Network measures for re-using problem information in EDAs

Roberto Santana, Concha Bielza, Pedro Larranaga
Departamento de Inteligencia Artificial, Universidad Politécnica de Madrid
28660 Boadilla del Monte, Madrid, Spain.
roberto.santana@upm.es, pedro.larranaga@fi.upm.es, mcbielza@fi.upm.es

June 30, 2010

Abstract

Probabilistic graphical models (PGMs) are used in estimation of distribution algorithms (EDAs) as a model of the search space. Graphical components of PGMs can be also analyzed as networks. In this paper we show that topological measures extracted from these networks capture characteristic information of the optimization problem. The measures can be also used to describe the EDA behavior. Using a simplified protein folding optimization problem, we show that the network information extracted from a set of problem instances can be effectively used to predict characteristics of similar instances.

1 Introduction

EDAs [9] are a class of optimization algorithms based on probabilistic modeling of the search space. These algorithms represent particular characteristics of high quality solutions using PGMs. Probabilistic modelling allows EDA to capture, represent, and use relevant interactions between the problem variables, increasing the efficiency of the search.

In many cases, the end user is not only interested in the solution of a given optimization problem, but also in reaching a better problem understanding. Usually, several runs of the EDA are done and their different outputs are constrasted. As a side product, these runs will produce a set of probabilistic models which store valuable information about the optimization problem. The models contain clues about the way in which the final solutions have been obtained. This information can in some cases be transformed into knowledge by the user. However, inspecting the models to detect characteristic patterns is not an easy task, being necessary a more automatic way to proceed.

Analysis of PGMs is not only applicable to the understanding of a single problem instance. Extracting and reusing problem information may have application in situations in which an optimization algorithm is expected to solve

several instances of the same class of problems. Commonly, these problem instances share some sort of (structural) similarity which would be beneficial to identify and exploit.

In this paper we treat two different but very related problems. Why type of information can be automatically extracted from the PGMs and how to employ this information? We approach these questions by analyzing the graphical structures of the learned PGMs as networks. Networks produced by EDAs are mined to extract a set of topological measures, that conveniently processed and fed to machine learning algorithms, are used to characterize similar optimization problems.

Automatic procedures for extracting and reusing information in the future of the solution of similar problems were presented in [6]. Two different approaches were introduced to extract the problem information. They were applied with good results to solve similar optimization problems. However, none of these approaches uses the problem information to infer, predict or characterize attributes of the related instances or the EDA's behavior for these intances. Using measures that contain information about the model structure and parameters [3] can be seen as a possible way to generalize structural modeling. However, these type of measures have not been yet applied to problem characterization or instance classification within EDAs.

We argue that the use of network measures computed from graphs representing problem structural information can serve as a basis for the application of transfer learning in optimization. Transfer learning [13] studies how the knowledge acquired while solving a given problem can be applied to solve different but related problems. Our contribution consists of adapting the results from network theory to the particular case of probabilistic graphical models used in EDAs and introducing network measures extracted from the PGMs learned by EDAs as a basis for transfer learning in optimization.

2 Estimation of distribution algorithms

Let X_i represent a discrete random variable. A possible value of X_i is denoted x_i. Similarly, we use $\mathbf{X} = (X_1, \ldots, X_n)$ to represent an n-dimensional random variable and $\mathbf{x} = (x_1, \ldots, x_n)$ to represent one of its possible values. We will work with positive probability distributions denoted by $p(\mathbf{x})$.

The type of probabilistic model, and the particular class of learning and sampling methods used are EDAs distinguished features. EDAs that use Bayesian networks (BNs) [4, 12] are among the most efficient algorithms able to represent higher order interactions. We use the estimation of Bayesian networks algorithm (EBNA) [4]. A pseudocode of EBNA is shown in Algorithm 1. The algorithm was implemented in Matlab using the MATEDA-2.0 software [14]. The scoring metric used by EBNA was the Bayesian metric with uniform priors, and each node was allowed to have a maximum number of 5 parents.

Algorithm 1: **EBNA**

1 Generate an initial population D_0 of individuals and evaluate them
2 t $\leftarrow 1$
3 **do** {
4 $D_{t-1}^{Se} \leftarrow$ Select N individuals from D_{t-1} using truncation selection
5 Using D_{t-1}^{Se} as the data set, apply local search to find one BN structure that optimizes the scoring metric
6 Calculate the parameters of the BN using D_{t-1}^{Se} as the data set
7 $D_t \leftarrow$ Sample M individuals from the BN and evaluate them
8 } **until** Stopping criterion is met

3 Analyzing graphs as networks

In recent years, results from graph theory have been developed and integrated into the modern theory of networks [1, 11]. Statistical network measures, that unveil the global structure of the network, and local measures, which serve to identify local patterns in the networks' topology are both useful tools to uncover and characterize the patterns of interactions in complex systems.

Most of the graphs used in EDAs (i.e. undirected, directed and weighted graphs) can be analyzed as networks. We conduct our analysis using the directed acyclic graphs (DAGs) learned in each generation of the EDA. We have computed several measures that serve to characterize these networks. A detailed description of these measures is beyond the space constraints of this paper and can be found in [5, 10, 11, 15, 16]. An account on the network measures used for our experiments follows.

1. *dagdif*: Number of different arcs between the DAGs learned at generations i and $i + 1$.
2. *Ndensity*: Connection density of the network, i.e. the number of connections present in the network out of all possible $(n^2 - n)$.
3. *indegree*: For a vertex, number of incoming arcs.
4. *outdegree*: For a vertex, number of outcoming arcs.
5. *betw. conn.*: Edge betweenness centrality. It is the fraction of all shortest paths in the network that traverse a given edge.
6. *pair dist.*: For a vertex, average distance to the rest of vertices. Disconnected vertices are assigned a very high, unattainable, distance value.
7. *reachability*: For a vertex, average reachability to the rest of vertices. The reachability value between vertices i and j is 1 if i is reachable from j, 0 otherwise.

8. *clust. coef.*: For a vertex, the clustering coefficient is the fraction of the existing number of vertex links to the total possible number of neighbor-neighbor links [16].

9. *shortcut prob.* The shortcut probability is the fraction of shortcuts in the graph [15]. Shortcuts are edges which significantly reduce the characteristic path length.

10. *n. motifs,* $M = 3$: Motif frequency for all motifs of size $M = 3$. A motif [11] is a connected graph consisting of M vertices and a set of edges with connectedness ensured forming a subgraph of a larger network. Its frequency is the number of times it appears in the network.

11. *n. motifs,* $M = 4$: Motif frequency for all motifs of size $M = 4$.

12. *max. modularity*: The maximum modularity gives a modularity value corresponding to a network module decomposition computed with Newman's spectral optimization method, generalized to directed networks [10]. A module is a densely connected subset of nodes that is only sparsely linked to the remaining network.

13. *vert. participation coef.*: The participation coefficient [5] defines how well distributed the links of a node are between different modules.

In the previous list, network measures 3, 4, 6, 7, and 8 are computed as the average of the local measures calculated for each vertex. Similarly, network measure 5 is the average of the measures computed for each edge. The rest of measures are global. In the learned DAGs, there are 4 different motifs ($M = 3$) and 24 different motifs ($M = 4$). Therefore, the total number of measures extracted from each graph is 39.

Our objective is to identify some properties in the networks generated by EDAs that support information about the problem being solved or serve as descriptors of EDAs behavior. In general, we would like that the analysis of the networks could serve to compare the difficulty of different problem instances and to extract problem information. The particular goal is to be able to predict problem features from the network measures derived from the DAGs learned by the EDAs.

We will start from a data set of characterized optimization problems. The problem characterization is given by a set of problem characteristics, e.g. the number of suboptima. We also have the previously described network measures computed from the DAGs generated by EBNA for each problem. The network measures of a subset of the problems are used to predict the characteristics of the rest of problems. This is a classical supervised classification problem. Classification accuracy is used as a measure of the informativeness of the used network descriptors. It also serves to evaluate the potentiality of our approach to reuse information extracted from the EDAs. We expect that our approach will enable us to find answers to the following general questions:

a) b)

Figure 1: (a): One possible configuration of sequence $HHHPHPPPPPH$ in the HP functional model. Hydrophobic proteins are represented by black beads and polar proteins, by white beads. There is one HH interaction (represented by a dotted line with wide spaces), one HP interaction (represented by a dashed line) and two PP interactions (represented by dotted lines) contacts. (b): Another possible configuration of the same sequence with a different pattern of interactions.

1. Can we predict the number of local optima the problem has?
2. Is it possible to determine whether the optimum has been found or not?
3. Can we identify the most similar and most different characterized problems with respect to a given uncharacterized problem?

4 Experiments

As problem benchmark we use a simplified protein model. The HP simplified protein model [2] is used in bioinformatics to investigate protein folding. In the HP model, a protein is considered a sequence of hydrophobic (H) and hydrophilic or polar (P) residues which are located in regular lattice models forming self-avoided paths. Figure 1 shows the graphical representations of two possible configurations for sequence $HHHPHPPPPPH$.

Interactions between neighbor residues (adjacent in the lattice but not connected in the sequence) contribute to the total energy of the HP lattice configuration. The energy values associated with the functional HP model [7] contain both attractive $\epsilon_{HH} = -2$ and repulsive interactions ($\epsilon_{PP} = 1$, $\epsilon_{HP} = 1$, and $\epsilon_{PH} = 1$). The HP problem consists of finding the solution (HP chain topological configuration) that minimizes the total energy. The energy that the functional model protein associates with the configuration shown in Figure 1a) is 1 because there is one HH interaction, one HP interaction and two PP interactions.

An HP protein configuration can be represented as a walk in the lattice (sequence of moves). In the sequence of moves, the two initial residues arelocated adjacent in the lattice. Each other residue is located to the left, to the righ, or forming a line with the previous two residues. For a given HP sequence and lattice, X_i will represent the relative move of residue i in relation to the previous two residues. Taking as a reference the location of the previous two residues in the lattice, X_i takes values in $\{0, 1, 2\}$. With respect to the location of the previous two residues, $x_i = 0$ means that residue i is located to left,

similarly $x_i = 1$ and $x_i = 2$ respectively mean that residue i will be located in line with the previous two residues and to their right. Values for X_1 and X_2 are meaningless, they are arbitrarily set to 0. This codification is called relative encoding [8]. The representations of configurations in Figure 1 a) and b) are $\mathbf{x}^i = (0, 0, 0, 2, 2, 0, 0, 2, 2, 0, 0)$ and $\mathbf{x}^j = (0, 0, 2, 2, 0, 1, 0, 2, 2, 0, 0)$, respectively.

Protein folds corresponding to proteins from the same family usually share common structural patterns. We expect that two similar HP sequences will have similar optimal lattice configurations. This fact explains the choice of this problem for the experiments.

4.1 Experimental framework

We use a data set[1] of 611 functional HP proteins corresponding to different sequences of 23 residues. These instances have a suitable characteristic. We know their optimal value, which is reached at a single configuration (disregarding symmetric representations like that shown in Figure 1 (b)). In addition, we know the closer suboptimal value and the number of configurations where this suboptimal value is reached. We use this information as a characterization of the problem. The optimal values of the 611 instances lie between -26 and -8. 374 instances have a number of suboptima in $\{1, \ldots, 4\}$ and the other 237 instances have a number of suboptima in $\{193, \ldots, 2532\}$.

To evaluate the EDA behavior and collect the networks, 30 independent runs of EBNA were run for each HP protein instance. For each instance, we computed how many times the optimum was found in the 30 experiments, the average generation at which it has been found and the average fitness of the best solutions found in all runs. For 310 of the 611 problems the optimum was found at least once. For 80 instances it was found only once and for 62 it was found 10 or more times. Most of the times the optimum is found, on average, between generations 10 and 15.

From each directed network corresponding to the structure (DAG) of the Bayesian network learned at each generation we compute the network descriptors[2] introduced in Section 3:

For the HP problem our general questions can be reformulated as follows:

1. Can we predict the number of local optimal for a given HP protein instance?

2. Can we predict whether EBNA has converged to the optimum value without knowing which the value of the optimum actually is?

3. Given a predefined similarity measure between instances, can we distinguish between the most similar and most different characterized HP instances to a given uncharacterized instance?

[1] This set is a subset of an original database introduced in [8].

[2] To compute them, we use the brain connectivity toolbox, available from http://sites.google.com/a/brain-connectivity-toolbox.net/bct/metrics

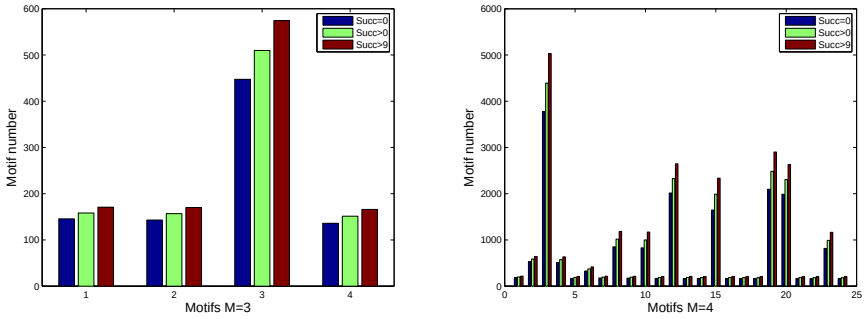

Figure 2: Motif frequencies computed from the networks of instances in which EBNA respectively has succesful rate 0 (blue), higher than 0 (green) and equal or higher than 9 (red).

We assume that prediction is done based on the networks learned from previous, characterized problems, and the networks obtained from the current, uncharacterized problem. Also, notice that the questions stated above address three distinct types of information about the problems: 1) Information about the problem characteristics. 2) Information about the algorithm behavior. 3) Information about the similarity between the problems.

The first problems considered are the determination of the algorithm convergence and the number of suboptima of the problem. For these two classification problems, we specify two classes. In the first case, classes are: 1A) Instances for which EBNA did not converge to the optimum in any of the 30 experiments. 1B) The rest of instances. For the second classification problem, classes are: 2A) Instances with 4 or fewer suboptima. 2B) The rest of instances, i.e. those with 193 or more suboptima. To get some clues about possible characteristic patterns associated to each of the classes, we computed and analyzed the average network descriptors from networks in each of the classes.

Figure 2 shows the motif frequencies for problems in classes 1A and 1B. In addition, we display information for a subset of instances of class 1B. This subset is comprised of instances where the EDA converged in 9 or more times from the 30 experiments. An initial observation is that the frequencies of all motif classes get higher for problems for which the EDA converges more often. A similar pattern is appreciated for the problem of classifying the number of suboptima (classes 2A and 2B) of the instance (data not shown), in which instances with a lower number of suboptima produce networks with a higher frequency of all types of motifs.

4.2 Numerical results

To evaluate predictors of the problem characteristics, we use a multivariate Gaussian classifier in which the conditional density of a solution given the class A_i is computed as

$$p(\mathbf{z}|A_i) = (2\pi)^{-\frac{n}{2}} |\Sigma_{A_i}|^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{z}-\mu_{A_i})^t {\Sigma_{A_i}}^{-1} (\mathbf{z}-\mu_{A_i})} \quad (1)$$

where $Z_i \in \{f_1, \ldots, f_m\}$, i.e. $\mathbf{Z}$ is a subset of components taken from the complete set of $m = 39$ features of the problem (network topological descriptors). A_i denotes a class of those described in the previous section, and μ_{A_i} and Σ_{A_i} are the parameters of a multivariate Gaussian distribution estimated from the points in class A_i. In the simplest case $|\mathbf{Z}| = 1$, i.e. only one network descriptor is used as predictor. In this case, equation (1) only involves univariate Gaussian distributions.

For a given set of features, we estimate the classifier accuracy using k-fold cross-validation with $k = 5$. The parameters of the multivariate Gaussians are learned using maximum likelihood estimation. To assign the classes, we use $p(A_i|\mathbf{z}) \propto p(A_i, \mathbf{z}) = p(\mathbf{z}|A_i)p(A_i)$ and assume all classes are a priori equiprobable. Therefore the assigned class is the one with highest $p(\mathbf{z}|A_i)$. The k-fold cross-validation procedure was repeated 50 times and from these experiments we computed the mean and standard deviation of the classifier accuracy.

For the first two classification problems, we independently computed the predicted accuracy given by each of the features. These results are shown in Table 1. For the sets of network motifs ($M = 3$ and $M = 4$), we only include in the table the accuracy corresponding to the network motif with the highest accuracy. It can be seen that the best accuracy is achieved by the betweenness connectivity in the first problem, and by the clustering coefficient in the second problem. Accuracies are higher for the second problem than for the first. It seems easier here to predict whether the problem has few or many suboptima than determining if the algorithm has converged to the optimum.

In order to improve the classification accuracy we consider interactions between the predictors. In this case, we search for a set of features that maximizes the classification accuracy. This feature subset selection problem, with 39 variables, is addressed using an EDA as implemented with MATEDA [14]. Only one run of the EDA was used to compute the best set of features, therefore solutions are likely to be improvable. The accuracies obtained with the best combination of features are shown in the last row of Table 1. For both problems, improvements over the best single classifiers were achieved. The classification accuracies of these sets of predictors are respectively above 70% and 90%.

We have empirically shown that the information learned during the optimization of past problems for which some particular features are known can be employed to predict characteristics of new problems for which we do not have the same kind of information.

In the next step, we intend to use the DAGs to distinguish, in a data set of characterized problems, similar from dissimilar problems (question 3). We use

		Convergence		Suboptima	
feature	*name*	*accuracy*	*std.dev.*	*accuracy*	*std.dev.*
1	*dagdif*	0.6023	0.0027	0.7601	0.0020
2	*Ndensity*	0.6635	0.0022	0.8841	0.0014
3	*indegree*	0.6637	0.0025	0.8838	0.0014
4	*outdegree*	0.6621	0.0031	0.8842	0.0018
5	*betw. conn.*	0.6789	0.0025	0.7323	0.0025
6	*pair dist.*	0.6151	0.0023	0.8593	0.0018
7	*reachability*	0.6137	0.0020	0.8581	0.0014
8	*clust. coef.*	0.6597	0.0026	0.8901	0.0017
9	*shortcut prob.*	0.6097	0.0043	0.6065	0.0068
10 : 13	*n. motifs, M=3*	0.6761	0.0025	0.8796	0.0024
14 : 37	*n. motifs, M=4*	0.6783	0.0022	0.8772	0.0016
38	*max. modularity*	0.6748	0.0034	0.7761	0.0020
39	*vert. mod. part.*	0.6376	0.0032	0.7875	0.0031
Best combination		0.7084	0.0065	0.9132	0.0035

Table 1: Classification accuracy and standard deviation for each single predictor and best combination for the EDA convergence to the optimum and for the prediction of the number of suboptima.

two different measures of similarity between instances. 1) The sequence similarity, which is the number of common residues in the two sequences and 2) The fitness correlation between problems, computed from 10000 random solutions.

To construct the database of cases, we identify, for each of the 611 instances, the most similar and most different instance in the set. Then for each pair of instances (i, j), we compute the difference $\mathbf{z}^i - \mathbf{z}^j$ between their corresponding network descriptors and associate the class value 1 if the pair is the most similar, or 0 if the pair is the most dissimilar. Notice that an instance may have more than one most similar or dissimilar matches. This is particularly the case for the sequence similarity measure. However, we select an arbitrary instance among those being closest (respectively most distant). As a result, for each similarity measure, there is a database of $611 \times 2 = 1222$ cases equally distributed between the two classes.

We use the same type of classifier and experimental protocols utilized in the previous classification experiments. Results are shown in Table 2. In this prediction problem the single classifiers have a more similar performance among each other. The best individual predictor when sequence similarity measure is used, is the reachability measure (feature 7). When the fitness correlation measure is used, the best predictor is the outdegree (feature 4). In general, single predictors do not provide a high accuracy. However, when interactions between features are considered, the accuracy in the prediction is much higher for both problems (an increase of 7% for the first problem and of 15% for the second). The main conclusion from the experiment is that information extracted from the networks can be used to distinguish similar and dissimilar pairs of instances.

		Based on seq. similarity		Based on fitness correlation	
feature	*name*	*accuracy*	*std.dev.*	*accuracy*	*std.dev.*
1	*dagdif*	0.5639	0.0039	0.5608	0.0052
2	*Ndensity*	0.6516	0.0019	0.6207	0.0035
3	*indegree*	0.6516	0.0021	0.6211	0.0036
4	*outdegree*	0.6514	0.0024	0.6683	0.0017
5	*betw. conn.*	0.6126	0.0031	0.6113	0.0033
6	*pair dist.*	0.6554	0.0021	0.6100	0.0037
7	*reachability.*	0.6558	0.0023	0.6457	0.0026
8	*clust. coef.*	0.6495	0.0026	0.6208	0.0030
9	*shortcut prob.*	0.5959	0.0025	0.6097	0.0039
10 : 13	*n. motifs, M=3*	0.6469	0.0026	0.6103	0.0030
14 : 37	*n. motifs, M=4*	0.6435	0.0024	0.6193	0.0027
38	*max. modularity*	0.6164	0.0028	0.6164	0.0030
39	*vert. part. coef.*	0.6056	0.0027	0.5822	0.0034
Best combination		0.7271	0.0041	0.8143	0.0043

Table 2: Classification accuracy and standard deviation of each single predictor and best combination for the prediction of the most similar and dissimilar pairs of instances.

5 Conclusions and future work

We have introduced a novel approach for re-using information in EDAs. It is based on the use of network measures computed from networks generated by EDAs and on the application of machine learning algorithms. We argue that the use of these measures could serve to devise "intelligent" optimization methods, able to learn from past experience to recognize and solve related problems.

References

[1] L. A. N. Amaral, A. Scala, M. Barthélémy, and H. E. Stanley. Classes of small-world networks. *Proceedings of the National Academy of Sciences (PNAS)*, 97(21):11149–11152, 2000.

[2] K. A. Dill. Theory for the folding and stability of globular proteins. *Biochemistry*, 24(6):1501–1509, 1985.

[3] C. Echegoyen, A. Mendiburu, R. Santana, and J. A. Lozano. A quantitative analysis of estimation of distribution algorithms based on Bayesian networks. Technical Report EHU-KZAA-IK-3, Department of Computer Science and Artificial Intelligence, University of the Basque Country, October 2009.

[4] R. Etxeberria and P. Larrañaga. Global optimization using Bayesian networks. In A. Ochoa, M. R. Soto, and R. Santana, editors, *Proceedings of the Second Symposium on Artificial Intelligence (CIMAF-99)*, pages 151–173, 1999.

[5] R. Guimera and L. A. N. Amaral. Functional cartography of complex metabolic networks. *Nature*, 433:895–900, 2005.

[6] M. Hauschild, M. Pelikan, K. Sastry, and D. E. Goldberg. Using previous models to bias structural learning in the hierarchical BOA. MEDAL Report No. 2008003, Missouri Estimation of Distribution Algorithms Laboratory (MEDAL), 2008.

[7] J. D. Hirst. The evolutionary landscape of functional model proteins. *Protein Engineering*, 12:721–726, 1999.

[8] N. Krasnogor, B. P. Blackburne, E. K. Burke, and J. D. Hirst. Algorithms for protein structure prediction. In *Parallel Problem Solving from Nature - PPSN VII*, volume 2439 of *Lecture Notes in Computer Science*, pages 769–778. Springer, 2002.

[9] P. Larrañaga and J. A. Lozano, editors. *Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation.* Kluwer Academic Publishers, Boston/Dordrecht/London, 2002.

[10] E. A. Leicht and M. E. J. Newman. Community structure in directed networks. *Physical Review Letters*, 100:118703, 2008.

[11] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298:824–827, 2002.

[12] M. Pelikan. *Hierarchical Bayesian Optimization Algorithm. Toward a New Generation of Evolutionary Algorithms*, volume 170 of *Studies in Fuzziness and Soft Computing.* Springer, 2005.

[13] R. Raina, A. Y. Ng, and D. Koller. Constructing informative priors using transfer learning. In *Proceedings of the 23rd International Conference on Machine Learning ICML-2006*, pages 713–720, New York, NY, USA, 2006. ACM Press.

[14] R. Santana, C. Bielza, P. Larrañaga, J. A. Lozano, C. Echegoyen, A. Mendiburu, R. Armañanzas, and S. Shakya. MATEDA: A Matlab package for the implementation and analysis of estimation of distribution algorithms. *Journal of Statistical Software*, 2010.

[15] O. Sporns. *Neuroscience Databases. A Practical Guide*, chapter Graph theory methods for the analysis of neural connectivity patterns, pages 171–186. Kluwer, 2002.

[16] D. J. Watts and S. Strogatz. Collective dynamics of small-world networks. *Nature*, 393(6684):440–442, 1998.