Finding Representative Nodes in Probabilistic Graphs

Laura Langohr and Hannu Toivonen

Department of Computer Science and
Helsinki Institute for Information Technology HIIT,
University of Helsinki, Finland
{laura.langohr,hannu.toivonen}@cs.helsinki.fi

Abstract. We introduce the problem of identifying representative nodes in probabilistic graphs, motivated by the need to produce different simple views to large BisoNets. We define a probabilistic similarity measure for nodes, and then apply clustering methods to find groups of nodes. Finally, a representative is output from each cluster. We report on experiments with real biomedical data, using both the k-medoids and hierarchical clustering methods in the clustering step. The results suggest that the clustering based approaches are capable of finding a representative set of nodes.

1 Introduction

Bisociative information networks (BisoNets) allow integration and analytical use of information from various sources [1]. However, information contained in large BisoNets is difficult to view and handle by users. The problem is obvious for BisoNets of hundreds of nodes, but the problems start already with dozens of nodes.

In this chapter, we propose identification of a few representative nodes as one approach to help users make sense of large BisoNets. As an example scenario of the approach, consider link discovery. Given a large number of predicted links, it would be useful to present only a small number of representative ones to the user. Or, representatives could be used to abstract a large set of nodes, e.g., all nodes fulfilling some user-specified criteria of relevance, into a smaller but representative sample.

Our motivation for this problem comes from genetics, where current high-throughput techniques allow simultaneous analysis of very large sets of genes or proteins. Often, these wet lab techniques identify numerous genes (or proteins, or other biological components) as potentially interesting, e.g., by the statistical significance of their expression, or association with a phenotype (e.g., disease). Finding representative genes among the potentially interesting ones would be useful in several ways. First, it can be used to remove redundancy, when several genes are closely related and showing all of them adds no value. Second, representatives might be helpful in identifying complementary or alternative components in biological mechanisms.

M.R. Berthold (Ed.): Bisociative Knowledge Discovery, LNAI 7250, pp. 218–229, 2012.

The BisoNet in our application is Biomine [2], an integrated network database currently consisting of about 1 million biological concepts and about 10 million links between them. Concepts include genes, proteins, biological processes, cellular components, molecular functions, phenotypes, articles, etc.; weighted links mostly describe their known relationships. The data originates from well known public databases such as Entrez[1], GO[2], and OMIM[3].

The problem thus is to identify few representative nodes among a set of them, in a given weighted network. The solutions proposed in this chapter are based on defining a probabilistic similarity measure for nodes, then using clustering to group nodes, and finally selecting a representative from each cluster.

In this framework, two design decisions need to be made: how to measure similarities or distances of nodes in a probabilistic network (Section 3), and which clustering method to use on the nodes (Section 4). Experimental results with real datasets are reported in Section 5, and we conclude in Section 6 with some notes about the results and future work.

2 Related Work

Representatives are used to reduce the number of objects in different applications. In an opposite direction to our work, clustering can be approximated by finding representative objects, clustering them, and assigning the remaining objects to the clusters of their representatives. Yan et al. [3] use k-means or RP trees to find representative points, Kaufman and Rousseeuw [4] k-medoids, and Ester et al. [5] the most central object of a data page.

Representatives are also used to reduce the number of datapoints in large databases, i.e., to eliminate irrelevant and redundant examples in databases to be tested by data mining algorithms. Riquelme et al. [6] use ordered projections to find representative patterns, Rozsypal and Kubat [7] genetic algorithms, and Pan et al. [8] measure the representativeness of a set with mutual information and relative entropy.

DeLucia and Obraczaka [9] as well as Liang et al. [10] use representative receivers to limit receiver feedback. Only representatives provide feedback and suppress feedback from the other group members. Representatives are found by utilizing positive and negative acknowledgments in such a way that each congested subtree is represented by one representative.

The cluster approximation and example reduction methods use clustering algorithms to find representatives, but are not applied on graphs. The feedback limitation methods again use graph structures, but not clustering to find representatives. Other applications like viral marketing [11], center-piece subgraphs [12], or PageRank [13] search for special node(s) in graphs, but not for representative nodes. The authors are not aware of approaches to find representatives by clustering nodes and utilizing the graph structure.

[1] www.ncbi.nlm.nih.gov/Entrez/
[2] www.geneontology.org/
[3] www.ncbi.nlm.nih.gov/omim/

3 Similarities in Probabilistic Graphs

Probabilistic graphs offer a simple yet powerful framework for modeling relationships in weighted networks. A probabilistic graph is simply a weighted graph $G = (V, E)$ where the weight associated with an edge $e \in E$ is probability $p(e)$ (or can be transformed to a probability). The interpretation is that edge e exists with probability $p(e)$, and conversely e does not exist, or is not true, with probability $1 - p(e)$. Edges are assumed mutually independent.

The probabilistic interpretation of edge weights $p(e)$ gives natural measures for indirect relationships between nodes. In this chapter we call these similarity measures, as is conventional in the context of clustering.

Probability of a Path. Given a path P consisting of edges $e_1, \ldots, e_k$, the probability $p(P)$ of the path is the product $p(e_1) \cdot \ldots \cdot p(e_k)$. This corresponds to the probability that the path exists, i.e., that all of its edges exist.

Probability of the Best Path. Given two nodes $u, v \in V$, a measure of their connectedness or similarity is the probability of the best path connecting them:

$$s(u, v) = \max_{P \text{ is a path from } u \text{ to } v} p(P).$$

Obviously, this is not necessarily the path with the least number of edges. This similarity function $s(\cdot)$ is our choice for finding representatives.

Network Reliability. Given two nodes s and t, an alternative measure of their connectivity is the probability that there exists at least one path (not necessarily the best one) between s and t. This measure is known as the (two-terminal) network reliability (see, e.g., [14]). A classical application of reliability is in communication networks, where each communication link (edge) may fail with some probability. The reliability then gives the probability that s and t can reach each other in the network.

Network reliability is potentially a more powerful measure of connectedness than the probability of the best path, since reliability uses more information — not only the best path. The reliability measure considers alternative paths between s and t as independent evidence for their connectivity, and in effect rewards for such parallelism, while penalizing long paths. The reliability is always at least as high as the probability of the best path, but can also be considerably higher.

However, computing the two-terminal network reliability has been shown to be NP-hard [15]. Fortunately, the probability can be estimated, for instance, by using a straightforward Monte Carlo approach: generate a large number of realizations of the random graph and count the relative frequency of graphs where a path from s to t exists. For very large graphs, we would first extract a smaller neighborhood of s and t, and perform the computation there. These techniques are described in more detail, e.g., by Sevon et al. [2] and Hintsanen and Toivonen [16]. Due to the complexity of computing the network reliability, we stick to the simpler definition of similarity $s(\cdot)$ as the probability of the best path.

4 Clustering and Representatives in Graphs

Our approach to finding representatives in networks is to cluster the given nodes, using the similarity measure defined above, and then select one representative from each cluster (Algorithm 1). The aim is to have representatives that are similar to the nodes they represent (i.e., to other members of the cluster), and also to have diverse representatives (from different clusters). In clustering, we experiment with two methods: k-medoids and hierarchical clustering. Both are well-known and widely used methods which can be applied to our problem of finding representatives; k-medoids is an obvious choice, since it directly produces representatives.

Algorithm 1. Find representative nodes

Input: Set S of nodes, graph G, number k of representatives
Output: k representative nodes from S
1: Find k clusters of nodes in S using similarities $s(\cdot)$ in graph G
2: For each of the k clusters, output its most central node (the node with the maximum similarity to other nodes in the cluster)

k-medoids. k-medoids is similar to the better known k-means method, but better suited for clustering nodes in a graph. Given k, the number of clusters to be constructed, the k-medoids method iteratively chooses cluster centers (medoids) and assigns all nodes to the cluster identified by the nearest medoid. The difference to the k-means clustering method is that instead of using the mean value of the objects within a cluster as cluster center, k-medoids uses the best object as a cluster center. This is a practical necessity when working with graphs, since there is no well defined mean for a set of nodes. The k-medoids method also immediately gives the representatives. The method is described in more detail, e.g., by Han and Kamber [17] and Kaufman and Rousseeuw [4].

For very large graphs, a straight forward implementation of k-medoids is not necessarily the most efficient. In our applications we use the Biomine database and tools to facilitate faster clustering. Given a set S of nodes, i.e., biological entities, to be clustered, and k, the number of clusters to be constructed, the method proceeds as follows. First, the Biomine system is queried for a graph G of at most 1000 nodes cross-connecting nodes in S as strongly as possible. The pairwise similarities between nodes are then calculated as the best path probabilities in G.

The Biomine system uses a heuristic to obtain G, details are omitted here. As the Biomine network consists of a million nodes, querying it for a graph exceeds by far the computational complexity of running k-medoids on the extracted graph. For brevity, we here omit discussion of the computational complexities of k-medoids and other approaches.

To start the actual clustering, k nodes from S are chosen randomly as initial medoids. Each remaining node in S is then clustered to the most similar medoid. If the pairwise similarity between a node and all medoids equals zero, the node

will be considered an outlier and is not assigned to any medoid in this iteration. Then, a new medoid is calculated for each cluster. The node that has a maximal product of similarities between each other node in the cluster and itself is chosen as the new medoid. The last two steps are then repeated until the clustering converges or the maximum number of iterations is reached.

Example. As an example, k-medoids was run with $k = 3$ and a set of nine genes. The genes belong to three known groups, each group of three genes being associated to the same phenotype. The three OMIM phenotypes used in the example are a pigmentation phenotype (MIM:227220), lactase persistence (MIM:223100), and Alzheimer disease (MIM: 104300).

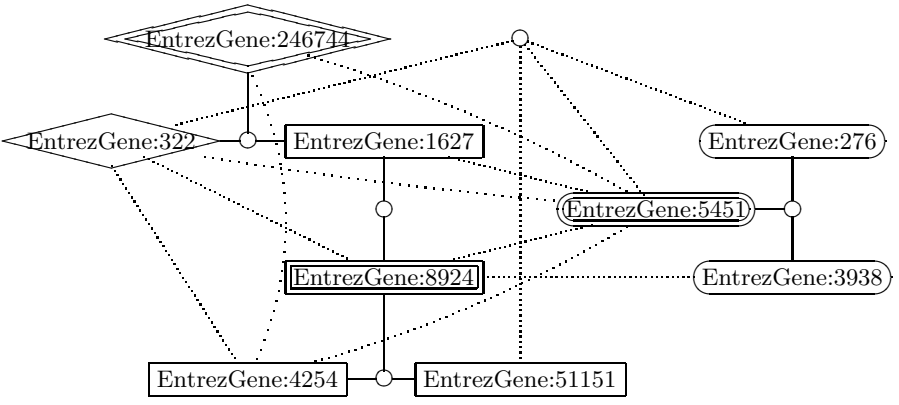

Fig. 1. Clusters (diamonds, boxes, ellipses) and representatives (double borders) of nine given nodes, and some connecting nodes (circles) on best paths between them. Lines represent edges between two nodes, dotted lines represent best paths with several nodes.

The algorithm converged in this case after two iterations. The result of the example run is shown in Figure 1. Looking at the quality of clustering, only one gene (EntrezGene:1627) was assigned to another cluster than it should with respect to the OMIM phenotypes. Apart from this gene, the clustering produced the expected partitioning: each gene was assigned to a cluster close to its corresponding phenotype. The three representatives (medoids) are genes assigned to different phenotypes. Hence, the medoids can be considered representative for the nine genes.

Hierarchical Clustering. As an alternative clustering method we use hierarchical clustering. Hierarchical clustering is a greedy clustering method that iteratively merges pairs of clusters. (Again, see, e.g., Han and Kamber [17] or Kaufman and Rousseeuw [4] for more information.) A possible problem with the k-medoids approach is that it may discover star-shaped clusters, where cluster members are connected mainly through the medoid. To give more weight on cluster coherence, we use the hierarchical clustering method with average linkage, as follows.

In the practical implementation, we again start by querying the Biomine system for a graph G of at most 1000 nodes connecting the given nodes S, and compute similarities of nodes in S as the probabilities of the best paths connecting them in G.

The hierarchical clustering proceeds in the standard, iterative manner, starting with having each node in a cluster of its own. In each iteration, those two clusters are merged that give the best merged cluster as a result, measured by the average similarity of nodes in the merged cluster. The clustering is finished when exactly k clusters remain.

After the clusters have been identified, we find the medoid in each cluster (as in the k-medoids method) and output them as representatives.

Random Selection of Representatives. For experimental evaluation, we also consider a method that selects representatives randomly. We again query the Biomine system for a graph G of at most 1000 nodes connecting the given nodes S, and compute similarities of nodes in S as the probabilities of the best paths connecting them in G.

We randomly select k medoids and cluster the remaining nodes of S to the most similar medoid. If the pairwise similarity between a node and all medoids equals zero, the node will be considered an outlier, as in k-medoids.

5 Experiments

Our goal in this section is to evaluate how successful the method is in finding representative nodes.

5.1 Test Setting

Test Data. We used data published by Köhler et al. [18], who defined 110 disease-gene families based on the OMIM database. The families contain three to 41 genes each; each family is related to one disease. Köhler et al. originally used the families in their experiments on candidate gene prioritization. Given a list of candidate genes they used a protein-protein interaction network to score the given genes by distance to all genes that are known to be related to a particular disease. Then they set up a ranking of the candidate genes based on their scores. Although their aim was different from ours, and the network they used was only a protein interaction network, the data sets also give a real test case for our problem.

Test Setting. In each test run, k gene families were randomly chosen as the nodes to find k representatives for. We performed 100 test runs for $k = 3$ and $k = 10$ of all three variants (k-medoids, hierarchical, random) of the method, and report averages over the 100 runs. As k-medoids is sensitive to the randomly selected first medoids, we applied k-medoids five times in each run and selected the best result. We applied the random selection of representatives 20 times in each run and used average values of the measures in order to compensate the random variation.

Measures of Representativeness. We use two measures of representativeness of the selected nodes. The first one is based on the similarity of nodes to their representatives, the second one on how well the k (known) families of nodes are covered by the k representatives.

The first measure is directly related to the objective of the k-medoids method. The idea is that each node is represented by its nearest representative, and we simply measure the average similarity of objects to their closest representative (ASR):

$$ASR = \frac{1}{|S| - K} \sum_{x \in S, x \neq m(x)} s(x, m(x))$$

where S is the set of given vertices, K is the number of clusters, $m(x)$ is the medoid most similar to x, and $s(x, m(x))$ denotes the similarity (probability of best path) between node x and medoid $m(x)$.

The second measure takes advantage of the known families of genes in our test setting. The rationale here is that a representation is better if it covers more families, i.e., contains a representative in more families. For this purpose, we calculate the fraction of non-represented classes (NRC):

$$NRC = \frac{1}{K} |\{k \mid \nexists j : m_j \in H_k,\ j = 1..K\}|,$$

where K is the number of classes and clusters (equal in our current test setting), m_j is the medoid of the jth cluster, and H_k is the kth original class.

For the k-medoids variant, we also report the number of outliers. Recall that the method outputs as outliers those nodes that are not connected (in the extracted subnetwork) to any medoid.

As additional characteristics of the methods we measure how good the underlying clusterings are. Again, we have two measures, one for the compactness of clusters, and one based on the known classification.

The first additional measure is the average compactness of clusters (ACC), where the compactness of a given cluster is defined as the minimum similarity of two objects in the cluster. The average is computed over clusters having at least two members:

$$ACC = \frac{1}{k'} \sum_{k=1}^{K} \min_{x,y \in C_k} s(x, y), \text{ where } k' = |\{k \mid |C_k| > 1, k = 1..K\}|,$$

i.e., k' is the number on non-trivial clusters. This measure is sensitive to outliers, and thus may favor the k-medoids variant.

The second additional measure compares the clustering to the known classes and measures their difference. We first identify the class best represented by each cluster, and then calculate how many objects were "wrongly assigned" (WAO):

$$WAO = \frac{1}{|S|} \sum_{k=1}^{K} \min_{k'=1..K} |C_k \backslash H_{k'}|.$$

Rand index could have been used here just as well.

5.2 Results

In terms of average similarity of nodes to their representative (ASR), the k-medoids method slightly but clearly outperforms the hierarchical method (Figure 2, left panels). The hierarchical method, in turn, is clearly superior to the random selection of representatives (Figure 2, right panels). For the k-medoids variant and $k = 3$, average similarities in the 100 test runs range from 0.3 to 0.8, and the total average is 0.51. For $k = 10$ the average is 0.55 and range is 0.4 to 0.8. For the hierarchical variant and $k = 3$, the average is 0.48 and range is 0.1 to 0.8. For $k = 10$ the average is 0.51 and range is 0.3 to 0.7. For the random variant and $k = 3$, average is 0.36 and range is 0.2 to 0.7. For $k = 10$ average is 0.43 and range is 0.3 to 0.6. These differences are no big surprise, since the k-medoids method more directly aims to maximize this measure than the hierarchical method, which however performs better than random choice of

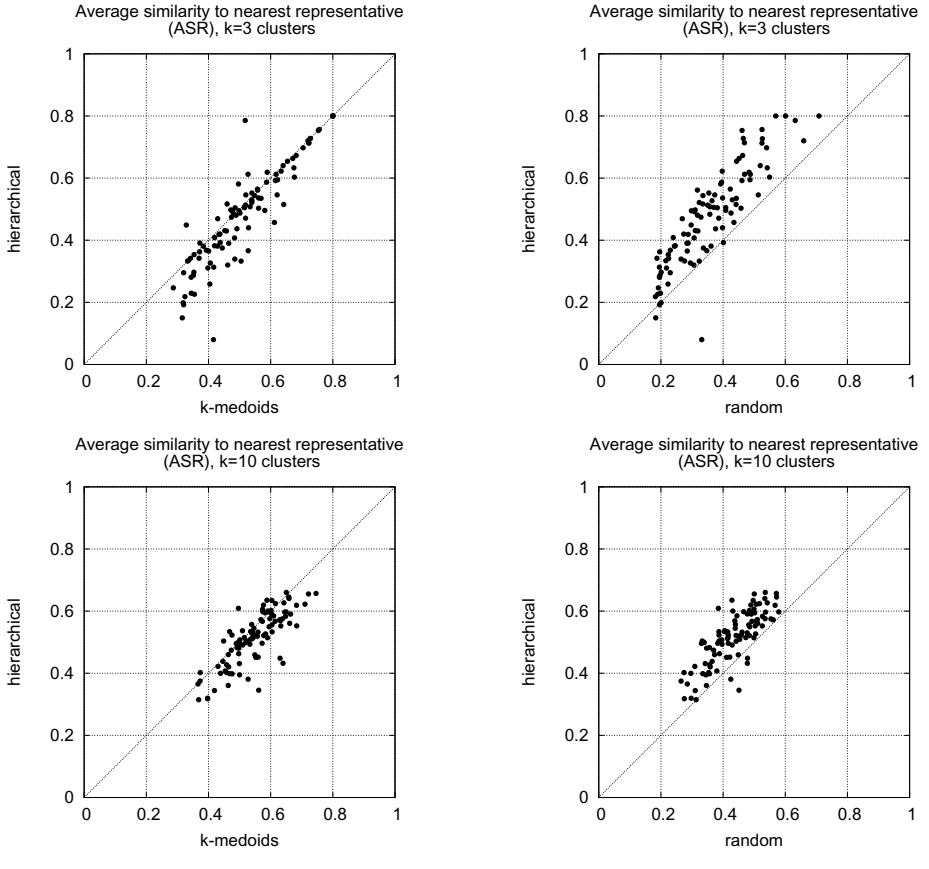

Fig. 2. Average similarity of objects to their nearest representative (ASR). In each panel 100 runs are visualized. Each point represents one run, thereby comparing ASR values of two variants (see x- and y-axis).

Table 1. Fraction of non-represented classes (NRC)

	k=3	k=10
k-medoids	14 %	29 %
hierarchical	16 %	21 %
random	34 %	39 %

representatives. Further, the k-medoids method may output some nodes as outliers. The average fraction of outliers in the experiments was 1.9 % for $k = 3$ and 4.5 % for $k = 10$.

The fraction of non-represented classes is a more neutral measure of performance since neither variant directly maximizes this. The results indicate that the k-medoids variant is slightly better with respect to this measure for $k = 3$

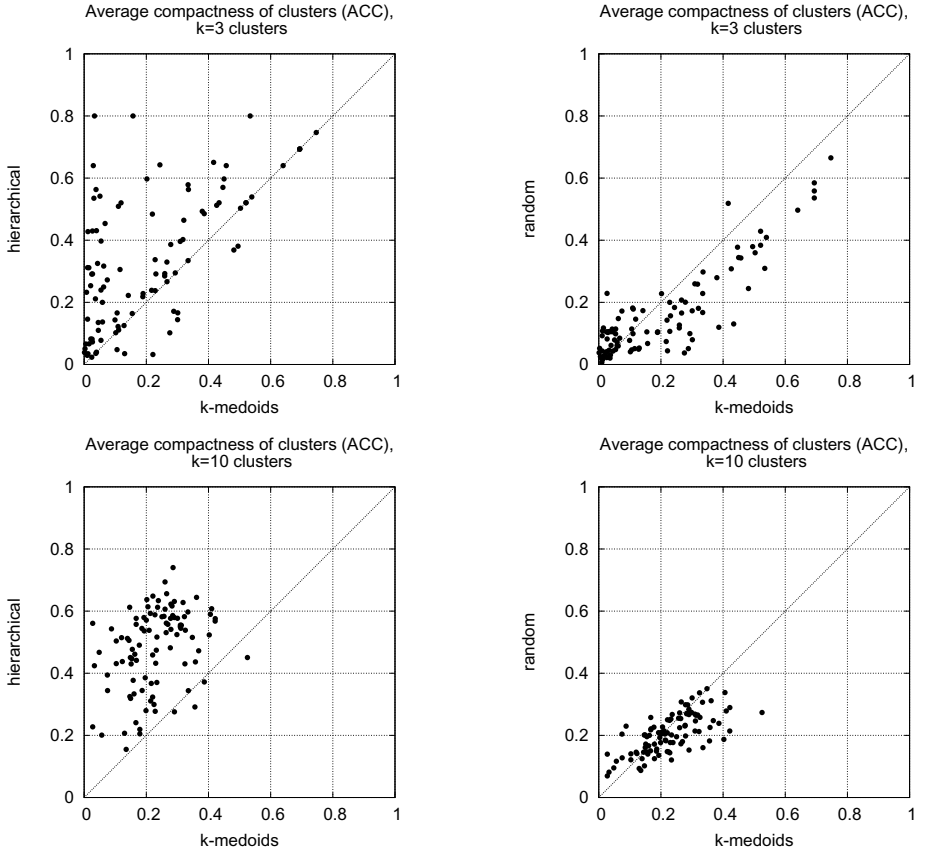

Fig. 3. Average compactness of nontrivial clusters (ACC). In each panel 100 runs are visualized. Each point represents one run, thereby comparing ACC values of two variants (see x- and y-axis).

(Table 1), but for $k = 10$ the hierarchical variant is clearly superior. Both methods clearly outperform the random selection of representatives.

To gain a better understanding of the performance of the methods, we look at the quality of clusterings produced. It is not surprising that clusters produced by the hierarchical method are on average more compact than those produced by the k-medoids method (Figure 3), as the hierarchical method more directly optimizes this measure. It is however somewhat surprising that k-medoids performs only slightly better than the random variant. The average compactness (minimum similarity within a cluster) is 0.20 ($k = 3$) and 0.23 ($k = 10$) for k-medoids, 0.33 ($k = 3$) and 0.48 ($k = 10$) for the hierarchical variant, and 0.16 ($k = 3$) and 0.21 ($k = 10$) for the random variant, with considerable spread and variance in all results.

In terms of wrongly assigned objects, the hierarchical variant clearly outperforms k-medoids (Table 2). The k-medoids variant outperforms the random selection of representatives, but for $k = 10$ only by a small difference.

Table 2. Wrongly assigned objects (WAO)

	k=3	k=10
k-medoids	18 %	44 %
hierarchical	15 %	25 %
random	27 %	46 %

6 Conclusions

We have described the problem of finding representative nodes in large probabilistic graphs. We based our definition of node similarity on a simple probabilistic interpretation of edge weights. We then gave a clustering-based method for identifying representatives, with two variants: one based on the k-medoids methods, one on the hierarchical clustering approach.

We performed a series of 100 experiments on real biomedical data, using published gene families [18] and the integrated Biomine network [2]. We measured the success of finding representatives with two measures: the similarity of nodes to their representatives, and the fraction of classes represented by the output.

In our experimental comparison, the k-medoids based variant and the hierarchical method are promising approaches. A look at the quality of the clusterings indicates that the success of the methods in identifying the underlying clusters depends on the measure used, and may also depend on the number of clusters to be constructed. According to the results, the hierarchical method is more robust, especially when looking for more than just a couple of representatives.

More work is needed to understand the reasons for the differences of the two approaches. Further, the problem of finding representative nodes needs to be validated in real applications. Based on the simple methods introduced here, and the initial experimental results, the clustering approach seems to be capable of reliably identifying a high quality set of representatives.

Acknowledgements. We would like to thank Lauri Eronen for his help with the test data and hierarchical clustering.

This work has been supported by the Algorithmic Data Analysis (Algodan) Centre of Excellence of the Academy of Finland and by the European Commission under the 7th Framework Programme FP7-ICT-2007-C FET-Open, contract no. BISON-211898.

References

1. Kötter, T., Berthold, M.R.: From Information Networks to Bisociative Information Networks. In: Berthold, M.R. (ed.) Bisociative Knowledge Discovery. LNCS (LNAI), vol. 7250, pp. 33–50. Springer, Heidelberg (2012)
2. Sevon, P., Eronen, L., Hintsanen, P., Kulovesi, K., Toivonen, H.: Link Discovery in Graphs Derived from Biological Databases. In: Leser, U., Naumann, F., Eckman, B. (eds.) DILS 2006. LNCS (LNBI), vol. 4075, pp. 35–49. Springer, Heidelberg (2006)
3. Yan, D., Huang, L., Jordan, M.I.: Fast approximate spectral clustering. In: 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2009), pp. 907–916. ACM, New York (2009)
4. Kaufman, L., Rousseeuw, P.: Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley Inc., New York (1990)
5. Ester, M., Kriegel, H.P., Xu, X.: Knowledge discovery in large spatial databases: focusing techniques for efficient class identification. In: 4th International Symposium on Advances in Spatial Databases (SDD 1995), pp. 67–82. Springer, London (1995)
6. Riquelme, J.C., Aguilar-Ruiz, J.S., Toro, M.: Finding representative patterns with ordered projections. Pattern Recognition 36(4), 1009–1018 (2003)
7. Rozsypal, A., Kubat, M.: Selecting representative examples and attributes by a genetic algorithm. Intelligent Data Analysis 7(4), 291–304 (2003)
8. Pan, F., Wang, W., Tung, A.K.H., Yang, J.: Finding representative set from massive data. In: The 5th IEEE International Conference on Data Mining (ICDM 2005), pp. 338–345. IEEE Computer Society, Washington, DC (2005)
9. DeLucia, D., Obraczka, K.: Multicast feedback suppression using representatives. In: 16th Annual Joint Conference of the IEEE Computer and Communications Societies. Driving the Information Revolution (INFOCOM 1997), pp. 463–470. IEEE Computer Society, Washington, DC (1997)
10. Liang, C., Hock, N.C., Liren, Z.: Selection of representatives for feedback suppression in reliable multicast protocols. Electronics Letters 37(1), 23–25 (2001)
11. Domingos, P., Richardson, M.: Mining the network value of customers. In: 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2001), pp. 57–66. ACM, New York (2001)
12. Tong, H., Faloutsos, C.: Center-piece subgraphs: problem definition and fast solutions. In: 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006), pp. 404–413. ACM, New York (2006)

13. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank citation ranking: bringing order to the web. Technical report, Stanford Digital Library Technologies Project (1999)
14. Colbourn, C.J.: The Combinatiorics of Network Reliability. Oxford University Press (1987)
15. Valiant, L.: The complexity of enumeration and reliability problems. SIAM Journal on Computing 8, 410–421 (1979)
16. Hintsanen, P., Toivonen, H.: Finding reliable subgraphs from large probabilistic graphs. Data Mining and Knowledge Discovery 17(1), 3–23 (2008)
17. Han, J., Kamber, M.: Data Mining. Concepts and Techniques, 2nd edn. Morgan Kaufmann (2006)
18. Köhler, S., Bauer, S., Horn, D., Robinson, P.: Walking the interactome for prioritization of candidate disease genes. American Journal of Human Genetics 82(4), 949–958 (2008)