Hardness of Approximate Sequence Similarity Search

S. Cenk Sahinalp

Joint work with Andrey Utis
Nearest neighbor search (NN)

- **Distance:** \( d: U^2 \rightarrow \mathbb{R} \) s.t. For all \( a, b \) in \( U 
  \begin{align*}
  d(a, b) & \geq 0, \\
  d(a, b) & = d(b, a)
  \end{align*} \)

- **Goal:** Preprocess \( a_1 \ldots a_p \) in \( U \), s.t.
  For each \( q \) in \( U 
  \begin{align*}
  \text{find } a_i & \text{ s.t. } d(a_i, q) \leq d(a_j, q), \text{ for all } a_j \text{ in } U 
  \end{align*} \)

- **Efficiency:** (dimensionality: \( \max_i |a_i| = n \))
  \begin{align*}
  \text{Preprocessing time: } & O(\text{poly}(n, p)), \\
  \text{Search time: } & O(\text{poly}(n, \log p))
  \end{align*}
Approximate Nearest Neighbor (ANN) Search

Given $a_1...a_p$ in $U$, for each $q$ in $U$,
Find $a_i$ s.t. for all $a_j$, $d(a_i,q) \leq c(n).d(a_j,q)$,
for small $c(n)$

FN (farthest neighbor search) and AFN can be similarly defined
Randomized ANN is easier

- $O(1)$-ANN algorithms for Hamming distance, $L_1$ and $L_2$ norms
- $O(\log \log n)$-ANN algorithm for $L_\infty$
- Hausdorff metric on finite sets
- Frechet metric (between sequences of points)
- Edit distance with block operations
Hardness Results for NN in Hamming Space

For $U=\{0,1\}^n$, $h(a,b)$ (Hamming distance)
several hardness results in cell probe model of Yao
[Chazelle et al, Borodin et al, Kumar et al, Liu, Chakrabati Regev]

All based on the richness technique [Miltersen et al],

None prohibitive BUT no positive result for exact or deterministic NN.
This talk: ANN/AFN in string spaces is as hard as ENN in Hamming space
Synopsis of the talk

- $c(n)$-ANN/AFN under set intersection (SI) is as hard as ENN/EFN under SI which is as hard as ENN/EFN under Hamming distance

- $c(n)$-AFN/ANN for LCS, DS, CVC and Edit Distance is at least as hard as ENN/EFN under Hamming distance

- ANN/AFN (furthest neighbors) for set intersection is inapproximable

- A linear time 2-approximation algorithm for permutation edit distance (PED) can be obtained
Oblivious reductions

Given \( d_1 \) and \( d_2 \), distance measures on \( U_1 \) and \( U_2 \), respectively

a pair of poly-time computable functions

\( f_A, f_B : U_1 \rightarrow U_2 \) provide a \((\kappa, \sigma)\)-oblivious reduction if:

\[
\kappa(n).d_1(a,b) \geq d_2(f_A(a),f_B(b)) \geq d_1(a,b)
\]

and

\[
|f_A(a)| \leq \sigma(|a|), \quad |f_B(b)| \leq \sigma(|b|)
\]
Reducibility among ANN (AFN) problems

• Suppose $d_1$ is $(\kappa,\sigma)$-obliviously reducible to $d_2$,
  Let $\sigma(n) = O(poly(n))$,
  Let $c(n)$ be a non-decreasing function

• If there is no efficient $c(n)$-ANN algorithm for $d_1$,
  then there is no $c(n)/\kappa(n)$-ANN algorithm for $d_2$
  i.e. $d_2$ is no easier than $d_1$ for ANN
Permutation edit distance (PED)

Given two permutations \(a, b\) from alphabet of size \(n\)
PED is the minimum number of character moves needed to transform \(a\) to \(b\)

e.g. \(\text{PED}(abc, cab) = 1\)
Hamming(ANN) is (1,2n)-obliviously reducible to PED(ANN)

- Given a,b in \{0,1\}^n,
  let \(\Sigma = \{x_1,\ldots,x_n,y_1,\ldots,y_n\}\)

- Define
  \[f_A(a_i=0) = f_B(b_i=0) = x_iy_i,\]
  \[f_A(a_i=1) = f_B(b_i=1) = y_ix_i\]

- Let
  \[f_A(a) = f_A(a_1)\ldots f_A(a_n)\]
  and
  \[f_B(b) = f_B(b_1)\ldots f_B(b_n)\]

  then
  \[h(a,b) = \text{PED}(f_A(a), f_B(b))\]
PED is (log n, n^2)-obliviously reducible to SI

**Sets:** binary vectors in \{0,1\}^n,

PED is (log n, n^2)-obliviously reducible to SI, i.e., dot product [Cormode Muthu S.]
SI is (1,3n)-obliviously reducible to LCS

• Given strings a,b over Σ,
  LCS: size of the longest subsequence that occurs in both a and b
  e.g. LCS (aabcd, abadc) = 3

• Given a,b in \{0,1\}^n, define Σ = \{x_1,…,x_n,y_1,…,y_n,z_1,…,z_n\}

• Let
  \( f_A(a_i=0) = x_i \), \( f_A(a_i=1) = y_i \)
  \( f_B(b_i=0) = z_i \), \( f_B(b_i=1) = y_i \)

  and
  \( f_A(a) = f_A(a_1)…f_A(a_n) \)
  \( f_B(b) = f_B(b_1)…f_B(b_n) \)

  then \( a\cdot b = LCS (f_A(a), f_B(b)) \)
Difference graphs

- Let $G_a, G_b$ be graphs on $n$ vertices labeled $1\ldots n$

- $GD(G_a,G_b)$ is a labeled graph, s.t. an edge $e$ is in $GD$ if and only if it occurs exactly in one of $G_a$ or $G_b$

- Difference Vertex Cover (DVC) is the size of the minimum vertex cover of $GD(G_a,G_b)$

- DVC is a metric
PED is \((1,n^2)\)-obliviously reducible to DVC

- Let \(a, b\) be permutations of \(\Sigma = \{1 \ldots n\}\)

- \(f_A(x) = f_B(x)\) is a graph on \(n\) vertices, with edge \((i,j)\) whenever \(i < j\) and \(i\) occurs before \(j\) in \(a\)

- \(GD(f_A(a), f_B(b))\) contains edge \((i,j)\) if and only if \(i\) and \(j\) occur in different order in \(a\) and \(b\)

- DVC is the minimum number of characters needed to remove to eliminate the differences between \(a\) and \(b\); which is \(PED(a, b)\)
2-approximation of PED

Use 2-approximation to VC:

Let \( a, b \) permutations of \( \Sigma = \{1 \ldots n\} \)
if \( a_1 \neq b_1 \), GD must have an edge \((a_1, b_1)\)

Delete \( a_1 \) and \( b_1 \), keep an array that indicates deleted characters, and repeat

\( O(n) \) time
Common graphs

- Let $G_a, G_b$ be graphs on $n$ vertices labeled by $1 \ldots n$

- $GC(G_a, G_b)$ is a labeled graph, s.t. an edge $e$ is in $GC$ if and only if it occurs in both $G_a$ and $G_b$

- Common Vertex Cover (CVC) is the size of smallest VC of $GC(G_a, G_b)$

- CVC is not a metric
DVC is (2,4n)-obliviously reducible to CVC

Let $G_a, G_b$ be graphs on $n$ vertices labeled $1 \ldots n$
Obtain $G_x`$ from $G_x$ by renaming each vertex $i$ as $(i+n)$

Define $f_A(G_a)$ as the concatenation of $G_a$ and $G_a^c$
Define $f_B(G_b)$ as the concatenation of $G_b^c$ and $G_b`$

Each edge of $GD(G_a, G_b)$ is either in $GC(G_a, G_b^c)$ or in $GC(G_a^c, G_b`)$. 
SI is \((1, 4n^2)\)-obliviously reducible to CVC

- Given \(a, b\) in \(\{0, 1\}^n\)
  \(f_A(x) = f_B(x)\) is a labeled graph on \(2n\) vertices

- Edge \((i, j)\) is in the graph whenever \(j = i+n\) and \(x_i = 1\)

- Each bit has a corresponding pair of vertices
More on the hardness of SI-ANN

Claim 1: For any constant $k$, $k$-ANN under SI is as hard as ENN under SI
A general vector transformation

- \( f: \mathbb{R}^n \rightarrow \mathbb{R}^{n^2} \) defined as follows

Given \( x = [x_1, \ldots, x_n] \)

Let \( y = f(x) = [y_{1,1}, y_{1,2}, \ldots, y_{1,n}, y_{2,1}, \ldots, y_{n,n}] \) s.t. \( y_{i,j} = x_i \cdot x_j \)

Key property:

\[
f(a) \cdot f(b) = \sum_{i=1}^{n} \sum_{j=1}^{n} f(a)_{i,j} \cdot f(b)_{i,j} = \sum_{i=1}^{n} \sum_{j=1}^{n} a_i \cdot a_j \cdot b_i \cdot b_j = \sum_{i=1}^{n} a_i b_i \sum_{j=1}^{n} a_j b_j = (a \cdot b)^2
\]
Inapproximability

- Suppose we have a k-ANN algorithm
- Apply f to the inputs $a_1...a_p, q$:
  \[(a_i \cdot q)^2 \geq k (a_j \cdot q)^2\]
  i.e., $a_i \cdot q \geq k a_j \cdot q$

- One O(1)-ANN algorithm implies all O(1)-ANN algorithms
- Identical proof for AFN
Orthogonality Query on \{-1,0,1\}

- Given $a_1...a_p$, and $q$ in $\{-1,0,1\}^n$
  
  if there exists at least one $a_i$ s.t. $a_i \cdot q = 0$, return it;
  otherwise return arbitrary $a_j$

- **Claim 2:** OQ(-1,0,1) is at least as hard as ENN for SI
Proof of claim 2

- Given $a_1...a_p$, and $q$ in $\{0,1\}^n$

  for each $0 \leq L \leq n$, construct $a_{1,L}...a_{p,L}$, $q_L$ in $\{0,1\}^{n+L}$ as follows:

- let $a_{i,L}$ be $a_i$ with $L$ 1's appended at the end

- let $q_L$ be $q$ with $L-1$'s appended at the end

  $q_L \cdot a_{i,L} = q \cdot a_i - L$

- The $L$-th data structure contains an orthogonal vector if and only if for some $a_i$, $a_i \cdot q = L$
Orthogonality Query on \{0,1\}

- Given \(a_1...a_p\) and \(q\) in \(\{0,1\}^n\)
  - if there is \(a_i\) s.t. \(a_i \cdot q = 0\), return one such \(a_i\);
  - otherwise return arbitrary \(a_j\)

- **Claim 3:** \(OQ(0,1)\) is at least as hard as \(OQ(-1,0,1)\)
Proof of claim 3

- Given $a_1...a_p$, and $q$ in $\{-1,0,1\}^n$

\[ f(a_i)\cdot f(q) = 0 \quad \text{iff} \quad a_i\cdot q = 0 \]

as $f(a_i)\cdot f(q) = (a_i\cdot q)^2$

- Observation:

\[ f(a_i)\cdot f(q) = (a_i\cdot q)^2 \geq 0, \quad \text{and} \quad f(a_i)\cdot f(a_j) = (a_i\cdot a_j)^2 \geq 0 \]

- The dot product of any two vectors is non-negative; i.e. the angle between any two vectors is at most 90 degrees
• **Conclusion:** there exists an $n^2$-dimensional cube with one vertex at the origin, which encloses $f(a_i)$ for all $i$, and $f(q)$

• This cube is only a function of $n$

• This cube defines a new orthogonal coordinate system, where all vectors have non-negative coordinates
• Two vectors with non-negative coordinates are orthogonal iff the product at every coordinate is 0.

• For each input vector v, compute v' by taking the projection of f(v) onto the edges of the cube, and setting each non-zero coordinate to 1.
  v' is a vector in \( \{0,1\}^{n^2} \)

• This preserves the pairwise orthogonality of vectors.

• Thus OQ(-1,0,1) reduces to OQ(0,1).

• The reduction is polynomial time and space, it squares the number of dimensions.
Proof of claim 1

- Suppose one can efficiently solve k-ANN for SI.
- Given $a_1 \ldots a_p$ and $q$ in $\{0,1\}^n$
  
  if there exists $a_j$ s.t. $a_j \cdot q = 0$,
  
  the only way to have $a_i \cdot q \leq k.a_j \cdot q = 0$ is
  
  to return $a_i$ with $a_i \cdot q = 0$

- This solves the OQ(0,1) problem which is as hard as ENN for SI
Stronger Claim 1

NN for SI with both multiplicative and additive errors:

\[ d(q,a_i) \gtrsim k \cdot d(q,a_j) + O(\text{polylog } n) \]

is at least as hard as ENN.
More oblivious reductions
• \( \min \{ a_i \cdot q \} \)

= \( \min \{ L : L^{\text{th}} \text{ data structure has an orthogonal vector} \} \)

• By trying \( L \) in order 0, 1, \ldots, \( n \),
  the first orthogonal vector is the NN of \( Q \)

• The reduction is polynomial time and space