The University of Texas at Austin
Computer Science

Bi-Lipschitz extensions and outlier embeddings into trees

Shuchi Chawla, Arnold Filtser*, **Kristin Sheridan**, Yonatan Trachtenberg*
*Department of Computer Science, Bar-Ilan University

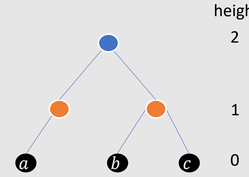
Problems and Definitions

Metric embeddings

A (probabilistic) **metric embedding** of (X, d) into (Y, d_Y) is a random map $\alpha: X \to Y$ We write $d_\alpha(x, y) \coloneqq d_Y(\alpha(x), \alpha(y))$, and we say α has distortion c if:

- $\forall x, y \in X$: $d_\alpha(x, y) \geq d(x, y)$ (for all possible α)
- $\forall x, y \in X$: $E_\alpha[d_\alpha(x, y)] \leq c \cdot d(x, y)$

For our purposes, Y will be HSTs



$d(a, b) = d(a, c) = 2^2 = 4$
$d(b, c) = 2^1 = 2$

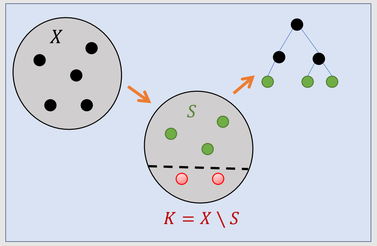
Hierarchically Separated Trees (HSTs)

- The distance between two distinct leaf nodes is 2 raised to the height of their least common ancestor
- [FRT03]: All n-point metrics embed into HSTs with $O(\log n)$ expected distortion

(Note: We use HSTs to refer to 2-HSTs.)

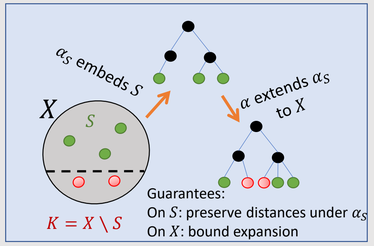
Outlier embeddings

Given a metric (X, d), how many points do I need to remove from X so the metric embeds into HSTs with c distortion?



Bi-Lipschitz extensions

Given an embedding of a subset $S \subseteq X$ into HSTs, can you "extend" the embedding to the rest of X without too much distortion?



A (k, c)-outlier embedding is a distortion-c embedding on all but at most k points of a metric

Outlier embedding approximations

source	Destination metric	Number of outliers (k')	Distortion (c')	Extra conditions
SWW15	ℓ_2^d for constant d, ultrametrics, trees	$O(k)$	c	Requires $c = 1$
CS20	The real line	$O(c^6 k \log^{\frac{5}{2}} n)$	$O(c^{13})$	Input is an unweighted graph metric
CS24	ℓ_2^n	$O\left(\frac{k \log^2 k}{\epsilon}\right)$	$(1+\epsilon)c$	
This work	**HSTs**	$\mathbf{O\left(\frac{k \log k}{\epsilon}\right)}$	$\mathbf{(32+\epsilon)c}$	**Probabilistic embedding**

References:
[CKR01] Gruia Calinescu, Howard Karloff, and Yuval Rabani. Approximation Algorithms for the 0-Extension Problem. SODA 2001.
[CS20] Karine Chubarian and Anastasios Sidiropoulos. Computing Bi-Lipschitz Outlier Embeddings into the Line. APPROX/RANDOM 2020.
[CS24] Shuchi Chawla and Kristin Sheridan. Composition of nested embeddings with an application to outlier embeddings. SODA 2024.
[FRT03] Jatin Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. STOC 2003.
[MST23] Kamesh Munagala, Govind S. Sankar, and Erin Taylor. Probabilistic Metric Embedding via Metric Labeling. APPROX/RANDOM 2023.
[SWW15] Anastasios Sidiropoulos, Dingkang Wang, and Yusu Wang. Metric embeddings with outliers. SODA 2015
Bowl/whisk icon: adapted from MScharwies, Wikimedia Commons, CC BY-SA 4.0.

Paper:
arXiv:2601.15470
APPROX 2026

Funding in part from:
and JPMorgan Chase & Co.

This work was funded in part by JPMorgan Chase & Co. Any views or opinions expressed herein are solely those of the authors listed, and may differ from the views and opinions expressed by JPMorgan Chase & Co. or its affiliates. This material is not a product of the Research Department of J.P. Morgan Securities LLC. This material should not be construed as an individual recommendation for any particular client and is not intended as a recommendation of particular securities, financial instruments or strategies for a particular client. This material does not constitute a solicitation or offer in any jurisdiction.

Results

Finding bi-Lipschitz extensions

Given $\alpha_S: S \to$HSTs, we can efficiently sample from a probabilistic extension $\alpha: K \cup S \to$HSTs with at most a multiplicative $\boldsymbol{O(\log|K|)}$ **increase in distortion** compared to α_S

Implied using a modified version of the [MST23] LP

Approximating outlier embeddings

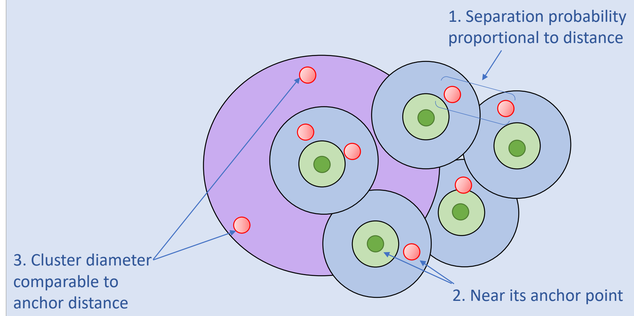
Given (X, d), we can efficiently:

- Correctly decide that (X, d) does not have a (k, c)-outlier embedding into HSTs, or
- Find an $\left(\boldsymbol{O\left(\frac{k}{\epsilon}\log k\right)}, \boldsymbol{(32+\epsilon)c}\right)$-outlier embedding

Onion partition

An **onion partition** of $K \subseteq X$ is a probabilistic partition $K_1, K_2, \dots, K_l$ of K with assigned anchor points $a_1, a_2, \dots, a_l \in X \setminus K$ such that:

1. The **probability that $x, y \in K$ are separated is small**, at most $O(\log k) \cdot \frac{d(x,y)}{d(\{x,y\}, X \setminus K)}$
2. The **anchor point a of x's partition is close to x**: $d(x, a) \leq 2d(x, X \setminus K)$, and
3. The **diameter of each partition is small** relative to its distance from its anchor point: for any $x \in K_i$: $d(x, a_i) \geq \Delta_{K_i}/6$, where Δ_{K_i} is the diameter of K_i



We can construct onion partitions for any metric (X, d) and $K \subseteq X$!

- [CKR01] partitions K and ensures properties (1) and (2)
- We can view [CKR01] as drawing balls around each point in $X \setminus K$ and assigning $x \in K$ to the first ball it falls in
- To ensure property (3), we divide each ball from [CKR01] into a set of nested balls, setting the radius of the ith ball to 2^i
- How to prevent the probability of separation from getting too large? Use randomness: pick a random $u \in [0,1]$ once and change the radius of each ith ball to 2^{i+u}

Bi-Lipschitz extension recipe

Ingredients:



- "Onion partition"
- Bounded diameter embedding
- Perfect merge function

Directions:

1. Given your metric (X, d) and embedding $\alpha_S: X \setminus K \to$ HSTs, partition K into sets $K_1, K_2, \dots, K_l$ with the **onion partition**.
2. Find a **bounded diameter embedding $\boldsymbol{\alpha_i}$** on each $K_i \cup \{a_i\}$, where a_i is the anchor point of K_i
3. Use the **perfect merge function** to combine each α_i with the initial embedding α_S

Key ideas:

- The bounded diameter embedding bounds distortion between points in the same cluster
- By the perfect merge function, if x and y are separated and assigned anchor points a_i, a_j, their distance is at least a constant factor times $d_\alpha(x, a_i) + d_\alpha(y, a_j)$
- If x and y are separated, their distance is at most the length of the $x - a_i - a_j - y$ path under α, and all distances on the path are bounded by the bounded diameter embedding
- The onion partition ensures the probability x and y are separated is proportional to $d(x, y)$

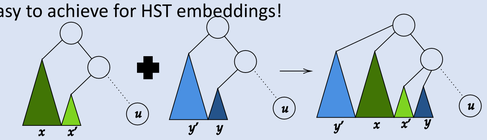
Perfect merge function

A perfect merge function "merges" two embeddings with one common point in their domains, **ensuring no distance is too small.**

If $\alpha_1: X_1 \to Y$ and $\alpha_2: X_2 \to Y$ with $X_1 \cap X_2 = \{u\}$, a perfect merge function uses them to produce $\alpha: X_1 \cup X_2 \to Y$ such that:

- $d_\alpha(x, y) = d_{\alpha_1}(x, y)$ for $x, y \in X_1$
- $d_\alpha(x, y) = d_{\alpha_2}(x, y)$ for $x, y \in X_2$
- $d_\alpha(x, y) \geq \frac{d_{\alpha_1}(x,u) + d_{\alpha_2}(u,y)}{2}$ for $x \in X_1, y \in X_2$

This is easy to achieve for HST embeddings!



Bounded diameter embedding

A **bounded diameter embedding** $\alpha: X \to Y$ has parameters c, c_Δ if:

- It is a distortion c embedding, and
- The diameter of the embedded points is at most c_Δ times the diameter of X

For HST embeddings, we can get $c = O(\log n), c_\Delta = O(1)$! [FRT03]