Ultrametrics Meet Fine-Grained Complexity

Karthik C. S.
(Tel Aviv University)

Joint work with

Vincent Cohen-Addad
(Google)

Guillaume Lagarde
(LaBRI)

Ultrametrics

- (Γ, Δ) is a metric space
    - $\Delta(a,b) \leqslant \Delta(a,c) + \Delta(b,c)$

Ultrametrics

- (Γ, Δ) is a metric space
  - $\Delta(a,b) \leqslant \Delta(a,c) + \Delta(b,c)$
- Ultrametric: $\forall a, b, c \in \Gamma$,

$$\Delta(a,b) \leqslant \max\{\Delta(a,c), \Delta(b,c)\}$$

Ultrametrics

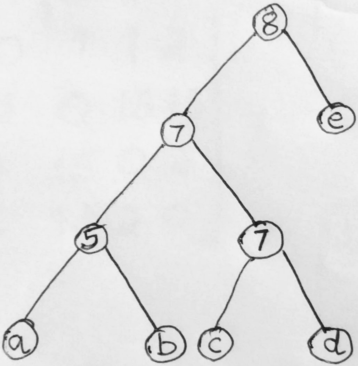
- (Γ, Δ) is a metric space
  - $\Delta(a,b) \leqslant \Delta(a,c) + \Delta(b,c)$
- Ultrametric: $\forall a, b, c \in \Gamma$,

$$\Delta(a,b) \leqslant \max\{\Delta(a,c), \Delta(b,c)\}$$

- Cool Property: $\forall a, b, c \in \Gamma$,

$$\Delta(a,b) = \Delta(a,c) \text{ or } \Delta(a,c) = \Delta(b,c) \text{ or } \Delta(a,b) = \Delta(b,c)$$

Example

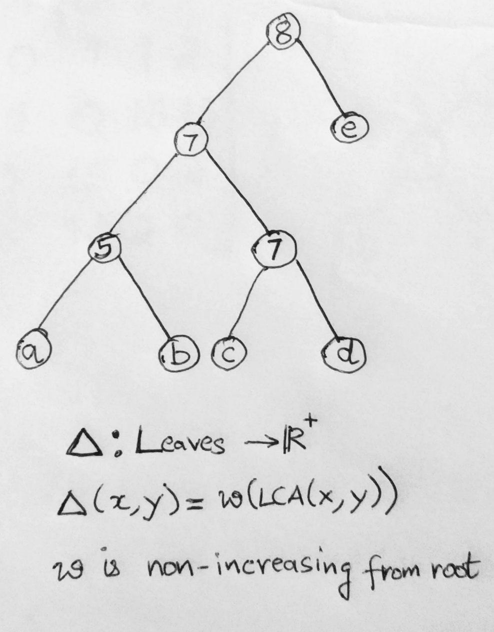


Δ: Leaves $\to \mathbb{R}^+$

$\Delta(x,y) = w(LCA(x,y))$

w is non-increasing from root

Example



Arises in:

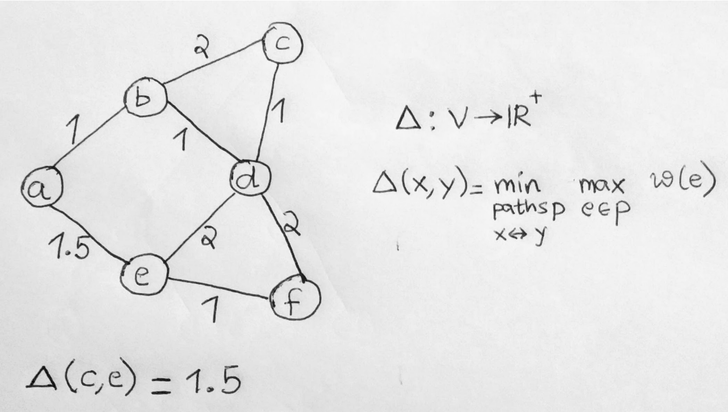
- Evolutionary Biology
- Hierarchical Clustering

Example

- Topology: Discrete metric
- Number Theory: p-adic numbers
- Graph Theory: Minmax paths

Example

- **Topology**: Discrete metric
- **Number Theory**: p-adic numbers
- **Graph Theory**: Minmax paths



Directions

- Focus on Embedding

Directions

- Focus on Embedding
- Embedding from Ultrametric

Directions

- Focus on Embedding
- Embedding from Ultrametric
  - Not today

Directions

- Focus on Embedding
- Embedding from Ultrametric
  - Not today
- Embedding to Ultrametric

Directions

- Focus on Embedding
- Embedding from Ultrametric
  - Not today
- Embedding to Ultrametric

$$\tau : X \longrightarrow L, \ \ \forall x, y \in X,$$

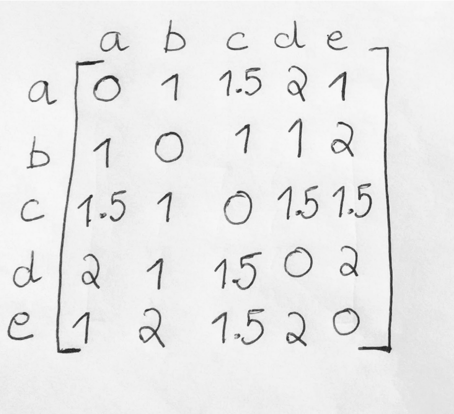
$$\|x - y\|_p \leq \Delta(\tau(x), \tau(y)) = w(\mathsf{LCA}(\tau(x), \tau(y))) \leq \rho_{\mathsf{OPT}} \cdot \|x - y\|_p$$

Motivation: Data Visualization

$$\{a, b, c, d, e\} \subseteq \mathbb{R}^{100}$$

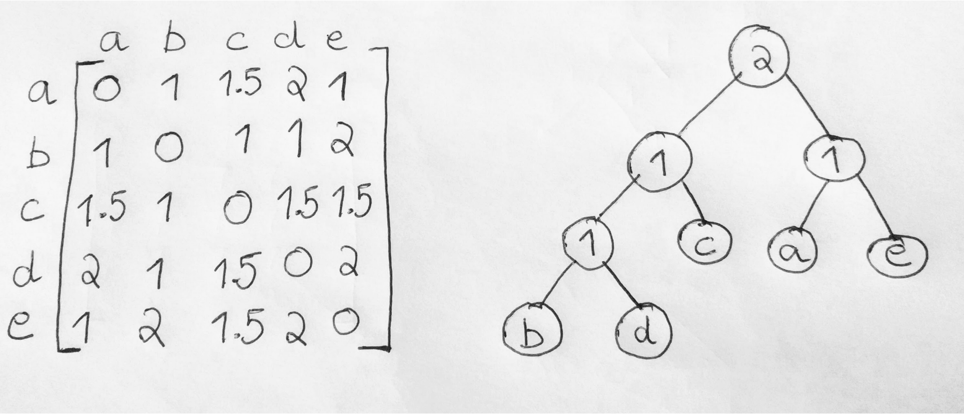
Motivation: Data Visualization

$$\{a, b, c, d, e\} \subseteq \mathbb{R}^{100}$$



Motivation: Data Visualization

$$\{a, b, c, d, e\} \subseteq \mathbb{R}^{100}$$



Results

Theorem (Farach–Kannan–Warnow'95)

Given the distance matrix of n points, the optimal ultrametric embedding can be computed in time $O(n^2)$.

Results

Theorem (Farach–Kannan–Warnow'95)

Given the distance matrix of n points, the optimal ultrametric embedding can be computed in time $O(n^2)$.

Theorem (Cohen-Addad–K–Lagarde)

- Assuming SETH, no 1.5 approximate embedding in $n^{1.99}$ time from ℓ_∞-metric.

Results

Theorem (Farach–Kannan–Warnow'95)

Given the distance matrix of n points, the optimal ultrametric embedding can be computed in time $O(n^2)$.

Theorem (Cohen-Addad–K–Lagarde)

- Assuming SETH, no 1.5 approximate embedding in $n^{1.99}$ time from ℓ_∞-metric.
- Assuming non-standard hypothesis, no 1.001 approximate in $n^{1+o(1)}$ time from Euclidean metric.

Results

Theorem (Farach–Kannan–Warnow'95)

Given the distance matrix of n points, the optimal ultrametric embedding can be computed in time $O(n^2)$.

Theorem (Cohen-Addad–K–Lagarde)

- Assuming SETH, no 1.5 approximate embedding in $n^{1.99}$ time from ℓ_∞-metric.
- Assuming non-standard hypothesis, no 1.001 approximate in $n^{1+o(1)}$ time from Euclidean metric.
- For any $\gamma \geq 1$, 5γ approximate embedding in time $O(n^{1+\frac{1}{\gamma^2}})$ for Euclidean metric.

Results

Theorem (Farach–Kannan–Warnow'95)

Given the distance matrix of n points, the optimal ultrametric embedding can be computed in time $O(n^2)$.

Theorem (Cohen-Addad–K–Lagarde)

- Assuming SETH, no 1.5 approximate embedding in $n^{1.99}$ time from ℓ_∞-metric.
- Assuming non-standard hypothesis, no 1.001 approximate in $n^{1+o(1)}$ time from Euclidean metric.
- For any $\gamma \geq 1$, 5γ approximate embedding in time $O(n^{1+\frac{1}{\gamma^2}})$ for Euclidean metric.

Performs Well in Experiments!

Farach–Kannan–Warnow'95: Algorithm

Input: An edge-weighted clique G

Output: An ultrametric tree T^{ULT}

Farach–Kannan–Warnow'95: Algorithm

Input: An edge-weighted clique G

Output: An ultrametric tree T^{ULT}

1. Compute Minimum Spanning Tree T^G

Farach–Kannan–Warnow'95: Algorithm

Input: An edge-weighted clique G

Output: An ultrametric tree T^{ULT}

1. Compute Minimum Spanning Tree T^G

2. Compute cut weights of T^G, i.e., $\forall e \in E(T^G)$:

$$P(e) = \{(i,j) \in V \times V \mid e \in \mathrm{Path}_{T^G}(i,j),\ \Delta_{\max}(i,j) = w(e)\}$$

$$C(e) = \max_{(i,j) \in P(e)} \|v_i - v_j\|_p$$

Farach–Kannan–Warnow'95: Algorithm

Input: An edge-weighted clique G

Output: An ultrametric tree T^{ULT}

1. Compute Minimum Spanning Tree T^G

2. Compute cut weights of T^G, i.e., $\forall e \in E(T^G)$:

$$P(e) = \{(i,j) \in V \times V \mid e \in \mathrm{Path}_{T^G}(i,j),\ \Delta_{\max}(i,j) = w(e)\}$$

$$C(e) = \max_{(i,j) \in P(e)} \|v_i - v_j\|_p$$

3. Build ultrametric tree:

Farach–Kannan–Warnow'95: Algorithm

Input: An edge-weighted clique G

Output: An ultrametric tree T^{ULT}

1. Compute Minimum Spanning Tree T^G
2. Compute cut weights of T^G, i.e., $\forall e \in E(T^G)$:

$$P(e) = \{(i,j) \in V \times V \mid e \in \mathrm{Path}_{T^G}(i,j),\ \Delta_{\max}(i,j) = w(e)\}$$

$$C(e) = \max_{(i,j) \in P(e)} \|v_i - v_j\|_p$$

3. Build ultrametric tree:
   - Leaves are V
   - Root is $e \in E(T^G)$ of max weight

Farach–Kannan–Warnow'95: Algorithm

Input: An edge-weighted clique G

Output: An ultrametric tree T^{ULT}

1. Compute Minimum Spanning Tree T^G
2. Compute cut weights of T^G, i.e., $\forall e \in E(T^G)$:

$$P(e) = \{(i,j) \in V \times V \mid e \in \mathrm{Path}_{T^G}(i,j),\ \Delta_{\max}(i,j) = w(e)\}$$

$$C(e) = \max_{(i,j) \in P(e)} \|v_i - v_j\|_p$$

3. Build ultrametric tree:
   - Leaves are V
   - Root is $e \in E(T^G)$ of max weight
   - Recursively build both children components of root

Farach–Kannan–Warnow'95: Algorithm

Input: An edge-weighted clique G

Output: An ultrametric tree T^{ULT}

1. Compute Minimum Spanning Tree T^G

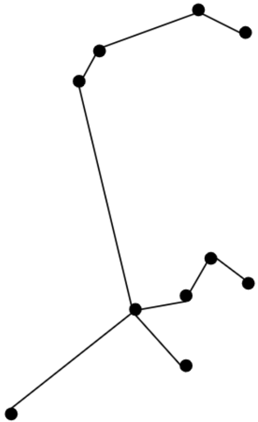
2. Compute cut weights of T^G, i.e., $\forall e \in E(T^G)$:

$$P(e) = \{(i,j) \in V \times V \mid e \in \mathrm{Path}_{T^G}(i,j),\ \Delta_{\max}(i,j) = w(e)\}$$

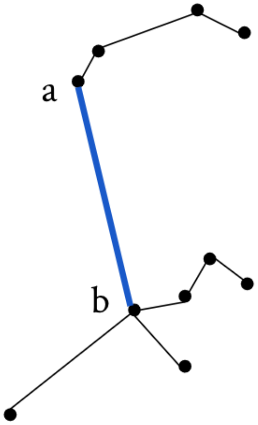
$$C(e) = \max_{(i,j) \in P(e)} \|v_i - v_j\|_p$$

3. Build ultrametric tree:
   - Leaves are V
   - Root is $e \in E(T^G)$ of max weight
   - Recursively build both children components of root
   - Weight of internal node e is $CW(e)$

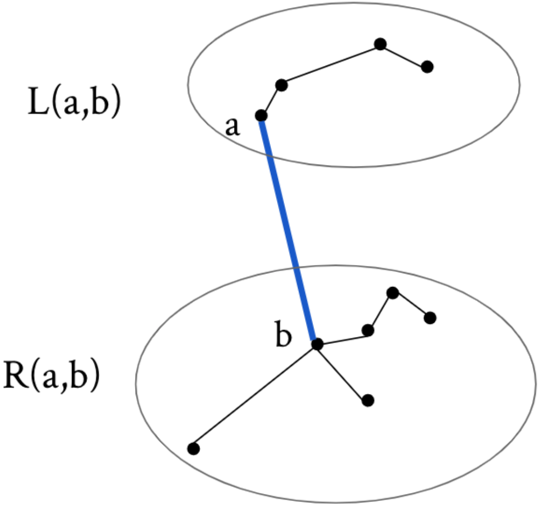
Cut weights: Illustration



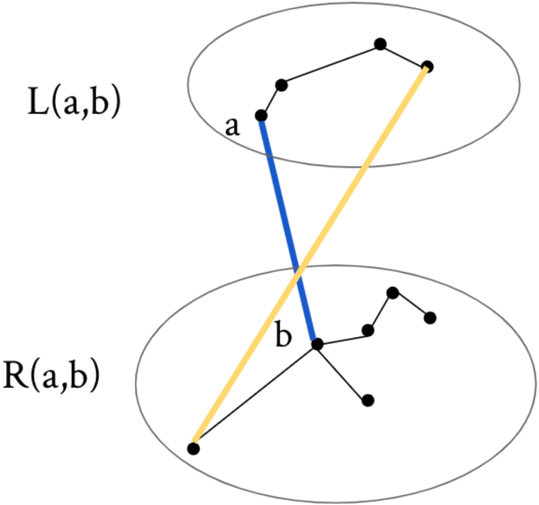
Cut weights: Illustration



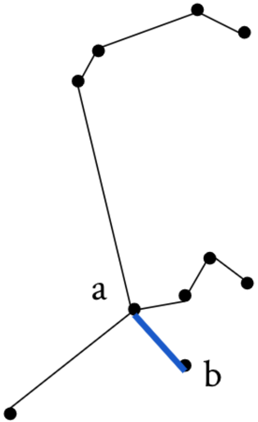
Cut weights: Illustration



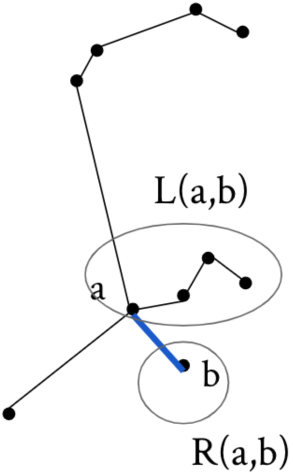
Cut weights: Illustration



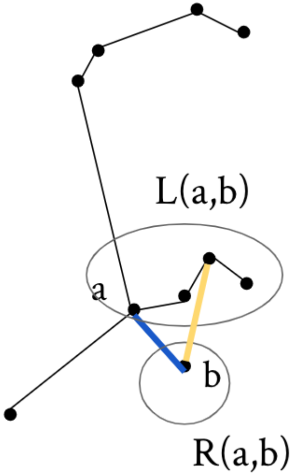
Cut weights: Illustration



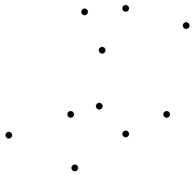
Cut weights: Illustration



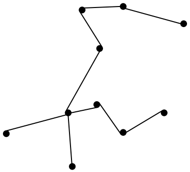
Cut weights: Illustration



Our Approximation Algorithm

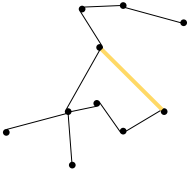


Our Approximation Algorithm



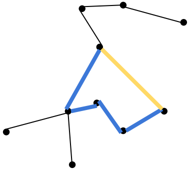
1. Compute a γ-approximate MST T^G over the complete graph G

Our Approximation Algorithm



1. Compute a γ-approximate MST T^G over the complete graph G

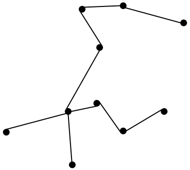
Our Approximation Algorithm



$w(\ ) \geq 1/\gamma \,.\, \max(\ )$

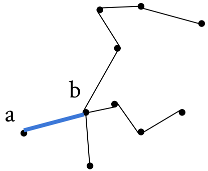
1. Compute a γ-approximate MST T^G over the complete graph G

Our Approximation Algorithm



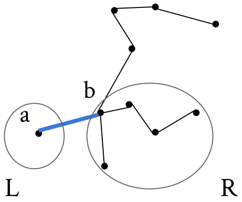
1. Compute a γ-approximate MST T^G over the complete graph G
2. Compute a β-estimate of the cut weights of the edges in T^G

Our Approximation Algorithm



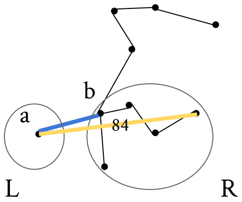
1. Compute a γ-approximate MST T^G over the complete graph G
2. Compute a β-estimate of the cut weights of the edges in T^G

Our Approximation Algorithm



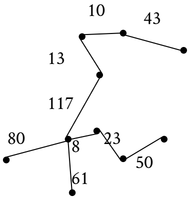
1. Compute a γ-approximate MST T^G over the complete graph G
2. Compute a β-estimate of the cut weights of the edges in T^G

Our Approximation Algorithm



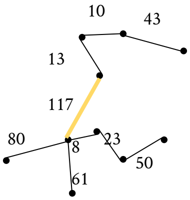
1. Compute a γ-approximate MST T^G over the complete graph G
2. Compute a β-estimate of the cut weights of the edges in T^G

Our Approximation Algorithm



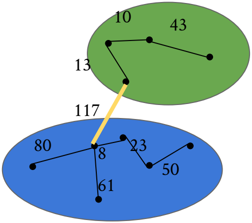
1. Compute a γ-approximate MST T^G over the complete graph G
2. Compute a β-estimate of the cut weights of the edges in T^G

Our Approximation Algorithm



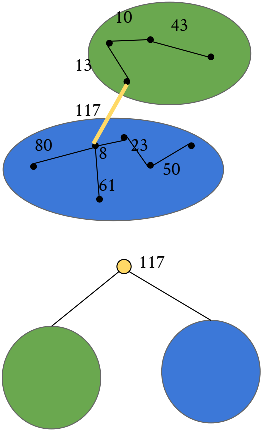
1. Compute a γ-approximate MST T^G over the complete graph G
2. Compute a β-estimate of the cut weights of the edges in T^G
3. Compute the ultrametric tree using T^G and β-estimates

Our Approximation Algorithm



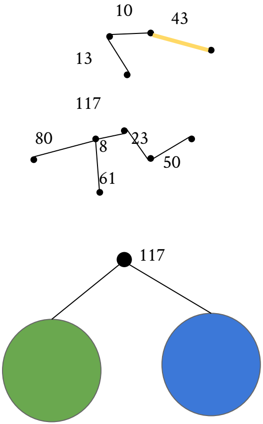
1. Compute a γ-approximate MST T^G over the complete graph G
2. Compute a β-estimate of the cut weights of the edges in T^G
3. Compute the ultrametric tree using T^G and β-estimates

Our Approximation Algorithm



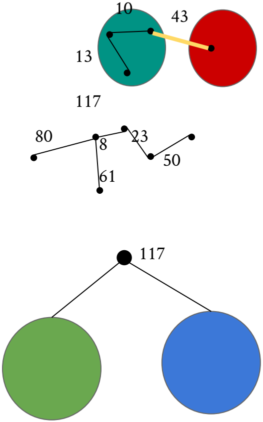
1. Compute a γ-approximate MST T^G over the complete graph G
2. Compute a β-estimate of the cut weights of the edges in T^G
3. Compute the ultrametric tree using T^G and β-estimates

Our Approximation Algorithm



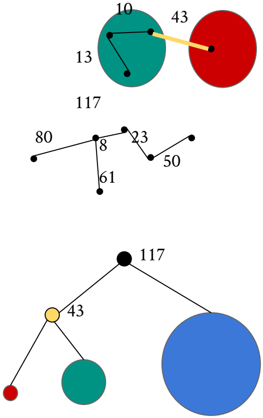
1. Compute a γ-approximate MST T^G over the complete graph G
2. Compute a β-estimate of the cut weights of the edges in T^G
3. Compute the ultrametric tree using T^G and β-estimates

Our Approximation Algorithm



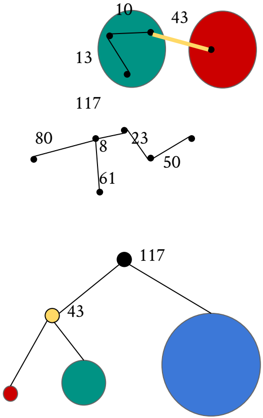
1. Compute a γ-approximate MST T^G over the complete graph G
2. Compute a β-estimate of the cut weights of the edges in T^G
3. Compute the ultrametric tree using T^G and β-estimates

Our Approximation Algorithm



1. Compute a γ-approximate MST T^G over the complete graph G
2. Compute a β-estimate of the cut weights of the edges in T^G
3. Compute the ultrametric tree using T^G and β-estimates

Our Approximation Algorithm



1. Compute a γ-approximate MST T^G over the complete graph G
2. Compute a β-estimate of the cut weights of the edges in T^G
3. Compute the ultrametric tree using T^G and β-estimates

→ **This gives a $\gamma \cdot \beta$-approximation**

Our Approximation Algorithm: Implementation

- For any $\gamma \geq 1$, γ-spanner constructions of Har-Peled, Indyk, Sidiropoulos in time$O(nd + n^{1+O(1/\gamma^2)})$

Our Approximation Algorithm: Implementation

- For any $\gamma \geq 1$, γ-spanner constructions of Har-Peled, Indyk, Sidiropoulos in time $O(nd + n^{1+O(1/\gamma^2)})$

- $\beta = 5$-estimate using a variant of union-find data structure

Hardness from SETH

Theorem (Cohen-Addad–K–Lagarde)

Assuming SETH, for every $\varepsilon > 0$, no algorithm running in $n^{2-\varepsilon}$ time, given $X \in \mathbb{R}^{O_\varepsilon(\log n)}$ ($|X| = n$) in ℓ_∞-space can distinguish:

YES: X can be embedded isometrically into an ultrametric.
NO: Distortion is at least $3/2$.

Hardness from SETH

Theorem (Cohen-Addad–K–Lagarde)

Assuming SETH, for every $\varepsilon > 0$, no algorithm running in $n^{2-\varepsilon}$ time, given $X \in \mathbb{R}^{O_\varepsilon(\log n)}$ ($|X| = n$) in ℓ_∞-space can distinguish:

YES: X can be embedded isometrically into an ultrametric.
NO: Distortion is at least $3/2$.

Morally Equivalent to Search Version

Hardness from SETH

Theorem (Cohen-Addad–K–Lagarde)

Assuming SETH, for every $\varepsilon > 0$, no algorithm running in $n^{2-\varepsilon}$ time, given $X \in \mathbb{R}^{O_\varepsilon(\log n)}$ ($|X| = n$) in ℓ_∞-space can distinguish:

YES: X can be embedded isometrically into an ultrametric.
NO: Distortion is at least $3/2$.

$\Uparrow$

Theorem (David–K–Laekhanukit'19)

Assuming SETH, for every $\varepsilon > 0$, no algorithm running in $n^{2-\varepsilon}$ time, given $A, B \in \mathbb{R}^{O_\varepsilon(\log n)}$ ($|A| = |B| = n$) can distinguish:

YES: $\exists (a, b) \in A \times B$ such that $\|a - b\|_\infty = 1$.
NO: $\forall (a, b) \in A \times B$ we have $\|a - b\|_\infty = 3$.

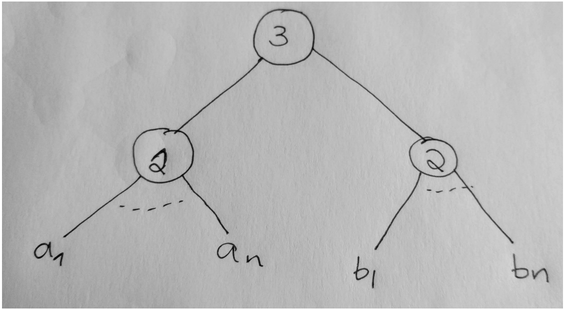
Moreover, in both cases $\text{dist}(A) = \text{dist}(B) = 2$ and $\text{dist}(A, B) \in \{1, 3\}$.

Hardness from SETH: YES case

- Input: $A, B \in \mathbb{R}^{O(\log n)}$ $(|A| = |B| = n)$
- Promise: $\forall a, a' \in A$ and $\forall b, b' \in B$: $\|a - a'\|_\infty = \|b - b'\|_\infty = 2$
- Case Assumption: $\forall (a, b) \in A \times B$ we have $\|a - b\|_\infty = 3$

Hardness from SETH: YES case

- **Input**: $A, B \in \mathbb{R}^{O(\log n)}$ $(|A| = |B| = n)$
- **Promise**: $\forall a, a' \in A$ and $\forall b, b' \in B$: $\|a - a'\|_\infty = \|b - b'\|_\infty = 2$
- **Case Assumption**: $\forall (a, b) \in A \times B$ we have $\|a - b\|_\infty = 3$



Hardness from SETH: NO case

- **Input**: $A, B \in \mathbb{R}^{O(\log n)}$ $(|A| = |B| = n)$
- **Promise**: $\forall a, a' \in A$ and $\forall b, b' \in B$: $\|a - a'\|_\infty = \|b - b'\|_\infty = 2$
- **Case Assumption**: $\exists (a, b) \in A \times B$ such that $\|a - b\|_\infty = 1$

Hardness from SETH: NO case

- Input: $A, B \in \mathbb{R}^{O(\log n)}$ $(|A| = |B| = n)$
- Promise: $\forall a, a' \in A$ and $\forall b, b' \in B$: $\|a - a'\|_\infty = \|b - b'\|_\infty = 2$
- Case Assumption: $\exists (a, b) \in A \times B$ such that $\|a - b\|_\infty = 1$
- Let $S : \{a, a', b\}$ such that $\|a - b\| = 1$ and $\|a' - b\| = 3$

Hardness from SETH: NO case

- **Input**: $A, B \in \mathbb{R}^{O(\log n)}$ $(|A| = |B| = n)$
- **Promise**: $\forall a, a' \in A$ and $\forall b, b' \in B$: $\|a - a'\|_\infty = \|b - b'\|_\infty = 2$
- **Case Assumption**: $\exists (a, b) \in A \times B$ such that $\|a - b\|_\infty = 1$
- Let $S : \{a, a', b\}$ such that $\|a - b\| = 1$ and $\|a' - b\| = 3$
- Let $\tau : S \to L$ be ultrametric embedding and ρ be distortion

Hardness from SETH: NO case

- **Input**: $A, B \in \mathbb{R}^{O(\log n)}$ $(|A| = |B| = n)$
- **Promise**: $\forall a, a' \in A$ and $\forall b, b' \in B$: $\|a - a'\|_\infty = \|b - b'\|_\infty = 2$
- **Case Assumption**: $\exists (a, b) \in A \times B$ such that $\|a - b\|_\infty = 1$
- Let $S : \{a, a', b\}$ such that $\|a - b\| = 1$ and $\|a' - b\| = 3$
- Let $\tau : S \to L$ be ultrametric embedding and ρ be distortion

$$
\begin{aligned}
3 = \|a' - b\|_\infty &\leq \Delta(\tau(a'), \tau(b)) \\
&\leq \max\{\Delta(\tau(a), \tau(b)), \Delta(\tau(a'), \tau(a))\} \\
&\leq \max\{\rho \cdot \|a - b\|_\infty, \rho \cdot \|a' - a\|_\infty\} = 2\rho
\end{aligned}
$$

Colinearity Problem

- YES case: Input is n points sampled from $\mathscr{B}_d$.

Colinearity Problem

- YES case: Input is n points sampled from $\mathscr{B}_d$.
- NO case:
  - Sample $(a_1, \ldots, a_n)$ from $\mathscr{B}_d$.

Colinearity Problem

- YES case: Input is n points sampled from $\mathscr{B}_d$.
- NO case:
  - Sample $(a_1, \ldots, a_n)$ from $\mathscr{B}_d$.
  - Pick distinct indices i, j, k in $[n]$ at random.

Colinearity Problem

- YES case: Input is n points sampled from $\mathcal{B}_d$.
- NO case:
  - Sample $(a_1, \ldots, a_n)$ from $\mathcal{B}_d$.
  - Pick distinct indices i, j, k in $[n]$ at random.
  - Let $a_{i,j}$ be the midpoint of a_i and a_j.

Colinearity Problem

- YES case: Input is n points sampled from $\mathscr{B}_d$.
- NO case:
  - Sample $(a_1, \ldots, a_n)$ from $\mathscr{B}_d$.
  - Pick distinct indices i, j, k in $[n]$ at random.
  - Let $a_{i,j}$ be the midpoint of a_i and a_j.
  - Let $\widetilde{a_k}$ be $(1 - \rho) \cdot a_k + \rho \cdot a_{i,j}$.

Colinearity Problem

- YES case: Input is n points sampled from $\mathcal{B}_d$.
- NO case:
  - Sample $(a_1, \ldots, a_n)$ from $\mathcal{B}_d$.
  - Pick distinct indices i, j, k in $[n]$ at random.
  - Let $a_{i,j}$ be the midpoint of a_i and a_j.
  - Let $\widetilde{a_k}$ be $(1 - \rho) \cdot a_k + \rho \cdot a_{i,j}$.
  - Input is $(a_1, \ldots, \widetilde{a_k}, a_{k+1}, \ldots, a_n)$.

Colinearity Hypothesis

- Colinearity Hypothesis: There exists constants $\rho, \varepsilon > 0$ such that no randomized algorithm running in time $n^{1+\varepsilon}$ can distinguish the two cases for every $d \geq O_{\rho,\varepsilon}(\log n)$.

Colinearity Hypothesis

- Colinearity Hypothesis: There exists constants $\rho, \varepsilon > 0$ such that no randomized algorithm running in time $n^{1+\varepsilon}$ can distinguish the two cases for every $d \geq O_{\rho,\varepsilon}(\log n)$.
- Worst Case variant is 3-SUM hard for even $d = 2$.

Colinearity Hypothesis

- Colinearity Hypothesis: There exists constants $\rho, \varepsilon > 0$ such that no randomized algorithm running in time $n^{1+\varepsilon}$ can distinguish the two cases for every $d \geq O_{\rho,\varepsilon}(\log n)$.
- Worst Case variant is 3-SUM hard for even $d = 2$.
- Related to Light bulb problem.

Colinearity Hypothesis

- Colinearity Hypothesis: There exists constants $\rho, \varepsilon > 0$ such that no randomized algorithm running in time $n^{1+\varepsilon}$ can distinguish the two cases for every $d \geq O_{\rho,\varepsilon}(\log n)$.
- Worst Case variant is 3-SUM hard for even $d = 2$.
- Related to Light bulb problem.

Theorem (Cohen-Addad–K–Lagarde)

Assuming CH, there exists $\varepsilon, \delta > 0$, no randomized algorithm running in $n^{1+\varepsilon}$ time, given $X \in \mathbb{R}^{O_{\varepsilon,\delta}(\log n)}$ ($|X| = n$) in Euclidean space can distinguish:

YES: Distortion is at most $1 + \delta$.
NO: Distortion is at least $1 + 2\delta$.

Results

Theorem (Farach–Kannan–Warnow'95)

Given the distance matrix of n points, the optimal ultrametric embedding can be computed in time $O(n^2)$.

Theorem (Cohen-Addad–K–Lagarde)

- Assuming SETH, no 1.5 approximate embedding in $n^{1.99}$ time from ℓ_∞-metric.
- Assuming Colinearity Hypothesis, no 1.001 approximate in $n^{1+o(1)}$ time from Euclidean metric.
- For any $\gamma \geq 1$, 5γ approximate embedding in time $O(n^{1+\frac{1}{\gamma^2}})$ for Euclidean metric.

Results

Theorem (Farach–Kannan–Warnow'95)

Given the distance matrix of n points, the optimal ultrametric embedding can be computed in time $O(n^2)$.

Theorem (Cohen-Addad–K–Lagarde)

- Assuming SETH, no 1.5 approximate embedding in $n^{1.99}$ time from ℓ_∞-metric.
- Assuming Colinearity Hypothesis, no 1.001 approximate in $n^{1+o(1)}$ time from Euclidean metric.
- For any $\gamma \geq 1$, 5γ approximate embedding in time $O(n^{1+\frac{1}{\gamma^2}})$ for Euclidean metric.

Performs Well in Experiments!

Open Problems

Improved Approximation Factor?

Open Problems

Improved Approximation Factor?

Euclidean Inapproximability under SETH?

Open Problems

Improved Approximation Factor?

Euclidean Inapproximability under SETH?

More Applications of Colinearity Hypothesis?

THANK
YOU!