MATOMIC

SDU

Edge-Wise Graph Alignment: Theory, Challenges, and Implementation

Akbar Davoodi

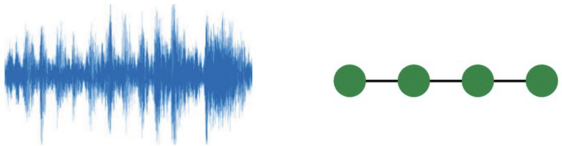
Joint work with Christoph Flamm, Daniel Merkle, and Peter F. Stadler

Department of Mathematics and Computer Science(IMADA)
University of Southern Denmark(SDU)

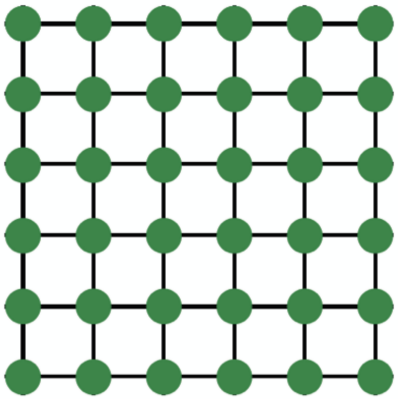
40^{th} TBI Winterseminar, Bled, Slovenia
Feb 09-14, 2025

Graph data structure

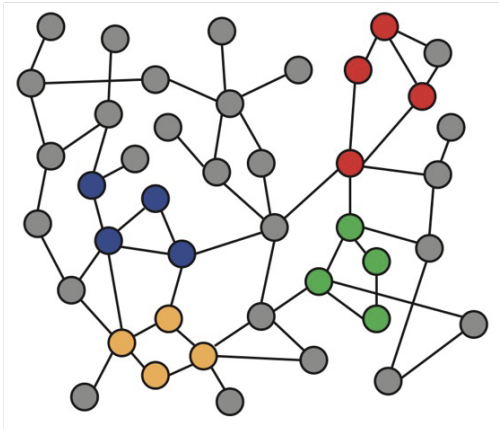
The term“alignment” was introduced in the context of aligning sequences (e.g., DNA, RNA, or protein sequences).



Text/ Speech/ Sequence



Image



Graph

Theory

Definition (Alignment of Spaces)

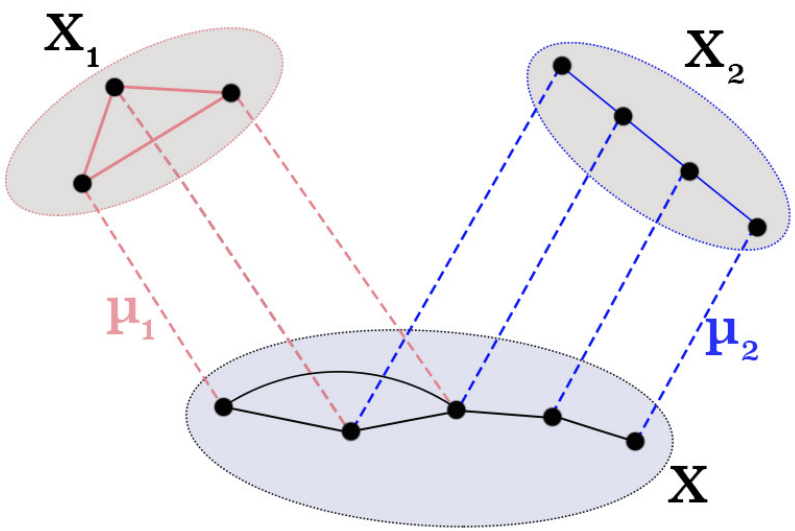
An alignment of spaces $(X_\alpha, \mathscr{S}_\alpha)$, $\alpha \in S$, $|S| \geq 1$ is a space $(X, \mathscr{S})$ such that

(i) there is a monomorphism $\mu_\alpha : X_\alpha \to X$ for every $\alpha \in S$;

(ii) for every $x \in X$, $\mu_\alpha^{-1}(x) \neq \emptyset$ for at least one $\alpha \in S$;

(iii) the restriction of $(X, \mathscr{S})[\mu_\alpha(X_\alpha)]$ is isomorphic to $(X_\alpha, \mathscr{S}_\alpha)$.

Example (alignment of two graphs):



Edge-wise Alignment

Questions

- What exactly do we require from a common substructure?
- Which properties should the common substructure preserve?
  - It should be an induced subgraph.
  - It should maximize the number of edges.
  - It should be connected.
  - ...

Application Context

In several chemoinformatics applications, the alignment must **maximize the number of matched edges**:

- Computing atom maps,
- Rule inference,
- ...

The "E-graph" concept

A simple graph $G = (V, E)$ consists of a set of vertices V and a set of edges $E \subseteq 2^V$, where each edge $e \in E$ connects exactly 2 vertices.

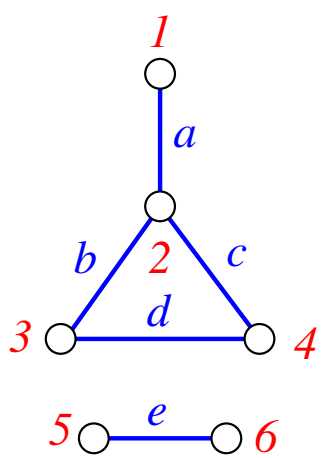
Definition(E-graph)

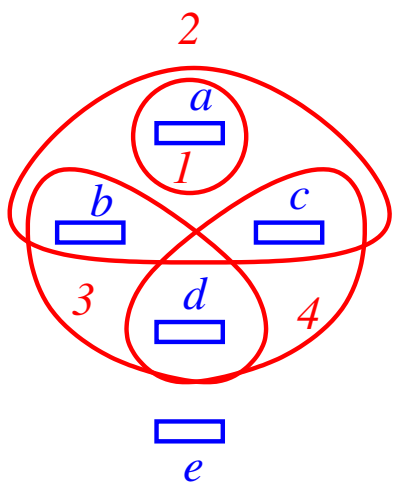
An *E-graph* is a pair (E, S), where $S \subseteq 2^E$, $\emptyset \notin S$,

- $\forall e \in E$, e is contained in exactly 2 or 0 sets in S,
- For all $\Upsilon, \Upsilon' \in S$, the intersection $|\Upsilon \cap \Upsilon'| \leq 1$.

Standard Graph Representation
Our object $= (V, E)$
$V = \{1, 2, 3, 4, 5, 6\}$
$E = \{\{1, 2\}, \{2, 3\}, \{2, 4\}, \{3, 4\}, \{5, 6\}\}$

E-graph Representation
Our object $= (E, S)$
$E = \{a, b, c, d, e\}$
$S = \{\{a\}, \{a, b, c\}, \{b, d\}, \{c, d\}\}$





Algorithm Example

Procedure *Match*($\mathcal{M}, \mathcal{P}$)

1: **if** $\mathcal{P} = \emptyset$ **then**
2: **return** $\mathcal{M}$
3: **end if**
4: **for all** $p \in \mathcal{P}$ **do**
5: **if** $\text{compatible}(\mathcal{M}, p)$ **then**
6: $\mathcal{M} \leftarrow \mathcal{M} \cup \{p\}$
7: Update $\mathcal{P}$
8: **Match**$(\mathcal{M}, \mathcal{P})$
9: **end if**
10: $\mathcal{P} \leftarrow \mathcal{P} \setminus \{p\}$
11: **end for**

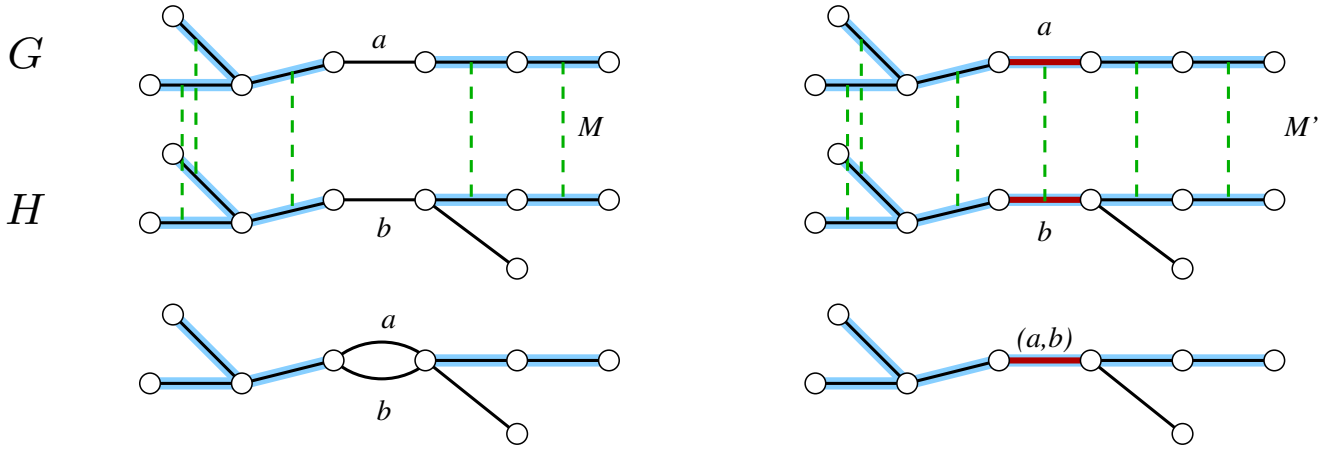
Notes:

- $\mathcal{M}$ represents the current alignment (already matched edges).
- $\mathcal{P}$ is the set of candidate matches.
- The procedure checks the compatibility of each candidate match before adding it.
- The recursive call to *Match* updates the alignment.

Discussion

What challenges might arise with this approach?

Implied Edges



$$\Upsilon_1 = \{(e_1, e_2),\ (a, -),\ (-, b)\} \qquad \Upsilon'_1 = \{(e_1, e_2),\ (a, b)\}$$
$$\Upsilon_2 = \{(f_1, f_2),\ (a, -),\ (-, b)\} \qquad \Upsilon'_2 = \{(f_1, f_2),\ (a, b)\}$$
$$|\Upsilon'_1 \cap \Upsilon'_2| > 1$$

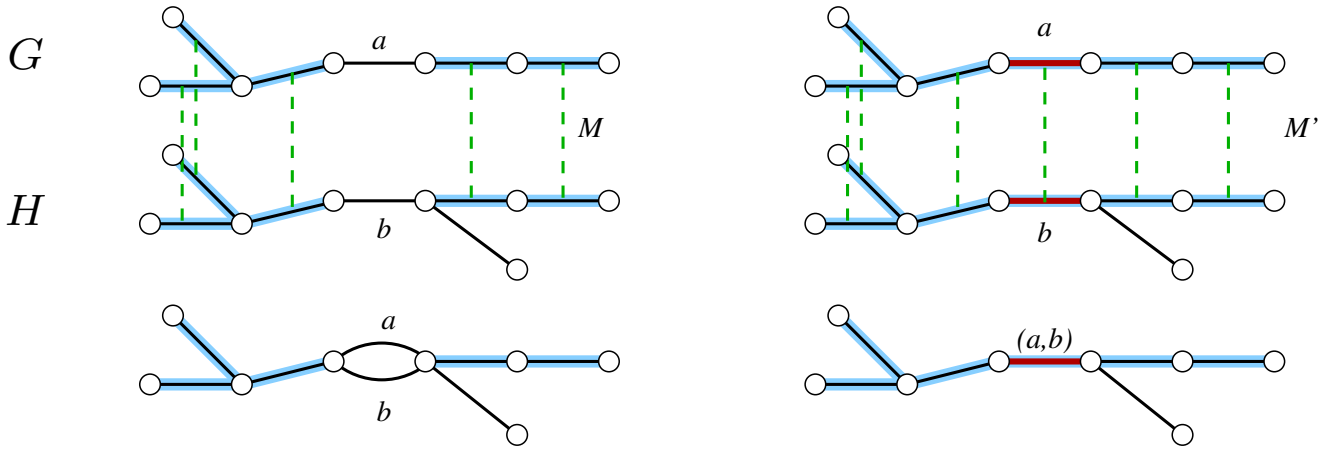
What to Do with Implied Edges?

Detect implied edges during the *Match* process. Whenever such an edge is created, add it immediately to the alignment.

Question

How can we detect implied edges?

Implied Edges



$$\Upsilon_1 = \{(e_1, e_2),\ (a, -),\ (-, b)\} \qquad \Upsilon'_1 = \{(e_1, e_2),\ (a, b)\}$$
$$\Upsilon_2 = \{(f_1, f_2),\ (a, -),\ (-, b)\} \qquad \Upsilon'_2 = \{(f_1, f_2),\ (a, b)\}$$
$$|\Upsilon'_1 \cap \Upsilon'_2| > 1$$

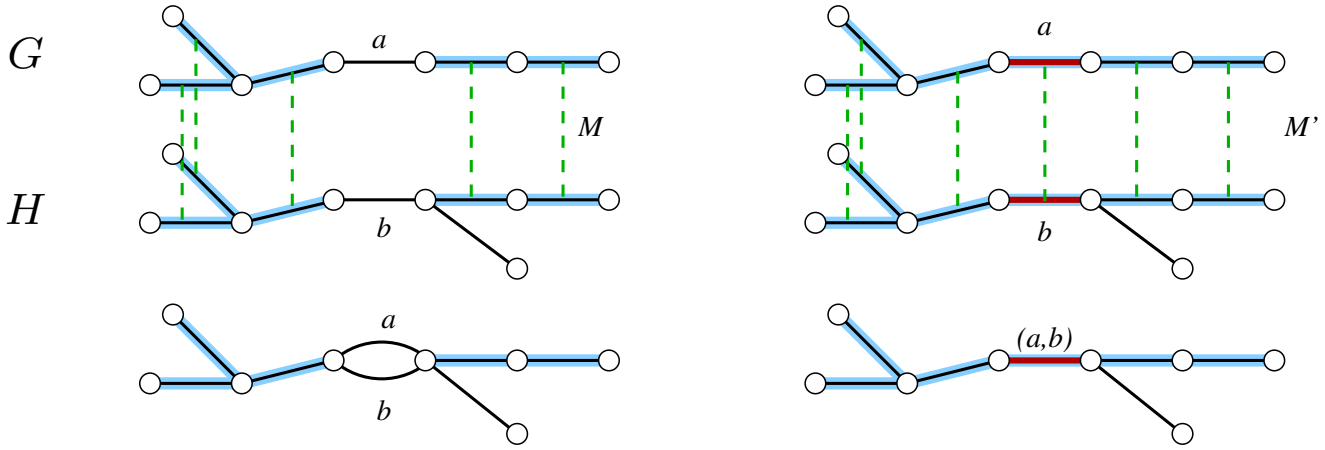
What to Do with Implied Edges?

Detect implied edges during the *Match* process. Whenever such an edge is created, add it immediately to the alignment.

Question

How can we detect implied edges?

Implied Edges



$$\Upsilon_1 = \{(e_1, e_2),\ (a, -),\ (-, b)\} \qquad \Upsilon'_1 = \{(e_1, e_2),\ (a, b)\}$$
$$\Upsilon_2 = \{(f_1, f_2),\ (a, -),\ (-, b)\} \qquad \Upsilon'_2 = \{(f_1, f_2),\ (a, b)\}$$
$$|\Upsilon'_1 \cap \Upsilon'_2| > 1$$

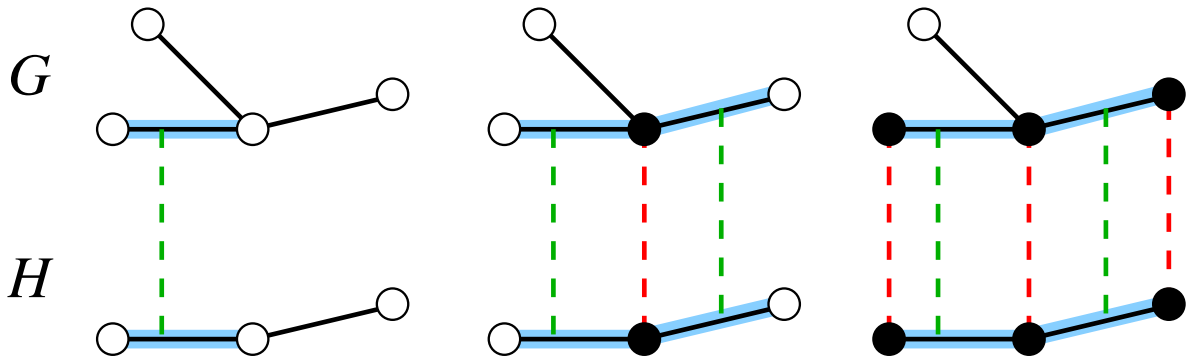
What to Do with Implied Edges?

Detect implied edges during the *Match* process. Whenever such an edge is created, add it immediately to the alignment.

Question

How can we detect implied edges?

How to Detect Implied Edges?



Observations

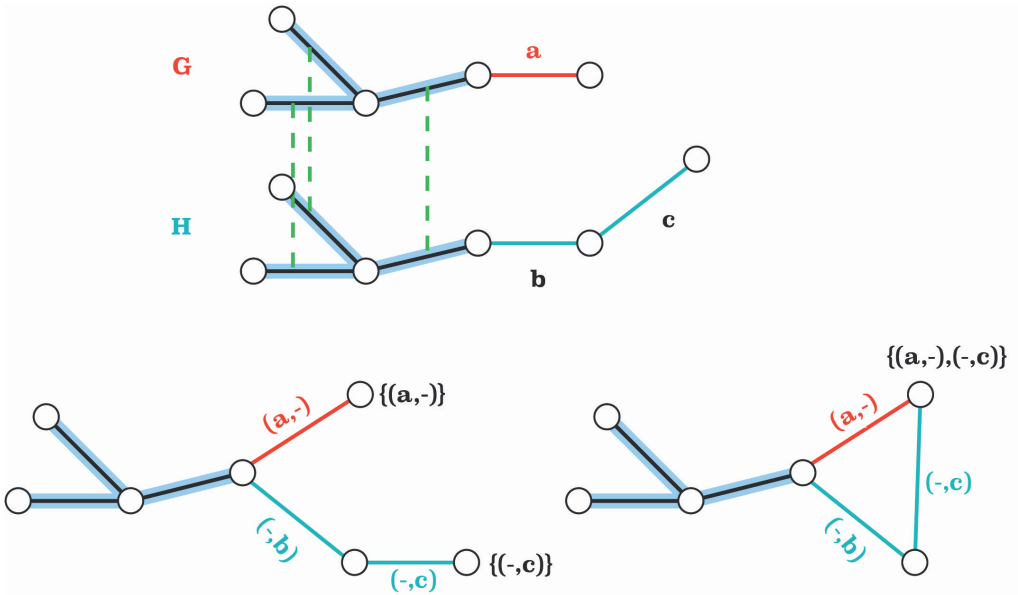
- Whenever two edges e and e' are incident in an E-graph, there exists a unique set Υ that contains both.
- We need to determine whether a newly added edge is isolated.

Solution

Incorporate *set-matches* in addition to standard edge matching during alignment.

Ambiguous Sets

Suppose that the common sub-E-graph of the two E-graphs G and H leaves a in G and b and c in H unmatched.



Depending on whether a and c are considered incident in the alignment, two distinct E-graph alignments can be constructed. In the alignment shown on the right, a and c are grouped together as $\{a, c\}$; in the alignment on the left, they appear as separate sets, $\{a\}$ and $\{c\}$.

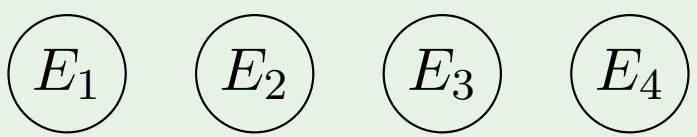
How to align several E-graphs?

$$Align(\overbrace{\underbrace{E_1,\ E_2}_{A},\ E_3}^{Align(A,E_3)}, \ldots, E_t)$$

How to align several E-graphs?

$$Align(\overbrace{\underbrace{E_1,\ E_2}_{A},\ E_3}^{Align(A,E_3)}, \ldots, E_t)$$

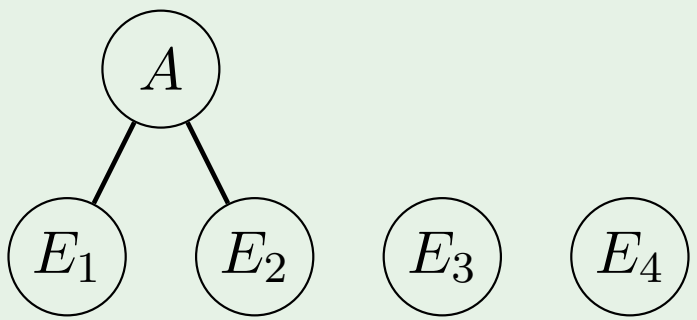
Example (aligning 4 E-graphs progressively):



How to align several E-graphs?

$$Align(\overbrace{\underbrace{E_1,\ E_2}_{A},\ E_3}^{Align(A,E_3)}, \ldots, E_t)$$

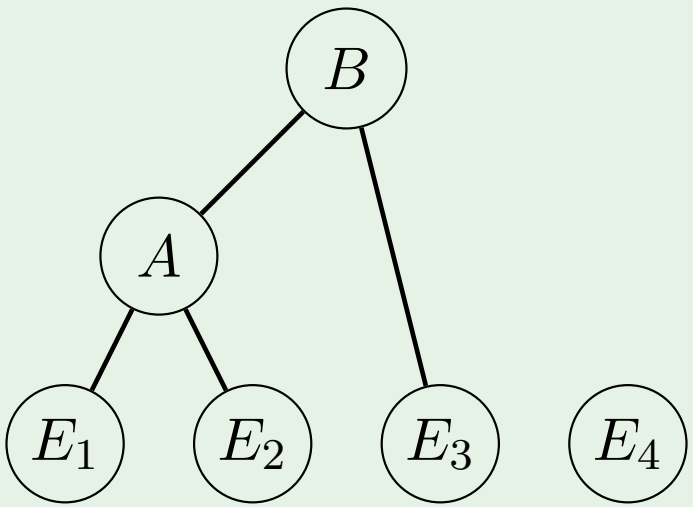
Example (aligning 4 E-graphs progressively):



How to align several E-graphs?

$$Align(\overbrace{\underbrace{E_1,\ E_2}_{A},\ E_3}^{Align(A,E_3)}, \ldots, E_t)$$

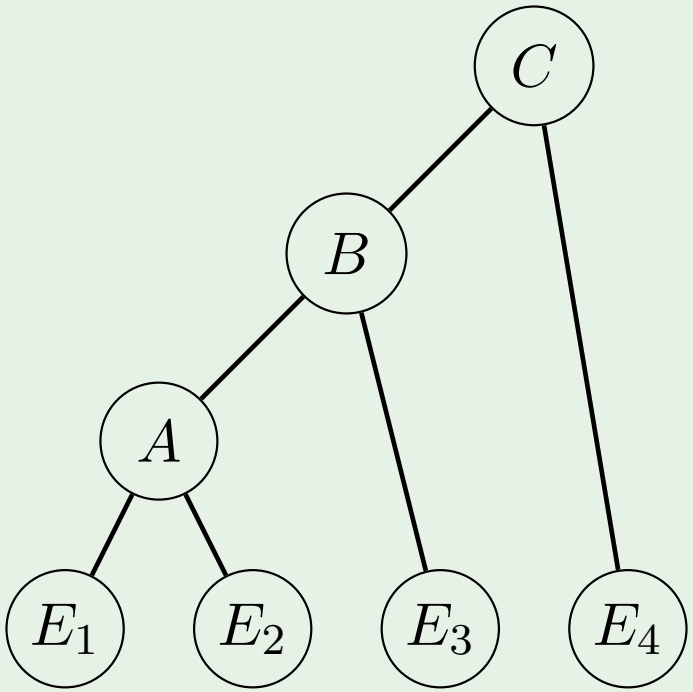
Example (aligning 4 E-graphs progressively):



How to align several E-graphs?

$$Align(\overbrace{\underbrace{E_1,\ E_2}_{A},\ E_3}^{Align(A,E_3)}, \ldots, E_t)$$

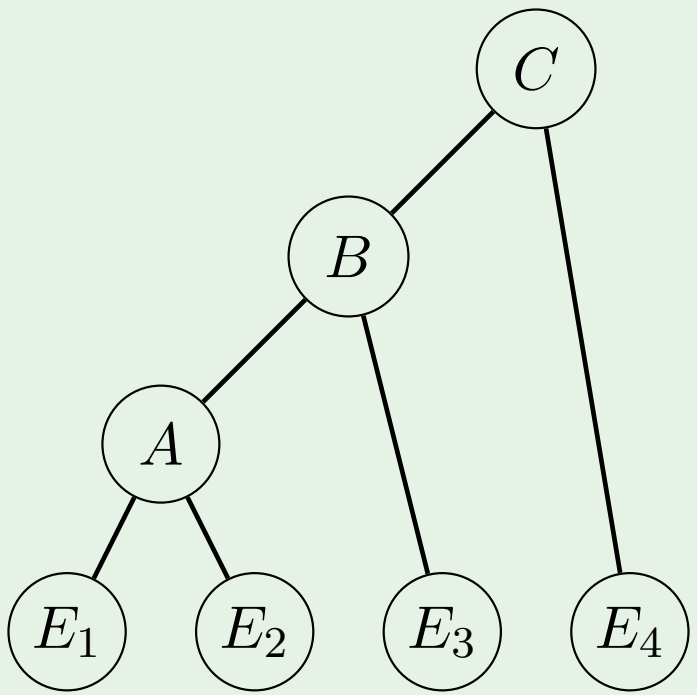
Example (aligning 4 E-graphs progressively):



How to align several E-graphs?

$$Align(\overbrace{\underbrace{E_1,\ E_2}_{A},\ E_3}^{Align(A,E_3)}, \ldots, E_t)$$

Example (aligning 4 E-graphs progressively):



Guide Tree

Guide Tree

The order of input E-graphs can affect the optimal solution in a progressive alignment framework.

- **Strategy 1.** Deterministic guide tree: compute pairwise distances between all E-graphs, then use a hierarchical clustering method (e.g., UPGMA or neighbor joining) to build a tree.
- **Strategy 2.** Online guide tree.
- **Strategy 3.** Two-phase online guide tree: first, select the pair of graphs with the lowest similarity as the initial pair; then, iteratively add the remaining graphs by choosing those with the highest similarity to the current alignment.

We built 10 random graphs on 10 vertices using the Erdős-Rényi model and computed the number of edges in the final alignment. We repeated this process 30 times and averaged the results to compare the three strategies. The results support Strategy 3.

Guide Tree

The order of input E-graphs can affect the optimal solution in a progressive alignment framework.

- **Strategy 1.** Deterministic guide tree: compute pairwise distances between all E-graphs, then use a hierarchical clustering method (e.g., UPGMA or neighbor joining) to build a tree.
- **Strategy 2.** Online guide tree.
- **Strategy 3.** Two-phase online guide tree: first, select the pair of graphs with the lowest similarity as the initial pair; then, iteratively add the remaining graphs by choosing those with the highest similarity to the current alignment.

We built 10 random graphs on 10 vertices using the Erdős-Rényi model and computed the number of edges in the final alignment. We repeated this process 30 times and averaged the results to compare the three strategies. The results support Strategy 3.

Implementation

- We have implemented edge-wise graph alignment, which will soon be publicly available.
- We also provide a comprehensive list of similarity measures for users to choose from, and four strategies are available for selection.
- When dealing with molecules, bond and atom types are defined as attributes, which simplifies the compatibility check between pairs.

Thank you for your attention.

Implementation

- We have implemented edge-wise graph alignment, which will soon be publicly available.
- We also provide a comprehensive list of similarity measures for users to choose from, and four strategies are available for selection.
- When dealing with molecules, bond and atom types are defined as attributes, which simplifies the compatibility check between pairs.

Thank you for your attention.