial difference between the two.

5.3. Spatial data experiments

In addition to social network experiments, we conducted some experiments to evaluate how well this method applies to clustering spatial multivariate data artificially modeled as a network. For our spatial data experiments, we transformed the well-known Fisher's Iris data, which has measurements on the sepal length, sepal width, petal length, and petal width of 150 iris specimens, 50 from each of three species.

For our experiments, we first created a network from the spatial data set with similarities defined between nodes. The typical approach is to find the k nearest neighbors of a point, and use the inverse of (or a similar function) the distance between its neighbors as the similarity measure. Because we need a connected graph to obtain a random-walk-based distance measure (the similarity matrix of a disconnected graph will be rank deficient), we also found the minimum spanning tree that connects all nodes together and added the edges to our artificial network. This intermediate network is based on the Euclidean distances, thus we call it the Euclidean distance network. Then, we obtained the random-walk-based network as explained in Section 3. We conducted genetic clustering experiments by varying the number of neighbors used in the modeling and compared the number of mistakenly clustered data points. As seen in Fig. 6 below, the random-walk-based network works well (with less than 10 misclassified points) on spatial data if the number of nearest neighbors is chosen small enough; e.g., about 25 for Fisher's data. The Euclidean distance network, on the other

Table 1
Six number summary of the minimum fitness values achieved from running the ACT and Euclidean distance based clustering done on 50 different generated data sets ($p_i = p_e = 0.1$)

Metric	Algorithm	Min.	1st Qu.	Median	Mean	3rd Qu.	Max
	Setting 1 ($n = 50, k = 7$)						
ACT	k-medoids	37	46	47	46.78	49	50
	$k-$ medoids$^+$	39	45	47	46.60	49	50
	h-ward	24	40	46	43.32	47	49
	h-single	9	13	15	17.28	19	45
Euclidean	k-medoids	21	23	25	25.70	28	34
	k-medoids$^+$	14	18	20	20.34	22	30
	h-ward	13	18	19.5	20.24	22	29
	h-single	14	16	19	19.18	22	27
	Setting 2 ($n = 100, k = 10$)						
ACT	k-medoids	70	91	93	92.46	95	100
	k-medoids$^+$	74	90	94	91.88	100	100
	h-ward	43	75	83.5	81.24	93	98
	h-single	15	18	20	22.14	24	64
Euclidean	k-medoids	35	47	50	50.28	54	64
	k-medoids$^+$	27	37	42	41.96	47	56
	h-ward	30	38	41.5	42.30	46	56
	h-single	23	29	32.5	34.24	39	51
	Setting 3 ($n = 150, k = 15$)						
ACT	k-medoids	99	134	137.5	133.44	140	145
	k-medoids$^+$	94	123	135.5	127.80	138	144
	h-ward	68	79	90.5	93.00	101	139
	h-single	17	20	22	22.56	25	35
Euclidean	k-medoids	39	53	58.5	59.20	66	78
	k-medoids$^+$	36	45	49	48.98	54	63
	h-ward	35	46	52	51.66	57	66
	h-single	29	36	42	41.92	47	59
	Setting 4 ($n = 200, k = 10$)						
ACT	k-medoids	156	170	182	179.68	188	197
	k-medoids$^+$	155	167	180	177.92	188	197
	h-ward	89	120.5	136	138.24	157	197
	h-single	39	90	112	112.60	145	182
Euclidean	k-medoids	76	111.75	127	126.08	140	169
	k-medoids$^+$	76	111	124	123.56	133.50	169
	h-ward	72	106	123	122.32	135.50	177
	h-single	52	89	101	110.36	124.25	175
	Setting 5 ($n = 250, k = 13$)						
ACT	k-medoids	158	204	233	221.44	239	245
	k-medoids$^+$	153	208	236	222.84	240	245
	h-ward	84	129	153	147.32	169.5	206
	h-single	29	54	84	82.44	104	172
Euclidean	k-medoids	99	111	133	132.48	148	184
	k-medoids$^+$	94	114	130	130.16	145	175
	h-ward	81	110	126	125.68	141	164
	h-single	60	84	103	103.48	123.5	144

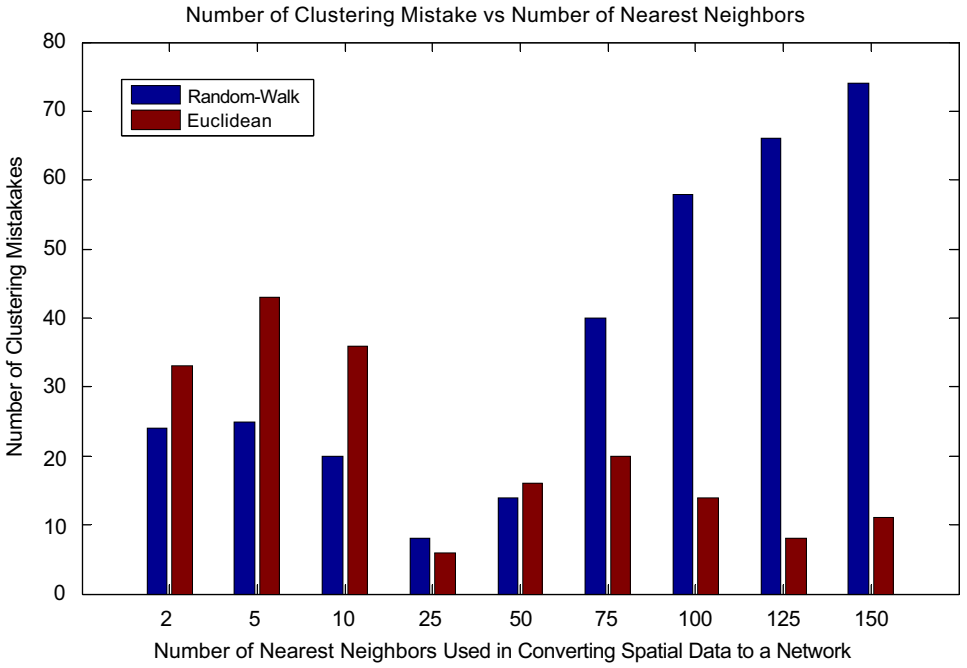


Fig. 6. Number of clustering mistakes Made with respect to number of nearest neighbors used: comparison between artificially created networks using random walk based distance vs. Euclidean distance.

hand, is much less sensitive to the number of neighbors, and works well if neighbors used are above 25. This is not too surprising in the light of Fig. 4, because with increasing number of connected neighbors the data set transforms into a random network and no longer has the characteristics of small-world networks.

5.4. Limitations

While our clustering results were exceptionally accurate using random-walk-based distances, there are some limitations of our work. First, given a network of n nodes with a $n \times n$ adjacency matrix A, construction of the random walk-based distance matrix has $O(n^3)$ time complexity for inverting matrix A. If matrix A is sparse, however, using recursive equations can be more efficient. This may limit the general applicability of using random-walk-based distances to very large social networks.

Second, our experiments produced excellent results when the number of clusters k was known, but we do not know a method of finding the right k when it is not given. This is, however, a more general problem associated with clustering, not specific to our approach, and requires further research.

6. Conclusions

In this study, we have examined the integration of genetic clustering with a promising distance measure based on random walks. Our experiments using genetic algorithms (GAs) in the domain of social networks confirms the reported success (Yen et al., 2005; Saerens and Fouss, 2004; Saerens et al., 2004) of this distance measure. Compared with the simple Euclidean approaches the clustering results are dramatically superior regardless of the specific clustering algorithm used. We also experimented with representing spatial data as a network, and clustering using random-walk-based distances. The clustering results were exceptionally good, when the conversion from spatial data to network representation was done with the optimum number of nearest neighbors. Finally, this study combines diverse parts of the literature including synthetic social network data creation, random-walk distance measures, and GAs, and thus offers an integrated view on clustering social networks using random-walk-based distances.

References

Barabási, A., 2005. Network Theory—the emergence of the creative enterprise. Science 308, 29.

Borgatti, S.P., Everett, M.G., 1997. Network analysis of 2-mode data. Social Networks 19, 243–269.

Chatterjee, S., Laudato, M., Lynch, L.A., 1996. Genetic algorithms and their statistical applications: an introduction. Comput. Statist. Data Anal. 22, 633–651.

Falkenauer, E., 1998. Genetic Algorithms and Grouping Problems. Wiley, Indianapolis, IN.

Freeman, L.C., 1993. Finding groups with a simple genetic algorithm. J. Math. Sociol. 17 (3), 227–241.

Freeman, L.C., 2003. Finding social groups: a meta-analysis of the southern women data. In: Ronald, B., Kathleen, C., Philippa, P. (Eds.), Dynamic Social Network Modeling and Analysis. National Academy Press, Washington, DC.

Fouss, F., Pirotte, A., Renders, J-M., Saerens, M., 2005. Markov-chain computation of similarities between nodes of a graph, with application to collaborative filtering. ⟨http://www.isys.ucl.ac.be/staff/marco/Publications/Saerens2005a.pdf⟩.

Hagen, L., Kahng, A., 1992. A new approach to effective circuit clustering. In: Proceedings of the IEEE/ACM International Conference on Computer-Aided Design, 422–427.

Harel, D., Koren, Y., 2001. On clustering using random walks. Lecture Notes in Compu. Sci. 2245, 8–41.

Jain, A.K., Murty, M.N., Flynn, P.J., 1999. Data clustering: a review. ACM Computing Surveys 31 (3), 264–323.

Milgram, S., 1967. The small world problem. Psychol. Today, 60–67.

Norris, J., 1997. Markov chains. Cambridge University Press, Cambridge.

Nv2D, 2005. Graph Visualization and Layout Software, ⟨http://web.mit.edu/bshi/Public/nv2d/⟩.

Pattarin, F., Paterlini, S., Minerva, T., 2004. Clustering financial time series: an application to mutual funds style analysis. Comput. Statist. Data Anal. 47, 353–372.

Saerens, M., Fouss, F., 2004. Computing Similarities Between Nodes of a Graph: Application to Collaborative Filtering, Submitted for publication.

Saerens, M., Fouss, F., Yen, L., Dupont, P., 2004. The principal components analysis of a graph, and its relationships to spectral clustering. In: Proceeding of the 15th European Conference on Machine Learning (ECML), pp. 371–383.

Tsvetovat, M., Carley, K.M., 2005. Generation of realistic social network datasets for testing of analysis and simulation tools. In: Carnegie Mellon University ISRI Technical Reports Series, CMU-ISRI-05-130.

Van Dongen, S.M., 2002. Graph clustering by flow simulation, Ph.D. Thesis, University of Utrecht, May.

Watts, D.J., Strogatz, S.H., 1998. Collective dynamics of 'small-world' networks. Nature 393, 440–442.

Yen, L., Vanvyve, D., Wouters, F., Fouss, F., Verleysen, M., Saerens, M., 2005. Clustering using a random walk based distance measure. In: Proceedings of the 13th Symposium on Artificial Neural Networks (ESANN 2005), pp. 317–324.