SparkAlign: A Star-Hopping Heuristic for Efficient Unsupervised Plain Graph Alignment

Anonymous Author(s)

Abstract

Unsupervised plain graph alignment (UPGA) seeks corresponding nodes between two graphs from topology alone, without attributes or seed anchors. The dominant embed-then-match paradigm is expensive on both fronts: it repeatedly retrains node embeddings and then solves a costly global assignment, so it scales poorly to large graphs. We present SparkAlign, a training-free heuristic that, inspired by *star-hopping* in celestial navigation, locates a node's counterpart by measuring its heat-diffusion proximity to a shared set of high-confidence *landmark* matches. Although heuristic in spirit, the descriptor is principled: diffusion makes structurally equivalent nodes comparable *across* graphs, and a short stability analysis justifies a finite diffusion depth. Matching then uses two sparse, highly parallel strategies that avoid dense global assignment, one favoring speed and one preserving one-to-one consistency on a restricted candidate set. On three benchmarks SparkAlign attains accuracy close to the structural identifiability limit while running up to two orders of magnitude faster than the strongest baseline and seeding markedly more accurate pseudo-anchors. Our code is available [1].

CCS Concepts

• **Information systems** → **Data mining**; • **Theory of computation** → *Graph algorithms analysis*; • **Computing methodologies** → Knowledge representation and reasoning.

Keywords

Unsupervised graph alignment, Heat diffusion, Training-free embedding, Sparse assignment

1 Introduction

Graph alignment identifies corresponding nodes across two graphs and underpins cross-platform user linkage [5, 14, 21, 41], biological network analysis [12, 15, 35], and knowledge-graph integration [16, 32, 33]. Most methods lean on side information—node attributes or a seed set of known anchor links [25, 30, 32, 35]—which in practice is often missing, unreliable, or withheld for privacy [27, 29, 34]. When neither attributes nor anchors are available the task becomes *unsupervised plain graph alignment* (UPGA): NP-hard, and reliant on topology alone.

State-of-the-art UPGA follows an *embed-then-match* loop: nodes are embedded, a similarity matrix is formed, a matcher commits the most confident pairs, and these pseudo-anchors seed the next round [10, 26, 34]. Both halves are costly. *Representation generation* repeatedly retrains or updates embedding models as pseudo-anchors accrue [12, 26, 34]. *Node matching* faces a quality–scalability tension: direct (row-wise nearest) matching is parallel but produces many-to-one collisions; greedy matching reduces collisions but is sequential; optimal one-to-one assignment is best but costs $O(n^3)$ [7, 27]. Yet prior methods relieve only one side—training-free representations paired with cubic matching (GRASP), or accelerated matching atop trained embeddings (MMNC)—so none is at once training-free, parallelizable, accelerated, and accurate.

We propose SparkAlign, a training-free *heuristic* that attacks both bottlenecks, and whose name reflects its guiding intuition. In celestial navigation an observer finds a faint target star by *hopping* from a few bright, already-identified stars along recognizable patterns. SparkAlign treats high-confidence matched pairs as such bright "landmarks" and locates every remaining node by its heat-diffusion proximity to them—an *anchor-relative* descriptor computed in closed form, with no model to train. Because diffusion is a deterministic function of topology, structurally equivalent nodes acquire near-identical descriptors even in different graphs (§3.3), so shared landmarks make the two graphs' representations comparable. A canonical feature-sorting step supplies permutation-invariant signatures for the first round, when no landmarks exist yet. For matching we contribute two sparse, fully parallel strategies—a fast collision-reduced matcher and a top-P *sparsified* optimal matcher—that retain one-to-one quality without dense global assignment.

Heuristic though it is, SparkAlign rests on a principled footing: we argue its diffusion descriptor is structurally discriminative, and a short stability result answers the natural objection that the diffusion operator is non-contractive, pinning down why a *finite* diffusion depth is optimal (§3.3). Our contributions are:

- a training-free, anchor-relative *heat-diffusion* descriptor for UPGA, with a structural-equivalence rationale and a stability result that locates the optimal diffusion depth (§3.3);
- two sparse matchers—a parallel fast matcher and a top-P sparsified Modified Jonker–Volgenant solver—that cut matching from $O(n^3)$ toward near-quadratic cost while keeping one-to-one consistency (§3.4);
- experiments on three benchmarks showing accuracy on par with the strongest baseline at up to 315× less runtime, with substantially better pseudo-anchor seeding (§4).

2 Related Work

Graph matching and spectral methods. Classical aligners relax the NP-hard quadratic assignment $\min_P \|\boldsymbol{P}\boldsymbol{A}_s\boldsymbol{P}^\top - \boldsymbol{A}_t\|_F$ over permutation matrices [11, 18, 31]. Spectral variants match leading eigenvectors of an alignment operator—EigenAlign [23], GRASP [15]—but are noise-sensitive and still need a separate matching step [29].
Embed-then-match. The prevailing paradigm learns node embeddings and then matches them [29]. Random-walk and GNN encoders—REGAL [14], CONE-Align [5], WAlign [12], MMNC [34]—capture structural proximity but require training and, on plain graphs, a way to make two independently trained spaces comparable. Optimal-transport aligners cast matching as a (Gromov–)Wasserstein coupling—GWL [39], S-GWL [38], SLOTAlign [32], FGWEA [33], CombAlign [4], FUGAL [2]—which is accurate but leans on iterative transport solvers that are costly at scale.

Iterative / gradual alignment. To avoid committing every match at once, gradual methods promote high-confidence pairs to pseudo-anchors and propagate [10, 26, 36]. They reuse pseudo-anchors by learning a transform between embedding spaces [12, 34], reweighting topology [10, 22, 36], or recalibrating similarities [23, 26, 41]—at the cost of re-embedding and re-matching each round.

Positioning. Many strong baselines mix regimes: GradAlign and CCNE consume seed anchors [26, 40], while CANA and attributed SLOTAlign use node features [30, 32]; we therefore separate *plain-unsupervised*, *anchor-supervised*, and *attributed* methods in our evaluation (§4). SparkAlign is plain-unsupervised and uniquely *training-free*: its only "embedding" is closed-form diffusion, and it improves by sparsifying—not retraining—the matcher.

3 The SparkAlign Method

3.1 Problem and Notation

Let $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s, \boldsymbol{A}_s)$ and $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t, \boldsymbol{A}_t)$ be a source and target graph with $n_s = |\mathcal{V}_s|$ and $n_t = |\mathcal{V}_t|$ unattributed nodes and binary adjacency matrices $\boldsymbol{A}_\star \in \{0,1\}^{n_\star \times n_\star}$; the two sizes may differ. UPGA seeks a *partial* one-to-one matching $\mathcal{M} = \{(u, v)\}$ between subsets $\mathcal{V}'_s \subseteq \mathcal{V}_s$ and $\mathcal{V}'_t \subseteq \mathcal{V}_t$ (so $n_s \neq n_t$ is allowed), using neither attributes nor *ground-truth* anchors. We distinguish these from the *pseudo-anchors* $\mathcal{M}^{(r)}$—pairs the algorithm has itself committed by round r and treats as temporary landmarks. We write $\boldsymbol{D}_\star = \mathrm{diag}(\boldsymbol{A}_\star \mathbf{1})$ for the degree matrix and $\|\cdot\|_2$ for the Euclidean norm.

3.2 Framework Overview

SparkAlign runs for R rounds; round r performs two stages,

$$\boldsymbol{H}^{(r)} = \underbrace{\mathrm{Embed}\Big(\mathcal{G}_s, \mathcal{G}_t, \mathcal{M}^{(r-1)}\Big)}_{\text{represent}}, \quad \mathcal{M}^{(r)} = \mathcal{M}^{(r-1)} \cup \underbrace{\mathrm{Match}\Big(\boldsymbol{H}^{(r)}\Big)}_{\text{match}}, \tag{1}$$

where $\boldsymbol{H}^{(r)} = (\boldsymbol{H}_s^{(r)}, \boldsymbol{H}_t^{(r)})$ is the round-r descriptor pair, Embed embeds the still-unmatched nodes relative to the current pseudo-anchors $\mathcal{M}^{(r-1)}$, and Match commits a few new high-confidence pairs (Fig. 1). Committed pairs join the landmark set, so later rounds resolve harder nodes against a richer reference frame. Here R is the number of rounds, L the diffusion steps, a_r the landmarks at round r, P the candidates per node, and N_{sel} the commits per round.

3.3 Anchor-Relative Diffusion Representation

Given the $a_r = |\mathcal{M}^{(r-1)}|$ pseudo-anchors $\mathcal{M}^{(r-1)} = \{(u_m, v_m)\}_{m=1}^{a_r}$, we seed a signal at each landmark: the columns of $\boldsymbol{Z}_s^{(0)} \in \mathbb{R}^{n_s \times a_r}$ (resp. $\boldsymbol{Z}_t^{(0)}$) are one-hot indicators of u_m (resp. v_m), and unmatched nodes start at zero. We diffuse this signal over each graph with its own symmetric operator

$$\boldsymbol{Z}_\star^{(\ell)} = \boldsymbol{Q}_\star \, \boldsymbol{Z}_\star^{(\ell-1)}, \quad \ell = 1, \ldots, L, \qquad \boldsymbol{Q}_\star = \boldsymbol{I} + \boldsymbol{D}_\star^{-\frac{1}{2}} \boldsymbol{A}_\star \, \boldsymbol{D}_\star^{-\frac{1}{2}}, \tag{2}$$

for $\star \in \{s, t\}$; the round-r descriptor is the raw diffusion output $\boldsymbol{H}_\star^{(r)} = \boldsymbol{Z}_\star^{(L)}$. Entry (i, m) of $\boldsymbol{Z}_\star^{(L)}$ measures how strongly heat from landmark m reaches node i—node i's coordinates *relative to the landmarks*, exactly the star-hopping intuition.

Why it is discriminative across graphs. Heat diffusion is a deterministic function of local topology, so two nodes that play the same structural role receive near-identical diffusion signatures regardless of where they sit [6, 9]; higher-order diffusion further denoises the structural signal against the edge differences between the two graphs [13]. Anchoring diffusion at a *shared* landmark set turns these per-graph descriptors into a common coordinate system—the diffusion analogue of landmark/pivot embeddings, where representing points by distances to common references yields globally consistent coordinates [8], and the same principle that lets structural-identity embeddings transfer across disjoint graphs [14]. A true pair (u, v) sees the same landmarks through the same local structure, so their anchor-relative descriptors coincide—which is what makes unsupervised matching possible. Unlike the fixed structural embeddings of GraphWave and REGAL, our descriptor is re-seeded from *discovered* landmarks each round and thus sharpens as alignment proceeds; unlike landmark/pivot embeddings, its landmarks are discovered rather than supplied.

Cold start. In the first round $\mathcal{M}^{(0)} = \emptyset$, so no landmarks exist. We instead diffuse from the identity, $\boldsymbol{Z}_s^{(0)} = \boldsymbol{I}_{n_s}$ and $\boldsymbol{Z}_t^{(0)} = \boldsymbol{I}_{n_t}$ (each node is its own source), then make the rows comparable across graphs by *canonical sorting*: we sort each row of $\boldsymbol{Z}_\star^{(L)}$ in descending order, keep its top-d entries ($d = \min(n_s, n_t)$, which equalizes the signature length across the two graphs), and ℓ_2-normalize to form $\boldsymbol{H}_\star^{(r)}$. Sorting discards the arbitrary node-index ordering, yielding permutation-invariant structural signatures that bootstrap the first set of landmarks.

Stability: why a finite depth is optimal. The operator $\boldsymbol{Q}$ in Eq. (2) is *not* contractive: $\boldsymbol{S} = \boldsymbol{D}^{-1/2}\boldsymbol{A}\boldsymbol{D}^{-1/2}$ has spectrum in $[-1, 1]$ with top eigenvalue 1, so $\boldsymbol{Q} = \boldsymbol{I} + \boldsymbol{S}$ has eigenvalues in $[0, 2]$ [17, 37]. Iterating $\boldsymbol{Q}$ is therefore a power iteration that amplifies the top eigenvector $\boldsymbol{u}_1 \propto \boldsymbol{D}^{1/2}\mathbf{1}$, a purely degree-determined signal.

Lemma 3.1 (Over-smoothing limit). *For a connected graph let $\boldsymbol{S}$ have eigenpairs $(\lambda_i, \boldsymbol{u}_i)$ ordered $1 = \lambda_1 > \lambda_2 \geq \cdots \geq \lambda_n \geq -1$. For any seed $\boldsymbol{x}_0$ with $\langle \boldsymbol{x}_0, \boldsymbol{u}_1 \rangle > 0$ (in particular any nonnegative seed, since $\boldsymbol{u}_1 > 0$ entrywise), the ℓ_2-normalized iterate obeys*

$$\left\| \frac{\boldsymbol{Q}^L \boldsymbol{x}_0}{\|\boldsymbol{Q}^L \boldsymbol{x}_0\|_2} - \boldsymbol{u}_1 \right\|_2 = O\Big(\big(\tfrac{1+\lambda_2}{2}\big)^L\Big) \xrightarrow{L \to \infty} 0.$$

Each landmark column of $\boldsymbol{Z}_\star^{(L)}$ is such an iterate, so by Lemma 3.1 every column aligns with $\boldsymbol{u}_1$ and $\boldsymbol{Z}_\star^{(L)}$ approaches a rank-one matrix $\boldsymbol{u}_1 \boldsymbol{c}^\top$: node i's descriptor row degenerates to $u_{1,i}\, \boldsymbol{c}^\top$—a single degree-determined scalar times a shared vector—so structurally distinct nodes become indistinguishable, the graph analogue of over-smoothing [20, 24]. This degeneration (together with the $\Theta(2^L)$ growth of the iterate) worsens with depth, whereas too small an L leaves descriptors local and noisy; diffusion depth is therefore a locality–smoothing trade-off with a finite optimum—the peak-then-decay behavior we observe when sweeping L (§4), and we operate at a small L where magnitudes stay bounded. We use $\boldsymbol{Q} = \boldsymbol{I} + \boldsymbol{S}$ rather than $\boldsymbol{S}$ because the self-loop keeps the spectrum in $[0, 2]$: this removes the sign oscillation a (near-)bipartite $\boldsymbol{S}$ would suffer and, since the power-iteration rate $(1+\lambda_2)/2$ exceeds λ_2, slows this degeneration, widening the useful range of L. A contractive scaling (e.g. a lazy walk $\alpha\boldsymbol{I} + (1-\alpha)\boldsymbol{S}$) shares $\boldsymbol{S}$'s eigenvectors and the same degeneration, only bounding magnitude and shifting this useful window—which our grid search over L already absorbs—so we keep the simplest operator.

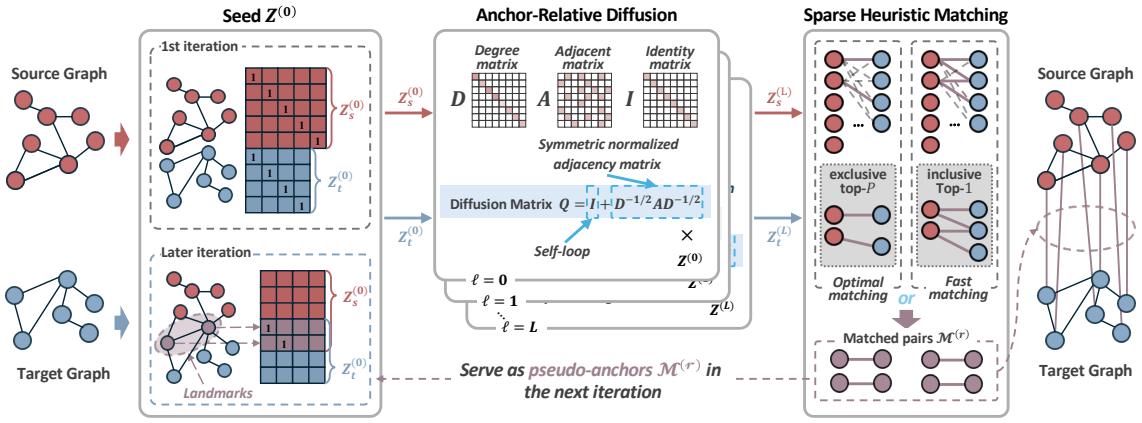

Figure 1: SparkAlign. Each round embeds unmatched nodes by heat diffusion from committed landmark pairs (§3.3), then commits a few confident matches with a sparse matcher (§3.4); commits become landmarks for the next round.

3.4 Sparse Heuristic Matching

Both matchers act on the unmatched sets $\mathcal{V}_s^{(r)}, \mathcal{V}_t^{(r)}$ (each of size n_r) and commit only the N_{sel} most confident pairs per round; confidence is the descriptor distance, which also induces the ranking used by Hits@1 and MRR.

Fast matching ($SparkAlign_f$). Each source u proposes its nearest target $v_u^\star = \arg\min_v \|\boldsymbol{h}_{s,u} - \boldsymbol{h}_{t,v}\|_2$; all proposals are computed in parallel. We globally sort proposals by distance and accept the best N_{sel} as new landmarks. This curbs nearest-neighbor matching's many-to-one collisions—a contested target is claimed by at most one pair per round—with no sequential dependency.

Sparse optimal matching ($SparkAlign_o$). For strict one-to-one quality we restrict each source u to its P nearest targets $\mathcal{N}_P(u)$ and set every other entry of the cost matrix to $+\infty$,

$$C_{u,v} = \begin{cases} \|\boldsymbol{h}_{s,u} - \boldsymbol{h}_{t,v}\|_2, & v \in \mathcal{N}_P(u), \\ +\infty, & \text{otherwise.} \end{cases} \tag{3}$$

A Modified Jonker–Volgenant solver [7] returns the optimal injective assignment—exact within the candidate set—committing only the N_{sel} lowest-cost pairs. The $+\infty$ pairs are never selected, so the solver's augmenting-path search skips almost all of the n_r^2 entries; in practice the matcher is dominated by forming the cost matrix, not by a cubic solve.

Complexity. Per round, diffusion is L sparse matrix products, $O(L|\mathcal{E}|\delta)$ for $|\mathcal{E}|$ edges and descriptor width δ (the landmark count a_r, or the cold-start dimension d); pairwise distances cost $O(n_r^2\delta)$; fast matching adds $O(n_r^2)$, and the sparse-optimal matcher's cost matrix is likewise $O(n_r^2)$, the candidate restriction ($P \ll n_r$) keeping its solver clear of a cubic blow-up in practice. In the *anchor* rounds $\delta = a_r \ll n$, so distances ($O(n_r^2\delta)$) dominate and SparkAlign is near-quadratic. Cold start is heavier—$\delta = d = \min(n_s, n_t)$ makes its diffusion, sort, and distances $O(L|\mathcal{E}|n)$, $O(n^2 \log n)$, $O(n_r^2 d)$—but cheap on a GPU, where wall-clock follows *work* and serial *depth* jointly: these steps are high-work yet embarrassingly parallel (shallow), the depth floor being the R sequential rounds and the augmenting-path solve. Reported runtimes include it.

4 Experiments

Datasets. We use three real-world alignment benchmarks: Facebook–Twitter [3] (~1.0K nodes), DBLP1–DBLP2 [28] (~2.2K), and Arxiv1–Arxiv2 [19] (~18.8K), plus randomly edge-perturbed variants.

Baselines, by regime. We separate methods by what they consume. *Plain-unsupervised* (our setting): FINAL [41], REGAL [14], GRASP [15], CONE-Align [5], MMNC [34], GWL [39], and Alpine [27]. *Anchor-supervised* (given 10% seed links, shown for reference only): GradAlign [26], WLAlign [21], DegUIL [22]. SparkAlign $_o/_f$ are the optimal and fast variants.

Metrics. We report Hits@1 and MRR; the ranking each metric needs comes from sorting candidates by descriptor distance (§3.4). We also report a *structural identifiability reference* (SIR): the fraction of nodes topologically distinguishable from all others (not tied under Weisfeiler–Lehman refinement). It is a *soft* reference, not a hard upper bound, since it credits a match anywhere within an indistinguishable class. All methods are timed on one machine: a 22-core Intel Xeon Platinum 8470Q, 2× NVIDIA RTX PRO 6000 (96 GB), and 110 GB RAM. SparkAlign's hyperparameters are few and forgiving: the diffusion depth is best kept at $L \in [5, 10]$—below 5 the descriptors stay too local, beyond ~10 diffusion over-smooths (Lemma 3.1)—and the per-round commit count is robust over $N_{\text{sel}} \in [10, 200]$, where a larger N_{sel} converges in fewer, faster rounds with accuracy stable up to ~200 before it degrades. We set L, N_{sel}, and the candidate budget P per dataset within these ranges and fix the rest (R, d); all baselines use their authors' recommended configurations.

4.1 Results

Accuracy near the identifiability limit. Table 1: $SparkAlign_o$ and $SparkAlign_f$ match or slightly edge every plain-unsupervised competitor on Hits@1, improving on the strongest baseline (Alpine) by a slim 0.10–0.32 points and approaching the identifiability reference—within 0.2 points of the SIR on DBLP. But the decisive advantages lie beyond accuracy: far lower runtime and markedly more accurate pseudo-anchors. Among the OT baselines, GWL collapses on all three benchmarks (Hits@1≈0), while the stronger

Table 1: Alignment accuracy (Hits@1 / MRR). Bold/underline = best/2nd among *plain-unsupervised* methods; anchor-supervised methods (10% seeds) are shown for reference. SIR is a soft structural-identifiability reference (§4), not a hard ceiling. CONE-Align needs equal graph sizes, so its FB–Tw entries are omitted.

Method	FB–Tw		DBLP		Arxiv	
	H@1	MRR	H@1	MRR	H@1	MRR
Plain-unsupervised						
FINAL	.1967	.2763	.1317	.2121	.1853	.2857
REGAL	.2650	.3245	.6992	.7874	.6379	.7203
GRASP	.2280	.3468	.0497	.0983	.0001	.0008
CONE	–	–	.8168	.8935	.5547	.6667
MMNC	.9140	.9344	.8010	.8858	.7289	.8072
GWL	.0000	.0102	.0023	.0073	.0001	.0006
FUGAL	.8670	.8757	.8703	.9237	.0001	.0006
Alpine	.9690	**.9765**	.8722	.9218	.7778	.8282
SparkAlign$_o$	**.9700**	.9723	**.8740**	**.9239**	**.7803**	.8347
SparkAlign$_f$	.9540	.9674	.8689	.9215	.7779	**.8393**
Anchor-supervised (reference; 10% seeds)						
GradAlign	.9033	.9359	.7676	.8450	.7367	.8115
WLAlign	.9141	.9350	.8191	.9001	.7641	.7913
DegUIL	.4720	.5876	.6188	.7422	.5490	.6497
SIR	.9920	–	.8763	–	.8076	–

Table 2: Efficiency: runtime (s) and peak memory (MB), on the same baselines as Table 1. Bold/underline = best/2nd among *plain-unsupervised* methods; the *vs. Alpine* row gives *SparkAlign*$_f$'s runtime speedup and *SparkAlign*$_o$'s memory ratio over Alpine (green: better).

Method	*Runtime (s)*			*Memory (MB)*		
	FB	DBLP	Arxiv	FB	DBLP	Arxiv
Plain-unsupervised						
FINAL	2.6	13.1	1,748	961	1,415	50,492
REGAL	1.1	2.4	102	**862**	**1,049**	18,736
GRASP	1.8	4.8	1,106	895	1,105	27,801
CONE	–	64.0	3,327	–	1,296	27,761
MMNC	2.8	5.8	517	931	1,123	28,163
GWL	4.9	15.4	51,132	1,472	1,751	17,707
FUGAL	133.7	358.7	13,601	990	1,223	30,884
Alpine	8.5	22.2	8,322	1,396	1,623	24,706
SparkAlign$_o$	3.0	7.7	1,132	1,450	1,524	**8,189**
SparkAlign$_f$	**0.59**	**0.72**	**26.4**	1,491	1,543	9,711
vs. Alpine	14.4×	30.8×	315.2×	0.96×	1.07×	3.02×
Anchor-supervised (reference; 10% seeds)						
GradAlign	16.0	35.3	13,837	1,186	2,199	82,523
WLAlign	108.4	168.5	3,968	1,796	2,257	34,013
DegUIL	9.7	11.9	478	1,956	2,193	16,831

FUGAL matches the best methods on the smaller graphs (.87 Hits@1 on Facebook–Twitter and DBLP) yet collapses on the 18.8K-node Arxiv pair—underscoring that OT solvers struggle to scale, whereas SparkAlign holds .78 Hits@1 there at a fraction of the cost.

Efficiency is the win. Table 2: *SparkAlign*$_f$ is fastest on all three datasets, 14–315× faster than Alpine, finishing the 18.8K-node Arxiv pair in 26.4 s versus Alpine's 2.3 hours; even the optimal variant *SparkAlign*$_o$ stays competitive, and both variants cut peak memory on Arxiv to ∼8–10 GB (∼3× less than Alpine). The training-free descriptor and sparsified matcher remove both efficiency bottlenecks identified in §1.

Table 3: Left: first-round pseudo-anchor accuracy (Hits@1)—the crux of iterative UPGA. Right: diffusion-operator ablation (Hits@1); our $I+S$ (self-loop plus symmetric normalization, $S=D^{-1/2}AD^{-1/2}$; $\hat{A}$ is the renormalized variant) wins, as Lemma 3.1 predicts.

Seed Hits@1				*Operator (Hits@1)*			
Method	FB	DBLP	Arx	Op.	FB	DBLP	Arx
REGAL	.770	.712	.655	$\boldsymbol{D^{-1}A}$	.8950	.8675	.7759
MMNC/CENA	.790	.841	.680	$\boldsymbol{S}$	.9260	.8624	.7727
SparkAlign$_o$	**.980**	**.885**	**.770**	$\hat{\boldsymbol{A}}$	.9320	.8689	.7791
SparkAlign$_f$	**.990**	.838	.760	$\boldsymbol{I+S}$	**.9700**	**.8740**	**.7803**

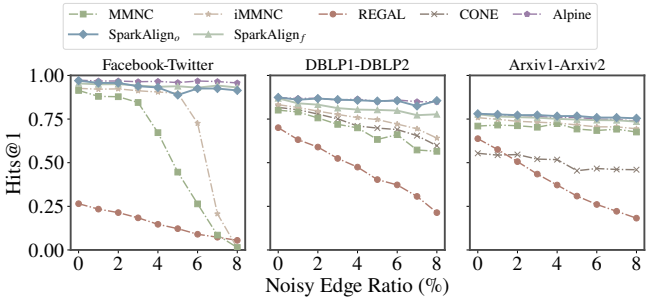

Figure 2: Hits@1 vs. removed-edge ratio (%). SparkAlign degrades gracefully on the larger DBLP and Arxiv pairs and stays competitive with Alpine.

Better seeds. Iterative UPGA lives or dies by its first pseudo-anchors. Table 3 (left): SparkAlign seeds them far more accurately than REGAL or neighborhood-consistency selectors (e.g. .99 vs. .79 on Facebook–Twitter), thanks to the permutation-invariant cold-start signatures. A diffusion-operator ablation (Table 3, right) confirms the design: $\boldsymbol{I+S}$ beats the random-walk, symmetric, and renormalized operators on Hits@1 across all datasets.

Stability and robustness. As Lemma 3.1 predicts, accuracy rises with L, peaks (near $L \approx 5$ on Facebook–Twitter, with a wider plateau on DBLP), then decays as descriptors over-smooth toward the degree signal; the budget P behaves similarly (full sweep in the extended version). Under random edge removal (Fig. 2), SparkAlign degrades most gracefully on the larger DBLP and Arxiv pairs—we attribute this to iterative refinement together with its more accurate seeds—while remaining competitive with Alpine.

5 Conclusion

We presented SparkAlign, a training-free *star-hopping* heuristic for unsupervised plain graph alignment that attacks both efficiency bottlenecks of the embed-then-match paradigm. Its anchor-relative heat-diffusion descriptor needs no training yet is principled—structurally discriminative and, by Lemma 3.1, stable at a finite, optimal diffusion depth—and its two sparse matchers preserve one-to-one quality without paying for dense $O(n^3)$ assignment. SparkAlign matches state-of-the-art accuracy at up to two orders of magnitude less runtime while seeding markedly better pseudo-anchors. Extending the stability analysis to error propagation from imperfect landmarks, and to edge-insertion noise, are natural next steps.

GenAI Usage Disclosure

Generative AI tools were used during preparation of this manuscript in two limited capacities: (i) language editing of author-written text for grammar and concision; and (ii) literature-search assistance to surface candidate related work, each result of which was subsequently read and verified by the authors against the primary sources. No part of the methodology, experimental design, results, or analysis was generated by AI, and the authors take full responsibility for the content.

References

[1] Anonymous Author(s). 2026. SparkAlign: Source Code and Configurations. https://anonymous.4open.science/r/SparkAlign-08C0/. Anonymized repository for double-blind review.

[2] Aditya Bommakanti, Harshith R Vonteri, Konstantinos Skitsas, Sayan Ranu, Davide Mottin, and Panagiotis Karras. 2024. Fugal: Feature-fortified unrestricted graph alignment. *Advances in Neural Information Processing Systems* 37 (2024), 19523–19546.

[3] Xuezhi Cao and Yong Yu. 2016. BASS: A Bootstrapping Approach for Aligning Heterogenous Social Networks. In *European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*. 459–475.

[4] Songyang Chen, Yu Liu, Lei Zou, Zexuan Wang, and Youfang Lin. 2025. CombAlign: Enhancing Model Expressiveness in Unsupervised Graph Alignment. *IEEE Transactions on Knowledge and Data Engineering* 38, 2 (2025), 956–968.

[5] Xiyuan Chen, Mark Heimann, Fatemeh Vahedian, and Danai Koutra. 2020. CONE-Align: Consistent Network Alignment with Proximity-Preserving Node Embedding. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*. 1985–1988.

[6] Fan Chung. 2007. The heat kernel as the pagerank of a graph. *Proceedings of the National Academy of Sciences* 104, 50 (2007), 19735–19740.

[7] David F. Crouse. 2016. On implementing 2D rectangular assignment algorithms. *IEEE Trans. Aerospace Electron. Systems* 52, 4 (2016), 1679–1696.

[8] Vin De Silva and Joshua B Tenenbaum. 2004. *Sparse multidimensional scaling using landmark points*. Technical Report. technical report, Stanford University.

[9] Claire Donnat, Marinka Zitnik, David Hallac, and Jure Leskovec. 2018. Learning structural node embeddings via diffusion wavelets. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1320–1329.

[10] Xingbo Du, Junchi Yan, and Hongyuan Zha. 2019. Joint Link Prediction and Network Alignment via Cross-graph Embedding. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. 2251–2257.

[11] Zhou Fan, Cheng Mao, Yihong Wu, and Jiaming Xu. 2020. Spectral Graph Matching and Regularized Quadratic Relaxations: Algorithm and Theory. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*, Hal Daumé III and Aarti Singh (Eds.). PMLR, 2985–2995. https://proceedings.mlr.press/v119/fan20a.html

[12] Ji Gao, Xiao Huang, and Jundong Li. 2021. Unsupervised Graph Alignment with Wasserstein Distance Discriminator. In *Proceedings of the ACM International SIGKDD Conference on Knowledge Discovery and Data Mining (SIGKDD)*. 426–435.

[13] Johannes Gasteiger, Stefan Weißenberger, and Stephan Günnemann. 2019. Diffusion improves graph learning. *Advances in neural information processing systems* 32.

[14] Mark Heimann, Haoming Shen, Tara Safavi, and Danai Koutra. 2018. Regal: Representation learning-based graph alignment. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*. 117–126.

[15] Judith Hermanns, Konstantinos Skitsas, Anton Tsitsulin, Marina Munkhoeva, Alexander Kyster, Simon Nielsen, Alexander M Bronstein, Davide Mottin, and Panagiotis Karras. 2023. Grasp: Scalable graph alignment by spectral corresponding functions. *ACM Transactions on Knowledge Discovery from Data* 17, 4 (2023), 1–26.

[16] Chuanyu Jiang, Yiming Qian, Lijun Chen, Yang Gu, and Xia Xie. 2023. Unsupervised deep cross-language entity alignment. In *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 3–19.

[17] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations (ICLR)*.

[18] Danai Koutra, Hanghang Tong, and David Lubensky. 2013. BIG-ALIGN: Fast Bipartite Graph Alignment. In *IEEE International Conference on Data Mining (ICDM)*. 389–398.

[19] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[20] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[21] Li Liu, Penggang Chen, Xin Li, William K. Cheung, Youmin Zhang, Qun Liu, and Guoyin Wang. 2024. WL-Align: Weisfeiler-Lehman Relabeling for Aligning Users Across Networks via Regularized Representation Learning. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 36, 1 (2024), 445–458.

[22] Meixiu Long, Siyuan Chen, Xin Du, and Jiahai Wang. 2023. DegUIL: Degree-aware graph neural networks for long-tailed user identity linkage. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 122–138.

[23] Huda Nassar, Nate Veldt, Shahin Mohammadi, Ananth Grama, and David F Gleich. 2018. Low rank spectral network alignment. In *Proceedings of the 2018 World Wide Web Conference*. 619–628.

[24] Kenta Oono and Taiji Suzuki. 2019. Graph neural networks exponentially lose expressive power for node classification. *arXiv preprint arXiv:1905.10947*.

[25] Jin-Duk Park, Cong Tran, Won-Yong Shin, and Xin Cao. 2022. GradAlign+: Empowering Gradual Network Alignment Using Attribute Augmentation. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*. 4374–4378.

[26] Jin-Duk Park, Cong Tran, Won-Yong Shin, and Xin Cao. 2023. On the Power of Gradual Network Alignment Using Dual-Perception Similarities. *IEEE Transaction on Pattern Analysis and Machine Intelligence (PAMI)* 45, 12 (2023), 15292–15307.

[27] Petros Petsinis, Konstantinos Skitsas, Sayan Ranu, Davide Mottin, and Panagiotis Karras. 2025. Alpine: Partial Unlabeled Graph Alignment. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*. 2315–2325.

[28] Adriana Prado, Marc Plantevit, Céline Robardet, and Jean-François Boulicaut. 2013. Mining Graph Topological Patterns: Finding Covariations among Vertex Descriptors. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 25, 9 (2013), 2090–2104.

[29] Shruti Saxena and Joydeep Chandra. 2024. A Survey on Network Alignment: Approaches, Applications and Future Directions. In *Proceedings of the International Joint Conference on Artificial Intelligence, (IJCAI)*. 8216–8224.

[30] Jiangli Shao, Yongqing Wang, Fangda Guo, Boshen Shi, Huawei Shen, and Xueqi Cheng. 2023. CANA: Causal-enhanced Social Network Alignment. In *Proceedings of the 32nd ACM international conference on information and knowledge management*. 2219–2228.

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

[32] Jianheng Tang, Weiqi Zhang, Jiajin Li, Kangfei Zhao, Fugee Tsung, and Jia Li. 2023. Robust Attributed Graph Alignment via Joint Structure Learning and Optimal Transport. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. 1638–1651.

[33] Jianheng Tang, Kangfei Zhao, and Jia Li. 2023. A Fused Gromov-Wasserstein Framework for Unsupervised Knowledge Graph Entity Alignment. In *Findings of the Association for Computational Linguistics: ACL 2023*, Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, Toronto, Canada, 3320–3334.

[34] Wei Tang, Haifeng Sun, Jingyu Wang, Qi Qi, Jing Wang, Hao Yang, and Shimin Tao. 2023. Multi-order Matched Neighborhood Consistent Graph Alignment in a Union Vector Space. In *Proceedings of the International Conference on Research and Development in Information Retrieval (SIGIR)*. 963–972.

[35] Huynh Thanh Trung, Tong Van Vinh, Nguyen Thanh Tam, Hongzhi Yin, Matthias Weidlich, and Nguyen Quoc Viet Hung. 2020. Adaptive Network Alignment with Unsupervised and Multi-order Convolutional Networks. In *IEEE International Conference on Data Engineering (ICDE)*. 85–96.

[36] Chenxu Wang, Peijing Jiang, Xiangliang Zhang, Pinghui Wang, Tao Qin, and Xiaohong Guan. 2023. Gtcalign: Global topology consistency-based graph alignment. *IEEE Transactions on Knowledge and Data Engineering* 36, 5 (2023), 2009–2025.

[37] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *International conference on machine learning*. Pmlr, 6861–6871.

[38] Hongteng Xu, Dixin Luo, and Lawrence Carin. 2019. *Scalable gromov-wasserstein learning for graph partitioning and matching*.

[39] Hongteng Xu, Dixin Luo, Hongyuan Zha, and Lawrence Carin Duke. 2019. Gromov-wasserstein learning for graph matching and node embedding. In *International conference on machine learning*. PMLR, 6932–6941.

[40] Hai-Feng Zhang, Guojing Ren, Xiao Ding, Li Zhou, and Xingyi Zhang. 2024. Collaborative cross-network embedding framework for network alignment. *IEEE Transactions on Network Science and Engineering* 11, 3 (2024), 2989–3001.

[41] Si Zhang and Hanghang Tong. 2016. FINAL: Fast Attributed Network Alignment. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*. 1345–1354.