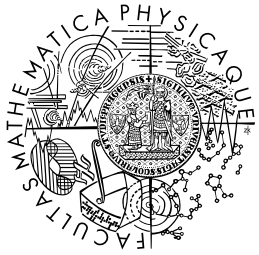

FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

BACHELOR THESIS

František Szczepanik

Centralities in Computation of Approximate Symmetries

Computer Science Institute of Charles University

Supervisor of the bachelor thesis: doc. Ing. et Ing. David Hartman, Ph.D. et Ph.D.
Study programme: Computer Science
Study branch: General Computer Science

Prague 2024

I declare that I carried out this bachelor thesis on my own, and only with the cited sources, literature and other professional sources. I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ....

Author's signature

I would like to thank my supervisor, David Hartman, and Anna Pidnebesna, for their advice and the time they spent discussing the topic with me. I would also like to thank my parents for their support during my studies.

Title: Centralities in Computation of Approximate Symmetries

Author: František Szczepanik

Institute: Computer Science Institute of Charles University

Supervisor: doc. Ing. et Ing. David Hartman, Ph.D. et Ph.D., Computer Science Institute of Charles University

Abstract: Network symmetry is a global characteristic of complex networks that helps understand their structure and properties. It has gained attention since MacArthur et al. showed that real-world networks contain surprisingly many symmetries. Traditional network symmetry definitions rely predominantly on graph automorphisms and are sensitive to minor changes to the network. This motivated the introduction of approximate network symmetry, a more robust metric. The thesis advances existing algorithms for approximate network symmetry computation by improving simulated annealing. Simulated annealing is enhanced to navigate the solution space by aligning similar vertices. Similarity of vertices is identified by graph centralities like PageRank, eigenvector centrality, and betweenness. We evaluate the improved annealing versions from multiple perspectives on random network models like the Barabási-Albert and Duplication-Divergence models.

Keywords: complex network, symmetry, automorphism, approximation, optimization, simulated annealing, graph centralities, random network models

Contents

Introduction

Complex networks are graphs that model the inner structure of natural and man-made systems such as the Internet [1], public transportation [2], the human brain [3], and social networks [4]. Analyzing these networks' local and global characteristics gives insight into their organization and evolution. One notable global property is network symmetry, which has gained significant attention since MacArthur et al. [5, 6] showed that real-world networks exhibit surprisingly high levels of symmetry. This property has proven useful in numerous research areas; for example, network symmetries reveal patterns of synchronized node clusters, helping understand various forms of collective behavior [7].

Most studies on network symmetry utilize automorphism groups to determine symmetry levels [8]. This approach is very fragile in the sense that even small changes to the network can drastically change the number of its automorphisms. Addressing this issue, Liu (2020) [8] proposed a definition of symmetry using approximate automorphisms, which allow imperfections.

Finding the best approximate automorphism for a given graph can be formulated as a combinatorial optimization problem suitable for metaheuristic approaches. Given the factorial complexity of the solution space, exhaustive search becomes computationally infeasible with growing graphs. Liu applied simulated annealing to search for the best approximate automorphism. Simulated annealing is an optimization probabilistic technique for approximating global minima that uses the concept of temperature to escape local minima.

We hypothesize that an ideal approximate automorphism will map "similar" vertices onto each other. Graph centralities, such as eigenvector centrality, PageRank, or betweenness, are numerical characteristics of vertices based on their position in the graph. We use centralities to guide simulated annealing to align similar vertices. We compare the performance of the original and guided versions of simulated annealing on several random network models, building on the experiments carried out by Straka [9] and Pidnebesna, Hartman et al. [10]. The primary aim of this thesis is to determine whether enhancing naive simulated annealing with graph centrality guidance improves the search for optimal graph symmetry.

Liu imposed a strict condition of no fixed points during the computation of approximate automorphisms, significantly reducing the solution space and potentially omitting promising candidates. Addressing this, Straka [9] proposed a version of annealing with the maximal number of fixed points as a parameter, which was further tested on random graphs by Pidnebesna, Hartman et al. [10]. However, it is unclear what the value of this parameter should be. We propose an alternative approach to fixed points that incorporates penalization. Specifically, we introduce an annealing variant that dynamically adjusts the number of fixed points by penalizing them in the objective function.

The thesis structure is organized as follows: Chapter 1 introduces Liu's notion of approximate symmetry. Chapter 2 outlines a description of simulated annealing and its implementation in the context of graph symmetries. Chapter 3 provides an overview of graph centralities we use to guide the search. Chapter 4 introduces several random network models, which serve as datasets for algorithm evaluation.

Chapter 5 introduces improved versions of simulated annealing and compares their performance on random network models. It also includes a comparison with a simple, greedy approach and examines the ability of improved annealing versions to find optimal symmetries when initialized close to a known solution.

1 Definitions

This chapter introduces graph automorphisms and uses them to establish the concept of approximate graph symmetry.

In the thesis, we extensively examine graphs, defined in discrete mathematics by a set of vertices V and edges E, denoted as $G = (V, E)$. All graphs considered are undirected, without loops, and unweighted. The variables n and m represent the number of vertices and edges of a given graph. To capture adjacencies, we use an adjacency matrix denoted as A, where $a_{ij} = 1$ if $\{v_i, v_j\} \in E$ and $a_{ij} = 0$ otherwise.

Network science involves the study of complex networks, which are modeled as graphs. Vertices (nodes) represent the actors or elements of the system, and edges (links) represent the connections between them. In network science, the terms networks, nodes, and links are used interchangeably with graphs, vertices, and edges, the only subtle difference being that the former usually refers to real-world systems, whereas the latter relates commonly to mathematical structures [11].

1.1 Graph automorphism

Graph automorphism is a permutation of vertices preserving adjacency:

Definition 1. *(*Graph automorphism*) An automorphism of a graph $G = (V, E)$ is a permutation π of V such that:*

$$\{\pi(v_x), \pi(v_y)\} \in E \iff \{v_x, v_y\} \in E.$$

To represent automorphisms and permutations in general, we use matrices:

Definition 2. *(*Permutation matrix*) Permutation matrix P for a permutation π on n elements is an $n \times n$ matrix where:*

$$P_{ij} = \begin{cases} 1 & \text{if } i = \pi(j), \\ 0 & \text{otherwise.} \end{cases}.$$

Permutation matrices have one non-zero entry per row and column and rearrange the vertices of the graph according to the given permutation when multiplied with its adjacency matrix (Figure 1.1). PA permutes A's rows and AP permutes its columns according to π.

The preservation of adjacency can be expressed by the condition that π is an automorphism for graph G with adj. matrix A if and only if $PA = AP$. Considering the orthogonality of permutation matrices, which implies $P^{\mathrm{T}} = P^{-1}$ and $PP^{\mathrm{T}} = I$, we obtain the following:

Lemma 1. *(*Characterization of automorphisms *[12]) Let A be the adjacency matrix of a graph $G = (V, E)$ and π a permutation on V represented by P. Then π is an automorphism of G if and only if $A = PAP^{\mathrm{T}}$.*

Proof: Let $v_h = \pi(v_i)$ and $v_k = \pi(v_j)$. Then:

$$(PA)_{hj} = \sum_{\ell=1}^{n} p_{h\ell} a_{\ell j} = a_{ij},$$

$$(AP)_{hj} = \sum_{\ell=1}^{n} a_{h\ell} p_{\ell j} = a_{hk}.$$

As a result, $AP = PA$ if and only if v_i and v_j are adjacent whenever v_h and v_k are adjacent. Therefore, $AP = PA$ if and only if π is an automorphism of G [12]. The form $A = PAP^{\mathrm{T}}$ follows from the orthogonality of permutation matrices. □

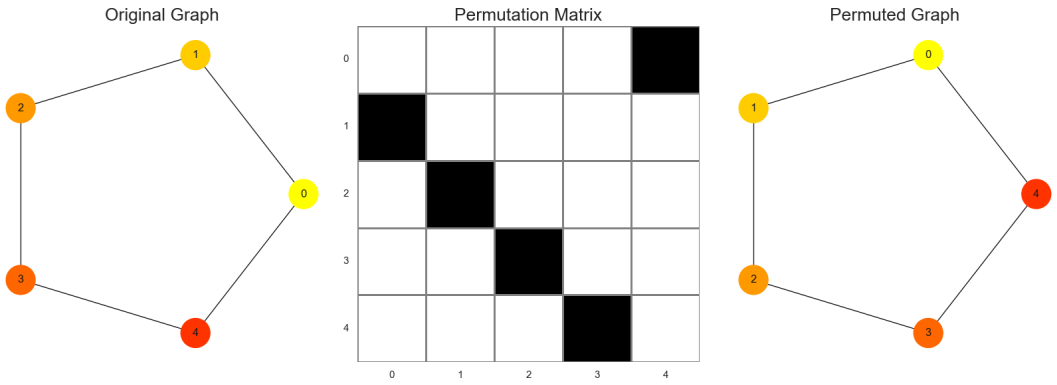

Figure 1.1 A cycle of length five, a permutation matrix for a rotational automorphism, and the cycle rotated along that automorphism. Black squares in the permutation matrix subplot correspond to 1, white to 0.

Every graph has at least one trivial automorphism: the identity that fixes every vertex onto itself and is represented by the identity matrix I.

Definition 3. *(*Fixed points, trivial and global permutations*) Let π be a permutation on a set V. The element $v \in V$ is a* fixed point *of π if $v = \pi(v)$. A* trivial permutation *fixes all points (the identity), whereas a* global *permutation has no fixed points.*

The number of fixed points can be expressed by the *trace* of the permutation matrix. A trace of a matrix P is the sum of elements on the diagonal and is denoted as $tr(P)$.

The number of fixed points of a permutation provides a basic intuition of how "valuable" the permutation is. If the trace is high, only a few vertices participate in the permutation, and the rest remain fixed. Imposing limits on the trace is one way of enforcing a certain "quality" of the automorphism.

The last piece of theory required to define approximate graph symmetry is a norm measuring the magnitude of elements in a matrix:

Definition 4. *(*L1 Norm*) The L1 norm of an $n \times m$ real matrix A is:*

$$\|A\|_1 = \sum_{i}^{m} \sum_{j}^{n} |a_{ij}|^2.$$

1.2 Approximate Symmetry

Most existing work on graph symmetry focuses heavily on global automorphisms and automorphism groups [8, 5, 6]. The issue with such symmetry

definitions is that they are very "fragile" in the sense that even small changes to the graph structure, such as edge deletions, can lead to complete asymmetry [8].

Consider the grid graph in Figure 1.2. It contains numerous symmetries, such as rotations, reflections along the diagonal, etc. Now consider the same grid with one edge removed. It is still somewhat symmetrical intuitively, yet has no global automorphism.

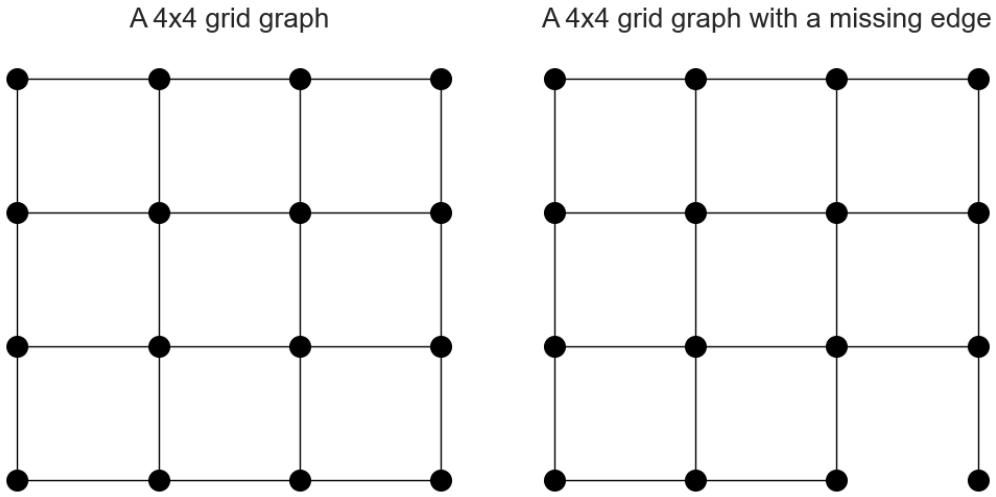

Figure 1.2 A grid graph is highly symmetric considering the number of global automorphisms. With just one edge removed, the graph loses all global automorphisms and will be evaluated as having no symmetry under traditional symmetry definitions, even though it still exhibits some intuitive non-zero level of symmetry [8].

Motivated by this fragility, Liu (2020) [8] introduced a new relaxed measure that allows "mistakes" in the (approximate) automorphism.

Definition 5. *(*Approximate graph symmetry*) The approximate symmetry of a graph $G = (V, E)$ with an adjacency matrix A is defined as* [1]*:*

$$E(A) = \frac{1}{4} \min_{P} (\left\| A - PAP^{\mathrm{T}} \right\|_1).$$

$E(A)$ minimizes the number of mismatches $A - PAP^{\mathrm{T}}$ over permutation matrices P, where $tr(P) = 0$. The higher the symmetry level, the lower $E(A)$ will be. If $A = PAP^{\mathrm{T}}$, P corresponds to an automorphism, rendering $E(A)$ zero. $E(A)$ also corresponds to the number of mismatched edges that differ after the permutation: $A_{ij} \neq A_{\pi(i)\pi(j)}$. For a visual representation, see Figure 1.3.

[1] Some literature also defines approximate symmetry using the Frobenius norm. The Frobenius norm can be derived from the L1 norm via square rooting, and, therefore, the two norms are interchangeable within the context of minimization problems.

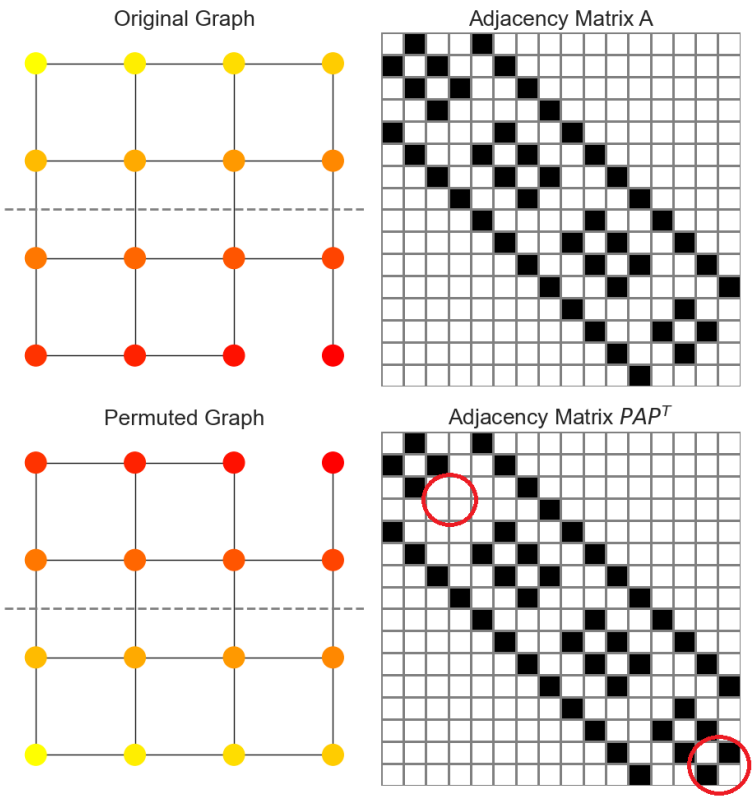

Figure 1.3 A visualization of $E(A)$ [8]. a) A 4 by 4 grid graph G with one missing edge. b) Its adjacency matrix A. c) G permuted along an optimal permutation P, a horizontal reflection. d) The adjacency matrix of permuted G. Highlighted are the differences from the original adjacency matrix A. Note that $E(A) = \frac{1}{4}(\|A - PAP^{\mathrm{T}}\|_1) = \frac{1}{4} \cdot 4 = 1$, corresponding to one mismatched edge.

Following Liu, we also define a parametrized symmetry version:

$$\epsilon(A, P) = \frac{1}{4}(\|A - PAP^{\mathrm{T}}\|_1),$$

yielding $E(A) = \min_P \epsilon(A, P)$. For comparability across graphs, Liu also introduced *normalized symmetry* scaled between 0 and 1; $E(A)$ is normalized by the maximal $\epsilon(A, P)$, which is achieved when the graph has exactly half of all possible edges, that is $\frac{1}{2}\binom{n}{2}$, and all of them are misplaced by P.

Definition 6. *(*Normalized approximate graph symmetry*) The normalized approximate symmetry of a graph $G = (V, E)$ with an adjacency matrix A is:*

$$S(A) = \frac{E(A)}{\frac{1}{2}\binom{n}{2}} = \frac{\min_P(\|A - PAP^{\mathrm{T}}\|_1)}{n(n-1)}.$$

Two trivial examples of graphs with perfect symmetry are empty and complete graphs, where every permutation yields $S(A) = 0$.

Identifying the optimal approximate automorphism P minimizing $\epsilon(A, P)$ is computationally hard, given that the growth of the number of permutations is factorial in the number of vertices. In the following chapters, we discuss simulated annealing, a metaheuristic approach used to navigate extensive solution spaces more efficiently.

2 Simulated Annealing

When computing the approximate symmetry of a graph with n vertices, evaluating all $n!$ permutations is computationally infeasible. We employ metaheuristics to facilitate the search. Metaheuristics are high-level procedures that effectively guide search in extensive solution spaces of hard optimization problems. Their goal is to find or approximate the global optimum, but they generally do not guarantee optimality. Confirming their correctness and efficiency can sometimes be only done empirically.

Simulated annealing was selected as the metaheuristic approach in Liu's paper. It is a probabilistic optimization technique inspired by statistical mechanics, specifically the annealing process in metallurgy [13]. The algorithm navigates the solution space by transitioning from one state to another while decreasing *energy*, which represents the optimization problem's objective function. The process is initialized at a random state and has high initial *temperature*, which allows the algorithm to sometimes explore worse states and avoid getting stuck in local optima. Temperature is gradually reduced according to a chosen *cooling schedule*. As it declines, the likelihood of accepting higher-energy states decreases, leading to eventual convergence to the final solution. The algorithm terminates if some threshold temperature is reached or the final iteration is completed. See Figure 2.1 for a visual representation of the algorithm computing the NP-Hard Travelling Salesman Problem (TSP).

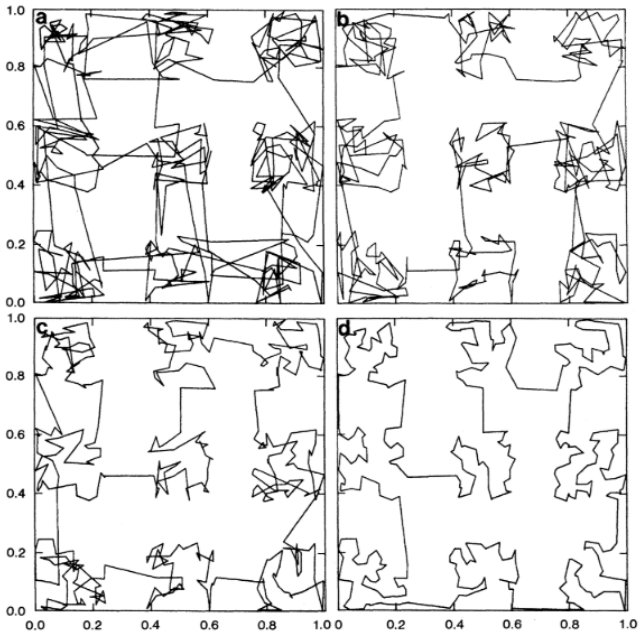

Figure 2.1 Snapshots of simulated annealing computing a TSP with 400 cities grouped into nine regions. TSP asks for the shortest route that visits each city once and returns to the origin. The temperature T decreases in each subplot: (a) $T = 1.2$, (b) $T = 0.8$, (c) $T = 0.4$, (d) $T = 0.0$. The energy declines throughout the run, which corresponds to the route becoming shorter [13].

2.1 Detailed description and implementation for optimizing S(A)

In the context of approximate graph symmetry, states correspond to permutations P, and the energy function to $\epsilon(A, P)$. Starting in a random permutation, we move to a neighboring permutation in each step, where P' is a neighbor of P if swapping two images under the permutation suffices for transitioning to one from another. We call this simple swap a *transposition*.

Below, we present the pseudocode of simulated annealing specifically adapted for optimizing approximate symmetries as described in Straka's thesis [9]:

Algorithm 1 Simulated annealing for approximate symmetry computation

1: $P \leftarrow$ random permutation matrix without fixed points
2: $T_1 \leftarrow t$
3: **for** steps $i = 1, 2, ...$ **do**
4: select P' from Neighbors(P)
5: **if** $\epsilon(A, P') \leq \epsilon(A, P)$ **then**
6: $P \leftarrow P'$
7: **else if** $\exp\left(\frac{\epsilon(A,P)-\epsilon(A,P')}{T_i}\right) \leq \text{uniform}[0, 1]$ **then**
8: $P \leftarrow P'$
9: **end if**
10: $T_{i+1} \leftarrow \frac{T_1}{\ln(1+k)}$
11: **end for**

A new state P' is always accepted if it improves energy, i.e., $\epsilon(A, P') \leq \epsilon(A, P)$. If this inequality does not hold, the algorithm may still accept P' with a probability of $e^{\frac{dE}{T_i}}$, where dE represents the energy change $\epsilon(A, P') - \epsilon(A, P)$, and T_i the current temperature at step i. T_i is adjusted at each step according to the cooling schedule. Liu applied a *logarithmic cooling schedule* as it is defined in line 12 of the pseudocode. Straka suggested it outperforms other cooling schedules, so we continue using it [9].

2.2 Implementation optimizations

Evaluating the objective function $\epsilon(A, P) = \frac{1}{4}(\left\|A - PAP^{\mathrm{T}}\right\|_1)$ involves two matrix multiplications, a costly operation with time complexity $O(n^\omega)$, where $\omega = 3$ for classic matrix multiplication, while modern algorithms achieve $\omega \approx 2.37$.

Straka introduced two significant optimizations. The first one is based on the observation that evaluating the full objective function in each step is unnecessary; rather, it suffices to compute its change. Since each transition corresponds to a transposition affecting only two rows and columns in PAP^{T}, we only calculate these differences. This reduces the time complexity to $O(n)$ per step instead of $O(n^\omega)$.

The second optimization builds on the fact that networks are often sparse [11], implying $deg(v) \ll n$. We evaluate the change in the energy function by associating each vertex with a set of its neighbors and computing the symmetric

difference of these sets upon transposition. This achieves $O(n \log n)$ per evaluation, but for sparse networks, it is, in practice, much faster than iterating on large matrices with many zeroes.

2.3 Moving between states

When transitioning from P, a neighbor is chosen uniformly randomly from $Neighbours(P)$, the set of permutations within one transposition from P. For instance, consider P representing the permutation π, where $\pi(i) = k$ and $\pi(j) = \ell$. A neighboring state P' is then achieved by swapping the images of i and j, resulting in $\pi(i) = \ell$ and $\pi(j) = k$.

We hypothesize that guiding the search during the next state selection could improve the results. In an automorphism or near-automorphism, "similar" nodes will likely be mapped onto each other (aligned). Assuming we could identify such similar nodes, we could choose swaps not randomly but in a way that aligns them. With that intention, we introduce graph centralities, numerical characteristics of nodes describing their position in the network.

3 Graph centralities

Thoroughly understanding a system often includes identifying its most important elements. For instance, when aiming to transmit some information across a social network effectively, it is crucial to identify influential individuals and maximize information spread through them [14].

Graph centralities address this question and help identify logical centers within a network. Developed in social sciences, the application of centralities has since expanded to fields such as biology and computer science [15]. A property of graph centralities that is especially interesting to us is that *automorphisms preserve centralities* [14, 16]. Therefore, in a permutation with high approximate symmetry, we expect vertices with similar centrality values to be aligned. In the description of graph centralities, we draw from a book on networks by Newman [15].

3.1 Degree centrality

The degree of a node, i.e., its number of neighbors, is the simplest and most intuitive measure of the node's importance. While it may seem logical to assume a node is significant whenever it has many connections, this approach fails to capture the node's broader position in the network beyond its imminent neighborhood. A high-degree node can be situated far from the actual center of the graph. Despite that, degree centrality in its naivety is still sometimes used. Consider the citation count of an academic paper, which is its degree in the citation network and is used to roughly determine its scientific impact [15].

3.2 Eigenvector centrality

Eigenvector centrality considers not only the number of a node's direct connections but also the "quality" of these connections. A node can thus become significant even with few links if these are with highly "prestigious" nodes[1].

For a network $G = (V, E)$ with an adjacency matrix A, eigenvector centrality x_i of a node v_i can be expressed as a sum of eigenvector centrality values of its neighbors:

$$x_i = \sum_{j=1}^{n} A_{ij} x_j.$$

Eigenvector scores are computed iteratively. Initially, all nodes start with a score of 1. In the second iteration, the score of each node becomes its degree. As the number of iterations continues $k \to \infty$, it can be shown that:

$$x_i = \frac{1}{\lambda} \sum_{j} A_{ij} \deg(j),$$

where λ is the greatest eigenvector of A [15]. See figure 3.1 for a visualization of eigenvector centrality values on a random graph.

[1] An interesting remark is that the earliest use of eigenvector centrality dates to 1895 in a paper about scoring chess tournaments [17].

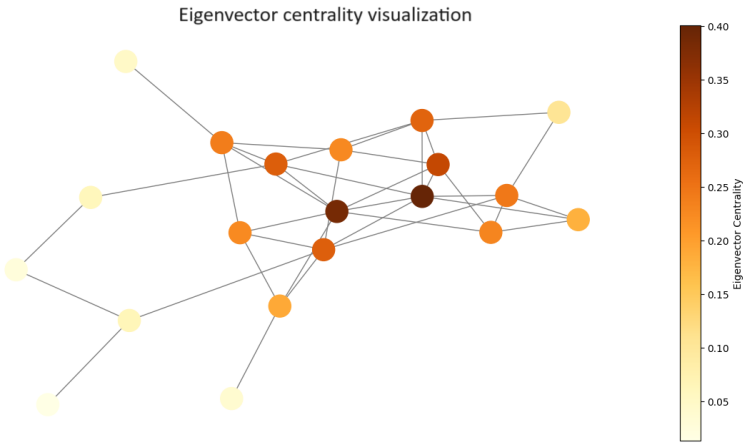

Figure 3.1 Eigenvector centrality values of vertices in a random graph

3.3 PageRank

PageRank is an algorithm introduced by Sergey Brin and Lawrence Page in their 1998 Stanford Paper [18] aiming to improve the quality of Web search engines. Like eigenvector centrality, PageRank evaluates the importance of a page by considering both the quantity and quality of its incoming links. It can be conceptualized as a model of a "random surfer" on the Internet who keeps skipping from one page to another through links. The page rank of a page then models the probability the surfer will visit it.

The formula includes a *damping factor* α, typically set to 0.85, representing the probability that the surfer will get bored with a given topic and restart on a random page. PageRank also adjusts by the number of outgoing links. This ensures that if one node with high centrality points to many others, it will pass on only a fraction of its importance. For instance, a link from Google will transfer less weight if Google also references a million other pages, even though Google has a high score itself [18].

Consider a network of pages (nodes) G with an adjacency matrix A containing a 1 if page v_i references page v_j. Let k_i be the number of outgoing references from page i (if it is zero, let the score for that page also be zero). PageRank x_i of the node v_i, in its simplest form, is defined as:

$$x_i = \alpha \sum_j A_{ij} \frac{x_j}{k_j}$$

While originally designed for directed graphs, PageRank is also applicable in the undirected case, where it sometimes comes close to the simple degree distribution but generally is not proportional to it [19].

3.4 Betweenness centrality

Betweenness centrality assesses a node's importance by counting the number of shortest paths between all pairs of nodes passing through it. Unlike previous centralities, it focuses less on how well-situated the node is and more on how necessary it is in connecting different parts of the network. It can be useful for identifying potential vulnerabilities of a network due to many critical paths passing through a specific node [14].

Formally, the betweenness of a node $B(v_i)$ is:

$$B(v_i) = \sum_{\substack{x,y \in V(G) \\ x \neq y \neq v_i}} \frac{\sigma_{xy}(v_i)}{\sigma_{xy}}$$

where σ_{xy} denotes the number of shortest paths between nodes x and y and $\sigma_{xy}(v_i)$ is the number of shortest paths between x and y passing through v_i [14].

3.5 Clustering coefficient

A cluster is a group of nodes more densely connected to each other than the rest of the network [11, 14]. The clustering coefficient is a measure of how clustered a neighborhood of a node is. The more interconnected a neighborhood of a node is, the higher the node's clustering coefficient. It can be considered a local variant of betweenness [15].

Formally, consider a network $G = (V, E)$ and N_i the neighbourhood N_i of node v_i: $N_i = \{v_j | \{v_i, v_j\} \in E\}$. Let $E(N_i)$ denote the edges of G induced by N_i. The clustering coefficient of a node v_i equals the ratio between the number of realized and possible links in N_i:

$$C(v_i) = \frac{2|E(N_i)|}{|N_i||N_i - 1|}$$

The clustering coefficient differs from previous centralities in that it does not assign higher values to more important nodes. On the contrary, when it is small, it implies a node is a critical connection for its neighbors [15].

4 Random network models

Network models serve as abstractions of real-world systems, and they should ideally reflect their inherent properties. Most real-world networks do not follow the regular structure of grids but rather appear much more random [11]. However, these networks are not entirely random without any recurring properties, and the laws that govern them are reflected in their structure. For example, many real-world networks exhibit the small-world phenomenon characterized by high clustering and low distances [20].

This chapter introduces several network models, some of which are not random or realistic and some of which more closely resemble real-world models. Following the methodologies of Straka [9] and Pidnebesna [10], these models form the primary dataset for comparing the improved (guided) and original versions of simulated annealing.

4.1 Grid

A grid is a square type of a *lattice graph*. Although not a random network model, we include it for comparison to highlight its extreme regularity against some of the latter models. We already saw an example of a 2D grid when defining approximate graph symmetry (Figure 1.2).

Formally, grids can be defined using graph products. $G \times H$ is the Cartesian product of graphs G and H if: 1) $V(G \times H) = V(G) \times V(H)$, and 2) $\{(u, u'), (v, v')\} \in E(G \times H)$ if $u = v$ and $\{u', v'\} \in E(H)$ or $u' = v'$ and $\{u, v\} \in E(G)$. A grid graph with d dimensions and lengths n_i in dimension i is then denoted as $R_{n_1,n_2,...,n_d}$ and defined as a graph product of path graphs $P_{n_1} \times P_{n_2} \times ... \times P_{n_d}$ making the number of vertices $\prod_{i=1}^{b} n_i$ [10].

4.2 Erdős–Rényi model

Introduced in 1959 by famous mathematicians Paul Erdős and Alfréd Rényi, the Erdős–Rényi model (ER) was the foundational piece in random graph theory [21]. The original application of the ER model was to probabilistically answer questions from extremal graph theory. When the term *random graph* is used without further context, it often refers to the ER model.

The ER model has two related definitions:

- $G(n, m)$: A graph with n vertices and m randomly placed edges.
- $G(n, p)$: A graph where each pair of vertices is connected with probability p.

Note that the expected number of edges of a $G(n, p)$ is $\binom{n}{2}p$. By the law of large numbers, with growing n, $G(n, p)$ will behave similarly as $G(n, E)$ for $E = \binom{n}{2}p$. Similarly, $G(n, m)$ will with high probability approximate $G(n, p')$ for $p' = \frac{m}{\binom{n}{2}}$. The parameter p then approximates edge density. We have already seen a $G(20, 0.2)$ ER graph when visualizing eigenvector centrality (Figure 3.1).

In Figure 4.1, we recreate Liu's [8] results and analyze the approximate symmetry S of a $G(100, m)$ ER graph as m varies from zero to the maximal number of edges. Interestingly, the S versus m curve is symmetric; initially, with no edges, every permutation of isolated vertices constitutes a perfect automorphism, rendering S zero. With growing m, the symmetry diminishes, reaching a minimum of approximately 0.35 when about half of the potential edges are present. It then declines as G becomes a fully connected graph, where any permutation again represents an automorphism. Notably, instances with m edges exhibit the same symmetry as instances with $\binom{n}{2} - m$ edges. This property is not coincidental and relates to an interesting and more general property of approximate symmetry:

Lemma 2. *(*Approximate symmetry of complements*) The approximate symmetry of a graph $G = (V, E)$ with an adjacency matrix A is the same as the approximate symmetry of its complement $\overline{G} = (V, \overline{E})$ with an adjacency matrix A', i.e.:*

$$S(A) = S(A').$$

The proof follows from the relation $A' = \mathbb{1} - A - I$, where $\mathbb{1}$ is an all-one matrix and I is the identity matrix. See [8] for the full proof.

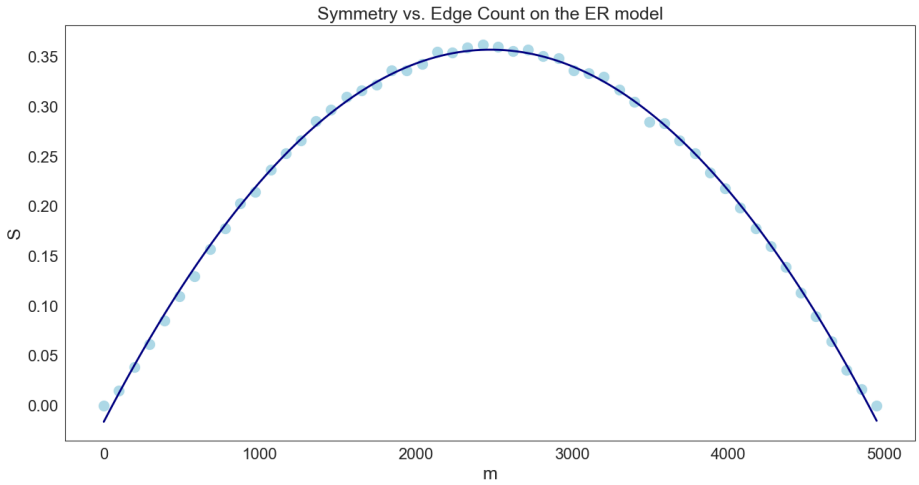

Figure 4.1 Approximate symmetry S of a $G(100, m)$ ER model for varying m. The data points are interpolated by a degree 2 polynomial.

This property interests us for additional reasons. As demonstrated in measurements by Liu [8] and Straka [9], even in some other types of random networks, it holds that sparser instances are more symmetric than denser instances.

4.2.1 Critique of the ER model

ER graphs were not developed in the context of network science nor were intended to replicate real-world networks. Unsurprisingly, they do not accurately reflect the characteristics of most real-world networks, e.g., the degree distribution. ER graphs have a binomial degree distribution that resembles a bell curve, meaning most nodes have a similar number of connections, and extreme outliers are very rare [11].

To demonstrate why this is inaccurate, consider modeling a population of $N \approx 7 \times 10^9$ individuals using an ER graph. The properties of the ER binomial degree distribution will imply that [11]:

- Most individuals would have between 968 and 1032 connections.
- The most connected individual would likely have no more than around 1185 connections.
- The least connected individual would likely still have around 816 connections.

Such a model conflicts with reality, as social networks commonly contain outliers with as many connections as 5000. The ER model assumes the number of vertices to be fixed and does not undergo any form of growth. In reality, networks dynamically expand as new nodes are introduced. Notably, these new nodes tend to connect preferentially to already well-connected nodes [11]. For example, in a citation network, a researcher is more likely to read and cite already frequently cited papers.

4.3 Barabási-Albert model

The Barabási-Albert model (BA), with its preferential attachment mechanism, provides an alternative that more accurately captures some of the real-world network characteristics. It produces *scale-free* networks, which are characterized by a degree distribution asymptotically following the power law with parameter γ. In the BA model, γ is typically set to 3. The probability a node will have degree k is then $k^{-\gamma}$. This distribution implies that while most nodes have few connections, a small number of nodes become highly connected, and we call those *hubs* [11].

The BA model has two parameters: n, the target number of nodes, and k, the number of links of a new node when it is added to the network. The generation process is then described as follows:

- Start with a random connected graph on k vertices.
- In each step, add a new node v with k connections. It links to an existing node u with probability p_u proportional to the degree d_u of u: $p_u = \frac{d_u}{\sum_{i=1}^{n} d_i}$.
- Repeat until the network contains n nodes.

The BA model mimics network evolution, where already well-connected nodes gain more connections, becoming hubs (for illustration, see Figure 4.2 depicting three BA graphs on 20 nodes with varying numbers of attachments of the new node. Dark nodes are hubs.). This description alone hints that BA graphs contain some symmetric substructures, unlike the ER model. Refer to Straka [9] for more detailed measurements confirming that the BA model really is more symmetric than the ER model.

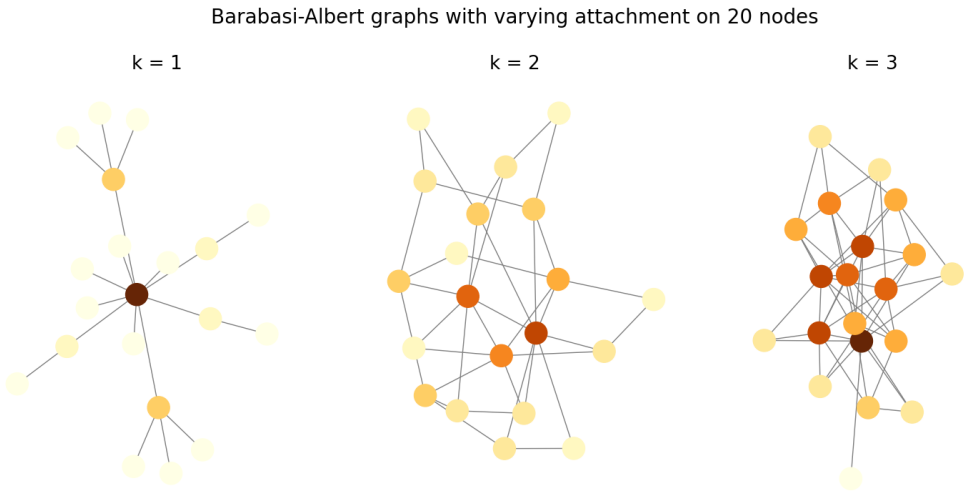

Figure 4.2 Instances of the BA model with varying attachment numbers k. Darker colors imply higher degrees.

4.4 Duplication-Divergence model

Protein-protein interaction (PPI) networks are graphs representing protein contacts. These networks evolve by the mechanism of gene duplication, which generates new proteins that are initially identical to pre-existing ones but gradually change, with most of the added proteins not surviving. The duplication-divergence (DD) model provides a simplified representation of this evolution and can be characterized by its growth mechanism [22]:

- **Duplication**: A node is randomly selected and duplicated. The newly created node inherits all links of the duplicated node.
- **Divergence**: Every link of the new node is retained with *divergence probability* σ. If no links survive, the duplication is unsuccessful, and the new node is discarded.

The process is initialized with a single node and grows until it reaches a target size. See Figure 4.3 for instances of the model with varying divergence probability.

In an extreme case, when $\sigma = 1$, the network evolves into a complete bipartite graph. When $\sigma \ll 1$, only one connection is most likely retained, resulting in each new node being a leaf and the graph resembling a tree. Studies of real-world PPI networks suggest σ to often be around 0.4 [22].

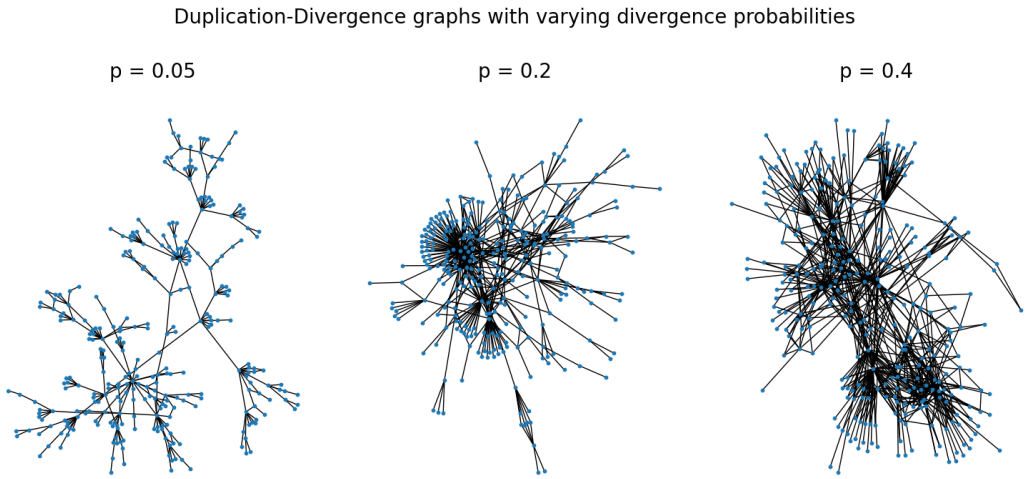

Figure 4.3 Three instances of the DD model on 300 nodes, where p is the divergence probability.

5 Results

In the final chapter, we introduce our results, starting with an alternative version of simulated annealing that dynamically determines fixed points by penalizing them in the objective function. After that, we present the most important result of this thesis: the effects of guiding annealing by graph centralities.

5.1 Annealing with dynamic fixed points

In the original paper on approximate network symmetry [8], Liu considered only permutations with no fixed points (FPs). This reduces the problem space at the risk of excluding potentially promising solutions that allow some FPs. This limitation motivated Straka [9] and later Pidnebesna et al. [10] to propose and experiment with a modified annealing variant that allows up to K FPs, where $K \in \mathbb{N}_0$ is a parameter. This variant, called *Annealing with Fixed Points* (AFP), introduces the uncertainty of what K is optimal.

Pidnebesna et al. experimented with $K = \frac{n}{2}$, allowing half the nodes to be fixed. However, this choice may seem a bit arbitrary, and furthermore, it is desirable to distinguish between near-global permutations and those involving only half vertices, the former being more "valuable." An alternative approach to FPs could work with penalization mechanisms rather than an arbitrarily set parameter. In this manner, we introduce an annealing variant that dynamically determines the number of FPs by penalizing them in the objective function.

Recall the definition of normalized approximate symmetry $S(A)$, which is essentially a ratio of mismatches $A - PAP^{\mathrm{T}}$ to the number of maximum possible mismatches. We observe that when two vertices are fixed in a permutation P, their adjacency will always be preserved, effectively reducing the mismatch "opportunities." Therefore, we reduce the denominator by subtracting pairs of all fixed points and obtain the following form, which we denote as $S_F(A)$:

$$S_F(A) = \frac{E(A)}{\frac{1}{2}\binom{n}{2} - \binom{F}{2}} = \frac{\min_P(\left\|A - PAP^{\mathrm{T}}\right\|_1)}{n(n-1) - 2F(F-1)},$$

Where F is the number of FPs in P. This is a penalization mechanism because increasing the number of FPs results in a smaller denominator, a higher value of $S_F(A)$, and, therefore, lower symmetry. It is still a valid measure in the sense that it scales between 0 and 1 and expresses the ratio of mismatched edges but only normalizes it by the actually relevant number of possible mismatches.

We propose a variant of simulated annealing with S_F as its objective function. This variant dynamically determines the number of FPs, and we call it *DFP-SA*. The penalization aims to ensure that adding FPs actually improves the solution rather than the algorithm arbitrarily reaching the $\frac{n}{2}$ limit (or whatever value the parameter was set to in AFP).

5.1.1 DFP-SA on BA and introduction of methodology

In Figure 5.1, we present comparisons of approximate symmetry S *as initially defined in Chapter 1*, measured by DFP-SA and AFP (even though DFP-SA takes

$S_F(A)$ as its objective function, we evaluate the results in the original measure $S(A)$, so the comparison makes sense). We denote AFP with K maximum fixed points as KFP-SA and set K to zero and half of the graph size, with the two versions denoted as 0FP-SA and N/2FP-SA.

The dataset consists of instances of the BA model with $n = 50, 100, 150$, and k, the number of connections of a new node, equal to $3, 5, 7$, following the methodologies of Pidnebesna et al. [10] (we distinguish between K in KFP-SA and k, the BA parameter). We present the measured symmetry for BA graphs on 50 nodes since the results on 100 and 150 nodes are similar (which can be verified in the more robust statistical evaluation that follows). The number of measurements carried out for every graph and annealing version is 50. The number of steps of each annealing run is 30000, which Straka declared to offer the best combination of performance versus time [9]. We will continue using this setting of annealing steps and number of measurements in all later evaluations throughout the rest of the thesis unless specified otherwise.

We use a *violin plot* to represent the distribution of measured approximate symmetry, which visualizes the mean and quartiles similarly to box plots but also includes the probability density of the data smoothed by a kernel density estimator. Apart from approximate symmetry, the average number of FPs of the given measurements is displayed above each corresponding plot.

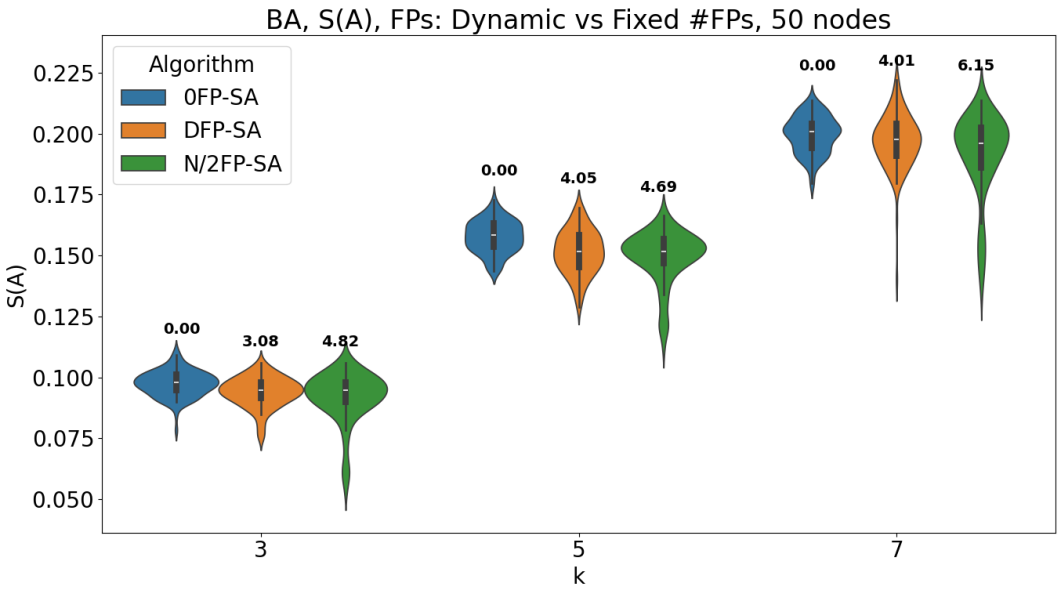

Figure 5.1 Comparison of the performance of DFP-SA and KFP-SA (with K set to 0 and N/2) on BA graphs with 50 nodes, where k is the number of attachments of a new node (we distinguish between k, the BA parameter, and K in KFP-SA).

A closer look at the results, specifically the mean values of the violin plots, suggests that 0FP-SA generally performed slightly worse than the two other variants and found the lowest symmetry. This is expected, considering it is the most restrictive version and disallows any FPs. When examining the average number of FPs, we observe that for DFP-SA, it is generally similar to or lower than that for N/2FP-SA. We also observe that sparser instances of the BA model, i.e., those with lower k, exhibit higher approximate symmetry, which is a common

trend, and we will later observe sparser instances having higher approximate symmetry in all other models.

For a more precise evaluation of these results, we follow the methodologies of Pidnebesna et al. [10] and use a paired two-sided t-test to determine whether the average measured approximate symmetry of two selected versions is the same. The null hypothesis, therefore, states that there is no significant difference in the means of approximate symmetry measured by the different annealing versions. Furthermore, if low p-values suggest the means are not the same, we use Cohen's d to measure the effect size. The results are visualized in Figure 5.2. We continue using this methodology and visual representation throughout the rest of the thesis. Therefore, we explain it now more thoroughly.

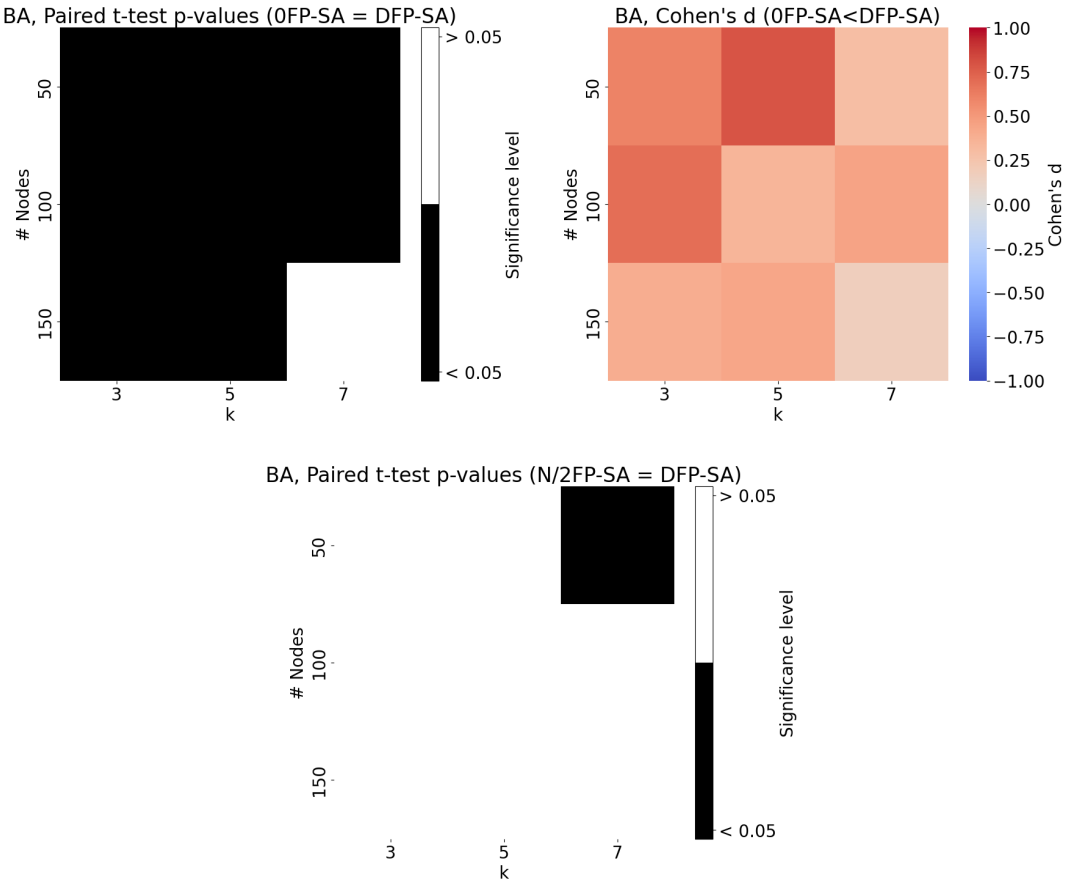

Figure 5.2 Statistical analysis of the performance of DFP-SA and KFP-SA (with K set to 0 and N/2) on BA graphs. k is the number of attachments of a new node.

The top-left subplot shows the t-test results assessing whether 0FP-SA and DFP-SA compute results with the same mean. A black square indicates a p-value lower than the standard significance level $\alpha = 0.05$. The top right subplot measures effect size by Cohen's d and is relevant only when the corresponding p-value is below α. A positive Cohen's d value (red) implies the mean of the first group is higher than the second group, while a negative value (blue) indicates the opposite. Values around 0.2 suggest a small effect size, 0.5 a medium effect size, and 0.8 a large effect size.

Assessing the top left sub-plot, we see p-values below α, with the exception of

the BA graph on 150 nodes and k equal to seven. We, therefore, cannot reject the null hypothesis in full generality, but we can still study the effect sizes in the cases when the p-value is below α. The predominantly red entries in the top-right subplot suggest that in these cases, the mean of 0FP-SA results is higher than DFP-SA results, implying that DFP-SA finds better symmetry, confirming our initial observation. The bottom sub-plot depicts t-test results assessing the equality of mean symmetry calculated by N/2FP-SA and DFP-SA. We cannot reject the null hypothesis nor study the effect size, considering almost all p-values are above α.

We replicate this comparison on other random models and larger graphs, reaching similar results (see appendix A.1). Taking into account all measurements, we cannot confidently and in full generality say that DFP-SA computes symmetry better or worse than 0FP-SA or N/2FP-SA, although, in many instances, it performs better than 0FP-SA, which is explained by 0FP-SA's restrictive nature. Also, DFP-SA includes a similar or lower average number of FPs than N/2FP-SA but computes them dynamically, which solves the uncertainty about setting the parameter K in KFP-SA. In conclusion, we introduced an annealing version with an alternative approach to FPs that reaches similar results as AFP. From now on, in all future measurements, we will continue using this dynamic approach to FPs, and all annealing implementations will operate with $S_F(A)$ as their objective function.

5.2 Centralities in annealing

This thesis primarily aims to investigate the effects of guiding simulated annealing to align similar vertices rather than performing steps by random transpositions, with the goal of improving symmetry detection. This approach is motivated by the fact that characteristics such as betweenness or PageRank are preserved by automorphisms, as discussed in Chapter 3; therefore, in an (approximate) automorphism, we can expect vertices with similar centralities to be aligned.

We introduce a general procedure for incorporating a centrality into the algorithm. We then plug in the centralities introduced in Chapter 3, but any numerical vertex characteristic can be used in this way. We call this new improved annealing version *guided annealing* and compare it with the original unguided annealing version on network models introduced in Chapter 4. All annealing versions, guided and unguided, determine the number of fixed points dynamically by penalization and operate with $S_F(A)$ as their objective function, as described in the previous section.

5.2.1 The move function

The important annealing function to be re-implemented is the *move* function, which generates a new permutation P' from the current permutation P. In the original version, this function performs a transposition by randomly swapping the images of two vertices under the current permutation (moving from $a, \pi(a)$ and $b, \pi(b)$ to $a, \pi(b)$ and $b, \pi(a)$). In our improved, guided version, the transposition is chosen with probability proportional to how similar pairs it leads to.

Similarity matrix and implementation description

We will use a *similarity matrix* M to express the similarity of vertex pairs in a chosen centrality Γ. M is an $n \times n$ matrix indexed by vertices.

To construct M, we start by computing a *difference matrix* D, where each element is defined as $d_{ij} = |\Gamma(i) - \Gamma(j)|$, i.e. the absolute difference of the centrality of v_i and v_j. To transform this into a similarity measure where higher values imply higher similarity, we compute inverses over the elements of D. Specifically, the entries of M are defined as $m_{ij} = (d_{ij} + \beta)^{-1}$, where $\beta > 0$ is a division constant preventing the denominator from being zero and moderating the variance of the matrix values; as β increases, the entries of M become more uniform, guiding the annealing less aggressively.

Re-implementing the move function

We now describe the re-implemented move function. We start by randomly choosing the first vertex a [1], where $\pi(a)$ is its image under the current permutation and $m_{a\pi(a)}$ their similarity. We evaluate a potential similarity increase by considering swaps with each vertex b and its image $\pi(b)$. The differ-

[1] We also experimented with a "one-step" approach, which does not select the first vertex randomly but instead considers all possible transpositions. However, this requires $O(n^2)$ evaluations in each step or a one-time pre-computation for all pairs of pairs, which has a complexity of $O(n^4)$, both practically unusable for growing graphs.

ence in the similarity of the original and proposed setting can be calculated as $\Delta M_b = m_{a\pi(b)} + m_{b\pi(a)} - m_{a\pi(a)} - m_{b\pi(b)}$.

Before normalizing ΔM into a probability distribution over vertices, we again introduce a smoothing parameter $\phi > 0$, which we call the probability constant, and set $\Delta E_b = max(\Delta M_b, \phi)$ for each b. Similarly to the division constant β, higher values of ϕ imply all vertices (even dissimilar pairs) have higher chances to be chosen, guiding the annealing less aggressively. We then draw b randomly from the created probability distribution.

For comprehension, we present the pseudocode of the re-implemented move function, where we assume π is the current permutation represented by P:

Algorithm 2 Move function in guided annealing.

1: $a \leftarrow$ random vertex
2: **for** every vertex b **do**
3: $\quad \Delta E_b = max(m_{a,\pi(b)} + m_{b,\pi(a)} - m_{a,\pi(a)} - m_{b,\pi(b)}, \phi)$
4: **end for**
5: $\Pi \leftarrow$ probability distribution computed by normalizing ΔE
6: $b \leftarrow$ vertex randomly drawn from Π
7: $P' \leftarrow$ permutation generated from P by transposing images of a, b
8: return P'

Time complexity of the new version

The re-implemented *move* function runs in time $O(n)$, as it involves iterating over all vertices to select the setting producing the most similarity. Recall the two optimizations from Chapter 2; the first one, stating that it suffices to compute differences in energy instead of evaluating the whole energy function in each step, remains valid. It also led to Complexity $O(n)$ per move (though better on average). The second optimization, yielding $O(n \log n)$ but a much better average by working with neighbor sets, is overshadowed by the complexity of choosing the right swap.

The similarity matrix is computed only once at the beginning of a run for each graph, and most of the complexity in this process lies in computing the centralities themselves. Consider betweenness, which is in practice (e.g., in the Python library NetworkX) computed by the Brandes Algorithm and has a time complexity of $O(nm)$ [23].

Optimization of parameters by grid search

When re-implementing the move function, we introduced two parameters, β and ϕ, i.e., the division and probability constants. To determine their (at least approximately) optimal values for each type of centrality, we carried out a grid search on about 400 random ER, BA, and Stochastic Block Model graphs with varying sizes (50 to 150 vertices) and parameters, although we later excluded Stochastic Block Model graphs from the analysis due to their structural similarity to ER graphs. The evaluated values of β and ϕ ranged from 0.001 to 10, and for every combination of parameter and graph, 30 simulations were conducted. We then selected the parameters with the best average results. When working with

PageRank, we also conducted a grid search to optimize the damping factor α, but without significant differences in results; therefore, we keep using the default value of $\alpha = 0.85$.

Generally, the few top combinations of parameters performed similarly, whereas the other combinations performed significantly worse. We do not claim the global optimality of the parameters we selected and recognize that different datasets could produce different parameters. However, the parameters we selected were sufficient to answer our hypothesis about guided annealing's improvement in symmetry calculation.

5.2.2 Erdős–Rényi model

We now examine guided annealing's improvements in detecting approximate symmetry, starting with the ER model. We compare the symmetry found by annealing variants on ER graphs with $n = 20, 50, 100$ and edge densities $p = 0.1, 0.3, 0.5$, loosely following the methodologies of Pidnebesna et al. [10].

As discussed in Chapter 4, the ER model is truly completely random; it lacks any internal structure stemming from a generation process and has no predictable substructures that can be reflected in each other. There are no nodes that play distinct roles, as in the case of hubs in the BA model. Given this total randomness, it is unsurprising that centrality-guided annealing resulted in no significant improvements, as can be seen in Figure 5.3. There were no statistically significant differences between results produced by the different annealing versions (in terms of t-test results checking the equality of the mean values of computed symmetry). This holds for all graph sizes, and so we present only results for ER graphs on 50 nodes. The results are visualized in violin plots, where each color represents a version of annealing guided by a different centrality.

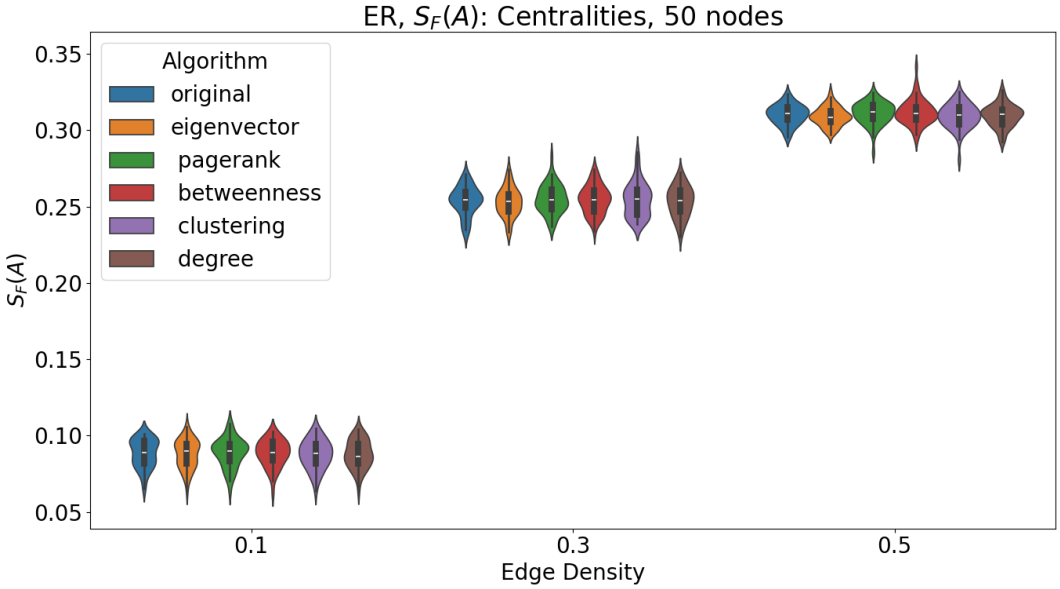

Figure 5.3 Comparison of the performance of simulated annealing guided by different centralities on ER graphs. For each configuration of parameters, we conduct 50 simulations (as stated in section 5.1.1).

5.2.3 Grid

In contrast with the ER model, grid graphs are highly regular. We evaluate improved annealing on grid graphs with $50, 100, 150$ vertices in 2 and 3 dimensions. In 2D, the lengths of sides are always set to $5 \times x$ and in 3D to $2 \times 5 \times x$ [10]. For all grid parameters, guided versions significantly improved computed symmetry, with betweenness yielding the highest improvements. Figure 5.4 presents the results for grids on 100 vertices. The remaining measurements on other grid sizes are included in the appendix A.5.

Looking at the results, we observe that annealing guided by the clustering coefficient underperforms and finds the lowest symmetry. Degree centrality performs slightly better than the clustering coefficient. This underperformance is common not only in grid graphs but also, to some extent, in the more sophisticated random network models, where we will see clustering performing the worst and degree centrality sometimes approaching the results of more complex centralities but rarely being the best. We explain this by the local scope of these two centralities; they focus only on a node's neighborhood without describing its broader position in the network and describe the node in less detail.

Looking again at Figure 5.4, we observe that betweenness improves symmetry detection the most. One possible explanation stems from the fact that in a grid graph, betweeenness has more unique values than other centralities. For instance, the vertices of a 5×20 grid have 48, 68, and 77 unique PageRank, eigenvector centrality, and betweenness values (compared to only 3 different degrees and a constant zero value of clustering). Betweenness thus grants the most nuanced differentiation of vertices and enables annealing to align vertices with higher precision.

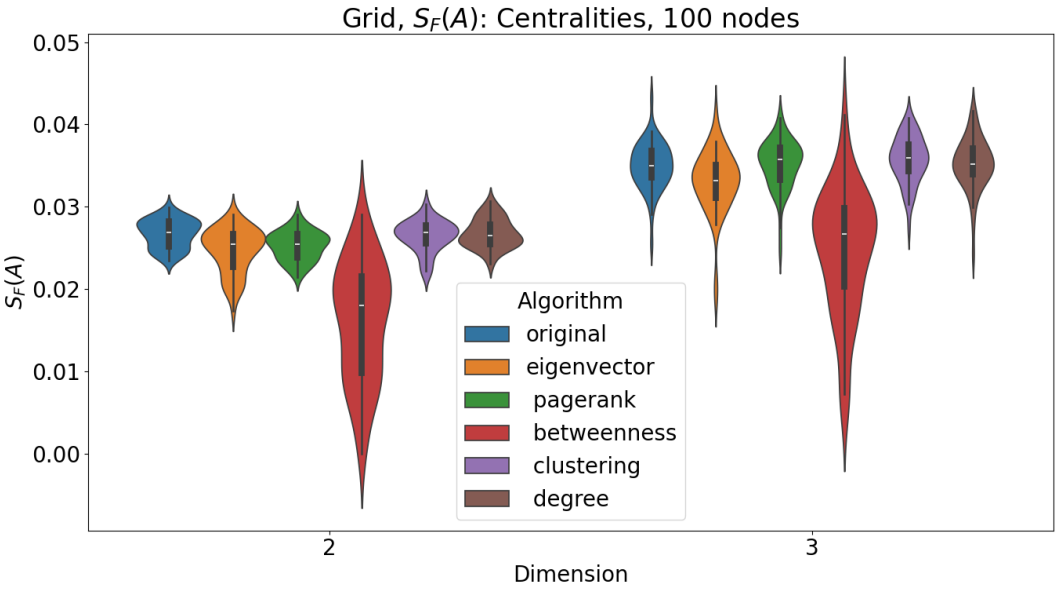

Figure 5.4 Comparison of the performance of simulated annealing guided by different centralities on grid graphs.

5.2.4 Barabási–Albert model

We now move to more realistic random network models, starting with the BA model. We conduct measurements on instances with 50, 100, and 150 nodes and k values of 3, 5, and 7.

We present the results for BA graphs on 150 nodes and k of 5 in Figure 5.5. The measurements on BA graphs with other parameters are included in the appendix A.6. Overall, the improvements are less pronounced than in the grid, especially for smaller graph sizes with $n = 50$. As graph size increases to $n = 100, 150$, the improvements become more significant. Versions of annealing guided by eigenvector centrality, betweenness, and PageRank yield some level of improvement over original annealing, with degree centrality following closely and clustering performing the worst.

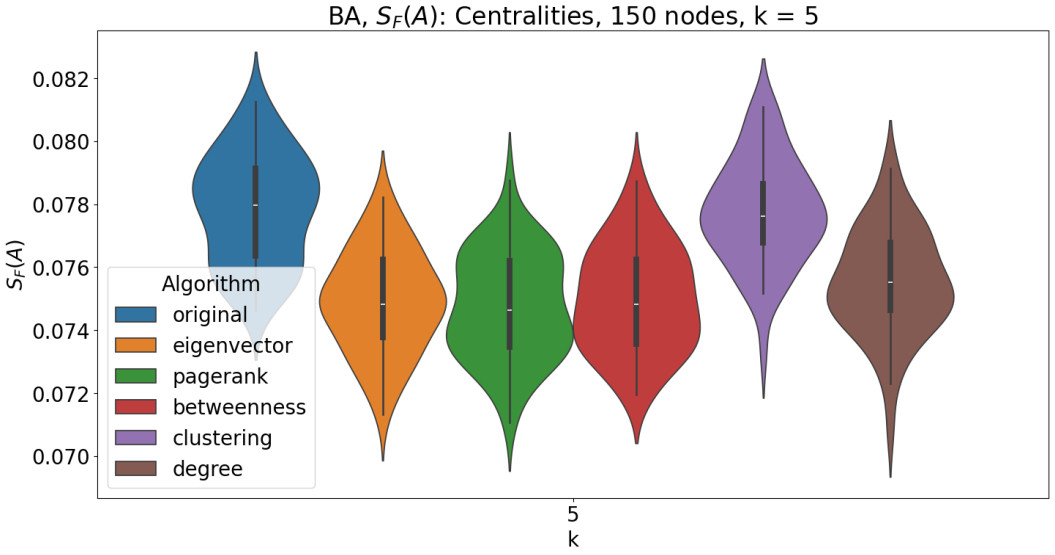

Figure 5.5 Comparison of the performance of simulated annealing guided by different centralities on BA graphs with 150 nodes and k equal to 5.

We select eigenvector centrality, which seems to perform similarly or better than other centralities, and rigorously evaluate its improvements, applying statistical methods outlined in Section 5.1. We compute paired t-tests to determine whether the two algorithms (eigenvector-centrality-guided and original annealing) compute symmetry values with the same mean, along with Cohen's d to measure the effect size, if relevant.

We present the results in Figure 5.6 and observe that apart from one combination of parameters, annealing guided by eigenvector centrality produces significantly better outcomes compared to original annealing, leading to p-values above 0.05. We also see that the larger and sparser the graph is, the greater the improvement, as evident by decreasing Cohen's d values.

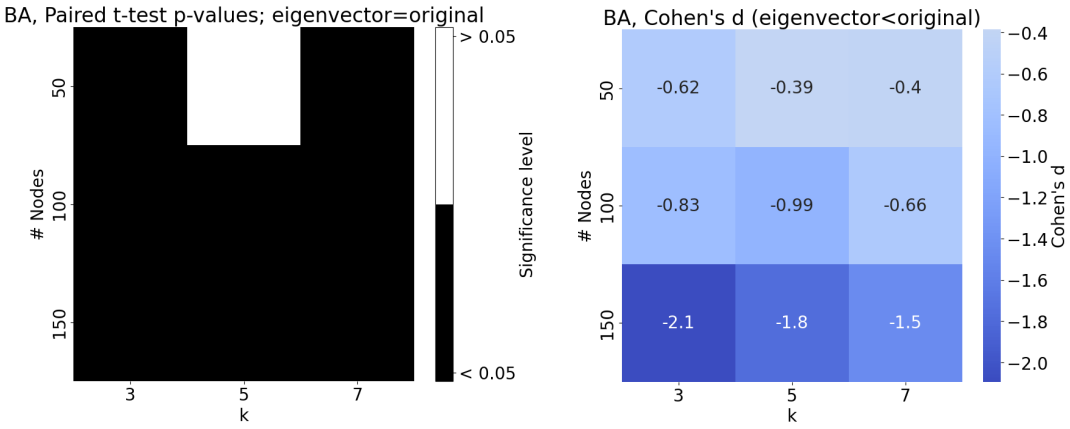

Figure 5.6 Comparison of the performance of original simulated annealing and simulated annealing guided by eigenvector centrality, BA model, where k is the number of links of a new node.

5.2.5 Duplication-Divergence model

Finally, we conduct measurements on the DD model. The dataset consists of DD graphs with node counts $n = 50, 100, 150$ and divergence probabilities $0.1, 0.3$. The results for DD graphs on 150 nodes and divergence probability 0.1 are presented in Figure 5.7, and the full measurements on all graph parameters are included in the appendix A.7. Our findings indicate that guided annealing significantly outperforms original annealing. It also holds that instances with divergence probability 0.3 are less symmetric than those with divergence probability 0.1, which we attribute to the fact that as σ decreases, the graph becomes sparser and starts to resemble trees, which are intuitively somewhat regular (at least in the sense of having many leaf nodes likely sharing similar properties).

We select PageRank for a more rigorous evaluation, which seems to perform at least as well as other centralities. Calculating paired t-tests and Cohen's d reveals that for smaller graph sizes, we cannot reject the null hypothesis, which states that the mean of symmetry values calculated by the two algorithms is the same. If we move to larger graph sizes, however, we see PageRank-guided annealing outperforming the original, unguided version, as evident in Figure 5.8. The improvements are more significant in sparser DD instances with lower σ and also become more pronounced with growing graph size.

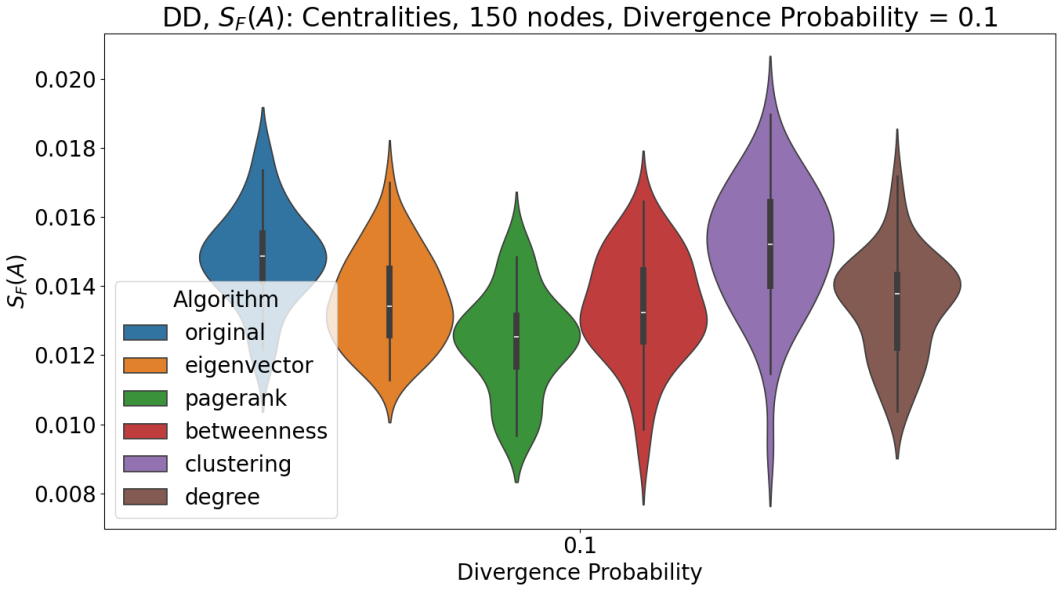

Figure 5.7 Comparison of the performance of simulated annealing guided by different centralities on DD graphs with 150 nodes and Divergence Probability 0.1.

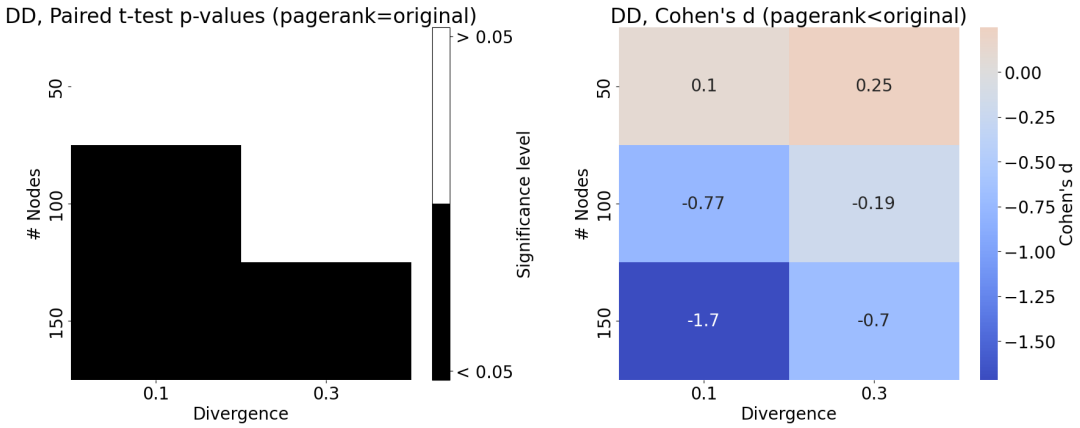

Figure 5.8 Comparison of the performance of original simulated annealing and simulated annealing guided by eigenvector centrality, DD model.

5.2.6 Larger graphs

In the previous section, we observed a trend in both the BA and DD models; improvements achieved by guided annealing became more pronounced with growing graph size. To further explore this trend, we extend our analysis to graphs of sizes 300 and 500. Specifically, we focus on the performance of PageRank-guided annealing on larger DD graphs (Figure 5.9) and eigenvector-centrality-guided annealing on larger BA graphs (Figure 5.10). In both of these settings, the t-test results allow us to reject the null hypothesis, stating that the two measured algorithms compute symmetry values with the same mean. Moreover, effect sizes confirm that the results of the improved, guided version are indeed better.

The results for DD graphs confirm that PageRank's improvements are more

significant in larger graphs (as evident by the effect sizes). The violin plot depicts improvements in found symmetry on DD graphs with $n = 500$ and divergence $p = 0.1$, and the two lower subplots provide data on effect sizes for other DD instances.

The trend also holds for the BA model, where, in the case of the $n = 500$, $k = 5$ graph (where k is the number of links of a new node), the guided version consistently outperforms the original one, almost to the extent that the worst results of the guided version surpass the best results of the original version.

It can be challenging to grasp the practical significance that improvements in measured $S(A)$ imply. We recall that $S(A)$ (and also $S_F(A)$) is proportional to the ratio of mismatched edges between the original and permuted graph. A more accessible way of presenting the improvements compares the average number of mismatched edges. While this method is limited since the value is incomparable across different graphs, it is useful when examining one specific instance. In a BA graph on 500 nodes with $k = 5$, which has 2475 edges, the average number of mismatched edges of the original, unguided annealing is ≈ 1916. In contrast, eigenvector-centrality-guided annealing achieves an average of ≈ 1845, translating to an improvement of about 71 fewer mismatched edges.

In conclusion, we can with confidence say that in both of these larger graph settings, guided annealing outperforms the original version. In the case of the BA model, we hypothesize that a possible explanation for this increasing improvement stems from the fact that sparsity increases along with size, assuming k is fixed. Sparsity also often leads to higher symmetry, and we have already seen that guided annealing's improvements are more significant in graphs that are symmetric rather than random (an example is non-existent improvement in the asymmetric ER graph). In the case of the DD model, where the probability of duplication is relative to graph size, this explanation cannot be applied, and we do not know exactly why improvements increase, although the fact that chance during the generation process plays a higher role in small graphs may be significant. Future work could analyze the trend of increasing improvement in even larger instances of random network models and reveal whether the improvements will further amplify.

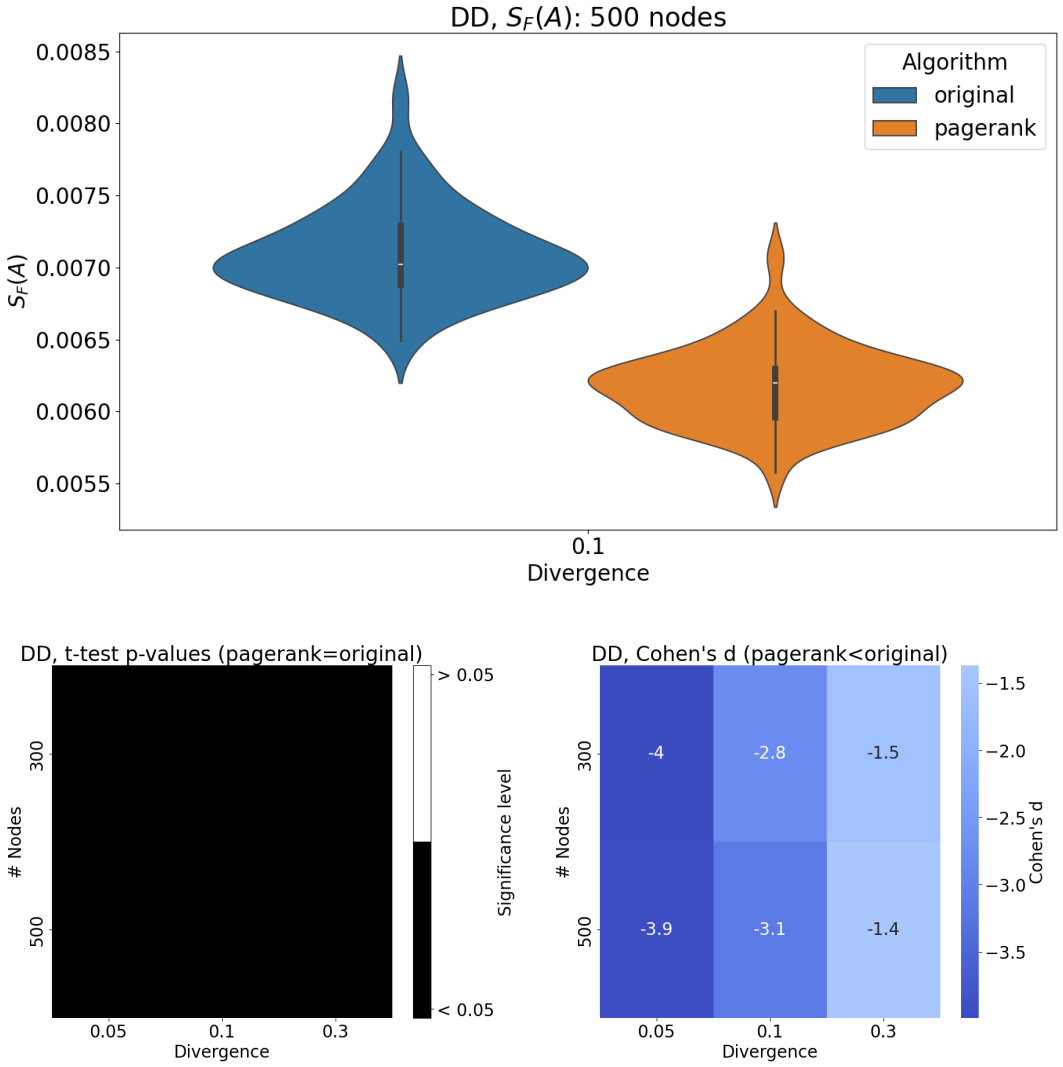

Figure 5.9 Original annealing vs. PageRank-guided annealing on large DD graphs

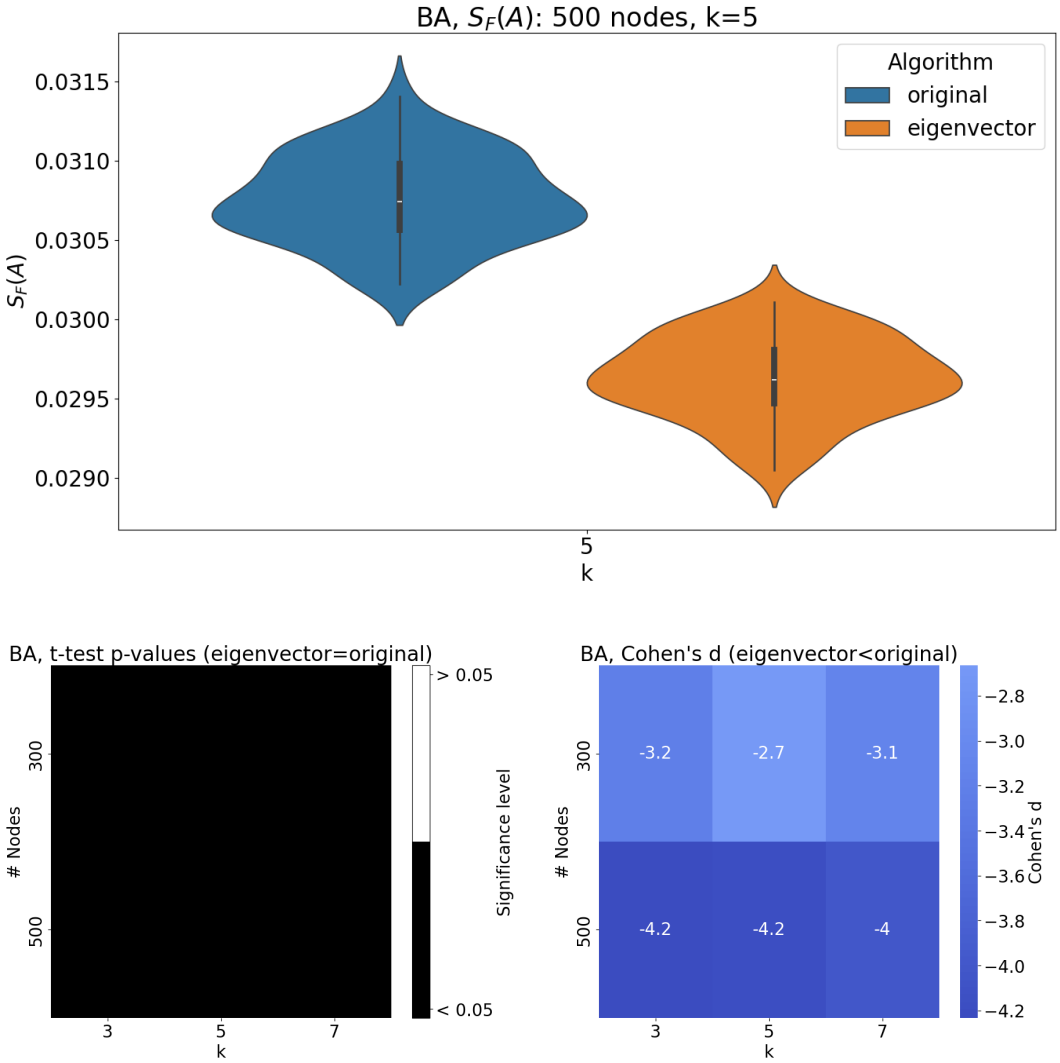

Figure 5.10 Original annealing vs. eigenvector-centrality-guided annealing on large BA graphs. k is the number of links of a new node.

5.3 A greedy approach

To evaluate centrality-guided annealing from another perspective, we compare its performance against a straightforward greedy approach. We introduce a variant of annealing called *gradient descent*, which loosely adheres to the core idea of gradient descent in mathematical optimization, i.e., always moving in the direction of the steepest improvement (this variant, therefore, is annealing only in name, since temperature plays no significant role in it).

The gradient descent version is defined by its re-implemented move function; based on the current permutation, we randomly select $\frac{n}{2}$ possible transpositions and choose the one minimizing energy.

While evaluating the energy function $\frac{n}{2}$ times is unnecessary, and only evaluating the proposed change in energy suffices, the time complexity of the new move function still increases by an order of $O(n)$. Altogether, combined with the orchestration, the gradient descent version becomes practically unusable for graphs of sizes above circa 100 vertices. Due to this limitation, we restrict our tests to graphs with 40 vertices.

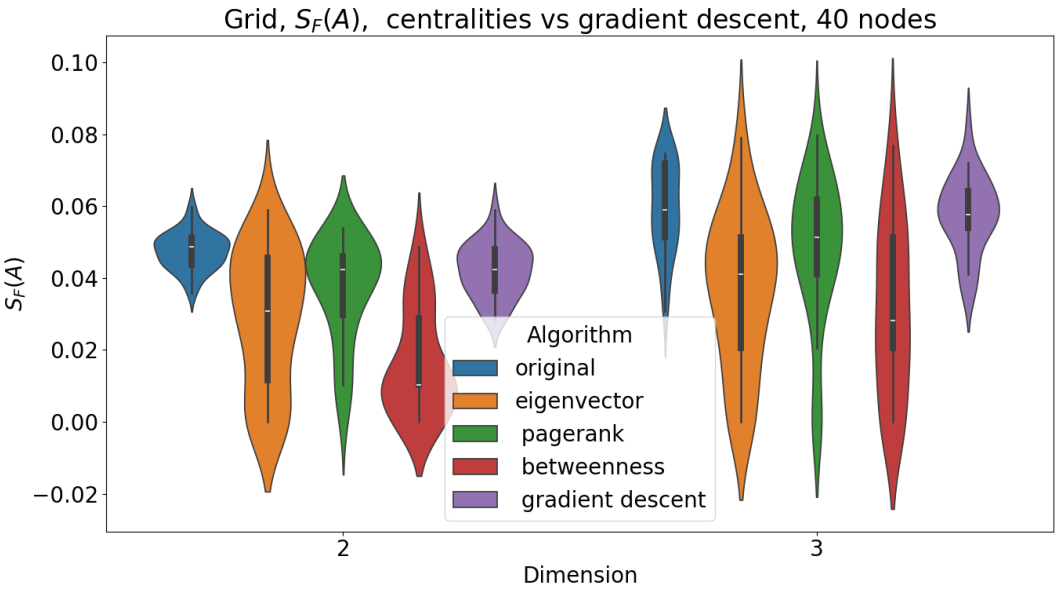

Figure 5.11 Comparison of $S_F(A)$ computed by Gradient descent, centrality-guided, and original versions of annealing on grid graphs with 40 vertices.

Figure 5.11 compares $S_F(A)$ computed by gradient descent, centrality-guided, and original versions on the grids. Similar comparisons on BA and DD graphs are included in the attachment A.8. We observe that even though the greedy approach did introduce some improvements, especially in the 2D case, centrality-guided annealing performs best on grid graphs. In the case of BA graphs, gradient descent performed marginally better than the original version, but its improvements were again not greater than those of eigenvector-guided annealing. The DD was the only model where gradient descent performed slightly better than guided annealing. We recall, however, that improvements of guided annealing became more pronounced for larger DD graphs, so we cannot say in full generality that gradient descent computes symmetry better than guided annealing in the DD model.

In conclusion, the gradient descent variant, characterized by higher computational demands and generally comparable or worse results, cannot be deemed more effective than guided annealing variants.

5.4 Initialized annealing

5.4.1 Lateral Random Model and reshuffled permutations

In previous sections, we analyzed the efficiency of new annealing implementations in detecting symmetry within some selected random models. A different approach considers a symmetric structure where some optimal symmetry is a priori known, initializes the algorithm at some selected distance from the optimum, and analyzes its ability to return to that optimum.

With that intention, we introduce the *Lateral Random Model* (LRM), which is essentially a union of two ER graphs and is motivated by the brain structure [10]. An LRM instance denoted as $L_{n,p,q}$ can be generated by the following procedure:

- Generate $G(\frac{n}{2}, p)$ an ER graph with vertices $V(G) = \{v_1, v_2, ..., v_n\}$
- Generate G', a copy of G with vertices $\{v'_1, v'_2, ..., v'_n\}$, where v'_i is a copy of v_i (including its edges).
- Let L be a union of G and G'.
- Connect each pair of the original vertex and its copy v_i, v'_i in L with probability q.

The resulting graph is an LRM denoted as $L_{n,p,q}$, and the automorphism $f : G \mapsto G'$ defined as $f(v_i) = v'_i$ represents the *left-right symmetry* (LR), which is the optimal symmetry of the LRM. To introduce some irregularity, we introduce a *k-rewired LRM*. One rewiring involves deleting a random edge and adding an edge between two randomly selected unconnected vertices. A k-rewired LRM is generated by rewiring an LRM instance k times. For small values of k, LR is still expected to be the dominant symmetry of the graph [10].

To analyze the ability of an annealing variant to return to optimal symmetry, we initialize it in permutations that are close to the optimum. Considering some permutation P, we alter it by selecting two random vertices and swapping their images under P. Repeating this ℓ times, we obtain an ℓ-reshuffled P [10].

5.4.2 Returning to LR

Figure 5.12 presents measurements comparing PageRank-guided annealing and the original unguided annealing in their ability to return to the LR of an LRM instance when initialized in an ℓ-reshuffled LR. Annealing guided by eigenvector centrality and betweenness performed similarly to PageRank-guided annealing, so we do not present them. The LRM instances are of size 200, and the number of simulations carried out for each instance is 30 [10].

The upper two subplots display the difference of approximate symmetry computed by the algorithm initialized in an ℓ-reshuffled permutation and approximate symmetry corresponding to LR, that is, $S_F(A) - S_F(LR)$. The x-axis represents the number of rewires, while the y-axis represents the number of shuffles ℓ in the initial ℓ-reshuffled LR. The lower two subplots compare the two algorithms by Hamming distance between LR and the computed permutation. An ideal

algorithm would converge to LR, making both the difference $S_F(A) - S_F(LR)$ and Hamming distance from LR zero [10].

The results indicate that the difference between computed symmetry and LR symmetry primarily depends on the number of swaps in the initial permutation. The effect of k rewires is less significant and can be attributed to the fact that as the instance becomes more random, the approximate symmetry corresponding to LR decreases [10]. Guided annealing performs better than original annealing on instances with 5, 50, and 100 swaps in the initial permutation, as signified by the lighter colors in these columns.

Similarly, guided annealing achieves slightly better results in terms of the Hamming distance from LR for some LRM instances. The improvements are visible only in limited parameter combinations, and within the same evaluation framework, guided annealing performs significantly worse than other algorithms computing approximate symmetry, such as approaches formulating the problem as a relaxation of a quadratic assignment problem [10].

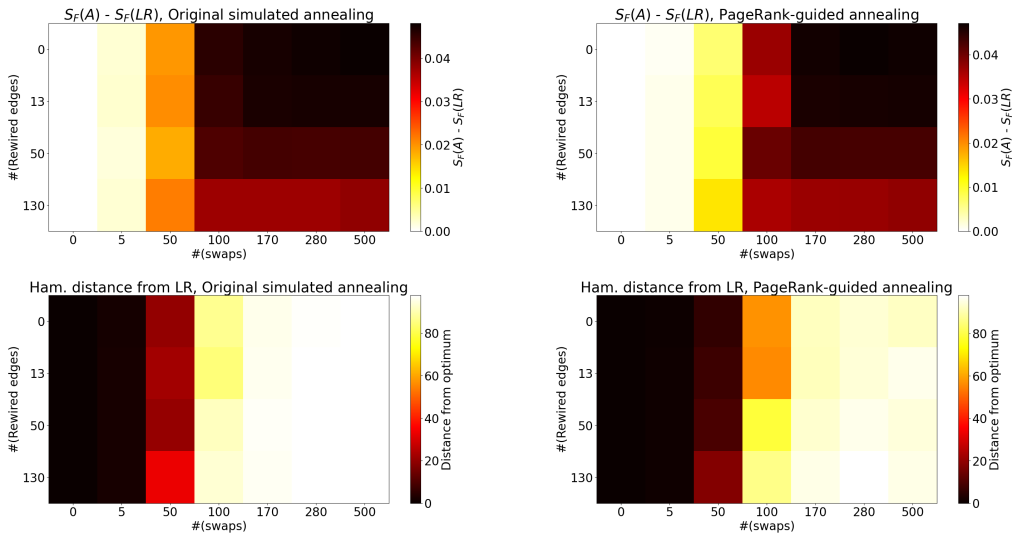

Figure 5.12 Comparison of the performance of original and PageRank-guided annealing when initialized at various distances from LR. The top two subplots depict the difference between computed symmetry and LR symmetry. The bottom two subplots depict the Hamming distance between computed symmetry and LR.

Conclusion

This thesis explored complex networks and algorithms computing approximate network symmetry. Specifically, we focused on the effects of enhancing simulated annealing by graph centralities to improve the computation of approximate network symmetry. We hypothesized that guiding annealing to align vertices with similar centralities would yield better results in finding symmetries than annealing randomly selecting the next state since graph centralities are preserved by automorphisms.

We discussed various graph centralities, including degree centrality, eigenvector centrality, PageRank, betweenness, and clustering coefficient. We later used them to enhance and guide simulated annealing to align similar vertices.

To evaluate new annealing versions, we constructed a dataset consisting of grids, the classic Erdős–Rényi model, the Barabási–Albert model with preferential attachment, and the Duplication-divergence network that models protein-protein interaction networks.

We proposed a variant of annealing dynamically determining the number of fixed points by penalizing them in the objective function instead of receiving the maximum number of fixed points as an arbitrarily set parameter. Our measurements demonstrated that this dynamic version performs similarly to parametrized annealing but eliminates the risk of arbitrarily fixing a large number of vertices. We continued using this dynamic approach to fixed points in all other measurements.

We re-implemented simulated annealing by guiding it to probabilistically align vertices with similar centralities instead of performing random transpositions. Our measurements showed that this enhancement of annealing with PageRank, betweenness, eigenvector centrality, and degree centrality improved symmetry calculation on some of the models in the dataset. The clustering coefficient generally yielded no significant improvements.

Specifically, no centrality improved symmetry calculation on the purely random Erdős–Rényi model, while betweenness provided substantial improvements on grid graphs. In the case of the Barabási-Albert and Duplication-Divergence models, which more closely resemble real-world networks, the improvements were less substantial than in the grid but still statistically significant. Interestingly, in these two models, improvements became more pronounced with growing graph size.

Comparisons with a simple greedy algorithm that always selects the next best state revealed that this greedy method did not outperform centrality-guided annealing and was computationally more demanding.

We measured guided annealing's ability to return to the optimal solution when initialized close to it. Its improvements were detectable for instances of the Lateral Random Model with few rewires. However, they were marginal and outmatched by approaches different from simulated annealing.

We confirmed that guiding annealing with graph centralities improves the computation of approximate symmetry. It has been shown other algorithms are, in some settings, more effective than simulated annealing, and future work could examine how other metaheuristic approaches, such as evolutionary algorithms, benefit from centrality-based enhancement. It also would be interesting to gain

a deeper understanding of certain features of the solution, e.g., examine how the orbits of the computed permutation differ on various network models and how introducing centrality guidance affects the orbits. Furthermore, it would be interesting to test how guided annealing performs on real-world datasets, such as brain connectome networks.

Bibliography

1. Abani, Noor; Braun, Torsten; Gerla, Mario. *Betweenness centrality and cache privacy in information-centric networks*. 2018. Available from doi: 10.1145/3267955.3267964.
2. Hong, Jungyeol; Tamakloe, Reuben; Lee, Soobeom; Park, Dongjoo. *Exploring the Topological Characteristics of Complex Public Transportation Networks: Focus on Variations in Both Single and Integrated Systems in the Seoul Metropolitan Area*. 2019. Available from doi: 10.3390/su11195404.
3. Xiaohu, Zhao; Yong, Liu; Xiangbin, Wang; Bing, Liu; Qian, Xi; Qihao, Guo; Hong, Jiang; Tianzi, Jiang; Peijun, Wang. *Disrupted small-world brain networks in moderate Alzheimer's disease: a resting-state FMRI study*. 2012. Available from doi: 10.1371/journal.pone.0033540.
4. Niewiadomska-Szynkiewicz, Ewa. *Application of Social Network Analysis to the Investigation of Interpersonal Connections*. 2012. Available from doi: 10.26636/jtit.2012.2.1268.
5. MacArthur, Ben; Sanchez-Garcia, Ruben; Anderson, James. *Symmetry in complex networks*. 2008.
6. Ball, Fabian; Geyer-Schulz, Andreas. *How Symmetric Are Real-World Graphs? A Large-Scale Study*. 2018. Available also from: https://www.mdpi.com/2073-8994/10/1/29.
7. Cho, Young; Nishikawa, Takashi; Motter, Adilson. *Stable Chimeras and Independently Synchronizable Clusters*. American Physical Society, 2017. Available from doi: 10.1103/PhysRevLett.119.084101.
8. Liu, Yanchen. *Approximate Network Symmetry*. 2020. Available from arXiv: 2012.05129.
9. Straka, Matej. *Approximative symmetries of complex networks*. [N.d.]. Available also from: https://dspace.cuni.cz/handle/20.500.11956/174639.
10. Pidnebesna, Anna; Hartman, David; Pokorná, Aneta; Straka, Matěj; Hlinka, Jaroslav. *Computing approximate symmetries of complex networks*. 2023. Available from arXiv: 2312.08042.
11. Barabási, Albert-László. *Network Science*. The Royal Society Publishing, [n.d.]. Available also from: http://networksciencebook.com/.
12. Biggs, Norman. *Algebraic Graph Theory*. Cambridge University Press, 1974. Cambridge Mathematical Library. Available also from: https://superoles.wordpress.com/wp-content/uploads/2015/09/n-biggs-algebraic-graph-theory-1993.pdf.
13. Kirkpatrick, Scott; Gelatt, Daniel C.; Vecchi, Mario M. *Optimization by Simulated Annealing*. 1983. Available from doi: 10.1126/science.220.4598.671.
14. Pokorná, Aneta. *Characteristics of network centralities*. [N.d.]. Available also from: https://dspace.cuni.cz/bitstream/handle/20.500.11956/118612/120362594.pdf?sequence=1&isAllowed=y.

15. Newman, Mark. *Networks: An Introduction*. Oxford University Press, 2010. Available from DOI: 10.1093/acprof:oso/9780199206650.001.0001.

16. Ghorbani, Modjtaba; Dehmer, Matthias; Lotfi, Abdullah; Amraei, Najaf; Mowshowitz, Abbe; Emmert-Streib, Frank. *On the relationship between PageRank and automorphisms of a graph*. 2021. Available from DOI: 10.1016/j.ins.2021.08.013.

17. Holme, Peter. *Firsts in network science*. [N.d.]. Available also from: https://petterhol.me/2019/04/15/firsts-in-network-science/.

18. Brin, Sergey; Page, Lawrence. *The anatomy of a large-scale hypertextual Web search engine*. 1998. Available from DOI: https://doi.org/10.1016/S0169-7552(98)00110-X.

19. Grolmusz, Vince. *A note on the PageRank of undirected graphs*. 2015. Available from DOI: https://doi.org/10.1016/j.ipl.2015.02.015.

20. Watts, Duncan; Strogatz, Steven. *Collective dynamics of 'small-world' networks*. 1998. Available from DOI: 10.1038/30918.

21. Erdös, Paul; Rényi, Alfred. *On Random Graphs I*. 1959. Available also from: https://snap.stanford.edu/class/cs224w-readings/erdos59random.pdf.

22. Ispolatov, Yaroslav; Krapivsky, Pavel; Yuryev, Anton. Duplication-divergence model of protein interaction network. *Physical Review E*. 2005. Available from DOI: 10.1103/physreve.71.061911.

23. Brandes, Ulrik. *A faster algorithm for betweenness centrality**. Routledge, 2001. Available from DOI: 10.1080/0022250X.2001.9990249.

A Appendix

A.1 DFP-SA vs. KFP-SA

As an extension to section 5.1.1, we present further analysis of the performances of DFP-SA and KFP-SA.

The results we reached are similar to those in section 5.1.1. We also compare the two approaches to FPs in the improved annealing implementation guided by centralities, meaning we compare the performance of guided annealing with parametrized FPs and guided annealing with dynamic FPs. The results again suggest similar performance of both versions, confirming that the dynamic approach is also valid in guided annealing.

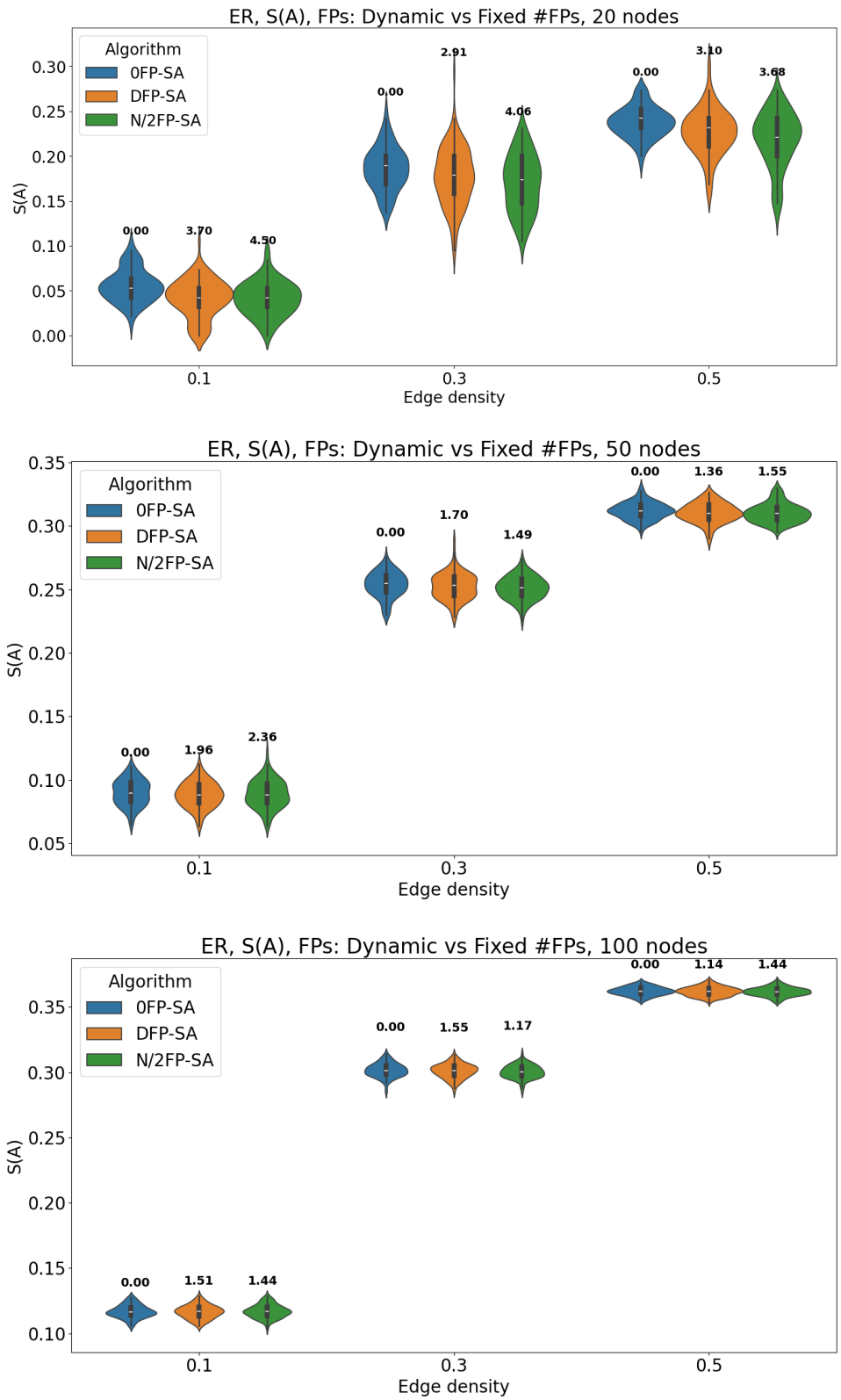

Figure A.1 Comparison of the performance of DFP-SA and KFP-SA (with K set to 0 and N/2) on ER graphs.

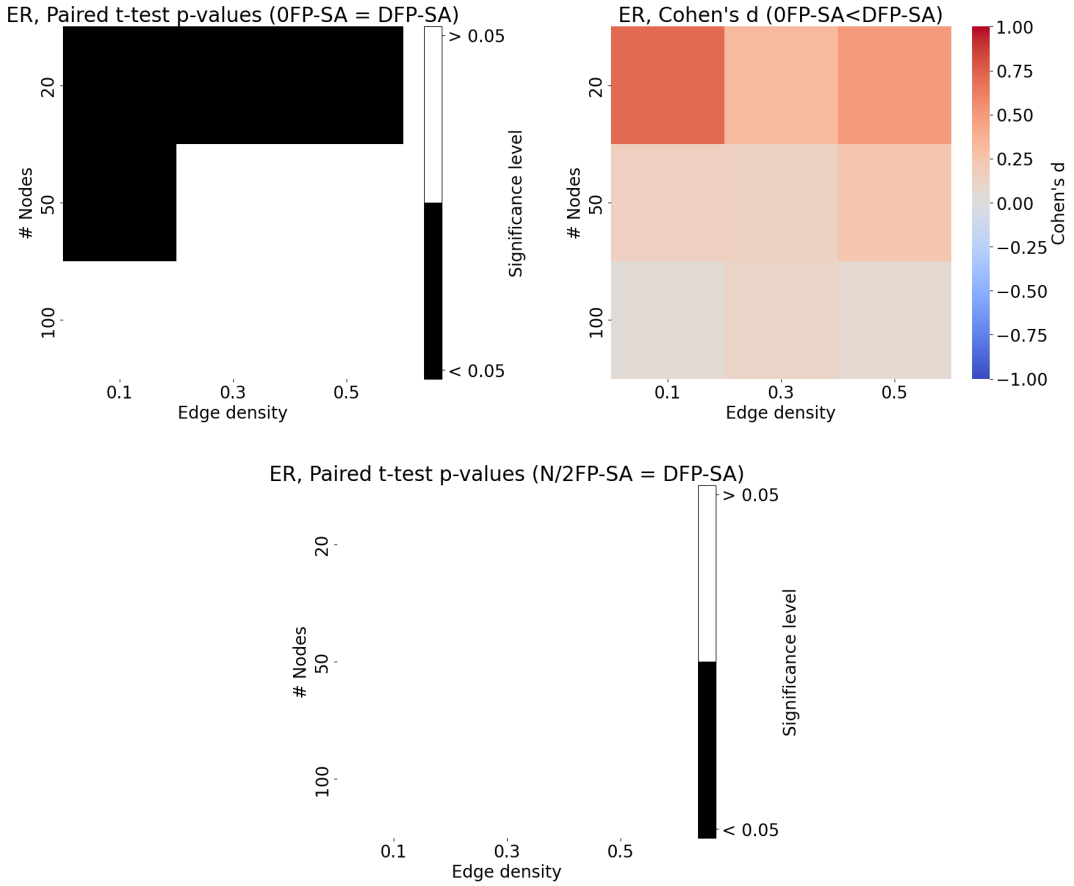

Figure A.2 Statistical analysis of the performance of DFP-SA and KFP-SA (with K set to 0 and N/2) on ER graphs.

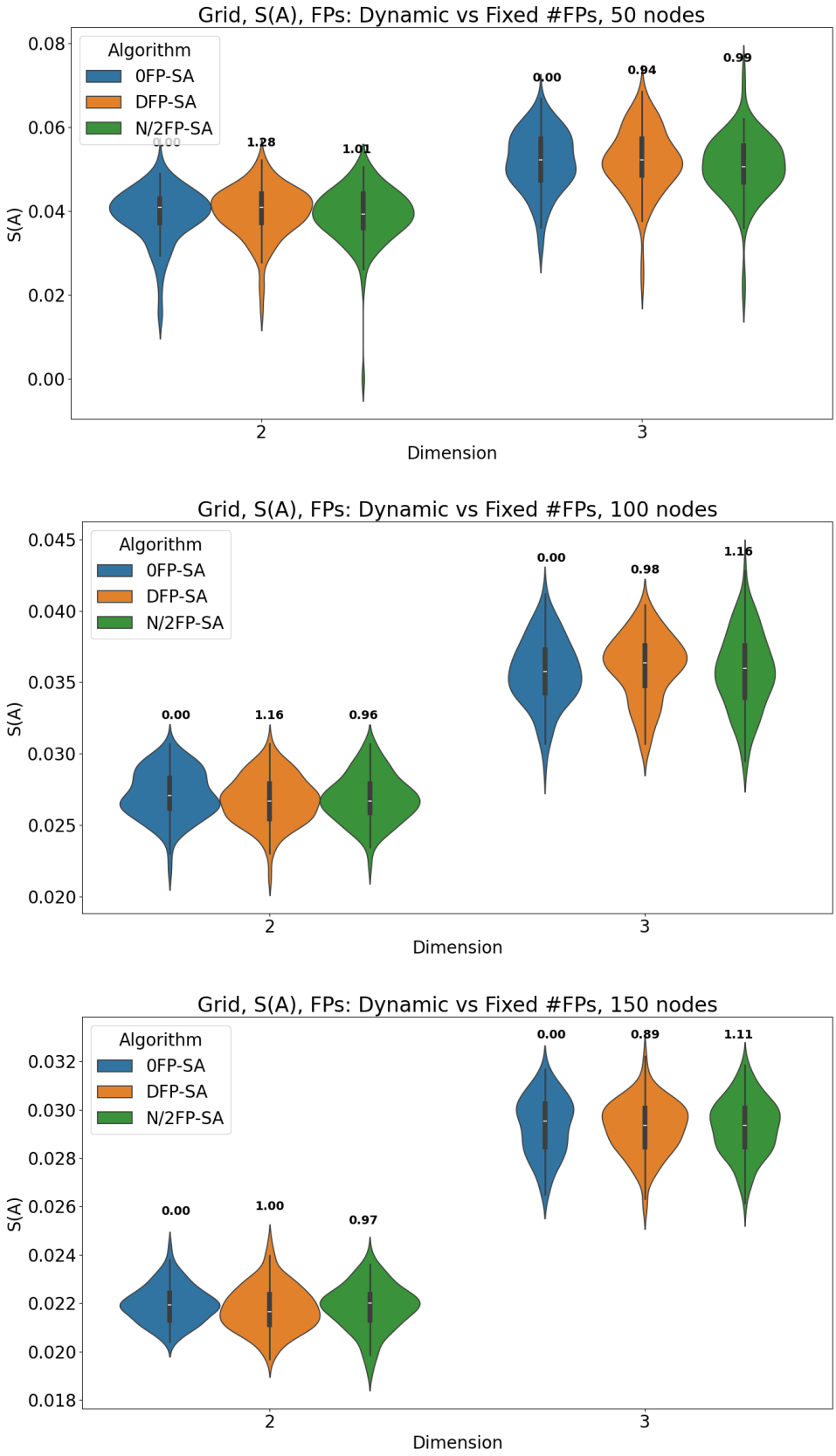

Figure A.3 Comparison of the performance of DFP-SA and KFP-SA (with K set to 0 and N/2) on grid graphs. The differences between the performance of the versions were not statistically significant.

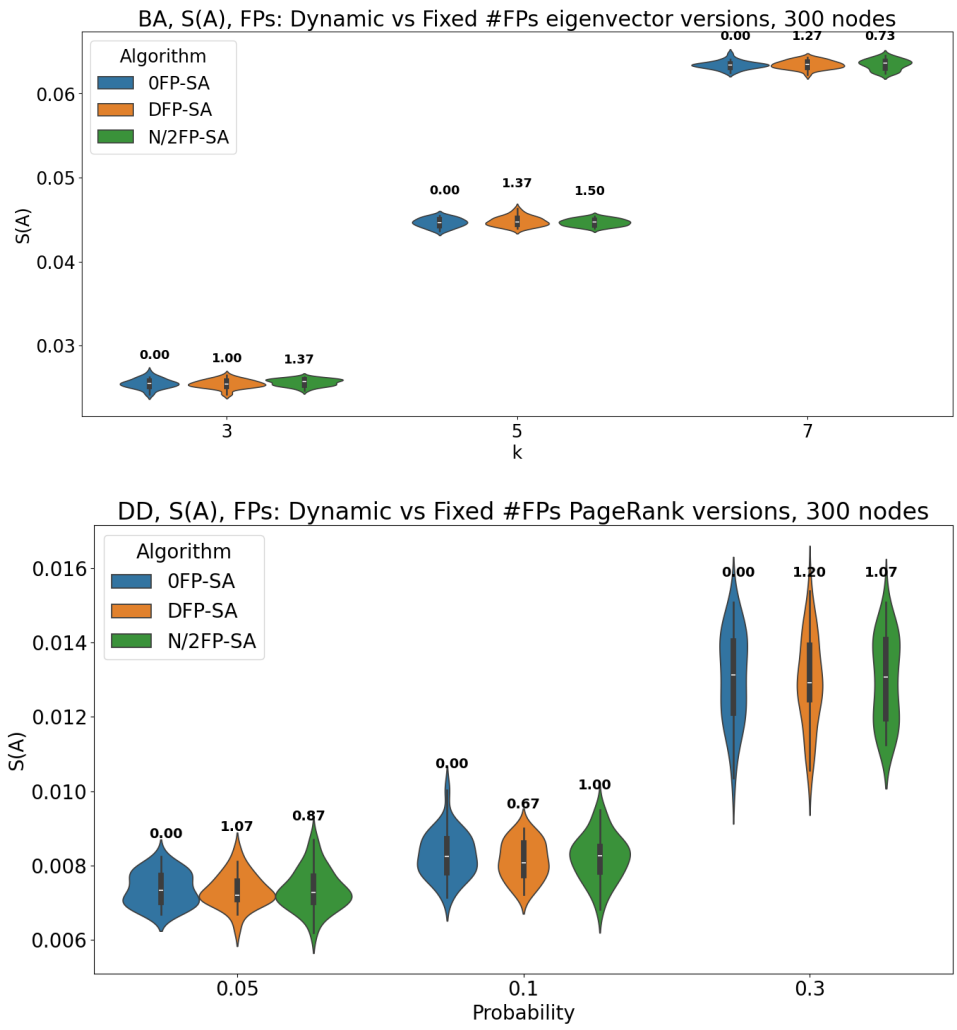

Figure A.4 Comparison of the performance of DFP-SA and KFP-SA (with K set to 0 and N/2) on larger BA and DD graphs. This is a comparison of improved annealing versions guided by centralities (as introduced in Chapter 5). The results are similar to previous measurements, meaning the dynamic approach performs similarly to the original version. There was no statistically significant difference between the performance of the two versions.

A.2 Comparison of annealing versions guided by different centralities

This section presents the measurements shown in section 5.2 in their entirety. These measurements compare approximate symmetry measured by annealing versions guided by different centralities on grid graphs, BA graphs, and DD graphs.

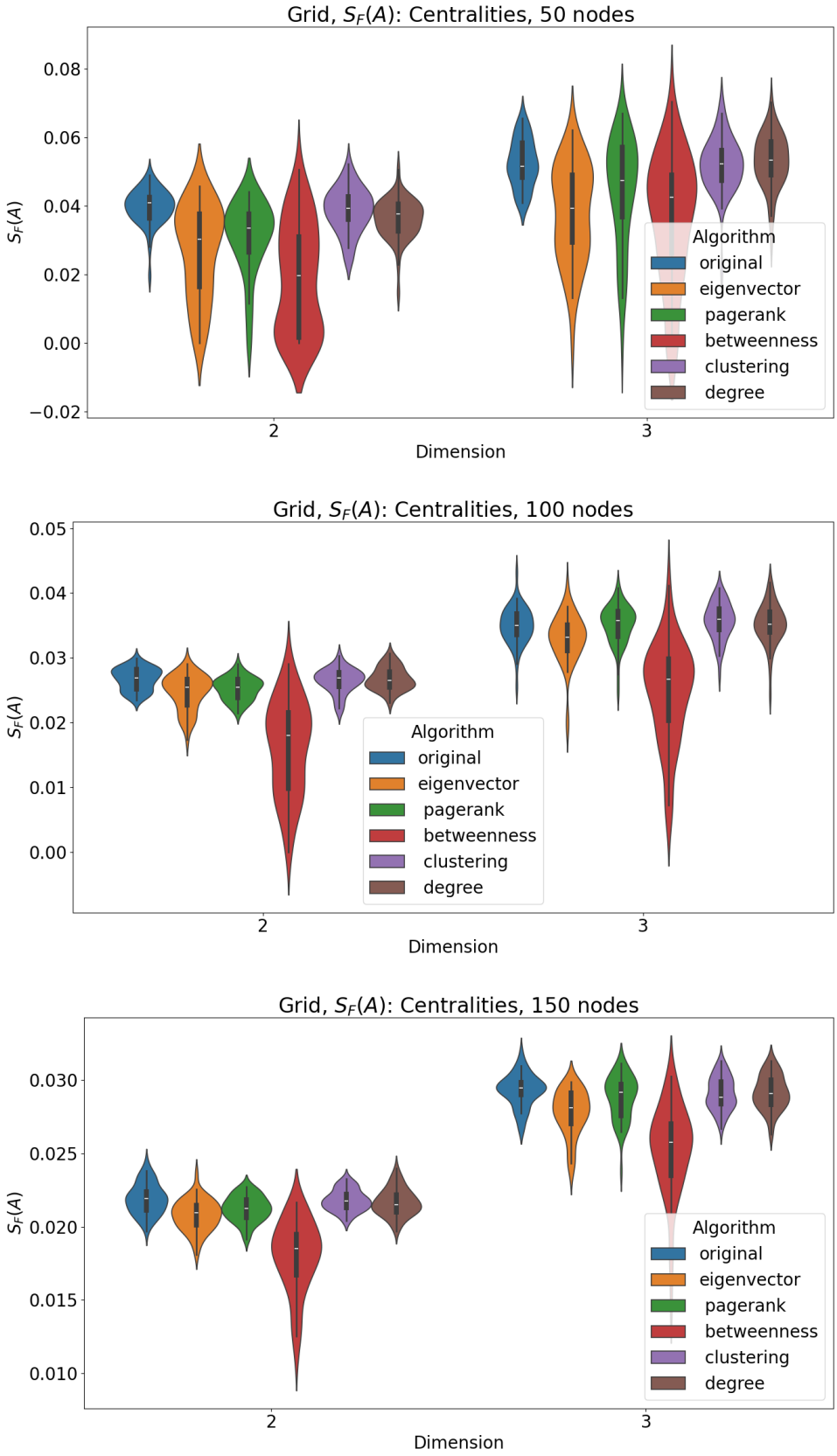

Figure A.5 Comparison of the performance of simulated annealing guided by different centralities on grid graphs.

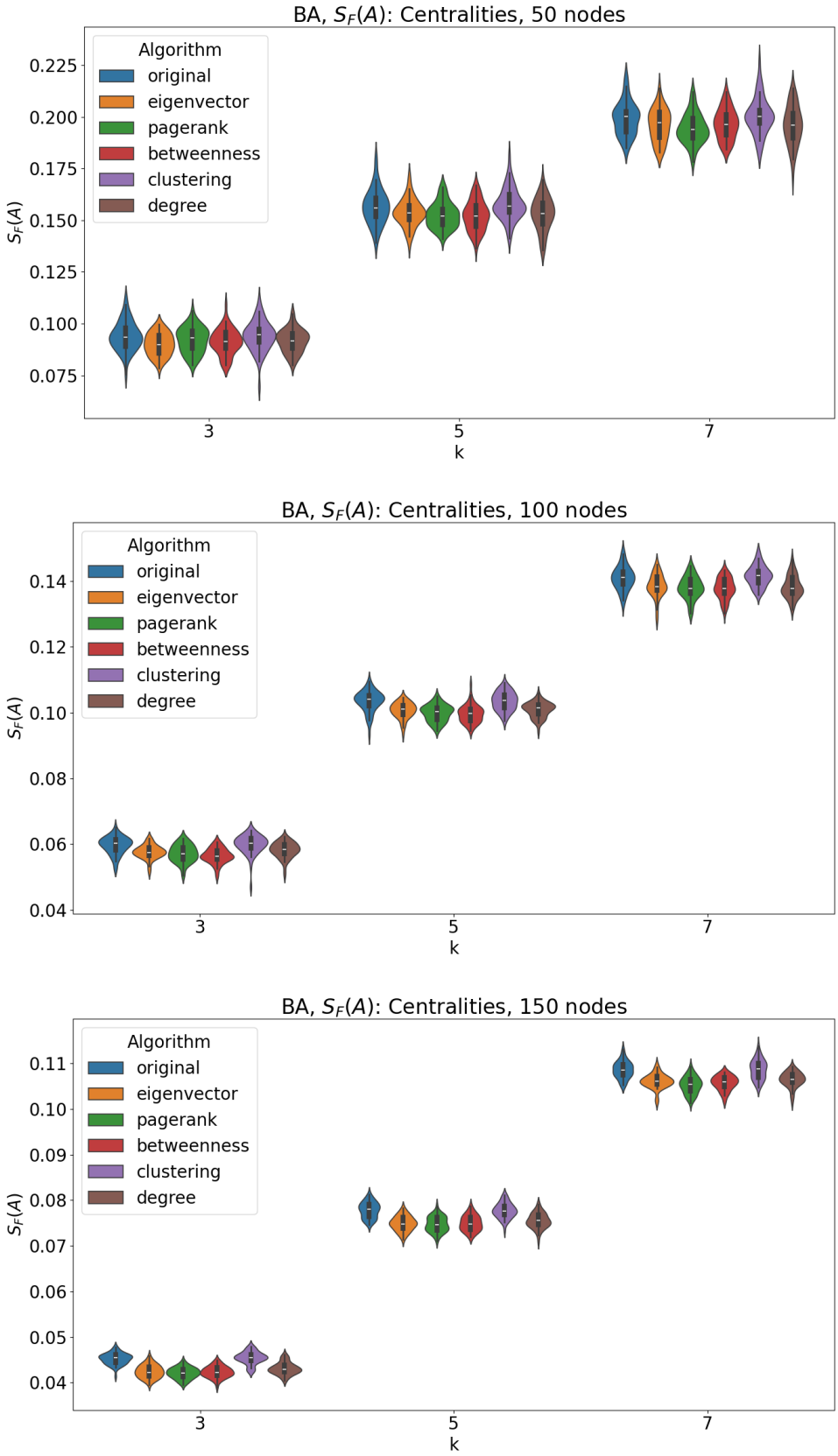

Figure A.6 Comparison of the performance of simulated annealing guided by different centralities on BA graphs.

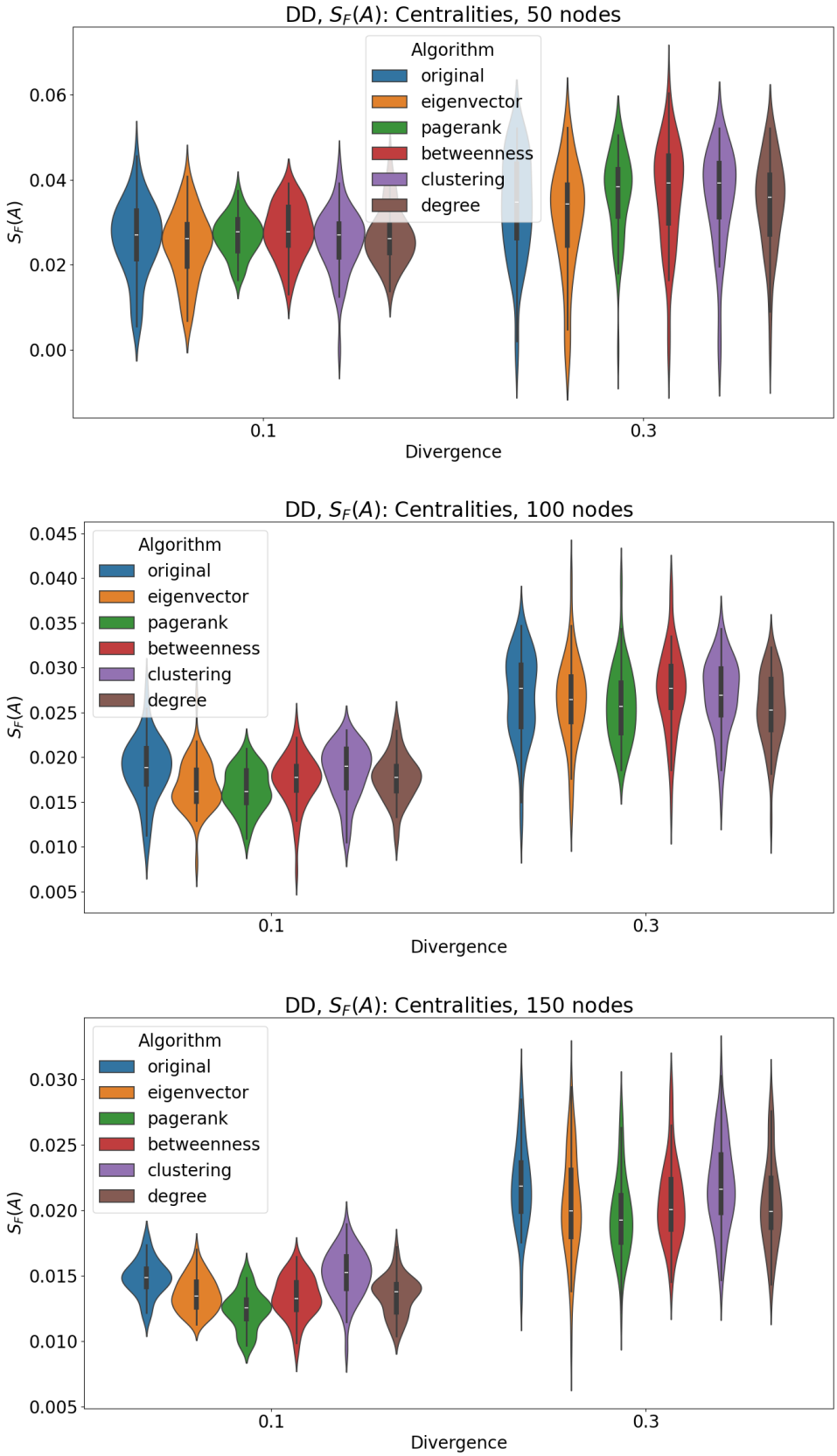

Figure A.7 Comparison of the performance of simulated annealing guided by different centralities on DD graphs.

A.3 Comparison of gradient descent and guided annealing

This section displays additional measurements comparing symmetry computed by the gradient descent annealing version and guided annealing.

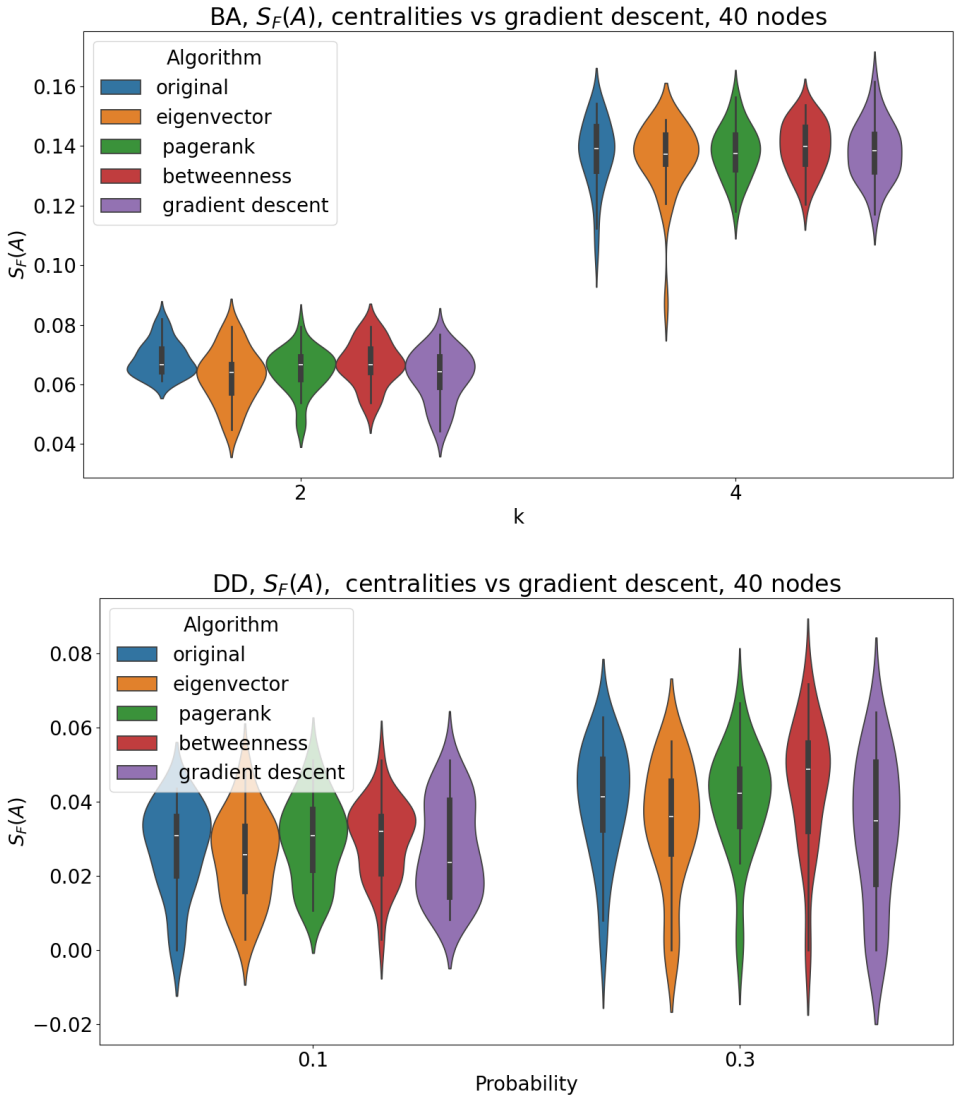

Figure A.8 Comparison of symmetry computed by gradient descent, centrality-guided, and original versions of annealing on BA and DD graphs on 40 vertices.