FUGAL: Feature-fortified Unrestricted Graph Alignment

Aditya Bommakanti
IIT Delhi
adityabommakanti2002@gmail.com

Harshith Reddy Vonteri
IIT Delhi
harshithreddyvonteri@gmail.com

Konstantinos Skitsas
Aarhus University
skitsas@cs.au.dk

Sayan Ranu
IIT Delhi
sayanranu@cse.iitd.ac.in

Davide Mottin
Aarhus University
davide@cs.au.dk

Panagiotis Karras
University of Copenhagen & Aarhus University
piekarras@gmail.com

Abstract

The necessity to align two graphs, minimizing a structural distance metric, is prevalent in biology, chemistry, recommender systems, and social network analysis. Due to the problem's **NP**-hardness, prevailing graph alignment methods follow a *modular* and *mediated* approach, solving the problem restricted to the domain of intermediary graph representations or products like embeddings, spectra, and graph signals. Restricting the problem to this intermediate space may distort the original problem and are hence predisposed to miss high-quality solutions. In this paper, we propose an *unrestricted* method, FUGAL, which finds a permutation matrix that maps one graph to another by directly operating on their adjacency matrices with judicious constraint relaxation. Extensive experimentation demonstrates that FUGAL consistently surpasses state-of-the-art graph alignment methods in accuracy across all benchmark datasets without encumbering efficiency.

1 Introduction and Related Work

Graph alignment seeks to match a pair of graphs to each other, i.e., to correlate nodes of one graph to those of the other. For instance, biological systems such as protein-protein interaction networks and gene regulatory networks can be represented as graphs. The alignment of such biological networks across species reveals *orthologous* proteins or genes (i.e., homologous genes that evolved from a common ancestor) and thereby conveys the biological function of uncharted genes in one species through their better-studied counterparts in another species [37, 36]. The same problem also arises in other high-impact network science tasks [11], such as identifying users in social networks [22] and feature matching in computer vision [5, 6]. The problem can be formulated as an instance of the *quadratic assignment problem* (QAP) [13, 24] between nodes of the two graphs, which treats the edges in one graph as units of *flow* and the edges in the other graph as *distances* between nodes. This relation renders the problem **APX**-hard to approximate even within an approximation factor that grows linearly with the number of nodes [30, 13].

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

1.1 Related Works

Owing to the problem's hardness, several heuristics have been proposed. Nevertheless, state-of-the-art graph alignment methods refrain from directly addressing the edge-aware QAP. Instead, they craft intermediate representations of nodes that allow for the computation of *similarities* and settle for solving an *assignment problem* over those representations. We call these methods *mediated* due to their restriction to *intermediary* graph representations. While the transformation from the original graph space to an intermediate space enables computational efficiency, the transformation incurs loss of information. In this work, we propose an *unrestricted* graph alignment method that avoids restricting the problem to an intermediate space, while also retaining efficiency. We call our method "unrestricted" rather than "unmediated" since, while we retain the full graph information in the core QAP, we also avail of help from mediated representations to solve the QAP. In the subsequent discussion, we summarize the various mediated and unmediated approaches in the literature.

Mediated Approaches: GWL [44] jointly learns embeddings and alignments using the dissimilarity notion of Gromov-Wasserstein discrepancy; it estimates distance matrices using the embeddings when learning the optimal transport, and regularizes the learning of embeddings using the learned transport. S-GWL [43] addresses the scalability drawback of GWL by adopting a partitioning method on the input graphs. CONE [4] models intra-network proximity with node embeddings and uses them to match nodes across networks after aligning embedding subspaces. REGAL [17] identifies node matchings by greedily aligning their latent feature representations learnt from graph structures. GRAMPA [13] constructs a similarity matrix as a weighted sum of outer products between all pairs of eigenvectors of the two graphs. GRASP [20] uses the spectral properties of the graphs grounded on the eigenvectors of their normalized Laplacian matrices. IsoRank [37] uses neighborhood similarity to extract structural graph information and recursively updates the score of a node pair using the score of their neighbors. GRAAL [25] is a greedy alignment method that matches nodes using a similarity score based on a dictionary of small frequent graph patterns. GOT [31] employs the probabilistic distribution of smooth graph signals defined with respect to the graph topology, and seeks alignments by minimizing the distance between these graph signal distributions. fGOT [32] adopts a dissimilarity metric that aligns two graphs using the probability distribution of data generated via graph filters. PARROT [46] presents a semi-supervised methodology which encodes graph topology through random walks with restart (RWR) for a position-aware transport cost and addresses a regularized Optimal Transport (OT) problem to determine node mappings. GW [35] and FGW [40] compute Gromov-Wasserstein discrepancy using similarity matrices of shortest path distances between nodes.

Unmediated Approaches: FAQ [42] is an unmediated algorithm that addresses the QAP by relaxing constraints to attain computational tractability. GLAG [14] proposes a problem formulation that retains the full graph information and relaxes the permutation constraints. As we will see in § 5, these methods of relaxing constraints lead to inferior accuracy.

Unrestricted Approaches: In addition to the full graph information, PATH [45] and FGM [49] also use feature matching, while DSPP [10] employs all-pairs-shortest-paths for graph alignment. We characterize these methods as unrestricted, since they also avail of help from mediated representations. As we will see in § 5, these methods are significantly inferior to FUGAL in terms of accuracy.

1.2 Contributions

Optimization problem formulation: We present FUGAL (*Feature-fortified Unrestricted Graph Alignment*), a graph alignment method that retains full graph information by integrating the quadratic assignment problem (QAP) in the optimization objective. To augment quality, we utilize a regularizer in the form of a linear assignment (LAP) supplement incorporating graph structural features.
Unrestricted solution: FUGAL relaxes the solution space to *doubly stochastic* matrices and uses a customized optimization strategy that guides the Frank-Wolfe algorithm [16] through a Sinkhorn distance objective [7] to steer the resulting doubly stochastic solution towards a *quasi-permutation* matrix. We call our approach *unrestricted*, as it does not rely *solely* on intermediate representations. On the other hand, it is not entirely *unmediated*, as the LAP regularizer using structural features is mediated. Thereby, we retain the full graph information and also enable mediating representations to efficiently guide the optimization process and thereby enable both efficacy and tractability.
Experimental evaluation: Through extensive experimentation with real-world and synthetic datasets across varying graph density and noise levels, we demonstrate that FUGAL outperforms state-of-the-art graph alignment methods in accuracy without a detrimental efficiency overhead.

2 Problem Formulation

Definition 2.1. Let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ denote an unlabelled, undirected *graph*, where $\mathcal{V}$ is the set of nodes each identified by a number $[n] = \{1, \ldots, n\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set. The adjacency matrix of $\mathcal{G}$ is $\mathbf{A} \in \{0,1\}^{n \times n}$ such that $a_{ij} = a_{ji} = 1$ if and only if $(i, j) \in \mathcal{E}$.

We denote an all-ones vector as $\mathbf{1}$, an all-ones square matrix as $\mathbf{J}$, and an all-zero square matrix as $\mathbf{O}$. The dimensions of entities are inferred from the equations employing them.

Definition 2.2. We denote the set of binary-valued *permutation matrices* as $\mathbb{P}^n = \{\mathbf{P} \in \{0,1\}^{n \times n} : \mathbf{P1} = \mathbf{1}, \mathbf{P}^\top \mathbf{1} = \mathbf{1}\}$ and that of real-valued *doubly stochastic* matrices as $\mathbb{W}^n = \{\mathbf{W} \in [0,1]^{n \times n} : \mathbf{W1} = \mathbf{1}, \mathbf{W}^\top \mathbf{1} = \mathbf{1}\}$.

Definition 2.3. Let $\mathbf{A} = [a_{ij}]_{i \in [n], j \in [m]} \in \mathbb{R}^{n \times m}$. We denote the *Frobenius norm* as the entry-wise 2-norm $\|\mathbf{A}\|_F = \left(\sum_{i=1}^{n} \sum_{j=1}^{m} |a_{ij}|^2\right)^{1/2}$.

Definition 2.4. We denote the *trace* of a matrix $\mathbf{A}$ as $\mathrm{tr}(\mathbf{A})$.

Theorem 2.5. *A doubly-stochastic matrix* $\mathbf{A}$ *with* $\mathrm{tr}(\mathbf{A}^\top(\mathbf{J} - \mathbf{A})) = 0$ *is a permutation matrix.*

Proof. From $\mathrm{tr}(\mathbf{A}^\top(\mathbf{J} - \mathbf{A})) = 0$ follows that $\sum_i \sum_j a_{ij} \cdot (1 - a_{ij}) = 0$. Since $\mathbf{A}$ is doubly-stochastic, $0 \leq a_{ij} \leq 1$ for all i and j. Thus, $a_{ij} \cdot (1 - a_{ij}) \geq 0$ for $1 \leq i, j \leq n$. Therefore, $a_{ij} \cdot (1 - a_{ij}) = 0$ for all i and j. As a consequence, $a_{ij} \in \{0, 1\}$ for each i and j. Given that $\mathbf{A}$ is doubly-stochastic and all its entries are either 0 or 1, by Definition 2.2, $\mathbf{A}$ is a permutation matrix. □

Problem 1 (Unmediated Graph Alignment)**.** *Consider two graphs* $\mathcal{G}_1 := (\mathcal{V}_1, \mathcal{E}_1)$ *and* $\mathcal{G}_2 := (\mathcal{V}_2, \mathcal{E}_2)$ *with adjacency matrices* $\mathbf{A}, \mathbf{B}$ *respectively. The objective of* unmediated graph alignment *is to identify a bijection* $f : \mathcal{V}_1 \to \mathcal{V}_2$ *between the two graphs that minimizes the number of edge disagreements. Formally, the problem is expressed as:*

$$\min_{\mathbf{P} \in \mathbb{P}^n} \|\mathbf{AP} - \mathbf{PB}\|_F^2, \tag{1}$$

where $\mathbb{P}^n$ *denotes the set of permutation matrices.*

The appellation *unmediated* denotes that we seek a correspondence among nodes without using any information other than the adjacency matrix. The problem is an instance of the **NP**-hard *quadratic assignment problem* (QAP) [24].

Due to the problem's hardness, a popular approximation path utilizes *intermediaries* such as node embeddings. A *mediated* graph alignment is thus expressed as a linear assignment between embeddings rather than a quadratic assignment between adjacency matrices:

$$\min_{\mathbf{P} \in \mathbb{P}^n} \|\mathbf{E}_1 - \mathbf{PE}_2\|_F^2 \tag{2}$$

where $\mathbf{E}_k \in \mathbb{R}^{|\mathcal{V}_k| \times F}$ is the embedding matrix of $\mathcal{G}_k$ and $\mathbf{E}_k[i, :]$ is the F-dimensional vector representation of node i of $\mathcal{G}_k$. The optimization problem in Equation (2) is a *linear assignment problem* (LAP), which is solvable optimally in $\mathcal{O}\left(N^3\right)$ by the Hungarian algorithm [26], while sub-optimal solutions reduce complexity to $\mathcal{O}\left(N^2\right)$.

Extension to graphs of unequal sizes. Consider two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ with node counts n_1 and n_2, respectively ($n_1 < n_2$). To enable alignment despite the size difference, we augment $\mathcal{G}_1$ with $(n_2 - n_1)$ isolated dummy nodes and discard mappings involving dummy nodes from the output.

3 FUGAL

To design FUGAL, we augment the core QAP of Eq. (1) with a LAP supplement that leverages simple *structural* graph features (§ 3.1) to form a unified optimization problem over the set of permutation matrices $\mathbb{P}^n$ (§ 3.2). As this problem is **NP**-hard, we relax its solution space to the set of *doubly stochastic* matrices $\mathbb{W}^n$ (§ 3.3), a superset of the set of permutation matrices. We refine the solution to obtain a *quasi-permutation matrix*, i.e., *almost* a permutation matrix, which we adjust to a permutation matrix that signifies a valid alignment by solving a simple LAP using the *Hungarian* algorithm [26]. We dub this approach "unrestricted" as it eschews the information loss incurred by mediated solutions, which rely solely on intermediary representations. However, we still employ supplementary mediating representations to ensure tractability and efficiency.

3.1 LAP Formulation

Here, we formalize the Linear Assignment Problem (LAP), which is auxiliary to our framework. We construct a node feature vector using four structural features proposed in NETSIMILE [2]. This includes **(1)** d_i, the degree of node v_i, **(2)** c_i, the clustering coefficient of v_i, **(3)** $\bar{d}_{N_i}$, the mean degree of v_i's neighbors, **(4)** and $\bar{c}_{N_i}$, the mean clustering coefficient of v_i's neighbours. Other features, such as betweenness centrality, PageRank, may also be used. Ultimately, the decision resides on the trade-off between the utility of including these features on alignment quality and the efficiency of computing these features.

Using these features, we construct a feature matrix $\mathbf{F}_k \in \mathbb{R}^{|\mathcal{V}_k| \times 4}$ for each graph $\mathcal{G}_k$ and, by the rationale that the structural features of corresponding nodes are similar, we formulate a Linear Assignment Problem for $\mathcal{G}_1$ and $\mathcal{G}_2$ as:

$$\min_{\mathbf{P} \in \mathbb{P}^n} \|\mathbf{F}_1 - \mathbf{P}\mathbf{F}_2\|_F^2 \tag{3}$$

By the Frobenius norm definition, Eq. (3) is equivalent to:

$$\min_{\mathbf{P} \in \mathbb{P}^n} \sum_i \|\mathbf{F}_1[i,:] - \sum_j \mathbf{P}_{ij}\mathbf{F}_2[j,:]\|_F^2 \tag{4}$$

Utilizing the property of permutation matrices that each row contains only one 1, we reformulate Eq. (4) to:

$$\min_{\mathbf{P} \in \mathbb{P}^n} \sum_{i,j} \mathbf{P}_{ij}\|\mathbf{F}_1[i,:] - \mathbf{F}_2[j,:]\|_F^2 = \min_{\mathbf{P} \in \mathbb{P}^n} \sum_{i,j} \mathbf{P}_{ij}\mathbf{D}_{ij} \tag{5}$$

where $\mathbf{D}$ is a distance matrix with $\mathbf{D}_{ij}$ denoting the squared Euclidean distance between $\mathbf{F}_1[i,:]$ and $\mathbf{F}_2[j,:]$. Since each row $\mathbf{P}[i,:]$ contributes exactly one term to this sum, being the element of $\mathbf{D}$ corresponding to the single 1 entry in $\mathbf{P}[i,:]$, the result is equal to the trace of the matrix product:

$$\min_{\mathbf{P} \in \mathbb{P}^n} \operatorname{tr}(\mathbf{P}^\top \mathbf{D}) \tag{6}$$

3.2 Optimization Problem

Our problem formulation augments the QAP of Eq. (1) with a LAP regularizing term as in Eq. (6):

$$\min_{\mathbf{P} \in \mathbb{P}^n} \|\mathbf{A}\mathbf{P} - \mathbf{P}\mathbf{B}\|_F^2 + \mu \cdot \operatorname{tr}(\mathbf{P}^\top \mathbf{D}) \tag{7}$$

where $\mathbf{A}$ and $\mathbf{B}$ denote the adjacency matrices of $\mathcal{G}_1$ and $\mathcal{G}_2$, respectively, $\mathbf{D}$ follows Eq. (6), and μ regulates the LAP's significance; since $\mathbf{P}\mathbf{P}^\top = \mathbf{I}$, this is expanded to:

$$\min_{\mathbf{P} \in \mathbb{P}^n} \operatorname{tr}(\mathbf{A}^\top \mathbf{A}) + \operatorname{tr}(\mathbf{B}^\top \mathbf{B}) - 2\operatorname{tr}(\mathbf{A}\mathbf{P}\mathbf{B}^\top \mathbf{P}^\top) + \mu \cdot \operatorname{tr}(\mathbf{P}^\top \mathbf{D}) \tag{8}$$

equivalently, ignoring constant terms and reversing the sign,

$$\max_{\mathbf{P} \in \mathbb{P}^n} \operatorname{tr}(\mathbf{A}\mathbf{P}\mathbf{B}^\top \mathbf{P}^\top) - \mu \cdot \operatorname{tr}(\mathbf{P}^\top \mathbf{D}) \tag{9}$$

In the case of $\mu = 0$, the first term alone corresponds to the maxQAP problem [30], which is $\mathbf{APX}$-hard to approximate even within an approximation factor that grows linearly with the number of nodes. Given this hardness of the QAP alone and the fact that relaxing combinatorial constraints often results in a substantial deterioration of solution quality, we introduce the LAP regularization to ground the QAP solution on pragmatic features and thereby guide it, even after we relax combinatorial constraints.

3.3 Approximating the Optimization Problem

The problem in Eq. (9) is $\mathbf{NP}$-hard, due to the non-convex nature of the space of permutation matrices [24]. A natural way to overcome this hardness is to enlarge the allowed solution space to the convex set of *doubly stochastic* matrices $\mathbb{W}^n$, as considered in FAQ [42]:

$$\min_{\mathbf{P} \in \mathbb{W}^n} -\operatorname{tr}(\mathbf{A}\mathbf{P}\mathbf{B}^\top \mathbf{P}^\top) + \mu \cdot \operatorname{tr}(\mathbf{P}^\top \mathbf{D}) \tag{10}$$

Since the problem in Eq. (10) calls to minimize a function subject to linear constraints implied by $\mathbf{P} \in \mathbb{W}^n$, the solution can be efficiently found [3] by algorithms such as *Adam* [23] and *Frank-Wolfe* [16]. The FAQ algorithm [42] follows such an approach to solve the relaxed optimization with the Frank-Wolfe algorithm and project the solution back onto $\mathbb{P}^n$, yet addresses exclusively the first, QAP term in Eq. (10). To further augment quality, as we elaborate later in Section 5, we include the LAP term in Eq. (10) and also add a regularizing term that guides the solution towards a quasi-permutation matrix. By Theorem 2.5, which establishes that a doubly-stochastic matrix $\mathbf{P}$ with $\operatorname{tr}(\mathbf{P}^\top(\mathbf{J}-\mathbf{P}))=0$ is a permutation matrix, we rewrite the problem in Eq. (9) as:

Algorithm 1 FINDQUASIPERMUTATION $(\mathbf{A}, \mathbf{B}, \mathbf{D}, \mu, T)$

Input: Adjacency Matrices $\mathbf{A}$, $\mathbf{B}$, Distance matrix $\mathbf{D}$, control parameter μ, num iters T
Output: Quasi-Permutation matrix $\mathbf{Q}$
Notation:
$f(\mathbf{P}) : -\operatorname{tr}(\mathbf{A}\mathbf{P}\mathbf{B}^\top\mathbf{P}^\top) + \mu \cdot \operatorname{tr}(\mathbf{P}^\top\mathbf{D})$
$g(\mathbf{P}) : \operatorname{tr}(\mathbf{P}^\top(\mathbf{J}-\mathbf{P}))$

$\mathbf{Q} \leftarrow \mathbf{1}\cdot\mathbf{1}^\top/n$
for $\lambda = 0$ **to** $T-1$ **do**
  for $it = 1$ **to** 10 **do**
    $grad \leftarrow \nabla f(\mathbf{Q}) + \lambda \cdot \nabla g(\mathbf{Q})$
    $q_{it} \leftarrow \arg\min_{q\in\mathbb{W}^n} \langle grad, q\rangle$ \\ Sinkhorn-Knopp
    $\alpha \leftarrow \frac{2}{2+it}$
    $\mathbf{Q} \leftarrow \mathbf{Q} + \alpha\cdot(q_{it} - \mathbf{Q})$
  end for
end for
return $\mathbf{Q}$

$$\begin{aligned} &\min_{\mathbf{P}\in\mathbb{W}^n} -\operatorname{tr}(\mathbf{A}\mathbf{P}\mathbf{B}^\top\mathbf{P}^\top) + \mu\cdot\operatorname{tr}(\mathbf{P}^\top\mathbf{D}) \\ &\text{Constraints: } \operatorname{tr}(\mathbf{P}^\top(\mathbf{J}-\mathbf{P})) = 0 \end{aligned} \tag{11}$$

and turn the constraint to a regularizer with parameter λ:

$$\min_{\mathbf{P}\in\mathbb{W}^n} -\operatorname{tr}(\mathbf{A}\mathbf{P}\mathbf{B}^\top\mathbf{P}^\top) + \mu\cdot\operatorname{tr}(\mathbf{P}^\top\mathbf{D}) + \lambda\cdot(\operatorname{tr}(\mathbf{P}^\top(\mathbf{J}-\mathbf{P}))) \tag{12}$$

Equivalently, by reformulating the constraints:

$$\begin{aligned} &\min_{\mathbf{P}} -\operatorname{tr}(\mathbf{A}\mathbf{P}\mathbf{B}^\top\mathbf{P}^\top) + \mu\cdot\operatorname{tr}(\mathbf{P}^\top\mathbf{D}) + \lambda\cdot(\operatorname{tr}(\mathbf{P}^\top(\mathbf{J}-\mathbf{P}))) \\ &\text{Constraints: } \mathbf{P}\mathbf{1}=\mathbf{1}, \mathbf{P}^\top\mathbf{1}=\mathbf{1}, 0\le\mathbf{P}_{ij}\le 1 \end{aligned} \tag{13}$$

We solve the problem in eq. (13) for $\lambda = 0$ by the Frank-Wolfe (FW) algorithm [16] with updates guided by an objective computed via the Sinkhorn-Knopp algorithm [7], due to the computational efficiency they confer. We use the solution to this optimization problem as a warm start, and refine it by gradually increasing λ over T iterations, each initiating with the solution obtained in the preceding one and solving the problem in Eq. (13) by FW. Alg. 1 outlines the process.

Rounding Algorithm: Alg. 1 yields a *quasi-permutation* matrix $\mathbf{Q}$. Next, to obtain an one-to-one mapping between nodes of $\mathcal{G}_1$ and $\mathcal{G}2$, we need to adjust $\mathbf{Q}$ to a permutation matrix by rounding. We pose this problem as an assignment problem, maximizing the sum of $\mathbf{Q}_{ij}$ entries selected for rounding up to 1, while rounding the rest down to 0, and solve it optimally by the Hungarian algorithm [26]. Alg. 2 in the appendix presents the complete FUGAL pseudocode.

4 Customized Optimization Strategy for Node Alignment

In this section, we elucidate the intricacies of Algorithm 1, which derives a quasi-permutation matrix, focusing on two pivotal steps: *(i)* initialization of the quasi-permutation matrix; *(ii)* finding the local solution for a given λ.

Initialization: Any doubly stochastic matrix is a viable option for initialization. However, we opt for an *uninformative flat* matrix, $\mathbf{1}\cdot\mathbf{1}^\top/n$. Our empirical observations indicated that this initialization consistently performs well across diverse datasets, contrary to *informative* initializations like the *identity matrix*, which exhibit inconsistency in performance, as we further elaborate in Section A.8.

Local Solution for a given λ: Given a specific λ, our objective is to solve the optimization problem of Eq. (13) under linear constraints. To achieve this, we employ the Frank-Wolfe algorithm (FW), a successive first-order optimization technique devised for solving convex quadratic programs [16]. While FW is a widely utilized solver as a subroutine for QAP algorithms, we tailor its application to FUGAL. Specifically, each iteration commences from the local solution obtained in the previous iteration and involves the following steps:
Computing the Gradient: The gradient of the objective function $f(\mathbf{P}) = -\operatorname{tr}(\mathbf{A}\mathbf{P}\mathbf{B}^\top\mathbf{P}^\top) + \mu\cdot\operatorname{tr}(\mathbf{P}^\top\mathbf{D})$ with respect to $\mathbf{P}$, evaluated at $\mathbf{Q}$, is $\nabla f(\mathbf{Q}) = -\mathbf{A}\mathbf{Q}\mathbf{B}^\top - \mathbf{A}^\top\mathbf{Q}\mathbf{B} + \mu\cdot\mathbf{D}$. Additionally,

the gradient of the constraint function $g(\mathbf{P}) = \mathrm{tr}(\mathbf{P}^\top(\mathbf{J} - \mathbf{P}))$ with respect to $\mathbf{P}$, evaluated at $\mathbf{Q}$, is $\nabla g(\mathbf{Q}) = \mathbf{J} - 2\mathbf{Q}$.

Updating Q: A critical step involves determining the doubly-stochastic matrix q_{it} that minimizes the inner product $\langle grad, q \rangle$, where $grad$ is the current gradient. Prior work [42] applies the Hungarian algorithm to obtain a permutation matrix to that end, which, however, may not yield the optimal answer and incurs $\mathcal{O}(n^3)$ cost. Contrarily, we obtain a proper doubly stochastic matrix to that end.

Definition 4.1 (Optimal Transport Distance Between r and c)**.** Given a $n \times n$ cost matrix $\mathbf{M}$, the cost of mapping an n-dimensional probability vector r to c, using a transportation matrix (or joint probability) $\mathbf{P}$ is quantified as $\langle \mathbf{P}, \mathbf{M} \rangle$. The following problem:

$$\min_{\mathbf{P} \in U(r,c)} \langle \mathbf{P}, \mathbf{M} \rangle. \tag{14}$$

is an *optimal transport* problem between r and c given cost $\mathbf{M}$, where

$$U(r, c) = \{\mathbf{P} \in \mathbb{R}_+^{n \times n} \mid \mathbf{P}\mathbf{1} = r, \mathbf{P}^\top \mathbf{1} = c\} \tag{15}$$

To render this optimal transport objective *strictly* convex and thus efficiently solvable by the *matrix scaling* Sinkhorn-Knopp fixed-point iteration algorithm [38] via matrix-vector products, we regularize it with an entropic penalty $h(\mathbf{P})$ that yields the *Sinkhorn distance* objective [7]:

$$\min_{\mathbf{P} \in U(r,c)} \langle \mathbf{P}, \mathbf{M} \rangle - \frac{1}{\kappa} h(\mathbf{P}) \tag{16}$$

where $h(\mathbf{P}) = -\sum_{i,j=1}^{n} \mathbf{P}_{ij} \log \mathbf{P}_{ij}$, $\kappa \in (0, \infty]$, which becomes equivalent to the transport distance for suitably large κ [7]. The method exhibits excellent performance in practice with $\mathcal{O}(n^2)$ empirical time complexity.

Setting r and c to $\mathbf{1}$ (the all-ones vector) in Eq. (15), $U(r, c)$ becomes the space of doubly stochastic matrices, hence Eq. (14) with $\mathbf{M} = grad$ captures our update step objective. Thus, we find the doubly stochastic matrix q_{it} that minimizes $\langle grad, q \rangle$ by the Sinkhorn-Knopp algorithm and update $\mathbf{Q}$ as $\mathbf{Q} \leftarrow \mathbf{Q} + \alpha \cdot (q_{it} - \mathbf{Q})$, with the step size α following the conventional choice $\alpha = 2/(2 + it)$.

Complexity Analysis: The complexity of FUGAL is $\mathcal{O}(n^3)$, which is in line with the majority of the baselines. A detailed derivation and comparison of FUGAL's complexity with baselines is provided in App. A.1.The $\mathcal{O}(n^3)$ complexity stems from the need to perform matrix multiplications, a core operation in FUGAL as well as the baselines. A thorough evaluation of empirical running times (§ 5.5) also demonstrates the practical scalability of FUGAL.

5 Experiments

In this section, we present a comprehensive evaluation of FUGAL vs. state-of-the-art graph alignment baselines on real and synthetic data sets with varying noise levels.

5.1 Datasets

Real Graphs. Table 1 summarizes the real-world datasets used to benchmark FUGAL.The last three data sets in the table are evolving graphs mandating challenging ground-truth alignments.

Synthetic Graphs. We employ Newmann-Watts (NW) [21] graphs, characterized by small-world properties and a high clustering coefficient. We generate NW graphs with 1000 nodes, number of neighbors per node k = 7, and a rewiring probability of p = 0.1. For each graph, we generate 5 noisy variants, perform alignments on each, and report average results. Given the obtained alignment set $\mathcal{P}$ and the ground truth set of alignments $\mathcal{P}_{real}$, we calculate *accuracy* as $\frac{|\mathcal{P} \cap \mathcal{P}_{real}|}{|\mathcal{P}|} \cdot 100$.

Table 1: Real-graph nodes n, edges m, and network type.

Dataset	n	m	Type
Arenas [27]	1 133	5 451	communication
inf-euroroad [1]	1 174	1 417	infrastructure
bio-celegans [9]	453	2 025	biological
ca-netscience [33]	379	914	collaboration
ACM [48]	9 872	39 561	citation
DBLP [48]	9 916	44 808	citation
MultiMagna [41]	1 004	8 323	biological
HighSchool [15]	327	5 818	proximity
Voles [8]	712	2 391	proximity

Noise Types. As in prior work [4, 20, 43], we introduce perturbations to the adjacency matrix by either removing or adding edges. We employ two noise types: *one-way* noise removes edges from the target graph, while *bimodal* noise removes and restores the same number of edges.

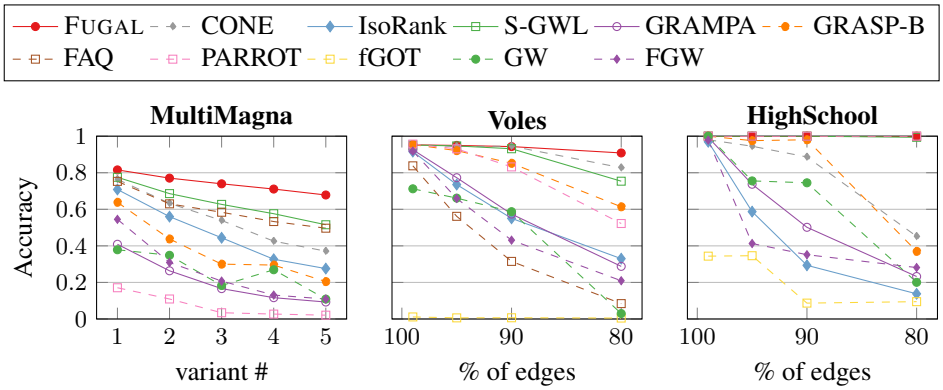

Figure 1: Accuracy, real graphs with real noise.

5.2 Experimental Setup

We ran all experiments on a 40-core Intel Xeon E5-2687W CPU machine @3.10GHz with Python implementations of FUGAL[1] and competitors;[2] the latter include CONE [4], IsoRank [37], S-GWL [43], GRAMPA [13], GRASP-B [19], FAQ [42], PARROT [46], fGOT [32], GOT [31], GW [35] and FGW [40]. Due to scalability limitations, we excluded fGOT from consideration for graphs with more than 1000 nodes, on which it failed to terminate within 5 hours. Moreover, due to the inability of GOT, PATH, and DSPP to scale for the smallest dataset in our analysis, we assess them separately on smaller graphs in Appendix A.4. We exclude GWL from evaluation in favor of its scalable and superior variant, S-GWL [39]. We omit from the comparison algorithms such as GRAAL [25], GLAG [14] and REGAL [17] due to their inferior performance [39, 29]. As we focus on non-attributed graphs, we exclude FINAL [47], which is equivalent to IsoRank on graphs without attributes. For the prerequisite similarity score in IsoRank, we devise a customized weight scheme as $sim(u, v) = 1 - \frac{|d_u - d_v|}{\max\{d_u, d_v\}}$, where $d_u = |N(u)|$ denotes the degree of node u. With all baselines, we use author-recommended parameters and derive node matchings from similarity scores using the Hungarian algorithm. In Appendix A.5, we benchmark FUGAL against S-GWL and CONE in terms of Matched Neighborhood consistency (MNC) [4] and the Frobenius norm between aligned graph adjacency matrices.

5.3 Accuracy on varying noise

Graphs with real Noise: We evaluate all algorithms on accuracy with three real-world networks: MultiMagna, Voles, and High School. MultiMagna represents a yeast protein-protein interaction (PPI) network and noisy variants incorporating an additional $q\%$ of low-confidence interactions, with $q \in \{5, 10, 15, 20, 25\}$. High School and Voles are temporal proximity networks; we align the last graph version to versions containing 80%, 85%, 90%, and 99% of edges. Figure 1 presents our results. FUGAL consistently achieves accuracy surpassing its counterparts across all datasets, with S-GWL being the closest baseline on average. On MultiMagna, FUGAL attains a 4% improvement over the next best algorithm, S-GWL, on the first graph variant, and this gap steadily increases to 16% on the last variant. On Voles, CONE and S-GWL follow FUGAL's accuracy with up to 90% of edges, yet with 80% of edges, they achieve 83% and 75% accuracy, respectively, vs. 90% of FUGAL. On the High School network, FUGAL, FAQ and PARROT align graphs perfectly, while S-GWL attains near-perfect alignment. IsoRank, GRAMPA, GRASP-B fGOT, GW and FGW fall short of FUGAL's performance across all three datasets. Despite performing comparably to FUGAL on the HighSchool dataset, FAQ and PARROT exhibit notably poorer performance on other datasets. The consistently superior performance of FUGAL underscores its robustness.

Large Real Graphs with Partially Aligned nodes: ACM and DBLP are two co-authorship networks of the ACM Digital Library and DBLP bibliography. In these networks, nodes represent authors, and an edge exists between two authors if they have collaborated on at least one publication. Across both networks, there are 6,325 authors who appear in both. Although both networks are attributed, we did not incorporate this information in our experiments. Note that S-GWL and GRASP-B are not scalable for networks of this magnitude, hence omitted from the analysis. Furthermore, the experiment was conducted in an unsupervised manner, meaning that the methods were not provided with any prior information regarding node alignment. Our results, detailed in Table 2, showcase

[1]Code and data at `https://github.com/idea-iitd/Fugal`.

[2]Source code from `https://github.com/constantinosskitsas/Framework_GraphAlignment`.

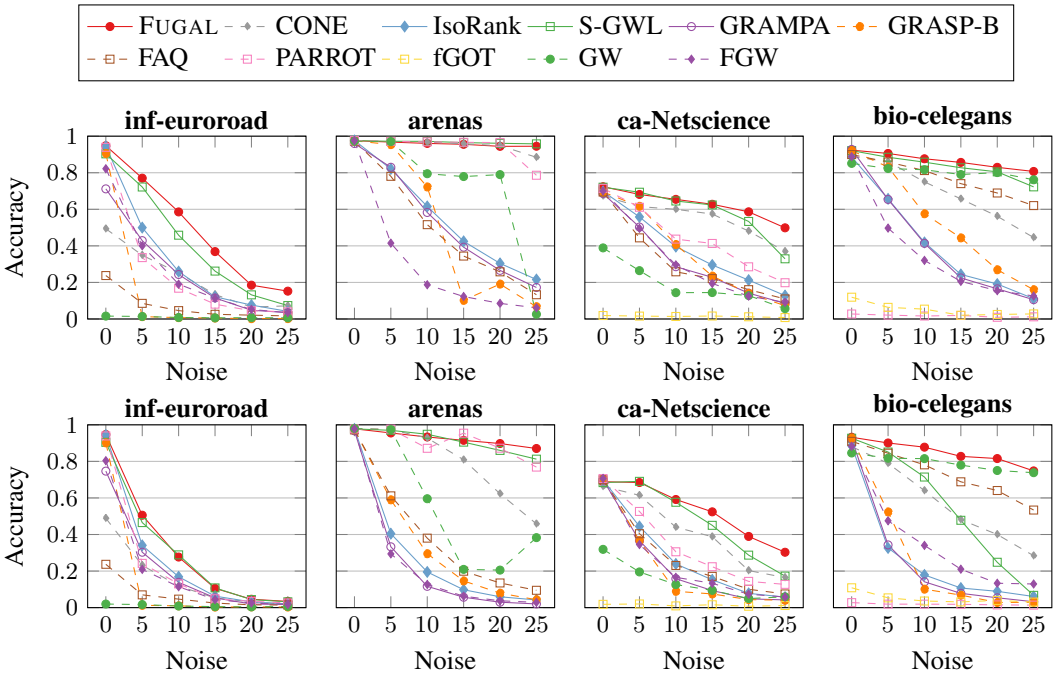

Figure 2: Accuracy, one-way (top) & bimodal (bottom) noise.

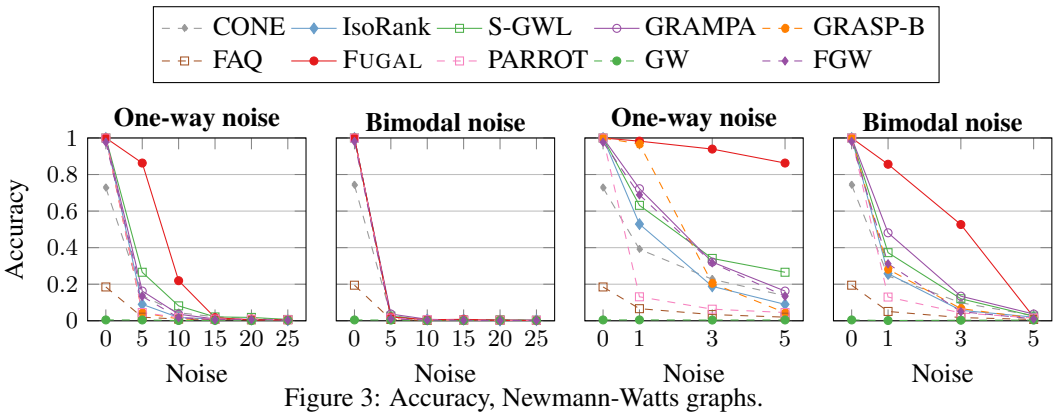

Figure 3: Accuracy, Newmann-Watts graphs.

the fraction of correctly aligned nodes out of the 6,325 aligned nodes. FUGAL demonstrated a significant improvement of 30% compared to the closest baseline, CONE. This underscores the superior scalability of FUGAL without compromising accuracy.

Table 2: Accuracy in alignment across ACM-DBLP.

	CONE	IsoRank	GRAMPA	FAQ	PARROT	GW	FGW	FUGAL
Accuracy	0.183	0.042	0.011	0.025	0.000	0.028	0.012	**0.487**

Real Graphs with Injected Noise: Figure 2 illustrates the results on real datasets subject to synthetic one-way and bimodal noise. Consistent with the trends observed in real noise, FUGAL exhibits superior performance across all evaluated networks and noise types. This consistent superior performance of FUGAL establishes it as a robust graph alignment solution. Appendix A.3 zooms in on the performance of FUGAL vs. baselines with noise levels in the range 0% to 5% to better highlight the performance superiority of FUGAL.

Synthetic Graphs: Figure 3 portrays accuracy results on Newmann-Watts graphs of 1000 nodes with node degree $k = 7$ and rewiring probability $p = 0.1$ subject to synthetic noise. Under one-way noise, all methods except CONE and FAQ achieve perfect alignment at 0% noise. With noise of 5% and 10%, FUGAL attains a 60% and 14% gain, respectively, over the 2nd-best method, S-GWL. Beyond these noise levels, all methods experience failures. Bimodal noise at 0% results in perfect alignment for most methods. However, alignment failures occur as noise grows. Figure 3 further zooms in noise levels in the range of 0% to 5%. FUGAL significantly outperforms all baselines under one-way noise, achieving a margin of 60% at 3% and 5% levels. Moreover, FUGAL performs

superiorly in the bimodal noise within the 5% noise threshold, gaining nearly 40% at 1% and 3% noise levels. These results underscore the efficacy of FUGAL in handling diverse graph structures.

5.4 Varying Density

Here, we examine performance under varying graph density. In Newmann-Watts graphs, the rewiring probability parameter p affects the edge density of sampled graphs for a fixed number of nodes n, while the parameter k, representing the number of nearest neighbors per node, affects the minimum and expected degree. Figure 4 shows our results when varying p and k in NW graphs comprising 1000 nodes. Methods other than FUGAL consistently fail to handle sparse graphs (low p). However, FUGAL attains accuracy 92% at $p = 0.25$, outperforming S-GWL, which achieves only 54%. Sparse graphs pose a challenge for alignment, as they provide less discriminating evidence in terms of density differentials. When varying k, FUGAL consistently achieves near-perfect alignment, surpassing all baselines. These findings corroborate the resilience of FUGAL across graph densities and its adaptability to varying degrees of connectivity.

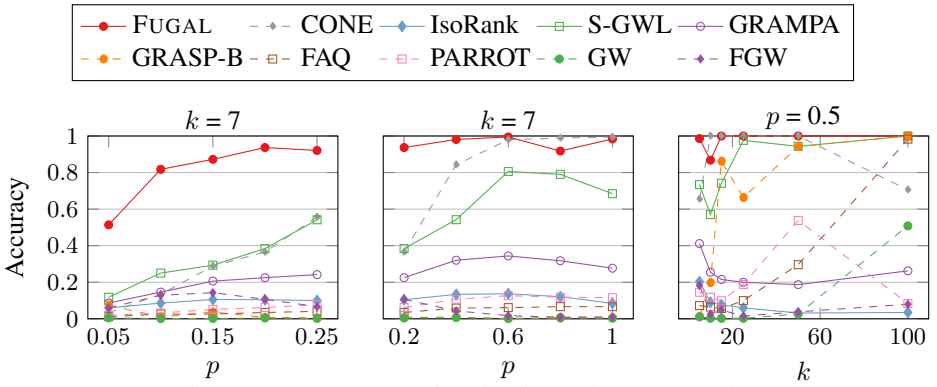

Figure 4: Accuracy varying density and one-way noise.

5.5 Efficiency

Here we compare the computational efficiency of FUGAL to that of S-GWL, which ranks as the second-best performer across most benchmark datasets.

Figure 5 plots running times *in logarithmic axes*. FUGAL achieves lower running times on MultiMagna, Voles, euroroad, arenas, and Newmann-Watts networks with an up to 3x speed up, highlighting its capacity to handle *large* networks. Conversely, S-GWL marginally outperforms FUGAL on *smaller* networks. S-GWL did not scale for ACM-DBLP, failing to terminate even after 5 hours. This discrepancy indicates S-GWL's incapacity to scale to large networks, which restricts its broader applicability. We emphasize that FUGAL achieves a substantial accuracy advantage without compromising efficiency, affirming its prowess as an efficient and effective solution. Appendix A.2 presents running times for all baselines.

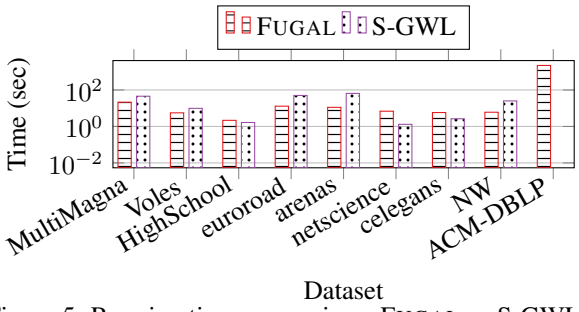

Figure 5: Running time comparison, FUGAL vs S-GWL.

5.6 Scalability

Given the results of Section 5.5, we delve into the scalability of FUGAL and S-GWL with Newmann-Watts graphs of increasing nodes. Figure 6 plots our findings. At 512 nodes, FUGAL and S-GWL have comparable running times. Still, as nodes grow, FUGAL outpaces S-GWL.

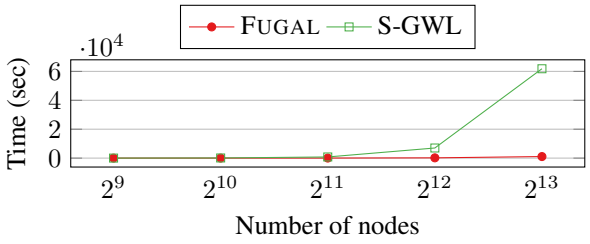

Figure 6: Scalability on NW graphs, $k = 7$, $p = 0.5$.

5.7 Parameters and Ablation

Table 3 in Appendix lists the parameters we employ in FUGAL with each dataset. We set the number of iterations T to 15 for all datasets. The parameter μ controls the sway of node features in the optimization. Sparser graphs, characterized by lower connectivity and less information in adjacency matrices, benefit from higher reliance on node features, hence we recommend a higher μ. Sparser datasets such as inf-euroroad, ca-netscience, and NW ($k = 7$, $p = 0.1$) benefit from higher values of μ (1–2), denser graphs from smaller values (0.1–0.5).

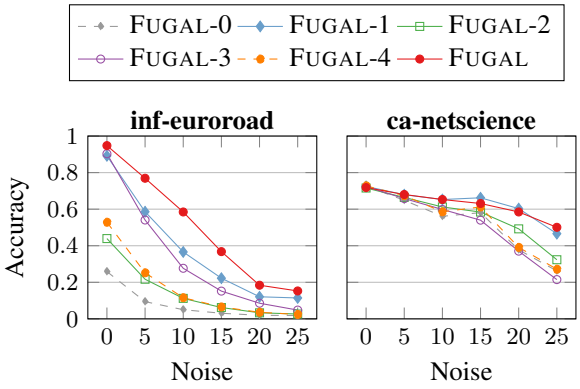

Figure 7: Accuracy of FUGAL variants, one-way noise.

We also conduct an extensive ablation study to assess the impact of the structural features outlined in Section 3.1. We craft five variants of FUGAL, where FUGAL-i utilizes only the i^{th} structural feature while excluding others. FUGAL-0 abstains from all structural features. Figure 7 juxtaposes the accuracy of these variants to that of FUGAL on two networks. Each variant employing structural features attains higher accuracy than FUGAL-0, corroborating the usefulness of these features. Further, FUGAL, leveraging all features, outperforms other variants. Notably, FUGAL-1 performs second-best, underscoring the significance of degree in identifying node alignments. We also investigate a variant setting $\lambda = 0$, denoted as FUGAL-DS (for *doubly stochastic*), instead of iteratively increasing it. As Figure 8 shows, FUGAL-DS attains worse accuracy.

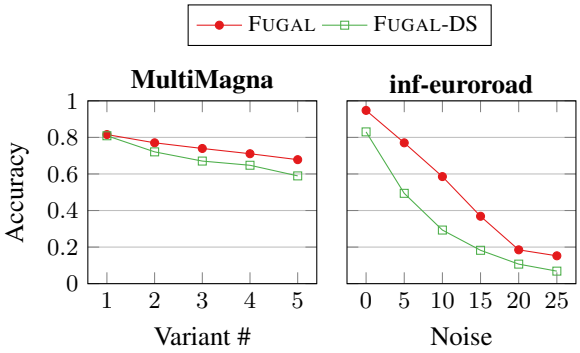

Figure 8: The effect of setting $\lambda = 0$ (FUGAL-DS) against the default option of iteratively increasing.

6 Conclusions

We introduced FUGAL, an *unrestricted* algebraic approach to graph alignment that works directly on graph adjacency matrices and identifies node correspondences by relaxing permutation matrix constraints and steering the solution to the desired form, followed by rounding. Through extensive experimentation, we established that FUGAL surpasses state-of-the-art graph alignment methods in accuracy across network types, noise conditions, and graph densities, even while maintaining a scalability advantage.

Broader Impact and Ethical consequences: FUGAL opens the way to improved solutions in graph alignment, as reflected in its performance across diverse networks, noise types, and graph density. This outcome can spark further research in optimization techniques and advances in bioinformatics, social network analysis, and infrastructure mapping. Still, advances in graph alignment also enhance the abilities of attackers attempting to de-anonymize sensitive social network and biological data. Therefore, preventing attacks on privacy is crucial, calling for the enforcement of advanced anonymization methods [34] before publishing such data.

7 Acknowledgements

AB was supported by Graviton Research Capital LLP. HRV was supported by the CSE Research Acceleration Fund of IIT Delhi. KS was supported by the Independent Research Fund Denmark.

References

[1] David A. Bader, Henning Meyerhenke, Peter Sanders, and Dorothea Wagner, editors. *Graph Partitioning and Graph Clustering. Proceedings of the 10th DIMACS Implementation Challenge Workshop.*, volume 588 of *Contemporary Mathematics*. American Mathematical Society, 2013.

[2] Michele Berlingerio, Danai Koutra, Tina Eliassi-Rad, and Christos Faloutsos. Network similarity via multiple social theories. In *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 1439–1440, 2013.

[3] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[4] Xiyuan Chen, Mark Heimann, Fatemeh Vahedian, and Danai Koutra. CONE-Align: Consistent network alignment with proximity-preserving node embedding. In *29th ACM International Conference on Information and Knowledge Management, CIKM*, pages 1985–1988, 2020.

[5] Donatello Conte, Pasquale Foggia, Carlo Sansone, and Mario Vento. Graph matching applications in pattern recognition and image processing. In *International Conference on Image Processing, ICIP*, pages 21–24, 2003.

[6] L.P. Cordella, P. Foggia, C. Sansone, and M. Vento. A (sub)graph isomorphism algorithm for matching large graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10):1367–1372, 2004.

[7] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *NeurIPS*, pages 2292–2300, 2013.

[8] Stephen Davis, Babak Abbasi, Shrupa Shah, Sandra Telfer, and Mike Begon. Spatial analyses of wildlife contact networks. *Journal of The Royal Society, Interface*, 12(102):20141004, 2015.

[9] Jordi Duch and Alex Arenas. Community detection in complex networks using extremal optimization. *Phys. Rev. E*, 72:027104, 2005.

[10] Nadav Dym, Haggai Maron, and Yaron Lipman. DS++: a flexible, scalable and provably tight relaxation for matching problems. *ACM Trans. Graph.*, 36(6):184:1–184:14, 2017.

[11] Frank Emmert-Streib, Matthias Dehmer, and Yongtang Shi. Fifty years of graph matching, network alignment and network comparison. *Inf. Sci.*, 346-347:180–197, 2016.

[12] Paul Erdős and Alfréd Rényi. On random graphs I. *Publicationes Mathematicae Debrecen*, 6:290, 1959.

[13] Zhou Fan, Cheng Mao, Yihong Wu, and Jiaming Xu. Spectral graph matching and regularized quadratic relaxations: Algorithm and theory. In *ICML*, volume 119, pages 2985–2995, 2020.

[14] Marcelo Fiori, Pablo Sprechmann, Joshua T. Vogelstein, Pablo Musé, and Guillermo Sapiro. Robust multimodal graph matching: Sparse coding meets graph matching. In *NeurIPS*, pages 127–135, 2013.

[15] Julie Fournet and Alain Barrat. Contact Patterns among High School Students. *PLOS ONE*, 9(9):1–17, 2014.

[16] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

[17] Mark Heimann, Haoming Shen, Tara Safavi, and Danai Koutra. REGAL: representation learning-based graph alignment. In *27th ACM International Conference on Information and Knowledge Management, CIKM*, pages 117–126, 2018.

[18] Keith Henderson, Brian Gallagher, Lei Li, Leman Akoglu, Tina Eliassi-Rad, Hanghang Tong, and Christos Faloutsos. It's who you know: graph mining using recursive structural features. In *17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD*, pages 663–671, 2011.

[19] Judith Hermanns, Konstantinos Skitsas, Anton Tsitsulin, Marina Munkhoeva, Alexander Kyster, Simon Nielsen, Alexander M. Bronstein, Davide Mottin, and Panagiotis Karras. Grasp: Scalable graph alignment by spectral corresponding functions. *ACM Trans. Knowl. Discov. Data*, 17(4), feb 2023.

[20] Judith Hermanns, Anton Tsitsulin, Marina Munkhoeva, Alexander M. Bronstein, Davide Mottin, and Panagiotis Karras. GRASP: graph alignment through spectral signatures. In *Web and Big Data - 5th International Joint Conference, APWeb-WAIM 2021*, pages 44–52, 2021.

[21] Petter Holme and Beom Jun Kim. Growing scale-free networks with tunable clustering. *Phys. Rev. E*, 65:026107, Jan 2002.

[22] Ehsan Kazemi, S. Hamed Hassani, and Matthias Grossglauser. Growing a graph matching from a handful of seeds. *Proc. VLDB Endow.*, 8(10):1010–1021, jun 2015.

[23] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[24] Tjalling C. Koopmans and Martin Beckmann. Assignment problems and the location of economic activities. *Econometrica*, 25(1):53–76, 1957.

[25] Oleksii Kuchaiev, Tijana Milenkovic, Vesna Memisevic, Wayne Hayes, and Natasa Przulj. Topological network alignment uncovers biological function and phylogeny. *Nature Precedings*, 4, 12 2009.

[26] Harold W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955.

[27] Jérôme Kunegis. KONECT: the Koblenz network collection. In *22nd International World Wide Web Conference, WWW*, pages 1343–1350, 2013.

[28] Jiajin Li, Jianheng Tang, Lemin Kong, Huikang Liu, Jia Li, Anthony Man-Cho So, and Jose Blanchet. A convergent single-loop algorithm for relaxation of Gromov-Wasserstein in graph data. In *ICLR*, 2023.

[29] Vince Lyzinski, Donniell E. Fishkind, Marcelo Fiori, Joshua T. Vogelstein, Carey E. Priebe, and Guillermo Sapiro. Graph matching: Relax at your own risk. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1):60–73, 2016.

[30] Konstantin Makarychev, Rajsekar Manokaran, and Maxim Sviridenko. Maximum quadratic assignment problem: Reduction from maximum label cover and lp-based approximation algorithm. *ACM Trans. Algorithms*, 10(4):18:1–18:18, 2014.

[31] Hermina Petric Maretic, Mireille El Gheche, Giovanni Chierchia, and Pascal Frossard. GOT: an optimal transport framework for graph comparison. In *NeurIPS*, pages 13876–13887, 2019.

[32] Hermina Petric Maretic, Mireille El Gheche, Giovanni Chierchia, and Pascal Frossard. fGOT: Graph distances based on filters and optimal transport. In *AAAI*, pages 7710–7718, 2022.

[33] Mark E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E*, 74:036104, 2006.

[34] Sadegh Nobari, Panagiotis Karras, HweeHwa Pang, and Stéphane Bressan. L-opacity: Linkage-aware graph anonymization. In *17th International Conference on Extending Database Technology, EDBT*, pages 583–594, 2014.

[35] Gabriel Peyré, Marco Cuturi, and Justin Solomon. Gromov-Wasserstein averaging of kernel and distance matrices. In *ICML*, pages 2664–2672, 2016.

[36] Roded Sharan and Trey Ideker. Modeling cellular machinery through biological network comparison. *Nature Biotechnology*, 24:427–33, 05 2006.

[37] Rohit Singh, Jinbo Xu, and Bonnie Berger. Global alignment of multiple protein interaction networks with application to functional orthology detection. *Proceedings of the National Academy of Sciences*, 105(35):12763–12768, 2008.

[38] Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21:343–348, 1967.

[39] Konstantinos Skitsas, Karol Orlowski, Judith Hermanns, Davide Mottin, and Panagiotis Karras. Comprehensive evaluation of algorithms for unrestricted graph alignment. In *Proceedings 26th International Conference on Extending Database Technology, EDBT*, pages 260–272, 2023.

[40] Vayer Titouan, Nicolas Courty, Romain Tavenard, Chapel Laetitia, and Rémi Flamary. Optimal transport for structured data with application on graphs. In *ICML*, pages 6275–6284, 2019.

[41] Vipin Vijayan and Tijana Milenković. Multiple network alignment via MultiMAGNA++. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 15(5):1669–1682, 2018.

[42] Joshua T Vogelstein, John M Conroy, Vince Lyzinski, Louis J Podrazik, Steven G Kratzer, Eric T Harley, Donniell E Fishkind, R Jacob Vogelstein, and Carey E Priebe. Fast approximate quadratic programming for graph matching. *PLOS one*, 10(4):e0121002, 2015.

[43] Hongteng Xu, Dixin Luo, and Lawrence Carin. Scalable Gromov-Wasserstein learning for graph partitioning and matching. In *NeurIPS*, pages 3046–3056, 2019.

[44] Hongteng Xu, Dixin Luo, Hongyuan Zha, and Lawrence Carin. Gromov-Wasserstein learning for graph matching and node embedding. In *ICML*, pages 6932–6941, 2019.

[45] Mikhail Zaslavskiy, Francis Bach, and Jean-Philippe Vert. A path following algorithm for the graph matching problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2227–2242, 2009.

[46] Zhichen Zeng, Si Zhang, Yinglong Xia, and Hanghang Tong. PARROT: position-aware regularized optimal transport for network alignment. In *ACM Web Conference, WWW*, pages 372–382, 2023.

[47] Si Zhang and Hanghang Tong. FINAL: fast attributed network alignment. In *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD*, pages 1345–1354, 2016.

[48] Si Zhang and Hanghang Tong. Attributed network alignment: Problem definitions and fast solutions. *IEEE Transactions on Knowledge and Data Engineering*, 31(9):1680–1692, 2019.

[49] Feng Zhou and Fernando De la Torre. Factorized graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(9):1774–1789, 2016.

A Appendix

Algorithm 2 FUGAL $(\mathcal{G}_1, \mathcal{G}_2)$

Input: Graphs $\mathcal{G}_1, \mathcal{G}_2$
Output: Permutation matrix $\mathbf{P}$

\\ STEP 1. Extract NetSimile Features
$\mathbf{F}_1 \leftarrow$ EXTRACTFEATURES$(\mathcal{G}_1)$
$\mathbf{F}_2 \leftarrow$ EXTRACTFEATURES$(\mathcal{G}_2)$
$\mathbf{D} \leftarrow$ EUCLEDIANDISTANCE$(\mathbf{F}_1, \mathbf{F}_2)$
\\ STEP 2. Approximate Optimization
$\mathbf{Q} \leftarrow$ FINDQUASIPERMUTATION$(\mathbf{A}, \mathbf{B}, \mathbf{D}, \mu, T)$
\\ STEP 3. Round to Permutation
$\mathbf{P} \leftarrow$ HUNGARIAN$(\mathbf{Q})$
returnP

Table 3: Parameters used in FUGAL.

Dataset	μ
Arenas	0.5
inf-euroroad	2
bio-celegans	0.1
ca-netscience	1
MultiMagna	0.5
HighSchool	0.5
Voles	0.5
Newmann-Watts	2
ACM-DBLP	0.1

A.1 Complexity Analysis

To assess the computational complexity of FUGAL, we examine the three primary components: (i) structural feature extraction; (ii) obtaining a quasi-permutation matrix; (iii) rounding to a permutation matrix. Let us consider source and target graphs $\mathcal{G}_1, \mathcal{G}_2$ with n nodes. Among NETSIMILE structural features, the clustering coefficient incurs a high computational cost of $\mathcal{O}(nM^2)$, where M is the maximum degree among vertices in the graph. Still, for real-world graphs conforming to a power-law degree distribution, the complexity for neighborhood features extraction is expected to be $\mathcal{O}(nM^\epsilon)$, with $0 < \epsilon < 1$ [18], while finding pairwise Euclidean distances between node features takes $\mathcal{O}(n^2)$. The quasi-permutation matrix derivation involves determining the gradient of the optimization problem, which requires $\mathcal{O}(n^3)$ due to matrix multiplications. The Sinkhorn-Knopp algorithm finds the doubly-stochastic matrix q minimizing $\langle grad, q \rangle$, is nearly $\mathcal{O}(n^2)$ [7], while the update to Q takes $\mathcal{O}(n^2)$ time. Thus, the time complexity for finding a quasi-permutation matrix is $\mathcal{O}(T \cdot n^3)$, where T is the number of iterations. We round the quasi-permutation matrix to a permutation matrix by the Hungarian algorithm, incurring a time complexity of $\mathcal{O}(n^3)$. Therefore, the time complexity of FUGAL is $\mathcal{O}(T \cdot n^3)$. Empirically, T typically ranges from 10 to 20, resulting in a cost of $\mathcal{O}(n^3)$ since $T \ll n$. We provide computational costs of baselines in Appendix A.6. Table 4 compares FUGAL's computational cost to that of baselines.

A.2 Running times for all baselines

Figure 9 presents the running times of various baselines on the benchmark datasets. It is noteworthy that while some of these baselines exhibit better running times than FUGAL, the substantial disparity in accuracy, as previously demonstrated in §. 5, renders a comparison skewed in favor of FUGAL.

A.3 Accuracy on Real Graphs with Synthetic Noise - Low Noise Range

In § 5, we have already demonstrated the superior performance of FUGAL in the presence of noise levels ranging from 0% to 25%. In this section, we extend the comparison to assess the performance

Table 4: Computational Complexity comparison with baselines. n, m denotes the number of nodes and edges respectively. T, L denote the number of loop iterations.

	CONE	IsoRank	S-GWL	GRAMPA	GRASP-B	FAQ	PARROT	fGOT	GOT	FUGAL	GW	FGW
Time $\mathcal{O}(.)$	n^2	n^4	$(n+m)\log n$	n^3	n^3	n^3	$Tmn + TLn^2$	n^3	n^3	n^3	n^3	n^3

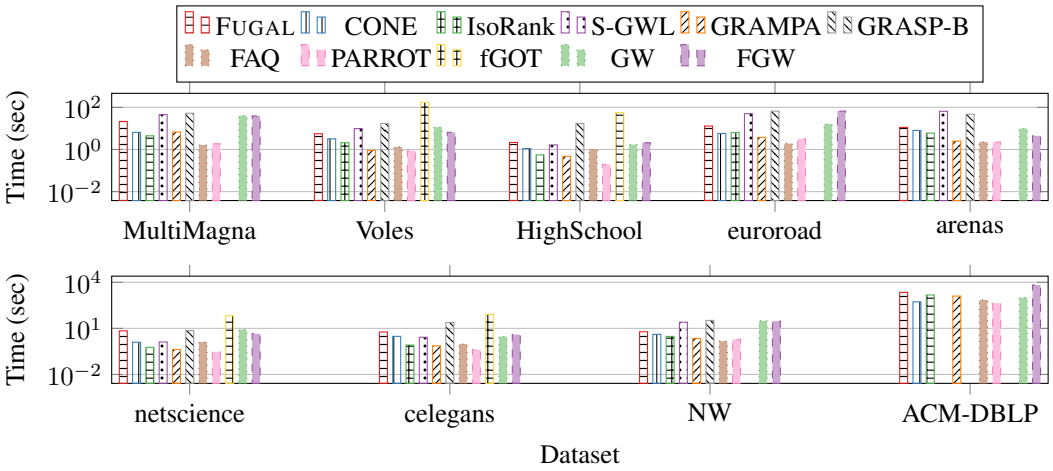

Figure 9: Running times of all baselines.

of FUGAL and other baseline methods on real datasets with synthetic noise, specifically of One-Way and Multi-Modal types, within the range of 0% to 5%. The results are presented in Figure 10. On the **inf-euroroad** dataset, FUGAL consistently outperforms all baselines across varying noise levels in both One-Way and Multi-Modal scenarios. For the **arenas** dataset, FUGAL, CONE, S-GWL, PARROT, and GW maintain accuracy levels exceeding 95% consistently across all noise levels, whereas other baselines exhibit a decline in performance with increasing noise levels, particularly evident in the Multi-Modal scenario. In the case of the **ca-netscience** dataset, both FUGAL and S-GWL achieve similar accuracy in both One-Way and Multi-Modal scenarios. On the **bio-celegans** dataset, S-GWL closely matches the performance of FUGAL in the One-Way scenario; however, with Multi-Modal noise, FUGAL achieves a 6% improvement over the next-best method, S-GWL. The consistent and superior performance of FUGAL across all benchmark datasets underscores its robustness.

A.4 Evaluation on Small Graphs

To compare the performance of non-scalable methods like GOT, fGOT, PATH and DSPP with FUGAL, we employed small Erdős-Rényi random graphs [12]. The limited scalability of these methods for larger graphs is empirically demonstrated in Section 5 (failing to terminate within 5 hours) as well as evidenced by the maximum graph size evaluated by the authors, which was 100. Following the methodology of fGOT [32], we varied the node count n from 20 to 100, with edges generated using a probability of $2\log(n)/n$. We also included S-GWL and PARROT in the analysis. The accuracy of these methods across different graph sizes is depicted in Figure 11. While FUGAL, S-GWL, and PARROT achieved perfect alignment across all graph sizes, other methods exhibited notably inferior performance. Following [32], the Frobenius distance between aligned graph Laplacian matrices across varying graph sizes is also reported in Figure 11. FUGAL, S-GWL, and PARROT maintained an L2 Distance of 0 across all graph sizes, indicating perfect alignment, whereas the performance of other methods deteriorated with increasing graph size. These distance values closely align with those reported by the original authors in [32], validating our experimental setup.

A.5 Additional Metrics

We assess FUGAL against S-GWL and CONE in terms of Matched Neighborhood Consistency (MNC) [4] and the Frobenius distance between aligned graph adjacency matrices. The results in Figure 12 indicate that FUGAL outperforms S-GWL and CONE across noise levels and noisy variants.

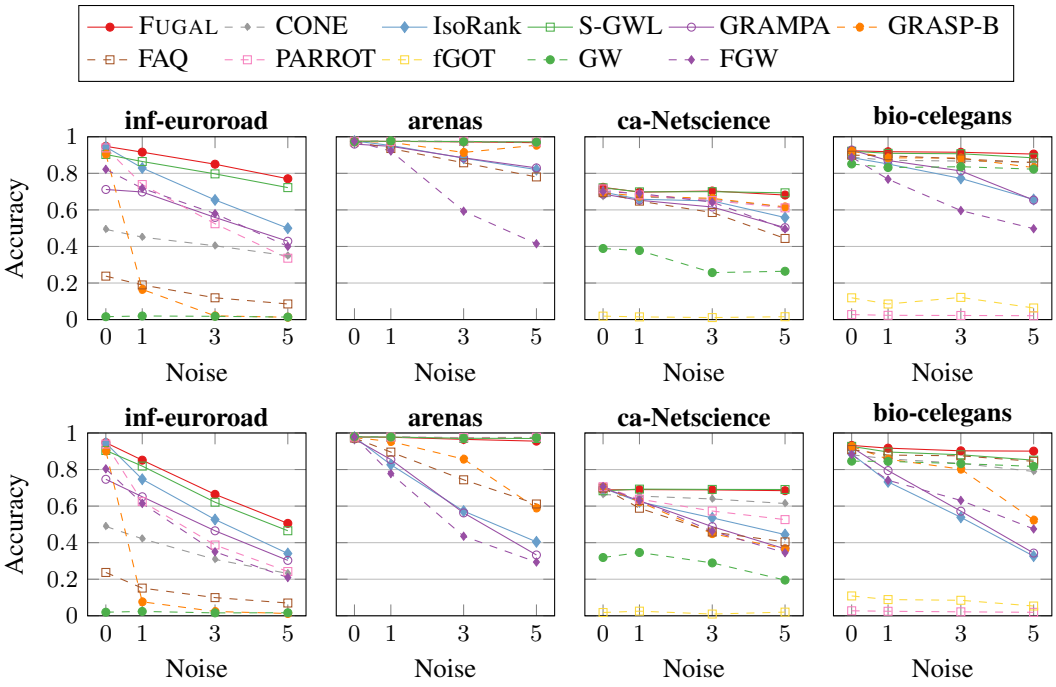

Figure 10: Accuracy comparison for real datasets with noise ranging from 0% to 5%. Top row represents One-Way noise, bottom row represents Multi-Modal noise scenarios.

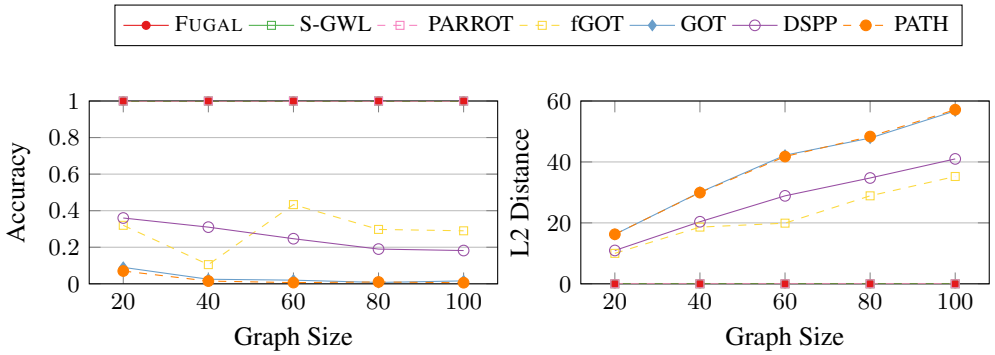

Figure 11: Performance comparison on Erdős-Rényi graphs. The performance is shown in terms of Accuracy (L) and the Frobenius distance between aligned graph Laplacian matrices (R) across different graph sizes (# of nodes).

A.6 Computational Complexity Analysis of Baselines

The computational costs incurred by the baseline methods and FUGAL are presented in Table 4. It is notable that all algorithms utilize the Hungarian algorithm to convert the similarity matrix into a permutation matrix, incurring a computational cost of $\mathcal{O}(n^3)$. However, the costs reported in Table 4 solely pertain to the computation of the similarity matrix, excluding this operation. Among the baselines, CONE, S-GWL, and PARROT exhibit superior time complexity compared to FUGAL. However, as demonstrated in Section 5, both CONE and PARROT fall significantly short of FUGAL in terms of performance. Despite the seemingly promising computational cost of S-GWL, our empirical analysis in Sections 5.5 and 5.6 revealed slower running times compared to expectations. Moreover, S-GWL fails to scale for larger graphs such as ACM-DBLP (failing to terminate within 5 hours), whereas FUGAL achieves superior accuracy within 40 minutes. The limited scalability of S-GWL has been underscored by various studies [46, 19, 39, 28]. Consequently, FUGAL emerges as a preferable option for attaining superior accuracy without incurring detrimental overhead.

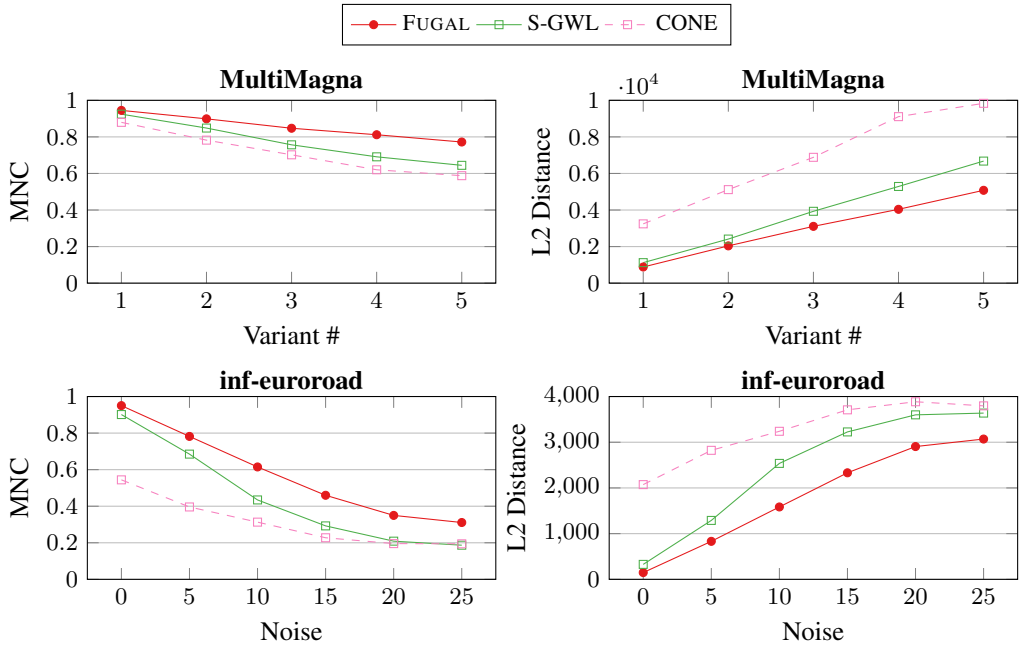

Figure 12: Comparison of FUGAL with S-GWL and CONE on MultiMagna and inf-euroroad datasets in terms of *Matched Neighborhood Consistency (MNC)* (L) and the *Frobenius distance* between aligned graph adjacency matrices (R) across different variants and varying noise respectively.

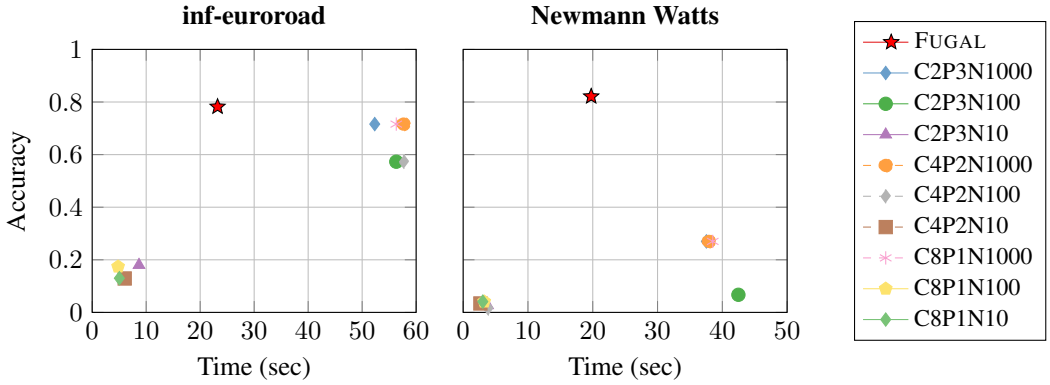

Figure 13: Accuracy vs. Running time, FUGAL vs S-GWL variants, one-way noise 5%.

A.7 Accuracy vs Running Time

S-GWL employs a recursive graph partition mechanism to accelerate graph alignment computations. An important question is how the performance of SGWL is affected by this recursive mechanism with respect to FUGAL. We have identified that the hyper-parameters cluster_num (**C**), partition_level (**P**), and node_prior (**N**) significantly influence the recursive partitioning process. Three variants of S-GWL were proposed by the authors [43] based on different combinations of **C** and **P**: (**C** = 2, **P** = 3), (**C** = 4, **P** = 2), and (**C** = 8, **P** = 1). Additionally, we vary the node_prior parameter within the set {10, 100, 1000}. These combinations result in nine distinct variants of S-GWL, denoted as CxPyNz where x, y, z represent the values of **C**, **P**, and **N** respectively. We conduct a comparative evaluation of FUGAL against these nine variants with respect to both accuracy and running times jointly. Figure 13 depict the outcomes on the inf-euroroad and Newmann Watts datasets. For the inf-euroroad dataset, variants achieving comparable accuracy to FUGAL exhibit significantly higher running times, while those with lower running times have accuracy less than 20%. In the Newmann Watts dataset, none of the S-GWL variants approach the accuracy of FUGAL, as previously established in Section 5. Notably, variants with reasonable accuracy tend to have longer running times, whereas those with better running times demonstrate poorer accuracy. These findings underscore the practical suitability of FUGAL for graph alignment, given its favorable balance between accuracy and running times.

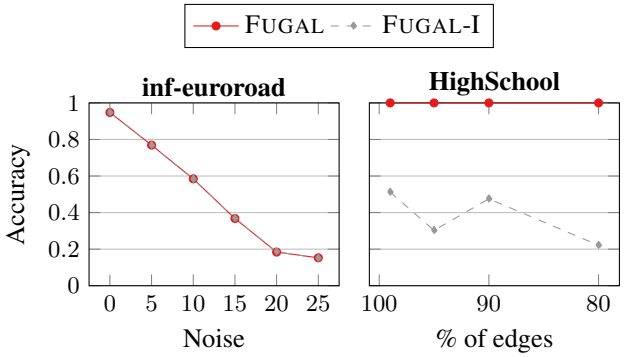

Figure 14: Accuracy, FUGAL vs FUGAL-I, one-way noise.

A.8 Effect of Initialization

Lastly, we empirically examine our decision to employ a *non-informative* flat doubly stochastic matrix, denoted as $\mathbf{1} \cdot \mathbf{1}^T/n$, as the initial quasi-permutation matrix in Algorithm 1, in contrast to alternatives. We try out a variant of FUGAL, FUGAL-I, which utilizes the *Identity* matrix as the initial quasi-permutation matrix instead. Figure 14 presents two instances of our comparative evaluation of FUGAL-I vs FUGAL. While FUGAL-I closely reaches the accuracy of FUGAL on the **inf-euroroad** network, it falls short of FUGAL on the **HighSchool** network, indicating its instability. These findings substantiate our selection of the flat doubly stochastic matrix as a robust choice for initialization.

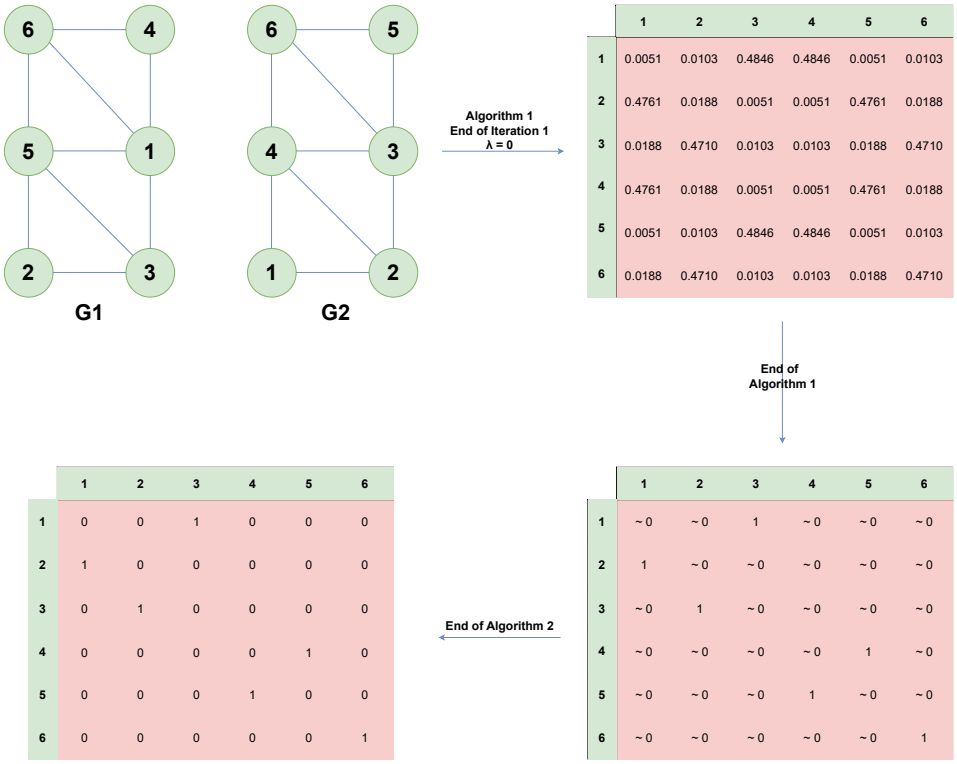

Figure 15: Operational stages of FUGAL. Values less than $1e^{-10}$ are approximated to 0.

A.9 Illustrative example of FUGAL's pipeline

We illustrate the functionality of FUGAL through an example in Figure 15. We create a source graph $\mathcal{G}1$ of 6 nodes and randomly permute it to a target graph $\mathcal{G}2$. We show three stages of FUGAL's operation: (1) the doubly stochastic matrix $\mathbf{Q}$, generated after the first iteration of Algorithm 1 ($\lambda = 0$); (2) the quasi-permutation matrix $\mathbf{Q}$ to which Algorithm 1 (Section 3) steers the doubly stochastic matrix; (3) the permutation matrix $\mathbf{P}$ into which Algorithm 2 refines this quasi-permutation matrix using the Hungarian algorithm. FUGAL aligns $\mathcal{G}1$ and $\mathcal{G}2$ perfectly.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: See Sections 3, 4, 5.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Section 5, 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: See Sections 2, 3, 4,Appendix A.1.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See Section 5, Appendix 5.7.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: See Section 5, Appendix 5.7.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: See Section 5, Appendix 5.7.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The error bars are compromising the visual interpretability of the plots due to the large number of baselines compared against.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Section 5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: [NA] .

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: See Section 6.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: [NA] .

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: See Section 5.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Link to codebase in Section 5.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: [NA] .

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: [NA] .

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.