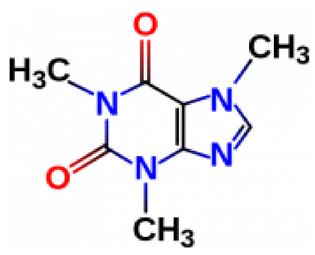
# Metric Indexing for Graph Similarity Search

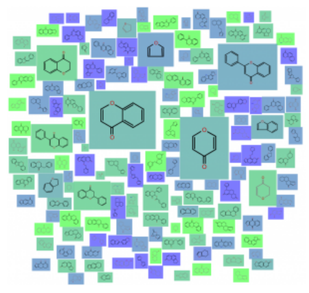
Franka Bause[1], David B. Blumenthal[2], Erich Schubert[3], Nils M. Kriege[1]

[1]University of Vienna [2]FAU Erlangen-Nürnberg [3]TU Dortmund University

## Searching in Large Graph Databases

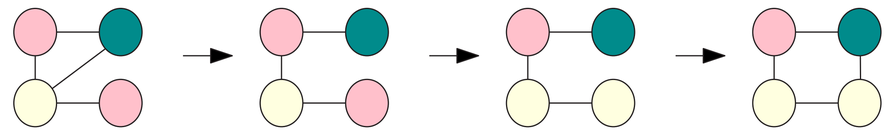
**Goal:** Find graphs similar to a query graph efficiently.



## Graph Edit Distance (GED)

- **Distance measure** for **graphs**
- Transform one graph into the other (**Edit path**)
- Cost of transformation $\hat{=}$ GED

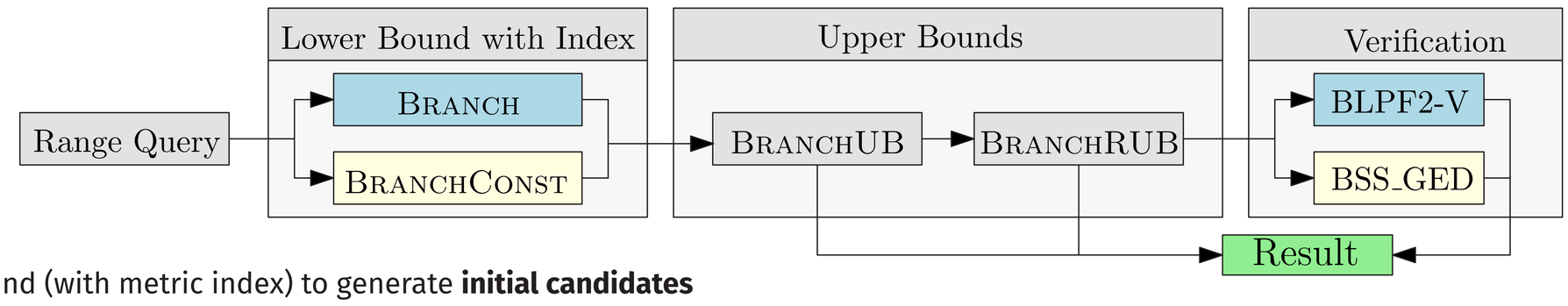


- **Costs** for individual edit operations **not uniform**
- *Assumption:* Edit costs are **metric**

## Our Contribution

- Filter-verification framework for **both** range and $k$-nn search
- Support of **arbitrary** (metric) edit costs
- **Speed-up** through metric index
- Usage of optimal multi-step $k$-nn search algorithm

## Efficient Filtering for the General Graph Edit Distance

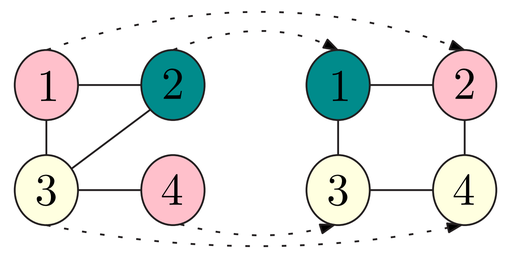



- Use lower bound (with metric index) to generate **initial candidates**
- Use upper bounds to **further filter** candidates
- **Verify remaining** candidates (compute exact GED in worst case)

## BRANCH – Assignment-based Lower Bound

- Compute **optimal assignment** between vertices
- Cost of assignment $\hat{=}$ Lower bound for the GED
- **Metric** if edit costs are metric
- Cost matrix $C$ (Extended with dummy vertices)
- $C_{ij}$: Cost for substituting vertex label
  - ✚ Minimum cost for substituting **edges**
    - $\rightarrow$ Optimal **assignment on edges** with halved edge costs to avoid overestimation
- Runtime: $O(n^2\Delta^3 + n^3)$
  - $n$: number of vertices
  - $\Delta$: maximum degree

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 2 | ∞ | ∞ | ∞ |
| 0 | 1 | 1 | 1 | ∞ | 2 | ∞ | ∞ |
| 1.5 | 1.5 | 0.5 | 0.5 | ∞ | ∞ | 2.5 | ∞ |
| 1.5 | 0.5 | 1.5 | 1.5 | ∞ | ∞ | ∞ | 1.5 |
| 2 | ∞ | ∞ | ∞ | 0 | 0 | 0 | 0 |
| ∞ | 2 | ∞ | ∞ | 0 | 0 | 0 | 0 |
| ∞ | ∞ | 2 | ∞ | 0 | 0 | 0 | 0 |
| ∞ | ∞ | ∞ | 2 | 0 | 0 | 0 | 0 |

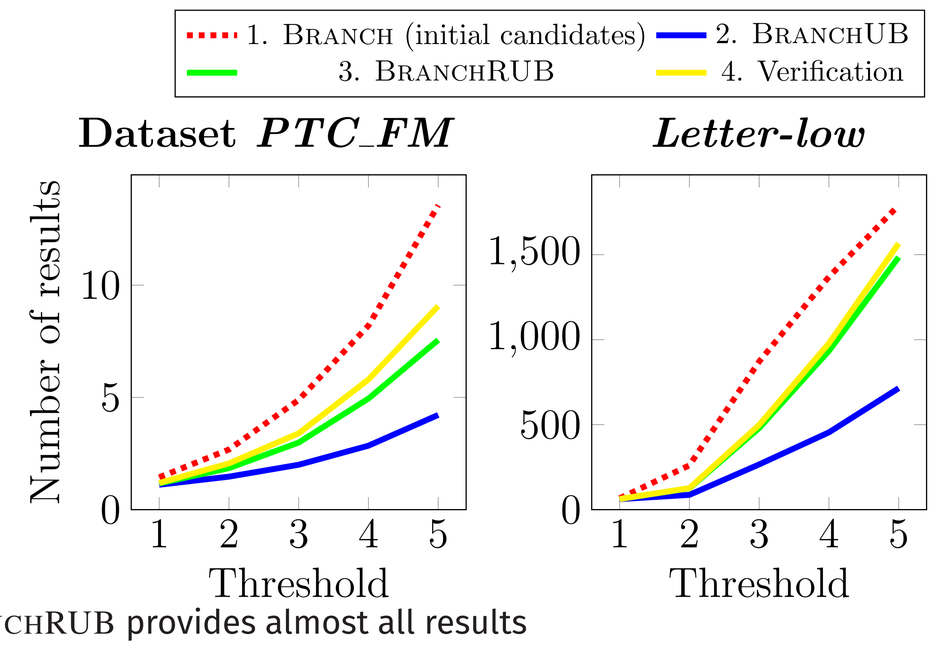



Optimal assignment

## BRANCHUB and BRANCHRUB – Upper Bounds Based on Edit Paths

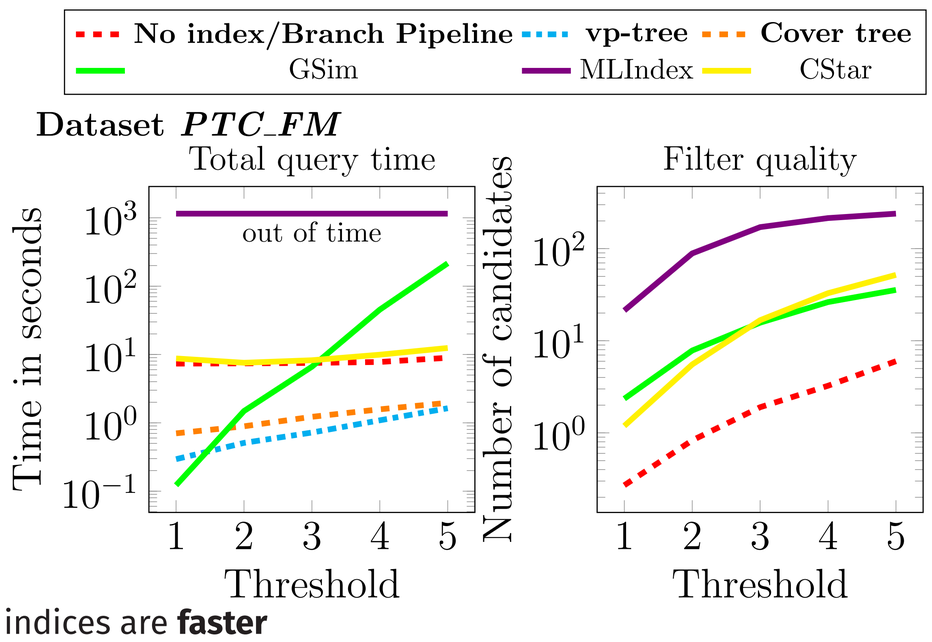
- Derive **sub-optimal edit path** from assignment computed by BRANCH
- **Swap** individual vertex **assignments** to improve edit path

## Comparison of Filter Quality




- BRANCHRUB provides almost all results
- BRANCH might be too loose for larger thresholds

## Comparison to State-of-the-Art




- Metric indices are **faster**
- BRANCH pipeline provides **best** filter quality

## Conclusion and Future Work

**Speed-up** through **metric index**!

**Future Work**

- Embed BRANCH into vectors (for LSH-based search)
- Tighter lower bound needed