# Edit Distances for Comparing Merge Trees


**Raghavendra Sridharamurthy, Adhitya Kamakshidasan, Vijay Natarajan**

Visualization and Graphics Lab , Department of Computer Science and Automation,

Indian Institute of Science, Bangalore.

http://vgl.csa.iisc.ac.in ; {raghavendrag, adhitya, vijayn}@iisc.ac.in


## Problem Statement

Design a distance measure to compare merge trees [CSA 2003].

- Prove theoretical guarantees.
- Provide efficient implementation.
- Applications to time-varying data.
- Applications to feature tracking.

## Motivation

***Applications***

- Topological shape matching.
- Symmetry and similarity detection in scalar fields.
- Feature tracking in time-varying data.
- Comparison between simulated and measured data.

***Why merge trees?***

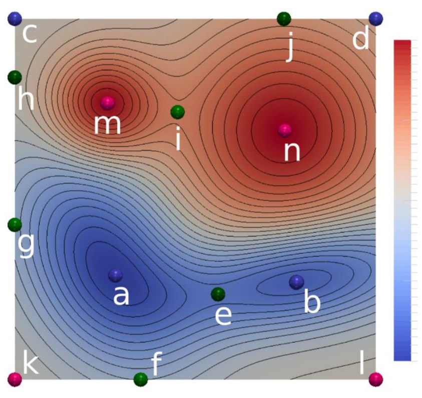
- Features in real data are either at local minima/maxima.
- Simple to implement.
- Easy mapping between regions in the domain and tree components.
- Well defined simplification strategy.

## Challenges

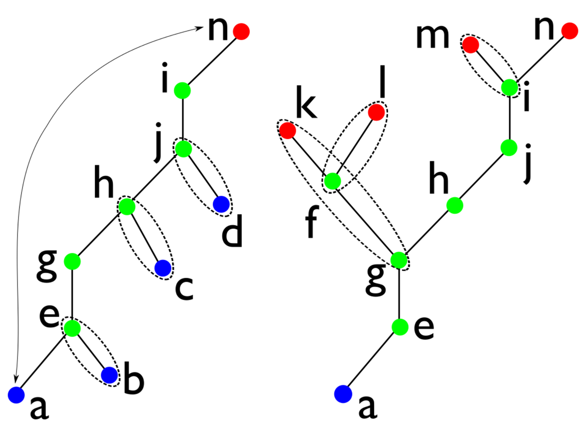
- **Efficiency:** Theoretical vs Practical.
- **Noise:** Small perturbations in the field results in significant changes in the tree structure.
- **Guarantees and Properties:** Hard to prove
  - Metric properties
  - Stability
  - Discrimination

## Background

***Scalar field topology***



Critical points in a 2D scalar field



Merge trees with persistent pairs [ELZ 2000]

***Tree edit distance***

Tree edit distance is inspired by edit distance for strings which results in alignments containing gaps. For example,

*apple*
*a_p_e*
*ap__e*

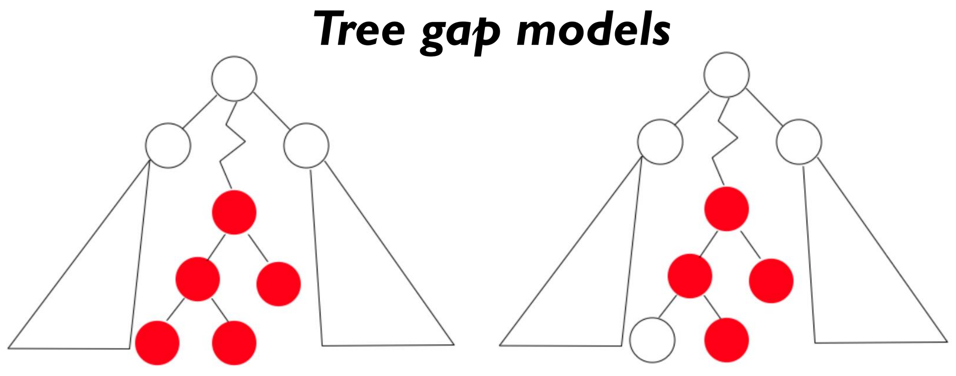
***Edit operations***

1. Relabel
2. Add or delete gap

***Distance***

Distance is given by minimum over all such sets of edit operations.

***Tree gap models***



*Complete subtree gap*     *General subtree gap*

***Observations***

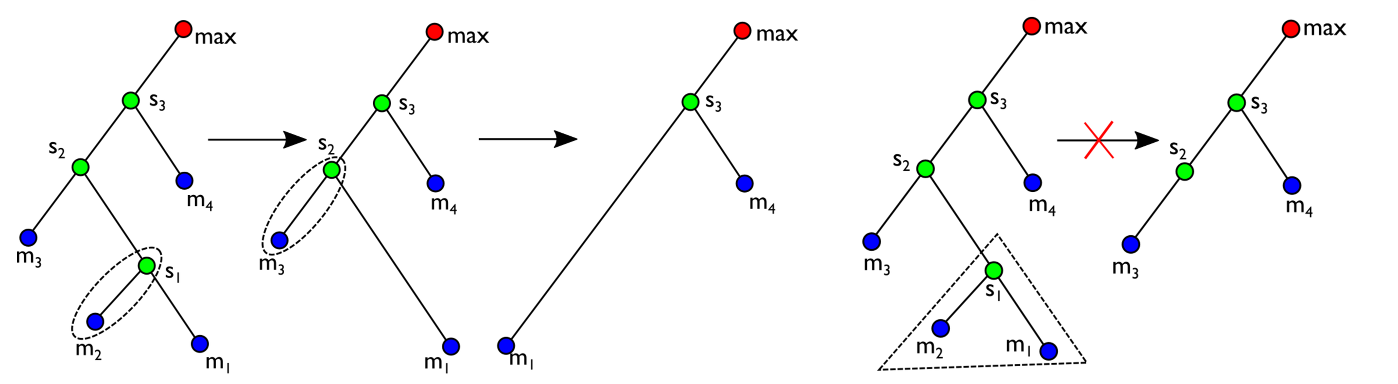
- NP-Hard for arbitrary trees [Touzet 2003]
- A DP based polynomial time algorithm exists for ordered labelled binary trees [Xu 2015].
- For merge trees, labels are function values, ordering is given by pre-order traversal.
- Neither gap model captures all the properties of merge trees.

## Tree Edit Distance based Measure

***Modified gap model***

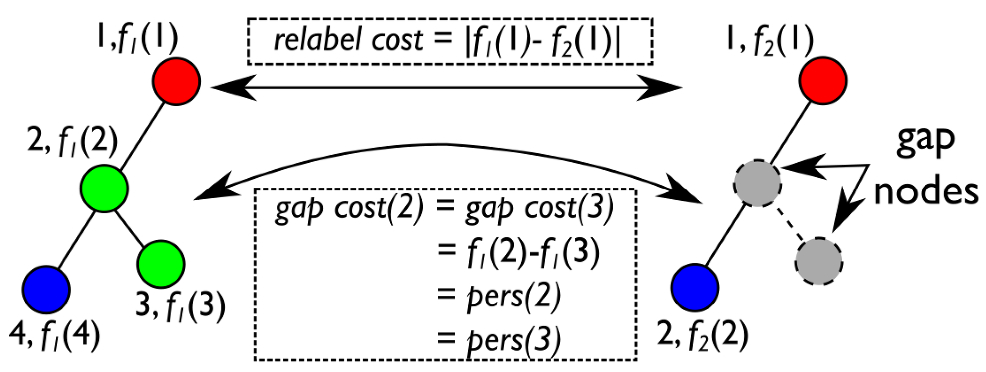
Categorise the set of edit operations using the properties of merge trees.

***I. Permissible set***     ***2. Non-permissible set***



***Modified cost model***

1. ***Relabel cost $r(i,j)$:*** Absolute difference in function values.
2. ***Gap cost $g(i)$:*** Persistence represented by the pairing.




***Measure***

- $R$ is set of all relabels.
- $G$ is set of all gaps.
- Cost $C = \sum_{(i,j)\in R} r(i,j) + \sum_{i\in G} g(i)$
- Distance $D = \min\{C\}$ over all allowed edit operations.

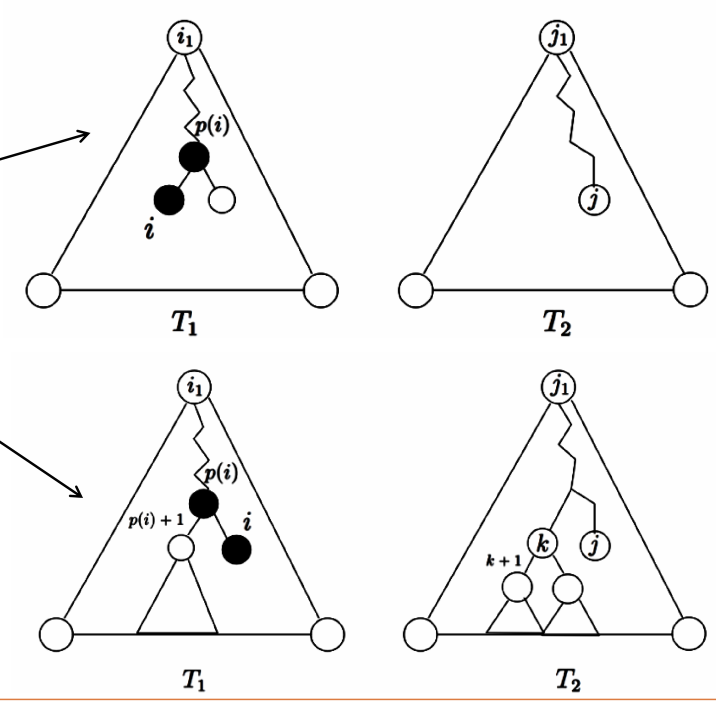
***DP algorithm and recurrences***

- $1, \dots, i', \dots, i, \dots, m$ pre-order of merge tree $T_1$ for function $f_1$
- $1, \dots, j', \dots, j, \dots, n$ pre-order of merge tree $T_2$ for function $f_2$
- Parent of the node $i$ is denoted by $p(i)$
- Case $D_2$ is similar to $D_1$
- Distance $D$ is given by [Xu 2015]

$$D[i'..i, j'..j] = min \begin{cases} D[i'..i-1, j'..j-1] + r(i,j), & \textit{relabel } i \textit{ to } j \\ D_1[i'..i, j'..j], & i \textit{ is mapped to a gap node} \\ D_2[i'..i, j'..j], & j \textit{ is mapped to a gap node} \end{cases}$$

$$D_1[i'..i, j'..j] = min \begin{cases} D[i'..i-1, j'..j] + g(i), \\ D_1[i'..i-1, j'..j] + g(i), \\ min_{j_1 \le k \le j}\{D_1[i'..p(i), j'..k] + \\ D[p(i)+1..i-1, k+1..j] + g(i)\} \end{cases}$$

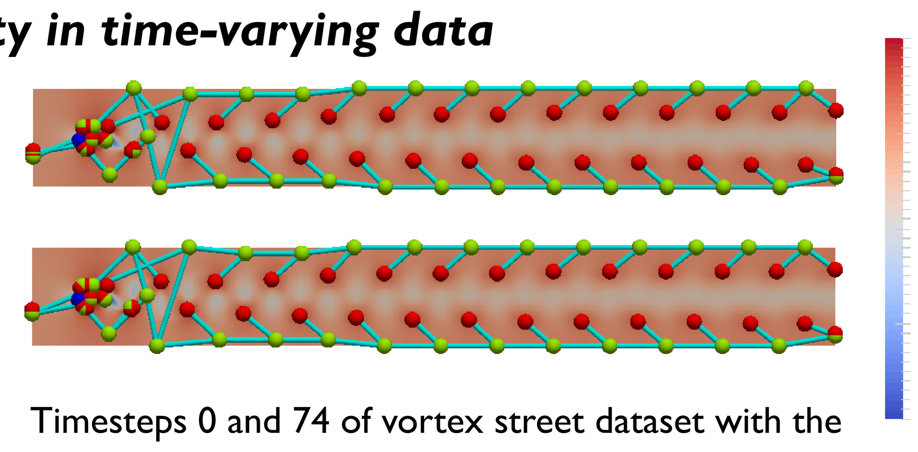
- Dynamic programming to compute $D[1..m, 1..n]$
- Choose canonical ordering of children.
- Running time $O(m^3n^2 + m^2n^3)$.



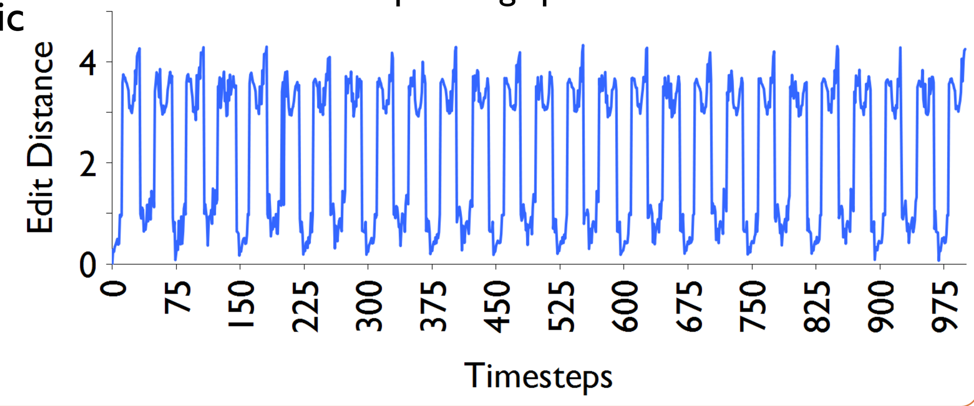
## Results and Future Work

***Periodicity in time-varying data***

- ***Data:*** Bénard von Kármán vortex street, 2D flow around a cylinder; $[400 \times 50]$, 1001 timesteps; Source: Weinkauf [2010].
- ***Features:*** Local maxima capture the vortex centres.
- ***Experiment:*** Study periodic vortex shedding, with known periodicity of 75.
- ***Key result:*** We use our distance measure $D$ and identify periodicity of 74-75.

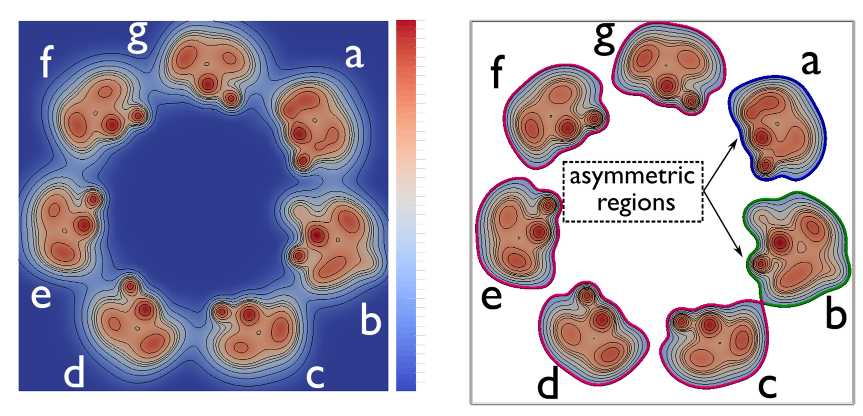


Timesteps 0 and 74 of vortex street dataset with the corresponding split trees.




***Detecting symmetry/asymmetry***

- ***Data:*** Synthetic data, contains both regions of symmetry and asymmetry.
- ***Features:*** Merge trees of the regions $(a, \dots, g)$.
- ***Experiment:*** Find whether $D$ is effective in capturing the symmetry/asymmetry.
- ***Result:*** $D \approx 0$ for symmetric regions (for example $D(c,d)$) , $D > 0$ in other cases (for example $D(c,b) = 0.53$) which is consistent with the premise of data synthesis.




***Future Work***

- Prove theoretical properties/guarantees.
- Introduce spatial overlap to enhance discrimination.
- Improve the efficiency, both in theory and in practice.


***Acknowledgements***

We thank the members of VGL for suggestions and active discussions. We acknowledge support from the Department of Science and Technology, India (DST/SJF/ETA-02/2015-16) and from the Joint Advanced Technology Programme, Indian Institute of Science (JATP/RG/PROJ/2015/16)



- [CSA 2003] H. Carr, J. Snoeyink, and U. Axen, "Computing contour trees in all dimensions," Computational Geometry, vol. 24, no. 2, pp. 75–94, 2003.
- [Xu 2015] H. Xu, "An algorithm for comparing similarity between two trees," arXiv preprint arXiv:1508.03381, 2015.
- [ELZ 2000] H. Edelsbrunner, D. Letscher, and A. Zomorodian. "Topological persistence and simplification". In Procs. 41st Annual Symp. on FOCS, pages 454–463., 2000.
- [Touzet 2003] H. Touzet. "Tree edit distance with gaps". Information Processing Letters 85(3):123–129, 2003.