Iterative graph alignment via supermodular approximation

Aritra Konar

and

Nicholas D. Sidiropoulos

Department of Electrical and Computer Engineering

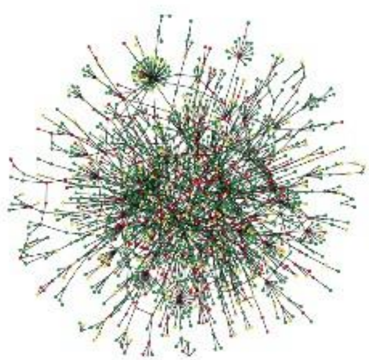
IEEE International Conference on Data Mining

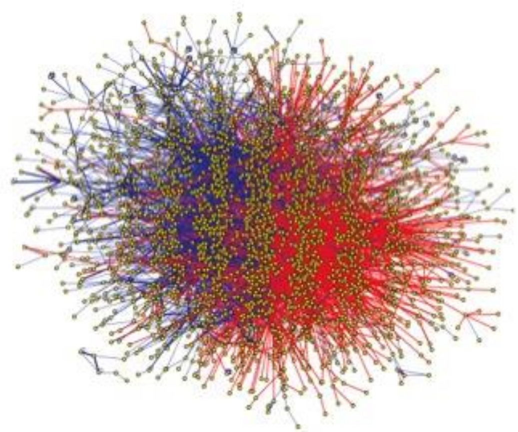
Beijing, China

November 9, 2019

Graph Matching

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$





$$\mathbf{B} = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$

$\mathbf{A} \in \mathbb{R}_+^{n_A \times n_A}$ $\quad$ $\mathbf{B} \in \mathbb{R}_+^{n_B \times n_B}$

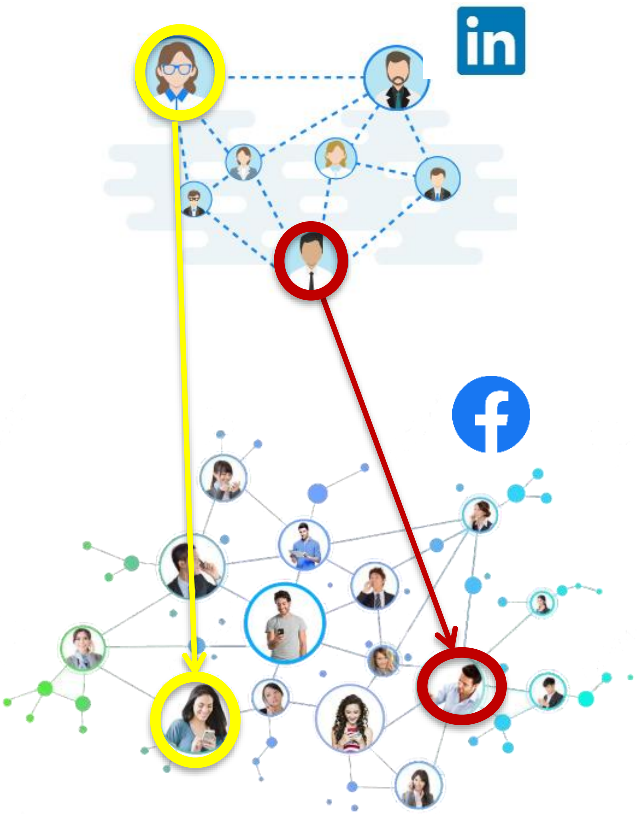
- Problem: Find correspondence mapping between vertex sets that best preserves adjacency relations

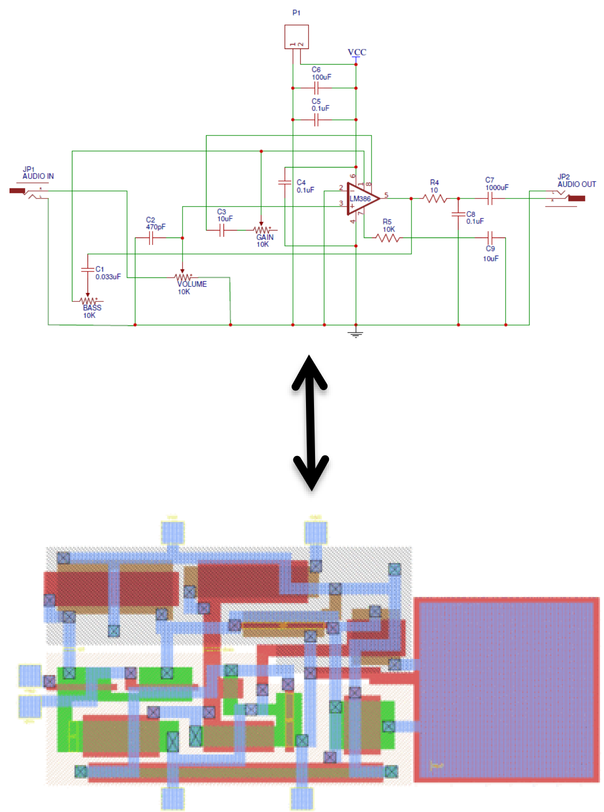
Minimize edge disagreements

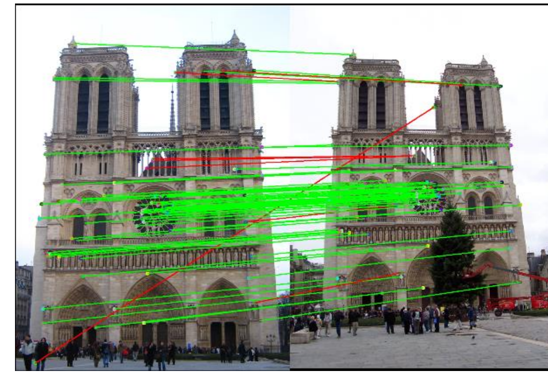
$$\begin{aligned} \min. \quad & \boxed{\|\mathbf{B} - \mathbf{P}\mathbf{A}\mathbf{P}^T\|_F^2} \\ \text{s.t.} \quad & \mathbf{P} \in \mathcal{P} := \{\mathbf{P} \in \{0,1\}^{n_B \times n_A} \mid \mathbf{P}^T\mathbf{P} = \mathbf{I}\} \end{aligned}$$

Correspondence mappings

Applications







Scene matching

Network de-anonymization

Electronic circuit design

Ontology alignment

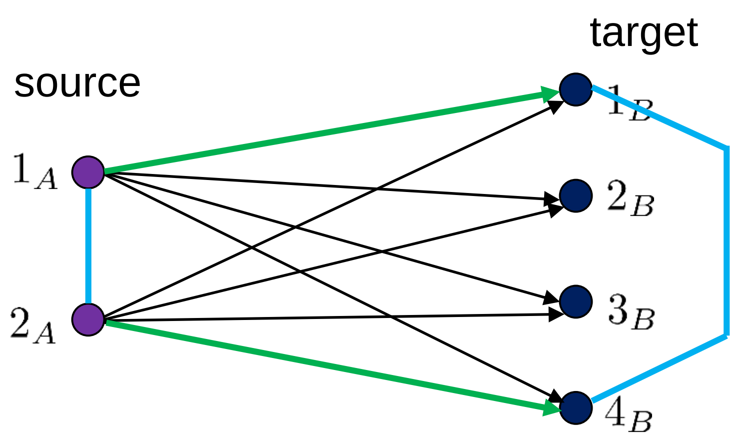
Graph Matching

- Vectorization: Define $\mathbf{x} = \mathrm{vec}(\mathbf{P}), n = n_A n_B$

$$\begin{aligned} \max. \quad & \boxed{\left\{ f(\mathbf{x}) := \mathbf{x}^T (\mathbf{A} \otimes \mathbf{B}) \mathbf{x} \right\}} \\ \text{s.t.} \quad & \mathbf{x} \in \mathcal{M} \end{aligned}$$

Maximize edge overlap

Set of maximum cardinality matchings in $\mathcal{G}_C$



$$\mathcal{G}_C = (\mathcal{V}_A \cup \mathcal{V}_B, \mathcal{E}_C)$$

Computational Challenges

- Graph Matching
  - Corresponds to a quadratic assignment problem [Koopmans-Beckmann 57]
- Theoretical: [Sahni-Gonzalez 76]
  - NP-hard (contains subgraph isomorphism as a special case)
  - NP-hard to approximate within constant-factor of optimum
- Practical:
  - Space and time complexity of computing + storing $\mathbf{A} \otimes \mathbf{B}$
  - Requires quadratic memory in the size of the graphs

Prior Art

Linear programming relaxation [Sherali-Adams 13]

Semidefinite programming relaxation [Sherali-Adams 13]

Spectral methods: [Umeyama 1988, Singh *et al.* 09, Feizi *et al.* 19]

Continuous domain

Branch and bound [Bazara 83]

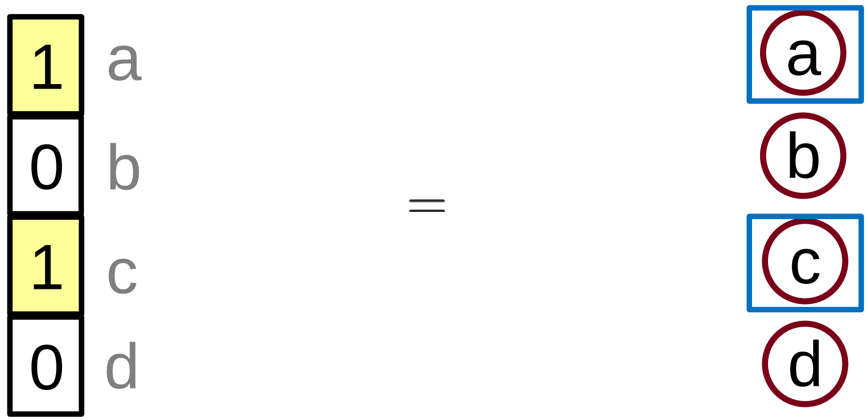
Message passing [Bayati *et al.* 10]

Discrete domain

Problem Reformulation

- Is there a more principled approach that works entirely in the combinatorial domain?
  - Represent discrete function as a set function

$\mathbf{x} \in \mathcal{M}$ $\qquad\qquad$ $\mathbf{x} = \mathbf{1}_{\mathcal{S}}$



$f : \{0,1\}^n \to \mathbb{R}$ $\qquad\qquad$ $F : 2^n \to \mathbb{R}$

  - Discrete problem = subset selection problem

Problem Reformulation

- Final formulation:

$$\boxed{\max_{\mathcal{S}\in\mathcal{I}_A\cap\mathcal{I}_B} \left\{ F(\mathcal{S}) := \mathbf{1}_{\mathcal{S}}^T(\mathbf{A}\otimes\mathbf{B})\mathbf{1}_{\mathcal{S}} \right\}}$$

where $\mathcal{I}_A = \{\mathcal{S}\subset\mathcal{E}_C, |\mathcal{S}\cap\delta(i)|\leq 1, \forall\, i\in\mathcal{V}_A\},$

$\mathcal{I}_B = \{\mathcal{S}\subset\mathcal{E}_C, |\mathcal{S}\cap\delta(j)|\leq 1, \forall\, j\in\mathcal{V}_B\}$

- Conventional wisdom:
  - Constraints are "harder" to handle compared to the objective

- Our perspective:
  - The opposite is true
  - Constraints: $\mathcal{S}\in\mathcal{I}_A\cap\mathcal{I}_B \Leftrightarrow$ matroid intersection

A closer look: objective function

- Key fact: $F(\mathcal{S})$ is a monotone, supermodular function [Konar-Sidiropoulos 19]

- Monotonicity: $\mathcal{A} \subseteq \mathcal{B} \implies F(\mathcal{A}) \leq F(\mathcal{B})$

- Supermodularity: For all $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{E}_C \setminus \{e\}$

$$F(\mathcal{A} \cup \{e\}) - F(\mathcal{A}) \leq F(\mathcal{B} \cup \{e\}) - F(\mathcal{B})$$

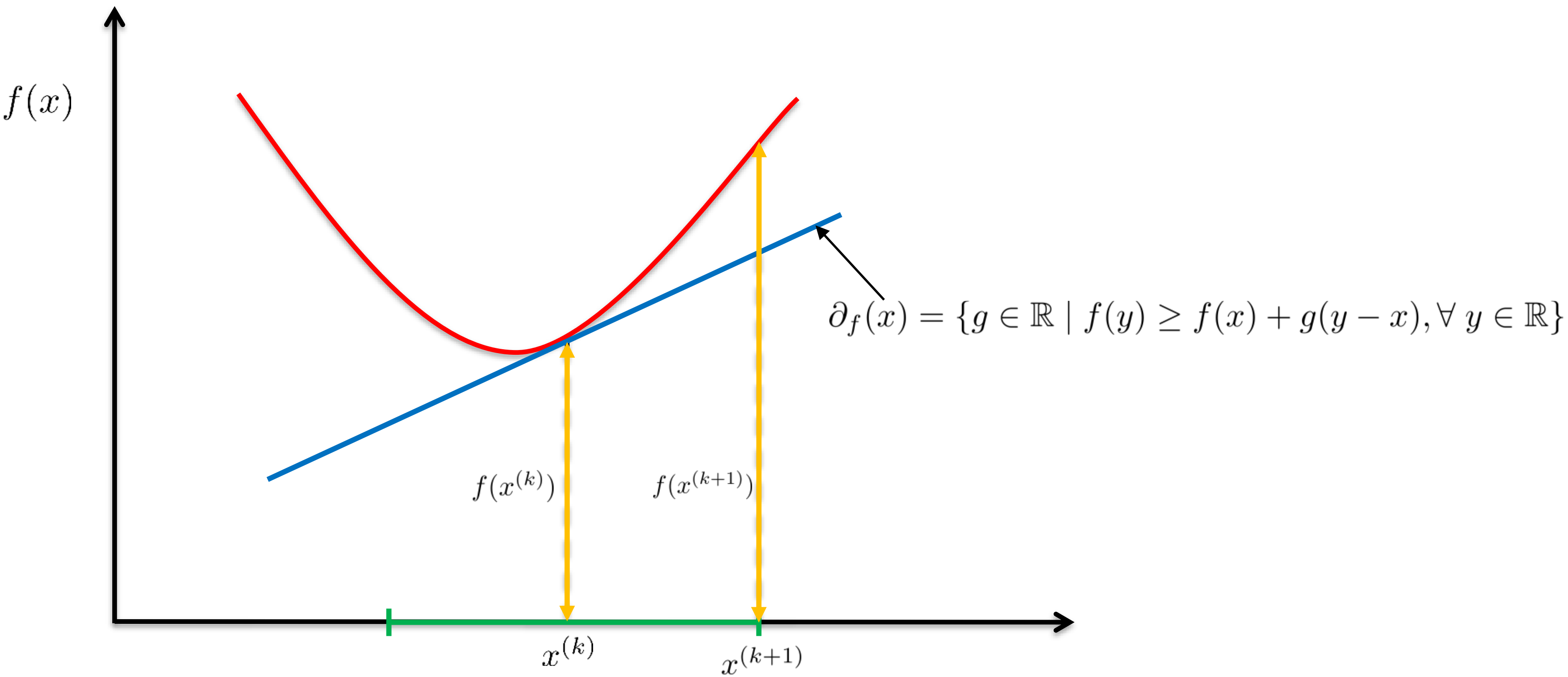
An improving returns property, reminiscent of convexity

Graph Matching

- Key Result:
  - Graph matching is a supermodular maximization problem subject to matroid intersection constraints!
- Take-away:
  - Constraints are manageable, but objective function is difficult to maximize
- Can we exploit supermodularity for approximate maximization?

Majorization Minimization

- Main Idea:
  - Iteratively maximize sequence of global lower bounds on reward



Does the idea carry over to the discrete domain?

Discrete Majorization Minimization

- **Key fact:** Supermodular functions possess (discrete) subgradients! [Jegelka-Bilmes 11]

$$\partial_F(\mathcal{X}) = \{\mathbf{g} \in \mathbb{R}^n \mid F(\mathcal{Y}) \geq F(\mathcal{X}) + G(\mathcal{Y}) - G(\mathcal{X}), \forall\, \mathcal{Y} \subseteq \mathcal{E}_C\}$$

where $G(\mathcal{Y}) = \mathbf{g}^T \mathbf{1}_\mathcal{Y} = \sum_{i \in \mathcal{Y}} g_i$

- **Construction of global lower bound:** [Bai-Bilmes 18]
  - Pick any $\mathbf{g} \in \partial_F(\mathcal{X})$, and define

$$m_\mathcal{X}(\mathcal{Y}) := F(\mathcal{X}) + G(\mathcal{Y}) - G(\mathcal{X})$$

  - Furthermore:

$m_\mathcal{X}(\mathcal{X}) := F(\mathcal{X})$ and $m_\mathcal{X}(\mathcal{Y}) \leq F(\mathcal{Y}), \forall\, \mathcal{Y} \subseteq \mathcal{E}_C$

A global lower bound on the reward function!

Discrete Majorization Minimization

- **Simplification:** For any given $\mathcal{S} \subseteq \mathcal{M}$

Option I: $$g_1(j) = \begin{cases} 2\mathrm{deg}_B(\pi(i))\mathrm{deg}_A(i), & \forall\, j \in \mathcal{S} \\ 2\mathbf{b}_{\bar{\pi}(i)}^T \mathbf{P}\mathbf{a}_i, & \forall\, j \notin \mathcal{S} \end{cases}$$

OR

Option II: $$g_2(j) = \begin{cases} 2\mathbf{b}_{\pi(i)}^T \mathbf{P}\mathbf{a}_i, & \forall\, j \in \mathcal{S} \\ 0, & \forall\, j \notin \mathcal{S} \end{cases}$$

- No Kronecker products required!
- In practice, use Option II (linear memory in size of input graph)

Discrete Majorization Minimization

- The algorithm:
  - Initialization: $\mathcal{S}^{(0)} \in \mathcal{M}$
  - Iterate: $k = \{0, 1, 2 \cdots\}$
    - Obtain subgradient $\mathbf{g}^{(k)} \in \partial_F(\mathcal{S}^{(k)})$
    - Compute update

$$\mathcal{S}^{(k+1)} \in \arg \max_{\mathcal{S} \in \mathcal{I}_A \cap \mathcal{I}_B}. \left\{ m_{\mathcal{S}^{(k)}}(\mathcal{S}) := F(\mathcal{S}^{(k)}) + \boxed{G_k(\mathcal{S})} - G_k(\mathcal{S}^{(k)}) \right\}$$

$$\updownarrow$$

$$\boxed{\mathcal{S}^{(k+1)} \in \arg \max_{\mathcal{S} \in \mathcal{I}_A \cap \mathcal{I}_B}. \left\{ G_k(\mathcal{S}) = (\mathbf{g}^{(k)})^T \mathbf{1}_{\mathcal{S}} \right\}}$$

Linear assignment / maximum weight bipartite matching problem

    - Repeat

Discrete Majorization Minimization

- Features:
  - Purely combinatorial - solves a few weighted bipartite matching problems
  - Guaranteed to improve the reward function:

$$F(\mathcal{S}^{(0)}) \leq F(\mathcal{S}^{(1)}) \leq F(\mathcal{S}^{(2)}) \leq F(\mathcal{S}^{(3)}) \leq \cdots$$

  - Guaranteed to maintain feasibility:

$$\mathcal{S}^{(k)} \in \mathcal{M}, \forall\, k \ \in \{0, 1, 2, \cdots\}$$

  - Complexity: Dominated by cost of solving weighted bipartite matching problem
    - (For $n_A = n_B$) Hungarian algorithm [Kuhn-Munkres 58] / Jonker-Volgenant algorithm [Jonker-Volgenant, 87]
    - (For $n_A < n_B$) Network-Simplex algorithm [Orlin 97]
    - Greedy matching
    - Sinkhorn Matrix Balancing [Cuturi 13, Sinkhorn 67]

Exact

Inexact

Experiments

❑ Setup:

- Given real world graph $\mathbf{A}$, generate noisy graph $\mathbf{B}$

$$\mathbf{B} = \underbrace{\mathbf{P}}_{\text{Ground-truth correspondence}}(\mathbf{A} + \underbrace{\boxed{(\mathbf{1} - \mathbf{A}) * \mathbf{Q}}}_{\text{Add extra edges}})\mathbf{P}^T$$

  - where $\mathbf{Q}$ is a random Erdos-Renyi noise graph

❑ Benchmarks:

  - Umeyama's Method: full EVD of each adjacency [Umeyama 1988]
  - Eigen-Align (EA): top eigen-vector of each adjancency [Feizi *et. al* 2016]
  - IsoRank: Random-walk based [Singh *et. al* 2008]
  - Feature Engineering (FE): local + egonet features [Berlingerio *et. al* 2012]
- Apply greedy matching on output of each algorithm to obtain final correspondence mapping

Experiments

- Implementation:
  - Initialization: Use output of FE
  - Regularization: Use node-level similarity matrix of FE
  - Inner-solver:
    - Exact: Jonker-Volgenant algorithm
    - Inexact: 5 iterations of Sinkhorn Matrix balancing + Greedy

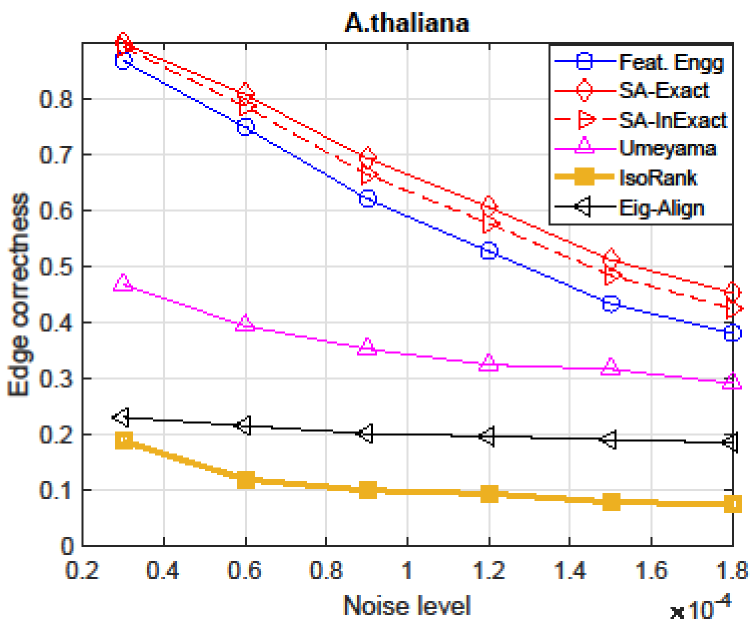
- Evaluation Metrics:
  - Edge correctness
  - Relative degree difference (by degree)

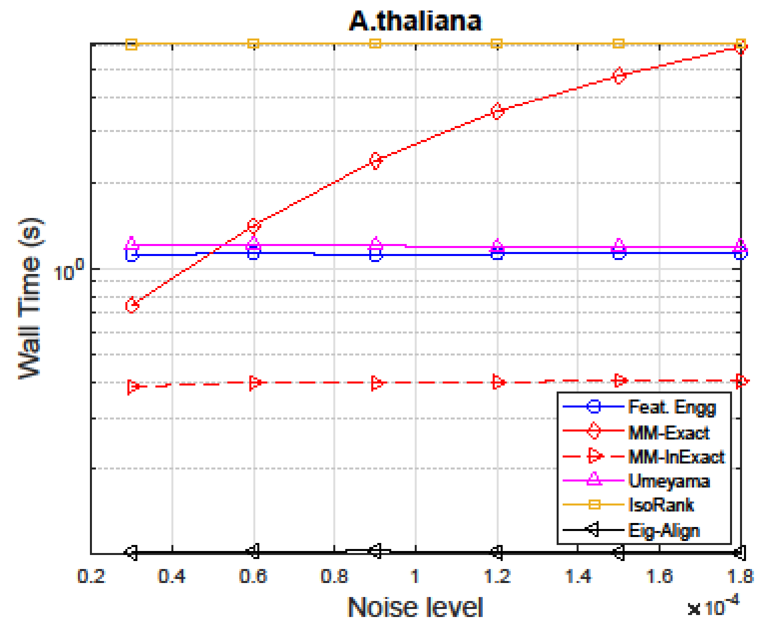
$$\mathrm{rdd}(i, \pi(i)) = \left(1 + \frac{|\deg(i) - \deg(\pi(i))|}{(\deg(i) + \deg(\pi(i)))/2}\right)^{-1}$$

  - Runtime

Results

A. Thaliana (PPI): n = 2,082, m = 4,145

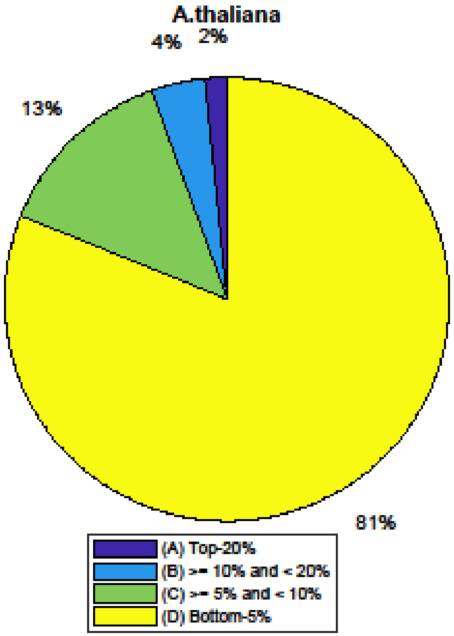


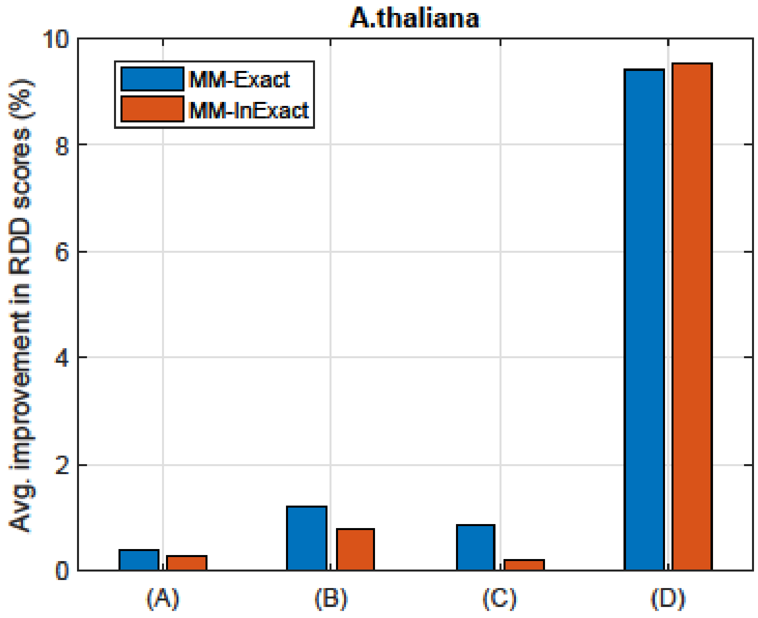


15 % accuracy improvement over FE, MM-Inexact best performance overall

Results

A. Thaliana (PPI): n = 2,082, m = 4,145





Significant improvement in RDD alignment scores for bottom 5% nodes

Conclusions

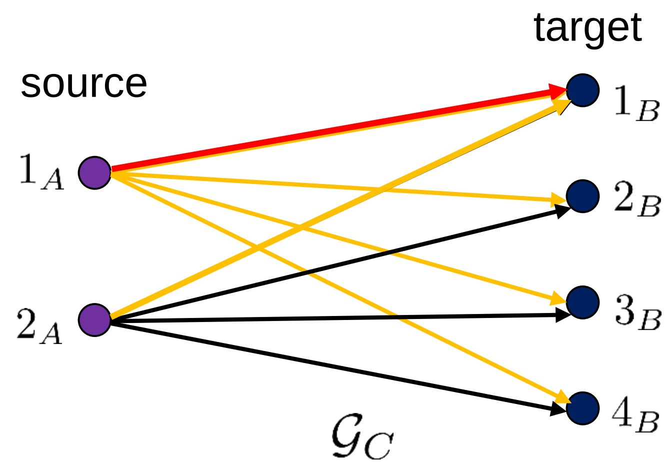
- Graph Matching through the lens of supermodularity:
  - Maximizing a supermodular function subject to matroid intersection constraints
  - Combinatorial local search based on discrete MM
    - Solve a sequence of bipartite matching problems
    - Does not require computing expensive Kronecker products
    - FE + Inexact version yields state-of-the-art performance on real-world data

- Future Work:
  - Instance specific approximation guarantees
  - Joint embedding + matching

Thank you!

A closer look: the constraints

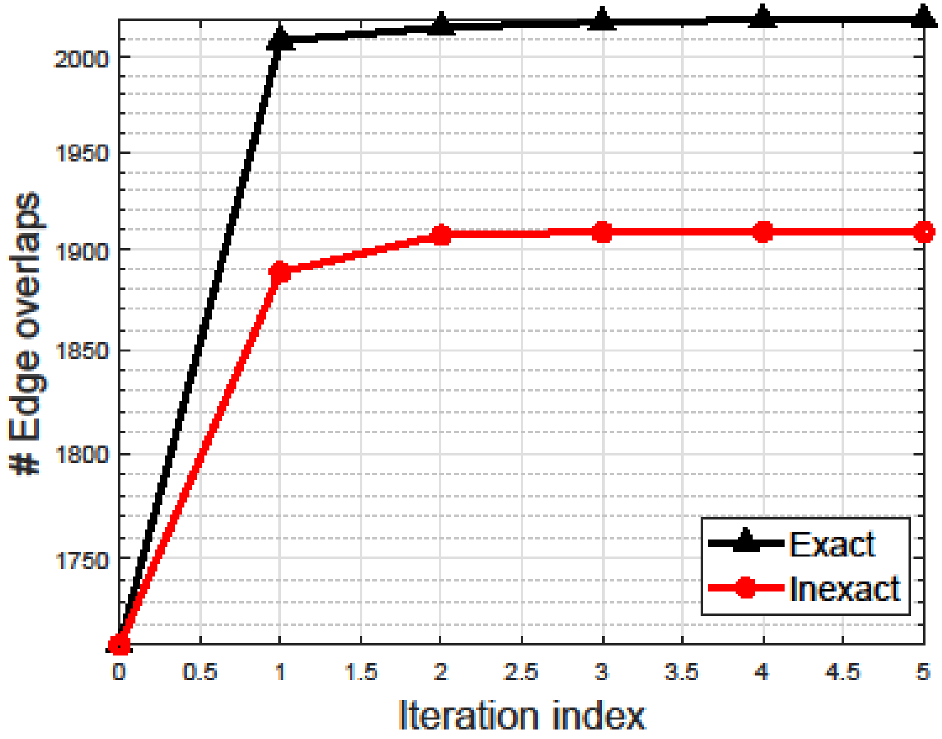
- Interpretation: $\mathcal{S} \in \mathcal{I}_A \cap \mathcal{I}_B$



- Set $\mathcal{I}_A$:
  - For every source vertex, only one outgoing edge can be selected
  - A partition matroid on the edges of $\mathcal{G}_C$
- Set $\mathcal{I}_B$:
  - For every target vertex, only one incoming edge can be selected
  - Also a partition matroid on the edges of $\mathcal{G}_C$
- Matching sets equivalent to intersection of partition matroids

Exact or Inexact?

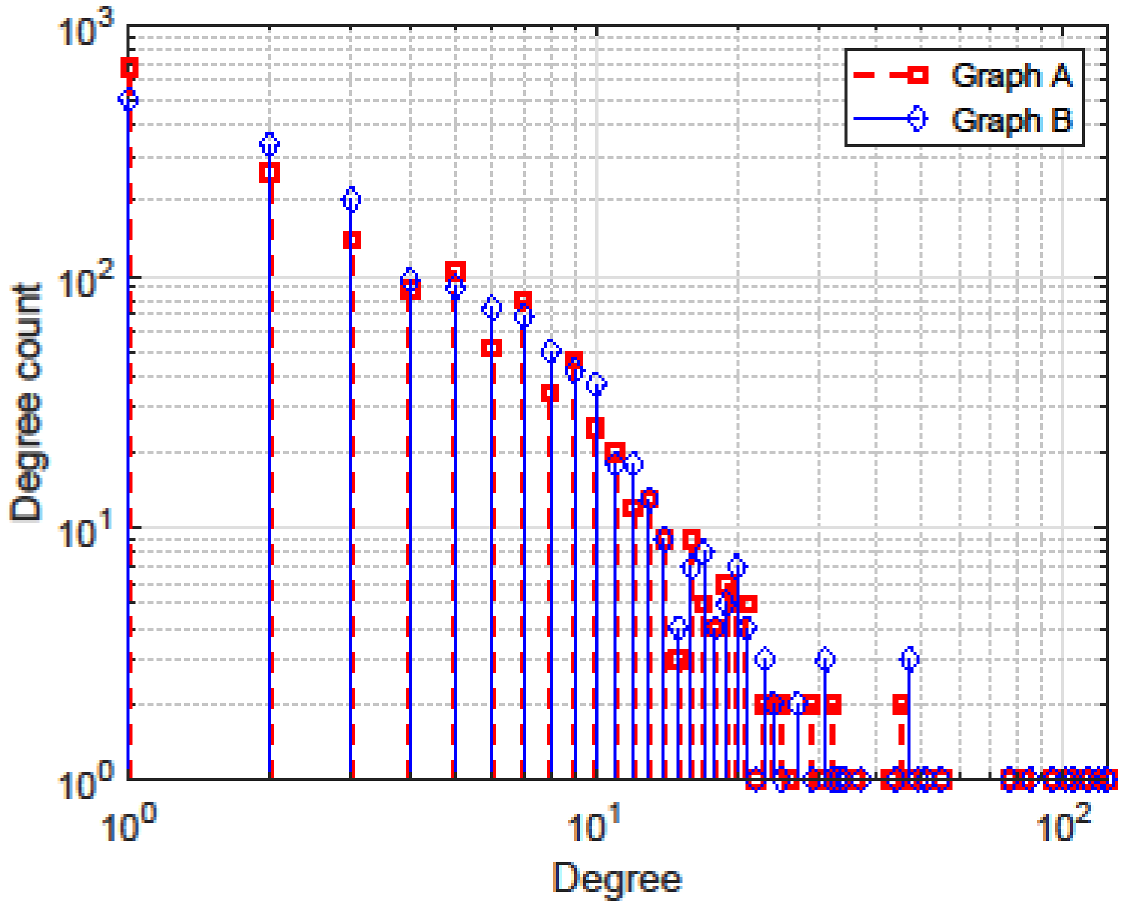
A. Thaliana (PPI): n = 2082, m = 4145



Inexact: 5 % performance loss, 10x speedup, approx. convergence in 1 iteration

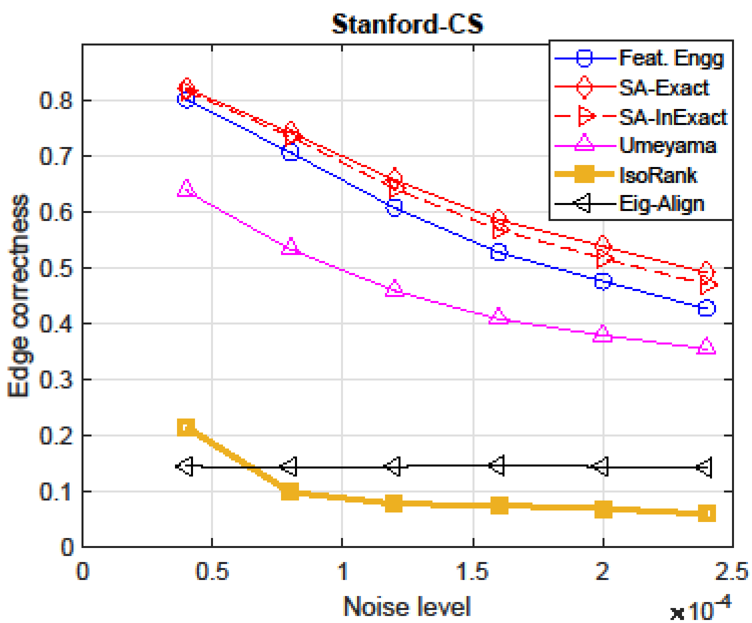
Setup

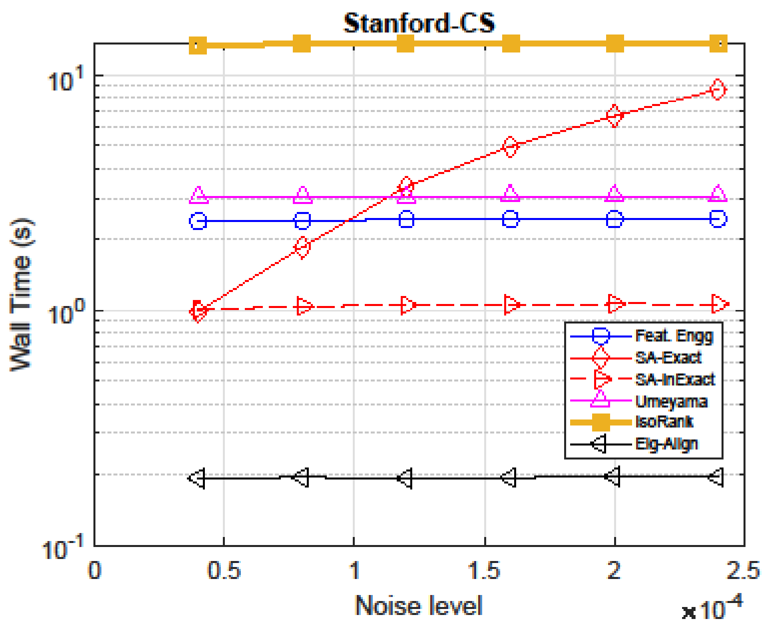
A. Thaliana (PPI): n = 2,082, m = 4,145



Results

Stanford-CS (web): n = 2,759, m = 10,270

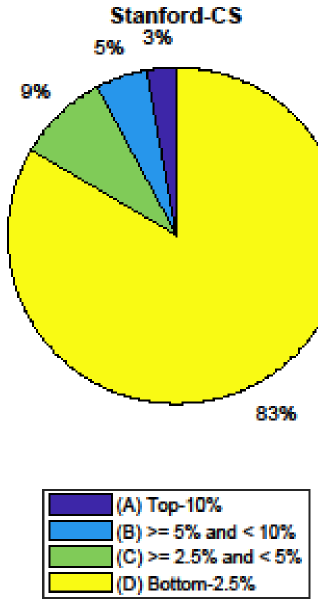


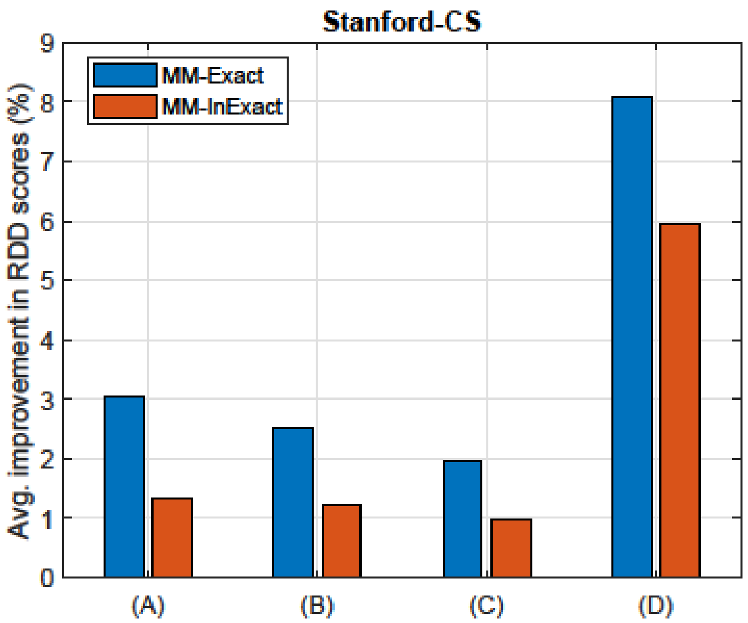


10 % accuracy improvement over FE, MM-Inexact best performance overall

Results

Stanford-CS (web): n = 2,759, m = 10,270

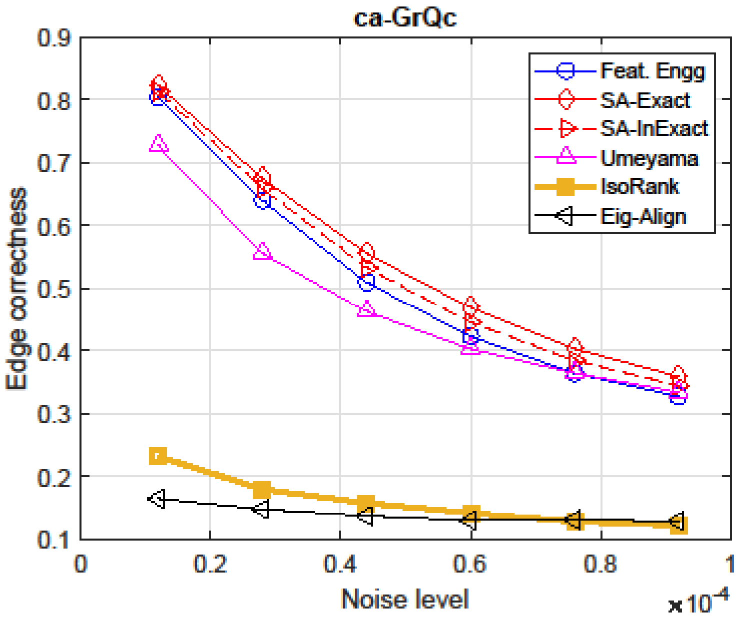


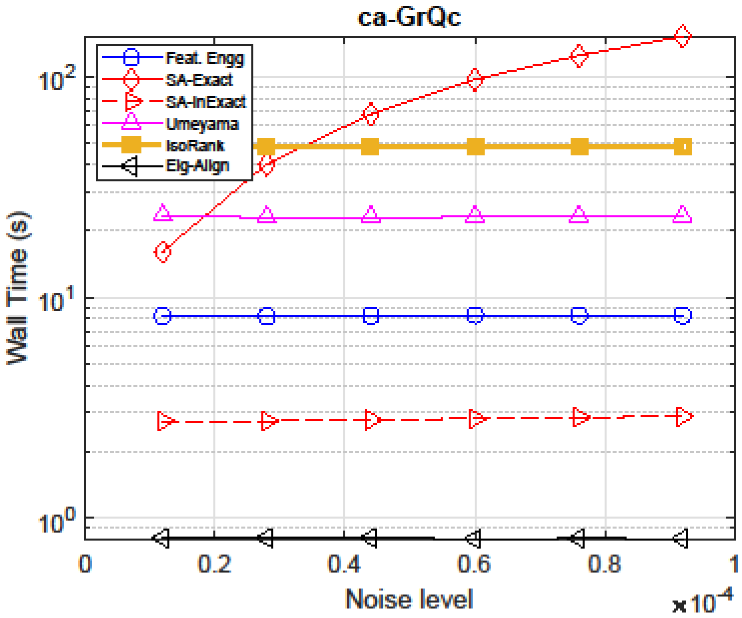


Significant improvement in RDD alignment scores of bottom 5% nodes

Results

Ca-GrQc (co-authorship): n = 5,242, m = 14,490

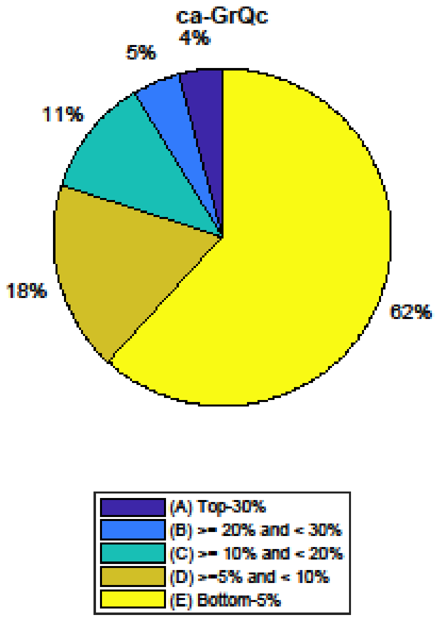


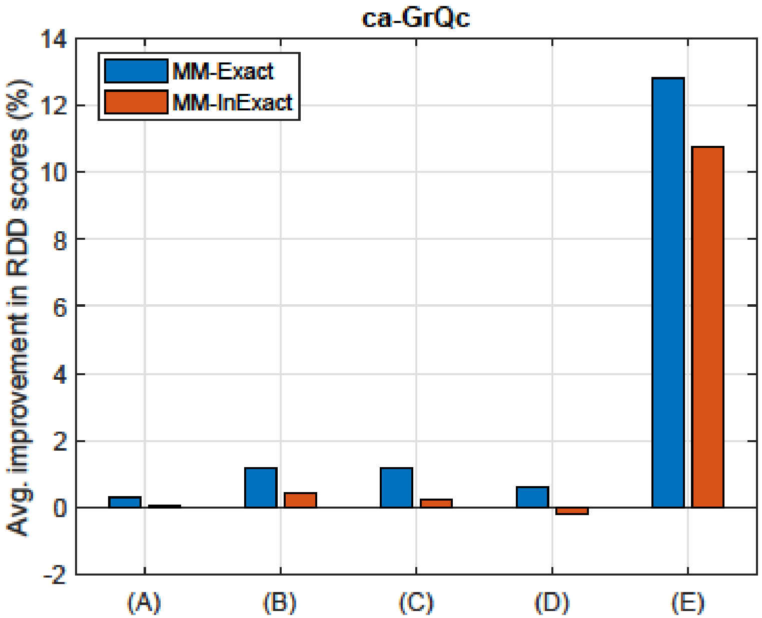


10 % accuracy improvement over FE, MM-Inexact best performance overall

Results

Ca-GrQc (co-authorship): n = 5,242, m = 14,490





Significant improvement in RDD alignment scores of bottom 5% nodes