EPJ manuscript No.
(will be inserted by the editor)

Do greedy assortativity optimization algorithms produce good results?

Wynand Winterbach[1,2,a], Dick de Ridder[2], Huijuan Wang[1], Marcel Reinders[2], and Piet Van Mieghem[1]

[1] Network Architecture and Services, Delft University of Technology, Faculty of EEMCS, 2628 CD Delft, Netherlands
[2] The Delft Bioinformatics Lab, Delft University of Technology, Faculty of EEMCS, 2628 CD Delft, Netherlands

Received: date / Revised version: date

Abstract We consider algorithms for generating networks that are extremal with respect to degree assortativity. Networks with maximized and minimized assortativities have been studied by other authors. In these cases, networks are rewired whilst maintaining their degree vectors. Although rewiring can be used to create networks with high or low assortativities, it is not known how close the results are to the true maximum or minimum assortativities achievable by networks with the same degree vectors.
We introduce the first algorithm for computing a network with maximal or minimal assortativity on a given vector of valid node degrees. We compare the assortativity metrics of networks obtained by this algorithm to assortativity metrics of networks obtained by a greedy assortativity-maximization algorithm. The algorithms are applied to Erdős-Rényi networks, Barabási-Albert and a sample of real-world networks. We find that the number of rewirings considered by the greedy approach must scale with the number of links in order to ensure a good approximation.

1 Introduction

Networks play an ever-larger role in the analysis of various systems. Examples are biological systems, social networks and computer networks. Comparison of such networks is difficult since they vary in size (both in node and link counts) and link configurations. Topological metrics provide one way of comparing different networks by encoding their properties as scalars or vectors: two networks with similar metrics could be considered equivalent, depending on the context.

Degree distributions of networks are an often-used metric for characterizing networks. Such first-order descriptions are not always enough to describe the topology of networks. Thus, it may be necessary to consider second-order measures in addition to degree distributions. One such measure is Newman's degree assortativity [9] (a special case of assortative mixing [10]), a relatively new metric that measures the extent to which nodes with similar degrees are connected by links. The limits of this metric are not yet as well studied as those of other metrics. Extremal graph theory provides a framework for studying these limits. A typical approach in extremal studies is the generation of networks that are extremal with respect to the metric being studied. As an example, in Wang *et al.* [14], the maximum and minimum assortativities achievable by networks with binomial degree distributions are shown to vary greatly with the densities of the networks. This is a non-obvious result, illustrating that assortativity measures have to be considered relative to a given network structure. We consider two methods for obtaining networks with maximal degree assortativity subject to fixed degree vectors: a greedy algorithm based on link rewiring and an exact algorithm based on weighted b-matching.

[a] *e-mail:* w.winterbach@tudelft.nl

Watts and Strogatz [15] introduced link rewiring as a technique for generating random networks. During rewiring, a link is chosen at random and one of its end-points is replaced by a random node in the same network provided that no self-loops or duplicate links are introduced (that is, the network must remain simple). Due to the way that rewiring works, the node and link counts are invariant. Evans [3] and Lindquist *et al.* [5] exploited this property and studied rewiring as a mechanism for optimizing metrics subject to fixed node and link counts.

Degree-preserving rewiring is a restriction of link rewiring where a pair of links is chosen at random and a random end-point from the first link is exchanged for a random end-point from the second link. Maslov and Sneppen [6] introduced degree-preserving rewiring as a technique for generating null models. Their aim was to determine the likelihood of features observed in protein-protein interaction networks (relative to the null models). By requiring that degrees are preserved, the rewiring procedure is able to generate random networks that can be characterized by their degree sequences. The utility of this is evident from the fact that two of the most well-known classes of random networks are characterized by their degree distributions: Erdős-Rényi networks and Barabási-Albert networks.

Degree-preserving rewiring forms the basis of a simple technique for optimizing the degree-assortativity of a network (with a constant degree vector): a number of such rewiring steps are applied such that each rewiring increases/decreases the assortativity. This is essentially the approach taken by our greedy algorithm. Menche *et al.* [7] implemented a heuristic degree-preserving rewiring algorithm based on simulated annealing that used to produce networks with maximized and minimized assortativities, focusing on the class of scale free networks. However, as they did not have an exact algorithm, they could not compare the results of their heuristic algorithm to exact results.

In this paper, we consider the open question of how good a simple greedy assortativity maximization approach is. To this end, we present a novel exact algorithm for computing the maximum degree-preserved assortativity of a network. Using ensembles of Erdős-Rényi and Barabási-Albert networks as well as a number of real-world networks, we compare results from the greedy algorithm to those of the exact algorithm. We show that while a greedy rewiring process does not, in general, attain optimum assortativity, it achieves very good approximations.

2 Assortativity maximization algorithms

2.1 Exact algorithm

Van Mieghem *et al.* [13] have shown that the assortativity $\rho(G)$ of a network $G(\mathcal{N},\mathcal{L})$ with $N = |\mathcal{N}|$ nodes and $L = |\mathcal{L}|$ links can be expressed as

$$\rho(G) = 1 - \frac{\sum_{i\sim j}(d_i - d_j)^2}{\sum_{i=1}^{N} d_i^3 - \frac{1}{2L}\left(\sum_{i=1}^{N} d_i^2\right)^2} \tag{1}$$

$$= 1 - \frac{\sum_{i=1}^{N} d_i^3 - 2\sum_{i\sim j} d_i d_j}{\sum_{i=1}^{N} d_i^3 - \frac{1}{2L}\left(\sum_{i=1}^{N} d_i^2\right)^2}. \tag{2}$$

where d_i is the degree of the i-th node and $i \sim j$ means that node i and node j are joined by a link. Under degree-preserving rewiring, $\sum_{i\sim j} d_i d_j$ is the *only* variable part of the expression, attaining a maximum when the assortativity of G is maximized. Now consider the weighted complete network K_G whose nodes have the same labels $n_1, n_2, \ldots, n_N$ as G and in which the link $\{n_i, n_j\} \in \mathcal{L}(K_G)$ has weight $w(i,j) = d_i d_j$. Thus, G is an unweighted subnetwork of K_G. Let G_w be equal to G except that it has the same link weights as K_G (thus, G_w is simply a weighted subnetwork of K_G). The sum of the link weights in G_w is exactly $\sum_{i\sim j} d_i d_j = \sum_{i\sim j} w(i,j)$. Thus, $\sum_{i\sim j} d_i d_j$ can be maximized by finding the maximum weight subnetwork in K_G whose degree vector matches that of G.

2.1.1 Assortativity optimization with b-matching

The maximum weight subnetwork problem in this context can also be viewed as a *weighted b-matching problem* (see [11]): every node n_i in K_G is matched with exactly d_i other nodes in K_G such that the weights of the links spanned by the matched nodes is maximized. In the terminology of graph matching, the number of nodes to be matched with n_i (d_i in this context) is known as the *capacity* of n_i. If every node is matched with a number of nodes equalling its capacity, the match is said to be *perfect.* Weighted b-matchings can be efficiently computed using the algorithm of Miller and Pekny [8] which has a worst-case time complexity of
$\max\{O(NL\log(\max_i\{d_i\})), O(N^2L)\}$. Since the algorithm is always applied to the network K_G, $L = O(N^2)$, rendering the running time $O(N^4)$.

2.1.2 Converting b-matching to 1-matching problems

We were unable to find a usable implementation of Miller & Pekny's algorithm and the algorithm is difficult to implement correctly. Hence we took a simpler route due to Shiloach [12], wherein we transform K_G into a network $T(K_G) = M$ such that a perfect weighted 1-matching on M specifies a perfect weighted b-matching on K_G. In a perfect 1-matching, each node is matched to exactly one adjacent node.

Every node n_i in K_G is represented by a *set* of d_i nodes in M which we denote by $T(n_i) = \mathbf{n}_i$; the nodes in $\mathbf{n}_i$ are denoted by n_{ik} for $k \in \{1, 2, \ldots, d_i\}$. A link $l = \{n_i, n_j\}$ in K_G is represented by subgraph $T(l)$ in M. Instead of connecting the nodes of $\mathbf{n}_i$ directly to the nodes of $\mathbf{n}_j$, we represent the link in M by two connected nodes, n_{i0} and n_{j0}, which we call *bridge nodes.* Each node in $\mathbf{n}_i$ is connected to n_{i0} whilst each node in $\mathbf{n}_j$ is connected to n_{j0}. Each link in $T(l)$ is assigned the weight $T(w(i,j)) = \sum_{\{s,t\}\in\mathcal{L}(K_G)} w(s,t) + w(i,j)$; as explained below, the addition of $\sum_{\{s,t\}\in\mathcal{L}(K_G)} w(s,t)$ ensures that maximum weight 1-matchings are also perfect matchings. This scheme is demonstrated in Figure 1; the black links are the links of $T(l)$.

The nodes in a set $\mathbf{n_i}$ can only be matched to bridge nodes (such as n_{i0}), since these are the only nodes to which they have links. The node n_i has $N-1$ links in K_G. Each of these links is transformed to a subgraph $T(l)$. Therefore, there are $N-1$ bridge nodes with which the $d_i \le N-1$ nodes in $\mathbf{n_i}$ can be matched. Now, if a matching is to be perfect, one of the following two cases must hold for each $T(l)$:

- n_{i0} and n_{j0} are matched with each other, or
- n_{i0} is matched with a node in $\mathbf{n}_i$ and n_{j0} is matched with a node in $\mathbf{n}_j$.

The second case is equivalent to b-matching the nodes n_i and n_j in K_G. Each set $\mathbf{n}_i$ is connected to every other set $\mathbf{n}_j, j \ne i$, allowing the second case to apply to any pair of node sets in M (in other words, every node in K_G can be b-matched to every other node in K_G). It is possible to 1-match each node in every set $\mathbf{n}_i$ because $|\mathbf{n}_i| = d_i$ which is the degree of n_i in the original network G. If this were not true, it would imply that the degree sequence $\mathbf{d}$ of G does not support a valid pairing (matching) of

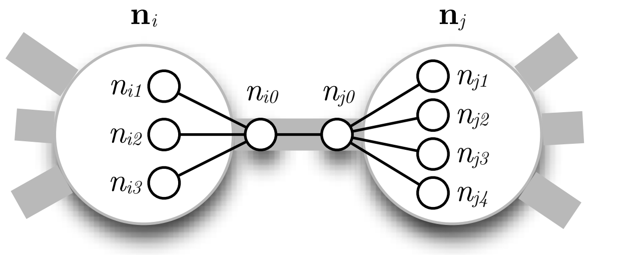


Figure 1: Representation of a link $\{n_i, n_j\} \in \mathcal{L}(K_G)$ in the network M.

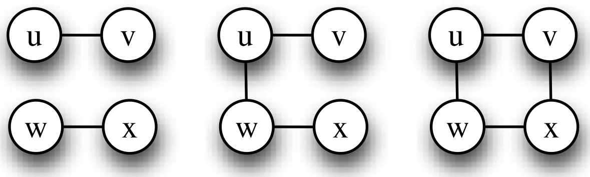


Figure 2: The only link configurations that permit link rewirings.

nodes into links such that the node degrees are equal to $\mathbf{d}$ – a contradiction. All nodes in M that are matched as in the first case correspond to nodes in K_G that are not b-matched. Thus, a perfect 1-matching on M is always possible and corresponds to a perfect b-matching on K_G.

If the cost for not matching a pair of nodes is smaller than the resulting gain, the 1-matching will not be perfect. By adding the constant $\sum_{\{s,t\}\in\mathcal{L}(K_G)} w(s,t)$ when transforming link weights, we ensure that cost for not matching exceeds any difference in weight that could have been gained. Thus in M, a matching is perfect if and only if it is a maximum matching.

We used Kolmogorov's [4] very fast $O(N^3)$ Blossom V 1-matcher. The network M has $N(N-1) + 2N(N-1) = O(N^2)$ nodes, resulting in a final running time of $O(N^6)$. As this grows very quickly with increasing N, we were limited in the sizes of the instances we could investigate.

2.2 Greedy algorithm

Like the exact assortativity maximization algorithm, the greedy algorithm modifies the topology of a given network in order to maximize the term $\sum_{i\sim j} d_i d_j$ in (2). As opposed to the exact algorithm which computes an entirely new link configuration, the greedy algorithm increases the term $\sum_{i\sim j} d_i d_j$ by rewiring pairs of links in a sequence of steps.

In an optimistic rewiring strategy, a pair of links $\{u,v\}$, $\{w,x\} \in \mathcal{L}(G)$ is selected such that u, v, w and x are distinct. If, in this *configuration* of four nodes, $\{u,x\} \notin \mathcal{L}(G)$ and $\{w,v\} \notin \mathcal{L}(G)$, $\{u,v\}$ and $\{w,x\}$ can be rewired to (that is, replaced by) $\{u,x\}$, $\{w,v\}$. Rewirable link pairs are always part of such four-node configurations (see Figure 2). Let d_u, d_v, d_w and d_x be the degrees of u, v, w, x in G. If $-d_u d_v - d_w d_x + d_u d_x + d_w d_v > 0$, the rewiring increases the term $\sum_{i\sim j} d_i d_j$ and therefore the change is made. Otherwise, the rewiring is rejected. There are eleven non-isomorphic four-node configurations of which only three – those in Figure 2 – permit pair-wise link rewiring. Inspection reveals that the symmetry of the first and last of these configurations allow for two possible rewirings, whereas the middle configuration allows only for one rewiring.

The greedy algorithm searches the input network for the configurations in Figure 2 whose links can be rewired to increase the assortativity. In each iteration of the algorithm, a random assortativity-increasing configuration is selected to ensure that different invocations of the greedy algorithm can sample different parts of the rewiring space. A simple way to facilitate this selection is to maintain a set R of rewirable link pairs from which selections can be made (R is in fact a network with links from the input network as its nodes; the links in R correspond to rewirable link pairs in the input network). After a pair of links $\{u,v\}$, $\{w,x\}$ is rewired, all rewirable configurations containing at least two nodes in $\{u,v,w,x\}$ have to be re-evaluated for rewirability. Those that are no longer rewirable are removed from R whilst those that become rewirable are added to R. The nodes of a rewirable link pair in R induce one of the rewirable configurations in Figure 2. The reason for focusing on rewirable link pairs as opposed to rewirable configurations, is that the first and last of the rewirable configurations in Figure 2 may be rewired in two ways and it is easier to consider each of the two rewirings as a separate element in the set R.

Explicitly maintaining R is expensive, at least initially (before any rewirings) when it may be that $|R| = O(N^4)$. However, when $|R|$ is large, keeping track of R is unnecessary as there is a good chance of finding rewirable link pairs when randomly sampling links from the network. Since not every random sampling will yield a rewirable link pair, sampling is repeated up to a pre-specified number of times s; if a valid rewiring is found, it is applied and the algorithm starts with a new iteration. As the greedy algorithm progresses, the number of rewirable link pairs $|R|$ decreases, rendering it less and less likely for a randomly sampled pair of links to be rewirable. Eventually, s random samplings will fail to discover rewirable link pairs.

At this point, R can be constructed explicitly, since $|R|$ should be small enough. From this point onwards, all link pairs are sampled from R and the algorithm proceeds until $|R| = 0$. The algorithm naturally decomposes into two states. In the first state, links are sampled at random from the input network; in the second, the set R is constructed and links are subsequently sampled from R. We refer to the first state as the *random selection* state and the second as the *exhaustive* state (since it continues until no more assortativity increasing configurations exist). Note that although $|R|$ may be small, constructing R requires $O(L^2)$ time, as all link pairs have to be enumerated.

The execution time on a large network is considerable and therefore such an exhaustive state is impractical for real-world assortativity-maximization algorithms.

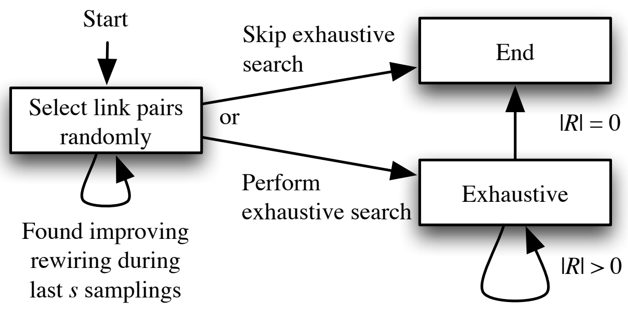

Start
Skip exhaustive search
End
Select link pairs randomly
or
|R| = 0
Perform exhaustive search
Exhaustive
Found improving rewiring during last s samplings
|R| > 0

Figure 3: State diagram for the greedy assortativity maximization algorithm.

Our motivation for including it was to study whether algorithms without exhaustive states might miss good, difficult to find solutions. The exhaustive step is optional in our greedy algorithm, allowing exhaustive and non-exhaustive results to be compared.

Combining all of this leads to the state diagram in Figure 3. When the exhaustive state is skipped, the greedy algorithm is a simple optimization algorithm whose results are unlikely to best those of more sophisticated algorithms, such as the algorithm of Menche *et al.* [7]. When the exhaustive state is engaged, our algorithm has the opportunity to find rewirings that will be missed by algorithms based on random link pair selection.

3 Approach setup

3.1 Data sets

We investigate ensembles of Erdős-Rényi and Barabási-Albert networks, as well as a number of real-world networks. Erdős-Rényi networks [2] are a 2-parameter family of random networks denoted $G_p(N)$. The parameter N is the number of nodes in the network whilst the parameter p is the probability that a pair of nodes are connected by a link. We considered networks of size $N \in \{25, 50, 80, 100, 150, 200\}$ and $p \in [0.05, 0.95]$. We also considered networks of size $N \in \{250, 300, 350, 400, 450, 500\}$ for $p = 0.05$; we were forced to limit p due to the excessive computation time required for larger p.

Barabási-Albert networks [1] are a 2-parameter family of random scale-free networks. As before, the parameter N denotes the number of nodes in the network. The parameter m represents the degree of nodes added in the growth process (Barabási-Albert networks are grown one node at a time). For these networks, we considered instances with $N \in [25, 1000]$ (including most values of N for which the Erdős-Rényi experiments were computed) and $m \in \{2, 3, 4\}$.

Random network ensembles were constructed for each pair of parameters: $\{N, p\}$ for Erdős-Rényi networks and $\{N, m\}$ for Barabási-Albert networks. With the exception of a few cases, at least 10^4 ensemble instances were generated for each parameter pair. Only 10^3 Erdős-Rényi networks with $N = 200$ and $p > 0.1$ were generated due to the long running times required on these networks.

The real-world networks that we considered come from a number of different domains and include protein-protein interaction networks, software call graphs, food webs, telecommunications networks and electronic circuits.

3.2 Algorithm setup

The greedy algorithm was executed in both its exhaustive and non-exhaustive modes. In the non-exhaustive mode, we considered various upper bounds to the number of random samplings: $s \in \{100, 1000, 10000, 100000\}$. In the exhaustive mode, $s = 100000$ random samplings were allowed before the algorithm switched to the exhaustive state.

3.3 Measured data

We considered the means and standard deviations of the differences between the assortativities as computed by the exact and greedy algorithms for each network instance (in a given ensemble of networks). A simple approach is to consider $\mathrm{E}[\rho - \rho']$ and $\mathrm{Std}[\rho - \rho']$. Here, ρ is a random variable representing the maximum assortativity of an ensemble of networks as computed by the exact algorithm. Similarly, ρ' is a random variable representing the maximum assortativity of the ensemble as computed by the greedy algorithm. Wang *et al.* [14] show that the range of degree assortativities achievable by networks with binomial degree distributions (which include Erdős-Rényi networks) vary greatly with their density and can often be much smaller than the possible assortativity range of $[-1, 1]$. In particular, as the density increases, the range shrinks. This variation in ranges skews the results, as the absolute differences may appear to be small whilst they are in fact large relative to the attainable assortativity range. To account for this, we normalize the mean and standard deviation by dividing the differences by $\rho - \rho_0$. Here, ρ_0 represents the assortativities of the original, unmodified networks. The normalized mean and standard deviation are given as $\mathrm{E}[(\rho - \rho')/(\rho - \rho_0)]$ and $\mathrm{Std}[(\rho - \rho')/(\rho - \rho_0)]$ respectively.

4 Results

4.1 Erdős-Rényi networks

4.1.1 Results as functions of N

First, we consider how the performance of the greedy algorithm changes as node counts increase. The normalized mean differences between the exact and greedy algorithms are shown in Figure 4 as functions of N for a few representative values of p. Instead of showing error bars, the

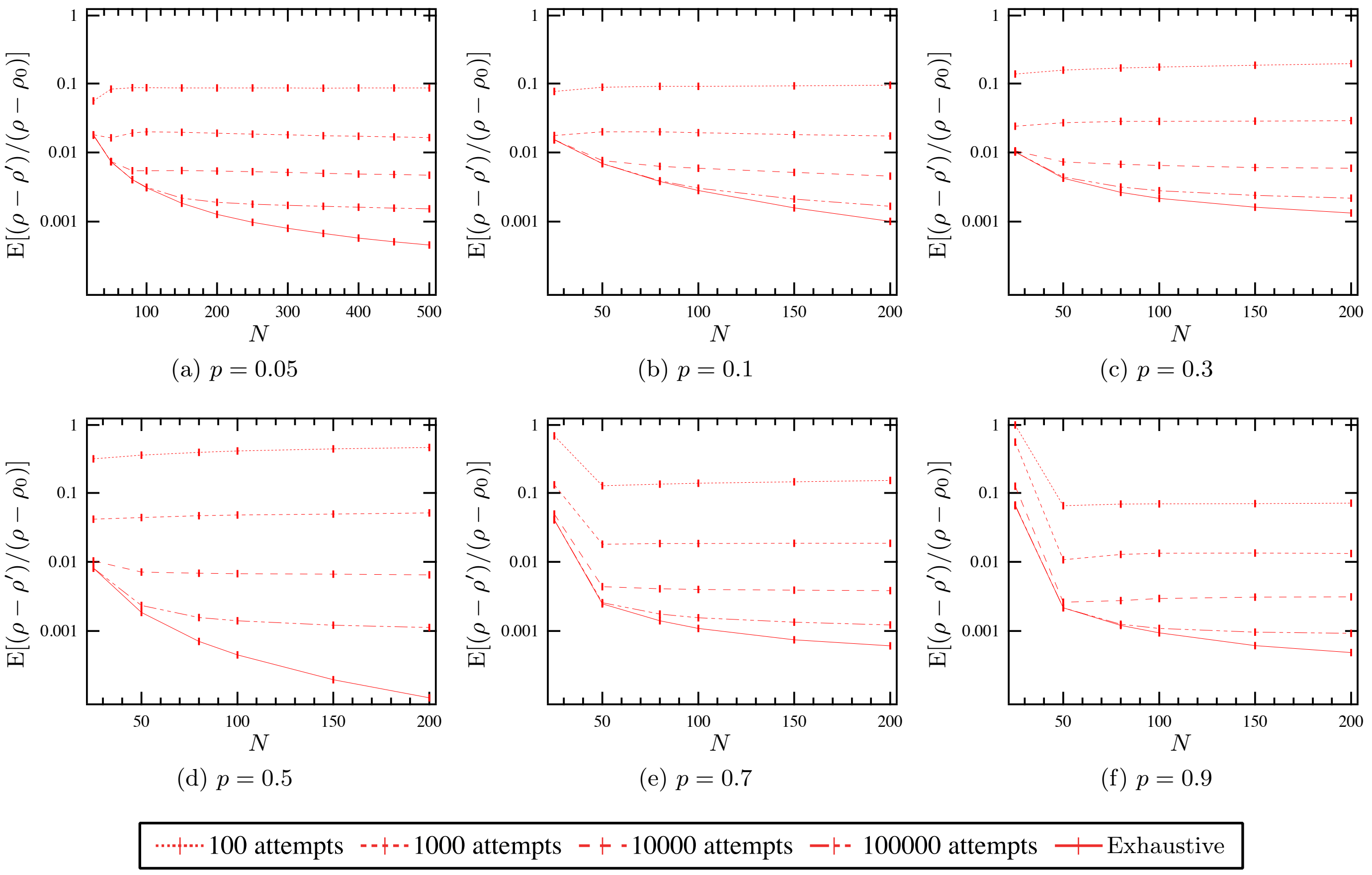


Figure 4: Means of relative differences in solutions obtained by the exact and greedy algorithms for various values of N and p. These plots apply to Erdős-Rényi networks. Each plot corresponds to a fixed p.

standard deviations corresponding to the means in Figure 4 are shown in Figure 5.

These plots paint a favorable picture for the greedy approach, as it performs well even when the number of random samplings s is small. The downward slopes corresponding to some of the non-exhaustive results seem to suggest that they improve as N increases. However, the Barabási-Albert, sparse Erdős-Rényi (Section 4.2) and real-world (Section 4.3) results show increasing trends. But in all cases, the exhaustive greedy algorithm tends towards the exact algorithm as N increases. The results also suggest that if s is chosen to be sufficiently large (for a given N), the non-exhaustive greedy algorithm can approach the exact algorithm.

The plots in Figure 5 suggest that the deviation from the mean tends to shrink with increasing N, regardless of the number of samplings s performed by the greedy algorithm. It seems reasonable to assume that these will all asymptotically approach 0.

4.1.2 Results as functions of p

In Section 4.1.1, we considered the performance of the greedy algorithm in terms of node counts. Here, we consider the performance relative to network density. The normalized differences between the exact and greedy algorithms are shown in Figure 6. Starting with $N \geq 50$, there are peaks and dips around $p = 0.5$. When the number of random selection trials s is small, the greedy results display peaks, whilst when s is large the results display dips. The same trends can be seen in the associated variance plots (which can be found in the supplementary material).

A partial explanation for why this happens lies in the number of rewirable configurations available in networks with $p = 0.5$ and in the probability of finding a rewirable link pair during random link selection. Consider the rewirable configurations in Figure 2. The left-most configuration depends on the links $\{u, v\}$ and $\{w, x\}$ being present, which has a probability of p^2, and on the other links being absent, which has a probability of $(1-p)^4$. This configuration can also arise when the links are replaced by $\{u, w\}$ and $\{v, x\}$ or by $\{u, x\}$ and $\{v, w\}$. Thus, the probability of finding a rewirable configuration with two links is $3(1 - p)^4 p^2$. By the same reasoning, one deduces that the probability of finding the rightmost configuration is $3(1 - p)^2 p^4$. The middle configuration has three links present, three links absent and can be constructed in *twelve* ways, rendering the probability of its appearing

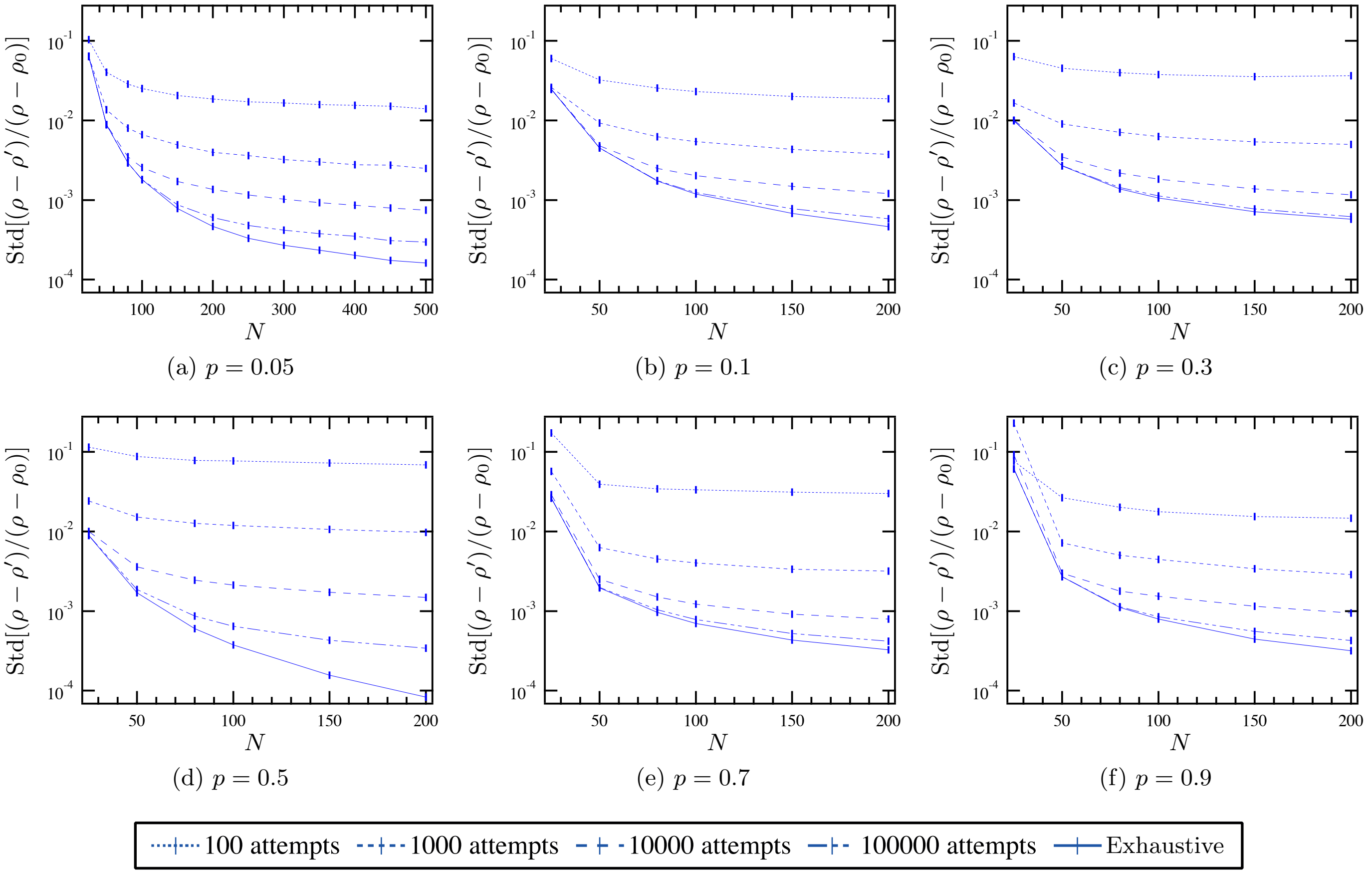


Figure 5: Standard deviations of relative differences in solutions obtained by the exact and greedy algorithms for various values of N and p. These plots apply to Erdős-Rényi networks. Each plot corresponds to a fixed p.

as $12(1-p)^3p^3$. There are $\binom{N}{4}$ possible configurations in a network on N nodes. In an Erdős-Rényi network, the probability of finding a rewirable configuration on four randomly chosen nodes is independent of the probability of finding a rewirable configuration on another set of four nodes (even if the sets overlap). Thus, the expected number $R_p(N)$ of rewirable configurations in an Erdős-Rényi network $G_p(N)$ is:

$$R_p(N) = \binom{N}{4}\left(3(1-p)^2p^4 + 12(1-p)^3p^3 + 3(1-p)^4p^2\right). \tag{3}$$

For all N, $R_p(N)$ attains a maximum at $p = 0.5$. Thus, an algorithm that is able to find all possible rewirings has ample opportunity for maximizing the assortativity and is less penalized for bad rewiring choices early in the rewiring process. As rewiring proceeds, the number of rewirable configurations decreases (non-linearly) and the probability of finding such rewirable configurations decreases to the point where the non-exhaustive greedy algorithm will fail to find them. Thus, while there may be many rewirable configurations, they are greatly outnumbered by the total number of link pairs.

Some caveats apply to $R_p(N)$. First, it is a mean-field approximation to the number of rewirable configurations. Second, the expression is not valid for networks that have been rewired (since these networks are no longer Erdős-Rényi networks). However, numerical simulations show that when $p = 0.5$, the number of rewirable configurations is indeed maximized (data not shown).

4.2 Barabási-Albert networks

To ensure that the results observed for Erdős-Rényi networks are not merely accidental, we also considered Barabási-Albert networks. It is difficult to fairly compare results on these two network classes, as link counts in Erdős-Rényi networks with fixed p grow by $O(N^2)$ whereas link counts in Barabási-Albert networks grow by $Nm + C = O(N)$ (where C is a small integer whose value depends on the link count of the seed network used to start generation of Barabási-Albert networks). By letting $p = (mN + C)/\binom{N}{2}$ one obtains Erdős-Rényi networks that have (on average) the same number of links as Barabási-Albert networks (constructed using the same N and m). The sparsity

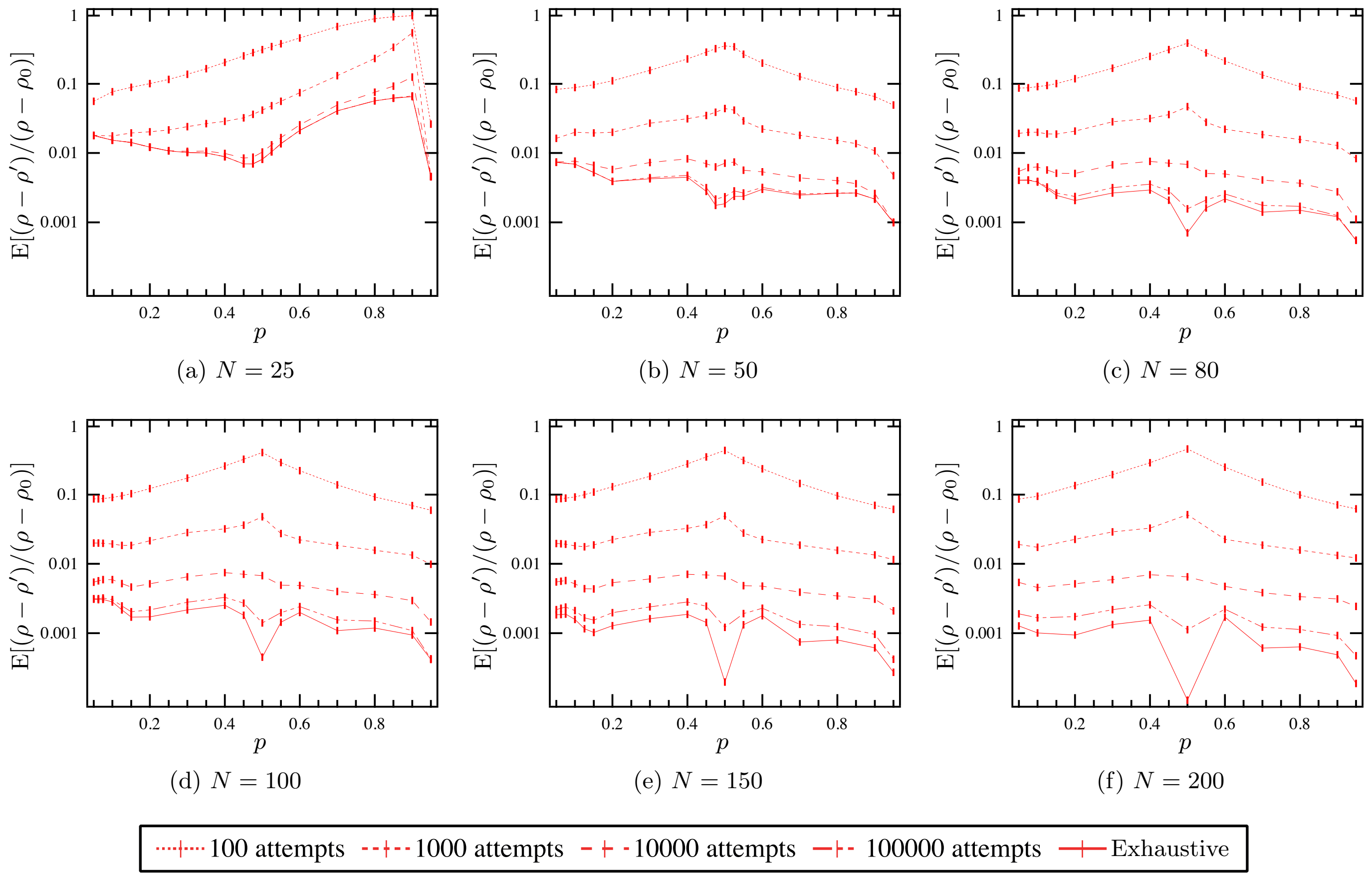


Figure 6: Means of relative differences in solutions obtained by the exact and greedy algorithms for various values of N and p. These plots apply to Erdős-Rényi networks. Each plot corresponds to a fixed N.

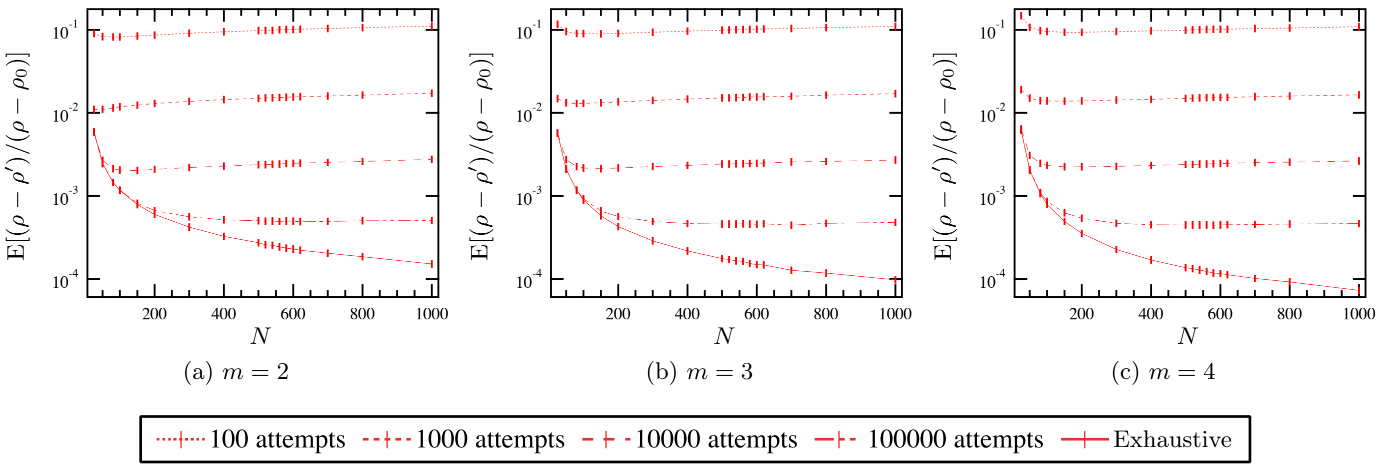
of these networks allowed us to investigate instances with up to $N = 1000$ nodes.

The means of the differences between the exact and greedy algorithms for Barabási-Albert networks are shown in Figure 7 as a function of N (for each m). Their Erdős-Rényi counterparts are shown in Figure 8. The overall trends are similar to those of the earlier Erdős-Rényi results (Figure 4): the greedy algorithm approximates the exact algorithm well and the exhaustive greedy results tend towards the exact greedy results as N increases. The fact that lines corresponding to the non-exhaustive greedy algorithm dip before rising (with increasing N) suggests that if s grows sufficiently with increasing N, non-exhaustive approaches will also tend towards exact solutions.

Overall, the greedy algorithm performs better on Barabási-Albert networks than it does on Erdős-Rényi networks with similar link counts. This is also true of the standard deviation plots corresponding to these networks (Figures 2 and 3 in the supplementary material).

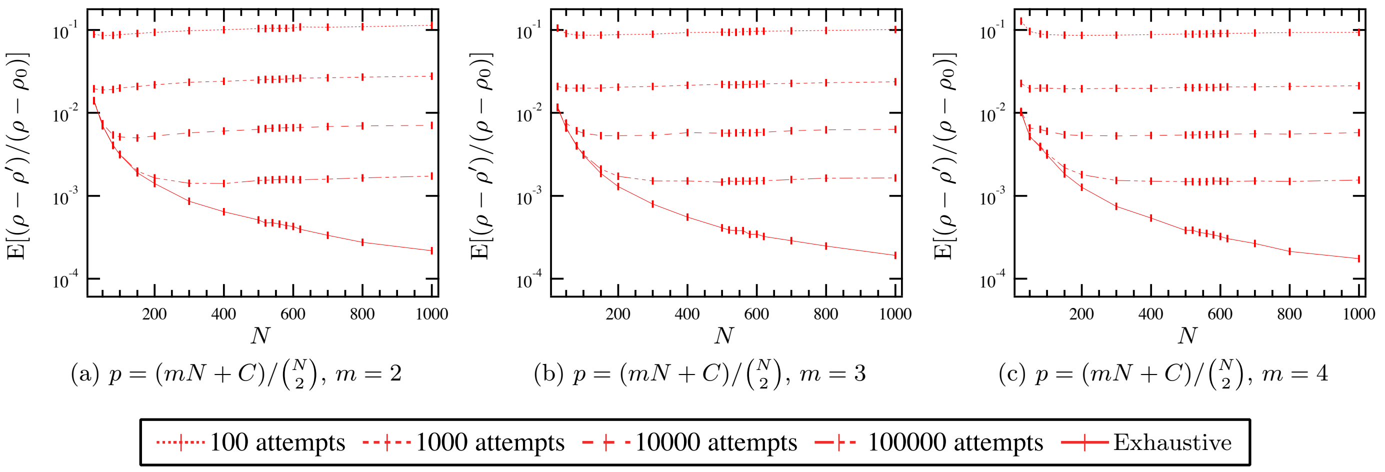
4.3 Real-world networks

Finally, we applied our algorithms to some real-world networks (see Supplemental Material at [URL will be inserted by publisher] for details). These networks are from diverse areas, making them a good testbed for confirming the trends observed for Erdős-Rényi and Barabási-Albert networks. The real-world network results are shown in Figure 9. The networks were sorted in terms of their link counts. These counts span two orders of magnitude, starting at 45 links at the left and ending with 5128 links on the right. The real-world network results confirm our earlier observations (albeit in terms of link counts). On the one hand, the exhaustive greedy algorithm fares progressively better as link counts increase. On the other hand, non-exhaustive runs of the greedy algorithm with fixed random sampling bounds s fare worse as L increases (although this is not so clear when $s = 100000$; this is likely because the link counts are not sufficient to show the same trends as for smaller s). Thus, for increasing link counts, the penalty incurred by the greedy algorithm requires increases in s. As a side note, there are a few networks for which the greedy algorithm manages to find exact solutions. These cases are coincidental and it remains an open question as to which network properties are associated with improved performance of the greedy algorithm (exact convergence being the golden standard). Our work so



(a) $m = 2$ (b) $m = 3$ (c) $m = 4$

Figure 7: Means of relative differences in solutions obtained by the exact and greedy algorithms for various values of N and m in Barabási-Albert networks. Each plot shows results as a function of N at fixed m.



(a) $p = (mN + C)/\binom{N}{2}$, $m = 2$ (b) $p = (mN + C)/\binom{N}{2}$, $m = 3$ (c) $p = (mN + C)/\binom{N}{2}$, $m = 4$

Figure 8: Means of relative differences in solutions obtained by the exact and greedy algorithms for Erdős-Rényi networks as functions of N. In each figure, probabilities are given by $p = (mN + C)/\binom{N}{2}$ where $m \in 2, 3, 4$ for the respective figures. The resulting networks have, on average, the same number of links as the Barabási-Albert networks in Figure 7.

far suggests that power law degree distributions and large numbers of rewirable configurations are important factors.

5 Conclusion

In this paper, we performed the first comparative study between greedy and exact algorithms for maximizing the assortativity of networks under the constraint that their degree vectors remain unchanged. We have focussed only on the maximization of assortativity but our results hold equally for the minimization of assortativity. A few sign changes in our algorithms is all that is required to convert them to minimization algorithms. We applied the algorithms to Erdős-Rényi, Barabási-Albert and real-world networks of varying sizes and link configurations. The overall theme is clear: the greedy assortativity-maximization algorithm approximates the exact algorithm well. We have shown that for all the considered Erdős-Rényi and Barabási-Albert networks, the average difference between the exhaustive greedy algorithm and exact algorithm decreases as the number of links is increased. The results support heuristic approaches such as those of Menche *et al.* [7], provided that the number of steps s is increased as the network size N increases. Our work raises some interesting questions:

- How many steps s does the greedy algorithm require to obtains results within a given tolerance of the exact algorithm?

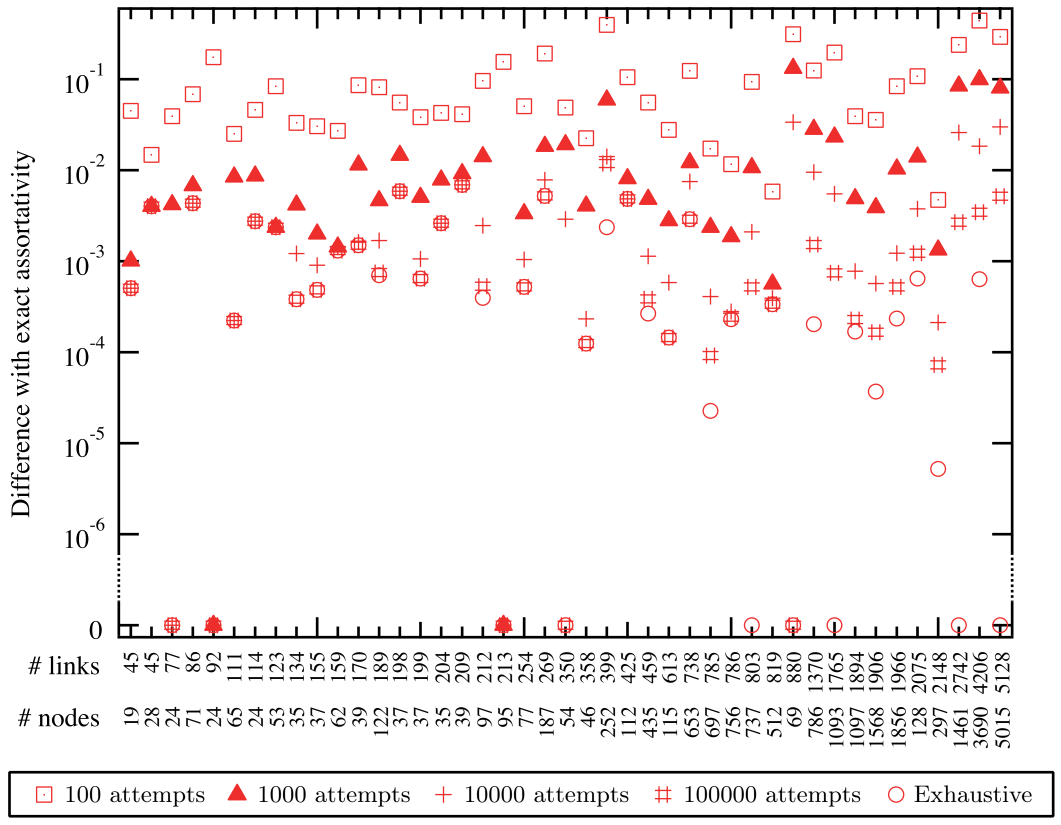


Figure 9: Means of relative differences in solutions obtained by the exact and greedy algorithms for a number of real-world networks.

– How bad can the results of a single greedy algorithm run be?
– How much better are sophisticated heuristic algorithms than our simple greedy algorithm?
– Why are greedy rewiring algorithms more effective on Barabási-Albert networks than on Erdős-Rényi networks?
– Under what circumstances can rewiring algorithms be expected to find exact solutions (as in the case of the real-world networks)?

Any approach to these questions would benefit from a faster exact assortativity-maximization implementation, such as the algorithm of Miller and Pekny [8]. Armed with such an implementation, one could investigate (hopefully much) larger networks.

The research leading to these results has been funded by the European Commission, under grant agreement no. 224619 (ResumeNet).

References

1. A. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509, 1999.
2. B. Bollobás. *Random Graphs*. Cambridge University Press, January 2001.
3. T. S. Evans. Exact solutions for network rewiring models. *The European Physical Journal B – Condensed Matter and Complex Systems*, 56(1):65–69, March 2007.
4. V. Kolmogorov. Blossom v: A new implementation of a minimum cost perfect matching algorithm. *Mathematical Programming Computation*, 1(1):43–67, 2009.
5. J. Lindquist, J. Ma, P. van den Driessche, and F. H. Willeboordse. Network evolution by different rewiring schemes. *Physica D: Nonlinear Phenomena*, 238(4):370–378, March 2009.
6. S. Maslov and K. Sneppen. Specificity and stability in topology of protein networks. *Science*, 296(5569):910–913, May 2002.
7. J. Menche, A. Valleriani, and R. Lipowsky. Asymptotic properties of degree-correlated scale-free networks. *Physical Review E*, 81(4):046103+, Apr 2010.
8. D. L. Miller and J. F. Pekny. A Staged Primal-Dual Algorithm for Perfect b-Matching with Edge Capacities. *INFORMS Journal on Computing*, 7(3):298–320, 1995.
9. M. E. J. Newman. Mixing patterns in networks. *Physical Review E*, 67(2):026126, 2003.
10. M. E. J. Newman. Mixing patterns in networks. *Physical Review E*, 67(2):026126, May 2003.
11. C. H. Papadimitriou and K. Steiglitz. *Combinatorial optimization: algorithms and complexity*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1982.
12. Y. Shiloach. Another look at the degree constrained subgraph problem. *Information Processing Letters*, 12(2):89–92, 1981.

13. P. Van Mieghem, H. Wang, X. Ge, S. Tang, and F. A. Kuipers. Influence of assortativity and degree-preserving rewiring on the spectra of networks. *The European Physical Journal B - Condensed Matter and Complex Systems*, pages 1–10–10, July 2010.
14. H. Wang, W. Winterbach, and P. Van Mieghem. Assortativity of complementary graphs. *The European Physical Journal B*, 83:203–214, 2011.
15. D. J. Watts and S. H. Strogatz. Collective dynamics of "small-world" networks. *Nature*, 393(6684):440–442, June 1998.