Ulam Metric k-Center in Parameterized Linear Time*

Matěj Kripner Václav Rozhoň Robert Šámal
Martin Šindelář Adrián Zámečník

hexagon:2611.00055v1

Abstract

Continuous Ulam Metric k-Center asks for at most k permutations whose radius-d balls cover the input rankings. For m rankings of n symbols, a deterministic algorithm takes $2^{O(kd\log(k+d+2))}mn$ word operations. Permanent reference rankings, short insertion lists, and small obstructions obtained by patience sorting control the search. For one center, this answers a question of Bai, Fomin, Golovach, More, and Wietheger with linear input dependence. An arbitrary-alphabet Hamming embedding with a nonexpansive retraction also gives the matching $2^{o(d\log d)}$ exclusion under the Exponential Time Hypothesis.

1 Introduction

1.1 Background and the source problem

A consensus ranking should remain close to the rankings it summarizes. Data with several distinct groups may require several centers. Ulam distance measures how many symbols must be moved while the other symbols retain their relative order. The radius is a useful parameter when each ranking disagrees with its group's consensus on only a few items. For example, moving the symbol 3 in 12345 gives 13245, so these rankings have Ulam distance one.

Let $m, n \geq 1$ and let Σ be an n-symbol alphabet. For permutations π, σ of Σ, define

$$\mathrm{dist}_{\mathrm{U}}(\pi, \sigma) = n - \mathrm{LCS}(\pi, \sigma),$$

where LCS denotes longest common subsequence length. Given m input permutations $\pi_1, \ldots, \pi_m$ and integers $k \geq 1$, $d \geq 0$, continuous Ulam Metric k-Center asks for at most k permutations $\sigma_1, \ldots, \sigma_k$ such that

$$\max_{1\leq i\leq m} \min_{1\leq j\leq k} \mathrm{dist}_{\mathrm{U}}(\pi_i, \sigma_j) \leq d.$$

The centers may be arbitrary permutations; they need not be inputs. This is the convention of Bai, Fomin, Golovach, More, and Wietheger [1, Section 1.1].

Bai et al. prove NP-hardness even for $d = 1$ when k is part of the input, and give an exact algorithm in time

$$2^{O(d^2k(d+k))}mn^{1+o(1)}$$

[1, Theorems 1 and 2]. For one center, the parameter dependence is $2^{O(d^3)}$. By comparison, the Hamming-distance consensus problem Closest String has $d^{O(d)}$ dependence over arbitrary alphabets, by the bounded-search-tree algorithm of Gramm, Niedermeier, and Rossmanith [7, Theorem 1]. Bai et al. ask whether the Ulam problem has the same dependence [1, Section 5, p. 19:21].

Question 1 (Bai–Fomin–Golovach–More–Wietheger)**.** Can continuous Ulam Metric 1-Center be solved in time $d^{O(d)}(mn)^{O(1)}$?

*This work was produced within the Bolzano project (https://bolzano.app), using AI systems for mathematical research and manuscript preparation. The listed authors contributed to the project and the preparation of this release. The manuscript received only limited human checking.

1.2 Contribution

Theorem 2. *For $k, d \geq 1$, continuous* Ulam Metric k-Center *can be decided deterministically in*

$$2^{O(kd\log(k+d+2))}mn \tag{1}$$

word operations, and a set of centers can be returned on every yes-instance. The permutations are given explicitly as arrays of symbols in $[n]$; a word holds an input address or a symbol.

For $k = 1$, this is $2^{O(d\log(d+2))}mn$, answering Question 1 while retaining linear dependence on the explicit input size. The boundary case $d = 0$ amounts to counting distinct input permutations and can be solved by direct comparisons to at most $k + 1$ representatives in $O(kmn)$ time.

The radius dependence for one center is also optimal in the following standard conditional sense.

Theorem 3. *Unless the Exponential Time Hypothesis fails, continuous* Ulam Metric 1-Center *has no algorithm with running time $2^{o(d\log d)}(mn)^{O(1)}$.*

Theorem 3 does not assert optimal dependence on k. The lower-bound statement was already made in the conclusion of Bai et al.; Section 5 supplies an arbitrary-alphabet reduction that justifies its use of the Closest String lower bound.

The main obstacle to the algorithm is that a radius-d ball can contain $n^{\Omega(d)}$ permutations. Guessing the moved symbols and their new positions would put the parameter in the exponent of the input size. Three ingredients avoid that enumeration.

1. A failed candidate has an obstruction on at most $2d + 2$ symbols. Stopping patience sorting after $d + 1$ replacements produces this obstruction within $O(n + (d + 1)\log(n + 1))$ operations (Lemma 4).

2. Relative to a fixed reference ranking, a symbol has at most $4d + 2$ relevant insertion cuts. Canonical cuts ensure that partial placements continue to agree with one fixed solution outside the still unguessed moved symbols (Lemmas 5–7).

3. Each center keeps its own permanent reference. An uncovered input either supplies a reference for a new cluster or exposes a new moved symbol for an existing center. At most k references and kd moved symbols are introduced (Section 4).

The source already uses a search that creates candidate centers from uncovered inputs. Its local steps use permutation graphs, short lists of one-symbol moves, and suitable two-colorings; the full version constructs a coloring family of size $2^{O(d^2)}\log n$ [2, Lemmas 11, 13, 16 and Algorithm 1]. Here the references remain fixed while partial placements accumulate. A small obstruction selects a new symbol directly, avoiding the coloring family. The compatibility argument preserves the association of each reference with a distinct true cluster, so the input partition need not be guessed. Intermediate candidates need not cover the same inputs at every stage; the final coverage test certifies the solution.

The linear input factor uses only a threshold version of the classical patience-sorting algorithm, not a faster unrestricted longest-increasing- subsequence algorithm. Almost all scanned symbols extend the list of pile tops in constant time; only $d + 1$ operations can require binary search before either success or an obstruction is known.

1.3 Related work

Bachmaier, Brandenburg, Gleißner, and Hofmeier proved NP-completeness of maximum rank aggregation under Ulam distance, among other metrics [3, Theorem 3]. Their bounded-modification-set framework also gives fixed-parameter algorithms for Kendall tau, Hamming, and Minkowski distances; Ulam distance is not among those algorithmic instantiations [3, Section 6]. Choosing an input

permutation gives the familiar factor-2 metric approximation. For m input rankings, Chakraborty, Gajjar, and Jha obtain a deterministic $(3/2 - 1/(3m))$-approximation in time $n^{O(m^2 \log m)}$ [4, Theorem 1]. More recently, Azgor, Chakraborty, Cuong, Das, Habib, and Nguyen prove that, for each $0 < \varepsilon < 1/34$, approximating the continuous center within $35/34 - \varepsilon$ is NP-hard even for four rankings [11, Theorem 1.3]. This rules out polynomial-time exact optimization for fixed $m = 4$, unless P equals NP; it is compatible with an exact algorithm parameterized by the radius.

The *discrete* center problem restricts the center to an input permutation. Abboud, Bateni, Cohen-Addad, Karthik C. S., and Seddighin study its fine-grained complexity and give a $(1 + \varepsilon)$-approximation [9, Theorems 1 and 3]. Fischer, Goldenberg, Habib, and Karthik C. S. subsequently establish conditional optimality, up to subpolynomial factors, of the pairwise-distance algorithm when the permutation length is polynomially related to the number of inputs [10, Theorem 2]. Those results concern a different solution space and polynomial input-size dependence, rather than the radius dependence of the continuous problem considered here.

The lower-bound comparison requires attention to the alphabet. Lokshtanov, Marx, and Saurabh exclude $2^{o(d \log d)}$ dependence for Closest String over unrestricted alphabets [8, Theorem 3.1]. The proof of Lemma 18 in Bai et al.'s full version uses a binary pair encoding [2, Section 3.3]. Fixed-alphabet Closest String has single-exponential algorithms [8, Section 3]; for the binary case, Fischer and Habib give a randomized $O^*(4^d)$-time algorithm [6, Theorem 1]. Thus the binary encoding alone does not transfer the unrestricted-alphabet lower bound. The marker construction in Section 5 handles any alphabet size and projects arbitrary permutation centers back to strings without increasing distances. It therefore completes the lower-bound argument anticipated in the source.

2 Deletion witnesses and small obstructions

For $S \subseteq \Sigma$, write $\pi|_S$ for the subsequence induced by S. A *deletion witness* between α and β is a set $D \subseteq \Sigma$ such that

$$\alpha|_{\Sigma \setminus D} = \beta|_{\Sigma \setminus D}.$$

Equivalently, $\mathrm{dist}_{\mathrm{U}}(\alpha, \beta) \le d$ if and only if such a witness of size at most d exists. In particular, a reference ranking ρ within distance d of a center $\sigma^\star$ agrees with that center outside some set X of at most d symbols. No uniqueness of X is assumed.

The following algorithm gives both the distance test and the obstruction needed for branching. The elementary patience-sorting principle is classical; see Aldous and Diaconis [5].

Lemma 4 (Threshold test and obstruction)**.** *Given permutations α, β, the inverse-position array of β, and $d \ge 0$, one can either certify $\mathrm{dist}_{\mathrm{U}}(\alpha, \beta) \le d$ or find $H \subseteq \Sigma$ such that*

$$|H| \le 2d + 2, \qquad \mathrm{dist}_{\mathrm{U}}(\alpha|_H, \beta|_H) > d. \tag{2}$$

This takes $O(n + (d+1)\log(n+1))$ operations and $O(n)$ memory.

Proof. Read the symbols of α and replace each by its position in β. A common subsequence of the permutations is exactly an increasing subsequence of this sequence of distinct ranks. Maintain piles whose tops $t_1 < \cdots < t_\ell$ are increasing. A new rank $r > t_\ell$ starts a new pile on the right; otherwise place r on the leftmost pile whose top exceeds r. Call the latter operation a replacement. For an empty list, the first rank starts the first pile.

After any prefix, t_j is the smallest possible final rank of an increasing subsequence of length j. To verify the invariant, suppose r replaces t_j. If $j > 1$, an increasing subsequence ending at $t_{j-1} < r$ extends to one of length j ending at r; for $j = 1$ the singleton suffices. There was no subsequence of length j ending below r, by the minimality of the old t_j. Nor can r end a subsequence of length greater than j, since that would require an old subsequence of length j ending below r. The other minima are unchanged. If r starts a new rightmost pile, the same extension gives a subsequence

of length $\ell + 1$, and no longer one can be created by one new rank. This proves the invariant by induction. Thus ℓ is the longest increasing subsequence length of the scanned prefix.

After i ranks, the number of replacements is $i - \ell$, because every other operation starts one pile. This deficit never decreases. Stop at the $(d+1)$-st replacement. If the whole sequence is scanned first, its deficit is the Ulam distance, so the test succeeds.

On stopping, let $P_1, \ldots, P_t$ be the non-singleton piles and let H contain their symbols. Each pile is decreasing in scan order, so an increasing subsequence of H contains at most one symbol from each pile. Moreover,

$$\sum_{j=1}^{t} (|P_j| - 1) = d + 1, \qquad t \le d + 1.$$

Consequently $|H| = d + 1 + t \le 2d + 2$ and its longest increasing subsequence has length at most t, proving (2).

Compare each rank first with the rightmost top. Appending costs constant time, and a replacement uses binary search in $O(\log(n+1))$ time. There are at most $d+1$ replacements. Store the symbols of each pile, and keep a list of the piles that have become non-singleton; their union can then be collected in $O(d+1)$ time. The claimed bounds follow. □

3 Small cut lists

Fix an arbitrary reference permutation ρ. A cut of the reference permutation ρ is an integer $c \in \{0, 1, \ldots, n\}$; it is the gap after the first c symbols of ρ. For a symbol x and a cut c, define

$$P_c^x = \{a \in \Sigma \setminus \{x\} : \mathrm{pos}_\rho(a) \le c\}.$$

For an input permutation π_i, define the predecessor set of x in π_i by

$$L_i(x) = \{a \in \Sigma \setminus \{x\} : a \text{ is before } x \text{ in } \pi_i\}.$$

Define the cut list

$$S_i(x) = \{c \in \{0, \ldots, n\} : |P_c^x \triangle L_i(x)| \le 2d\}.$$

The key point is that, although there are $n+1$ possible cuts, only $O(d)$ of them can be relevant for any fixed input and symbol.

Lemma 5 (Small cut lists)**.** *For every input π_i and every symbol x,*

$$|S_i(x)| \le 4d + 2.$$

Proof. Put $\ell = |L_i(x)|$ and $h(c) = |P_c^x|$. For every listed cut,

$$|h(c) - \ell| \le |P_c^x \triangle L_i(x)| \le 2d.$$

Thus $h(c)$ lies in an interval containing at most $4d+1$ integers. As c ranges from 0 to n, the value $h(c)$ runs through $0, \ldots, n-1$. Each value occurs once except $\mathrm{pos}_\rho(x) - 1$, which occurs twice, immediately before and after the reference position of x. Consequently

$$|S_i(x)| \le 4d + 2.$$

For computation, begin with $|P_0^x \triangle L_i(x)| = |L_i(x)|$ and scan ρ. Passing a symbol of $L_i(x)$ decreases the value by one, passing a symbol outside $L_i(x) \cup \{x\}$ increases it by one, and passing x leaves it unchanged. The inverse-position array of π_i gives membership in $L_i(x)$ in constant time. Thus the list can be computed in $O(n)$ time, without any assumption on $\mathrm{dist}_{\mathrm{U}}(\rho, \pi_i)$. □

Now fix a permutation $\sigma^\star$ with $\mathrm{dist}_{\mathrm{U}}(\rho, \sigma^\star) \le d$, a deletion witness X of size at most d between them, and put $R = \Sigma \setminus X$. For $x \in X$, let

$$B_x = \{a \in R : a \text{ is before } x \text{ in } \sigma^\star\}.$$

Because $\sigma^\star|_R = \rho|_R$, the set B_x is a prefix of $\rho|_R$. Define the *canonical cut* of x by

$$c_x = \begin{cases} 0, & B_x = \emptyset, \\ \max\{\mathrm{pos}_\rho(a) : a \in B_x\}, & B_x \neq \emptyset. \end{cases} \tag{3}$$

Then $P^x_{c_x} \cap R = B_x$. In particular, the exact definition (3) assigns the same cut to all moved symbols occupying the same gap of the stable order R.

Lemma 6 (Canonical cuts are listed)**.** *Let $\sigma^\star$ and X be as above, and let $x \in X$. Let D_i be a deletion witness of size at most d between $\sigma^\star$ and π_i. If $x \notin D_i$, then the canonical cut c_x belongs to $S_i(x)$.*

Proof. Consider a symbol $a \in \Sigma \setminus (X \cup D_i)$. Since $a \in R$, the definition of the canonical cut gives $a \in P^x_{c_x}$ if and only if $a \in B_x$, which means that a precedes x in $\sigma^\star$. Also $a, x \notin D_i$, so $\sigma^\star$ and π_i order a and x in the same way. Therefore

$$a \in P^x_{c_x} \quad \Longleftrightarrow \quad a \in B_x \quad \Longleftrightarrow \quad a \in L_i(x).$$

All possible disagreements between $P^x_{c_x}$ and $L_i(x)$ are therefore contained in $X \cup D_i$. Neither predecessor set contains x, so the disagreements are in $(X \setminus \{x\}) \cup D_i$, a set of size at most $2d$. Hence $|P^x_{c_x} \triangle L_i(x)| \le 2d$, so $c_x \in S_i(x)$. □

Example. Let $\rho = 12345$, let $\pi_i = 13245$, and take $d = 1$. For the symbol $x = 3$, the predecessor set in π_i is $L_i(3) = \{1\}$. Direct inspection gives

$$S_i(3) = \{0, 1, 2, 3, 4\};$$

cut 5 is excluded because its predecessor set $\{1, 2, 4, 5\}$ differs from $L_i(3)$ in three symbols. The permutation $\sigma^\star = 13245$ has reference-deletion set $X = \{3\}$, and the canonical cut of 3 is cut 1, immediately after the symbol 1. Thus the cut needed by this permutation is indeed present in the local list, as Lemma 6 guarantees in general.

We next formalize partial placements. A state consists of

$$(A, \gamma, \lambda),$$

where $A \subseteq \Sigma$, $|A| \le d$, is a set of symbols guessed to be moved relative to ρ, $\gamma : A \to \{0, \dots, n\}$ assigns a full-reference cut to every guessed symbol, and λ is a total order of A. The candidate permutation σ_A is obtained by deleting every symbol of A from ρ and reinserting each $x \in A$ at cut $\gamma(x)$: smaller numerical cuts come first, and equal cuts are ordered according to λ. Formally, write $\lambda(x)$ for the rank of x in λ, note that $\mathrm{pos}_\rho(z) \in \{1, \dots, n\}$, and sort all symbols lexicographically by

$$\kappa_A(z) = \begin{cases} (\mathrm{pos}_\rho(z), 0, 0), & z \notin A, \\ (\gamma(z), 1, \lambda(z)), & z \in A. \end{cases}$$

This lexicographic definition remains unambiguous when a cut endpoint is itself a symbol of A deleted from the reference.

Lemma 7 (Reconstruction of compatible states)**.** *Let $\lambda^\star$ be the order induced by $\sigma^\star$ on X. Suppose that $A \subseteq X$, that $\gamma(x) = c_x$ for every $x \in A$, and that $\lambda = \lambda^\star|_A$. Put $U = X \setminus A$. Then*

$$\sigma_A|_{\Sigma \setminus U} = \sigma^\star|_{\Sigma \setminus U}. \tag{4}$$

Proof. Set $R = \Sigma \setminus X$, so that $\Sigma \setminus U = R \cup A$. Both permutations induce the order $\rho|_R$ on R.

Let $x \in A$ and $r \in R$. In σ_A, the symbol r precedes x if and only if $\mathrm{pos}_\rho(r) \le c_x$, equivalently $r \in B_x$. By the definition of B_x, this is also equivalent to r preceding x in $\sigma^\star$.

It remains to compare two symbols $x, y \in A$. The sets B_x and B_y are prefixes of the same order. If $B_x \subsetneq B_y$, then the definition (3) gives $c_x < c_y$, so x precedes y in σ_A. Choosing $r \in B_y \setminus B_x$ also gives $x < r < y$ in $\sigma^\star$. The case $B_y \subsetneq B_x$ is symmetric. Finally, if $B_x = B_y$, then $c_x = c_y$, and both permutations order x, y according to $\lambda^\star|_A = \lambda$. Thus every pair in $R \cup A$ has the same relative order in the two permutations, proving (4). □

The reconstruction can be implemented without sorting all n symbols. Sort the at most d moved symbols by $(\gamma(x), \lambda(x))$, then scan the cuts $0, \dots, n$. At a positive cut c, first output the reference symbol in position c if it is not moved; next output the moved symbols assigned to cut c in their prescribed order. At cut zero only the moved symbols are output. This realizes the lexicographic keys above in $O(n + d \log(d+1))$ time.

4 The search over several centers

A search node contains $s \le k$ records

$$(\rho_j, A_j, \gamma_j, \lambda_j), \qquad 1 \le j \le s.$$

Each reference ρ_j is an input permutation and remains fixed for the lifetime of its record. The other data are a partial placement relative to that reference, with $|A_j| \le d$; denote its reconstructed candidate by σ_j. Initially there is one record with $\rho_1 = \pi_1$ and $A_1 = \emptyset$.

At a node, perform the following steps.

1. Construct the candidates. If every input is within distance d of some candidate, return these candidates and accept.

2. Otherwise choose an uncovered input π_i. If $s < k$, create one child that adds the record $(\pi_i, \emptyset, \emptyset, \emptyset)$.

3. For each existing record j with $|A_j| < d$, use Lemma 4 to find an obstruction H_j between σ_j and π_i. For each $x \in H_j \setminus A_j$, each $c \in S_i^{(j)}(x)$, and each of the $|A_j| + 1$ insertion positions in λ_j, create a child that adds x with this cut and order position to record j. Here $S_i^{(j)}$ is the cut list computed relative to ρ_j. Keep all other records and all earlier placements unchanged.

4. Reject if every child rejects.

The algorithm does not reject an instance merely because two inputs are farther than $2d$ apart: they may be covered by different centers.

Proof of Theorem 2. If $k \ge m$, the inputs themselves are a solution. If $d \ge n - 1$, any input is a center. Hence assume $1 \le k < m$ and $1 \le d < n - 1$. Soundness follows from the coverage test at every accepting node.

For completeness, fix a feasible set of at most k true centers and assign each input to one true center that covers it. Empty clusters can be omitted. A compatible node associates its records injectively with true centers, with each reference belonging to its associated cluster. Write $\sigma_j^\star$ for the center associated with record j. When a record is created, fix a deletion witness X_j, $|X_j| \le d$, between ρ_j and $\sigma_j^\star$. Compatibility further requires

$$A_j \subseteq X_j, \qquad \gamma_j(x) = c_x^{(j)} \quad (x \in A_j), \qquad \lambda_j = \sigma_j^\star|_{A_j},$$

where $c_x^{(j)}$ is the canonical cut defined using ρ_j, X_j, and the last equality identifies a total order with its ordered sequence. Lemma 7 gives, with $U_j = X_j \setminus A_j$,

$$\sigma_j|_{\Sigma\setminus U_j} = \sigma_j^\star|_{\Sigma\setminus U_j}. \tag{5}$$

The initial node is compatible by associating its sole record with the true center assigned to π_1.

Consider an uncovered input π_i at a compatible node. If its assigned true center is not represented by a record, there are fewer records than true centers and hence $s < k$. The new-record child is compatible: associate it with that true center and choose its witness X_{s+1}. Its moved set is empty, so every placement condition is satisfied.

Otherwise let j be the record associated with the assigned true center. Choose a deletion witness D, $|D| \le d$, between π_i and $\sigma_j^\star$. Since π_i is uncovered, the obstruction H_j satisfies (2). It contains a symbol in $U_j \setminus D$. Indeed, if it contained no such symbol, then $H_j \setminus D$ would be disjoint from U_j. Equation (5) and the witness property of D would give

$$\sigma_j|_{H_j\setminus D} = \sigma_j^\star|_{H_j\setminus D} = \pi_i|_{H_j\setminus D}.$$

Deleting $H_j \cap D$, at most d symbols, would therefore make the restricted permutations equal, contradicting (2).

Choose $x \in H_j \cap (U_j \setminus D)$. In particular, $|A_j| < |X_j| \le d$, so this record is eligible for extension. The symbol is moved relative to its reference and is kept by the witness D. Lemma 6 places its canonical cut in $S_i^{(j)}(x)$. The algorithm includes the child selecting this cut and the insertion position that extends the current order to $\sigma_j^\star|_{A_j\cup\{x\}}$. This child is compatible.

Thus every compatible nonaccepting node has a compatible child. Each transition either adds a record or increases one moved set. There are at most $k-1$ record additions and kd moved-symbol additions, so no path has more than $k(d+1)-1$ transitions. A compatible path must therefore accept. This proves completeness without any assumption that coverage is monotone along the path.

There is at most one new-record child, and the extension of each record has at most $(2d+2)(4d+2)(d+1)$ choices. The branching factor is at most

$$b = 1 + k(2d+2)(4d+2)(d+1).$$

The depth bound gives $1 + b + \cdots + b^{k(d+1)}$ nodes. Since $d \ge 1$, this is $2^{O(kd\log(k+d+2))}$.

Precompute the inverse-position arrays of all inputs in $O(mn)$ time. At a node, reconstructing all candidates costs $O(k(n + d\log(d+1)))$. Coverage testing uses at most km applications of Lemma 4. For the uncovered input, the same routine provides each required obstruction. There are $O(k(d+1))$ relevant symbols, and each cut list takes $O(n)$ time. These costs are bounded by a polynomial in k, d times mn, using $\log(n+1) = O(n)$. Creating or copying a child record adds only a polynomial factor in k, d, already absorbed in the search bound. This proves (1) in the stated word model. □

5 Hamming retractions and the radius lower bound

An isometric encoding alone would not suffice for the lower bound: a permutation center outside the encoded strings could conceivably have a smaller radius. The following construction provides a simultaneous projection for all inputs. Write d_{H} for Hamming distance.

Lemma 8 (Arbitrary-alphabet retraction)**.** *For every $q \ge 2$ and $L \ge 1$, there are maps*

$$E : \{0,\dots,q-1\}^L \longrightarrow S_{Lq}, \qquad r : S_{Lq} \longrightarrow \{0,\dots,q-1\}^L$$

computable in polynomial time such that $r \circ E$ is the identity and

$$\mathrm{dist}_{\mathrm{U}}(E(s), E(t)) = d_{\mathrm{H}}(s,t), \tag{6}$$

$$d_{\mathrm{H}}(r(\alpha), r(\beta)) \le \mathrm{dist}_{\mathrm{U}}(\alpha,\beta) \qquad (\alpha,\beta \in S_{Lq}). \tag{7}$$

Consequently, replacing each input string by its encoding preserves the optimal radius with any prescribed number of arbitrary centers.

Proof. For coordinate i, use a disjoint block of symbols $B_i = \{z_i, a_{i,1}, \ldots, a_{i,q-1}\}$. Define

$$C_i(b) = a_{i,1} \cdots a_{i,b}\, z_i\, a_{i,b+1} \cdots a_{i,q-1} \qquad (0 \le b < q),$$

with an empty initial or final anchor string at the endpoints. Distinct values of b give distinct block permutations that share all $q-1$ anchors in order. Thus their Ulam distance is one. Set

$$E(s) = C_1(s_1)C_2(s_2)\cdots C_L(s_L).$$

Both encodings place the disjoint blocks in the same order, so their longest common subsequence lengths add over the blocks. This proves (6).

For an arbitrary permutation α, restrict it to each block B_i. If $\alpha|_{B_i} = C_i(b)$ for some b, put $r(\alpha)_i = b$; otherwise put $r(\alpha)_i = 0$. The canonical block permutations are distinct, so this is well-defined and $r(E(s)) = s$. If $r(\alpha)_i \neq r(\beta)_i$, then the two block restrictions differ, since equal restrictions receive equal decoded letters. Their Ulam distance, being a nonnegative integer, is at least one. Therefore

$$d_{\mathrm{H}}(r(\alpha), r(\beta)) \le \sum_{i=1}^{L} \mathrm{dist}_{\mathrm{U}}(\alpha|_{B_i}, \beta|_{B_i}).$$

Any common subsequence of α, β, restricted to B_i, is a common subsequence of their block restrictions. Summing these upper bounds on the subsequence length and subtracting from Lq gives

$$\sum_{i=1}^{L} \mathrm{dist}_{\mathrm{U}}(\alpha|_{B_i}, \beta|_{B_i}) \le \mathrm{dist}_{\mathrm{U}}(\alpha, \beta),$$

which proves (7).

An encoded family of string centers achieves the same radius by (6). Conversely, decode each arbitrary permutation center. For every input string t, applying (7) with $\beta = E(t)$ shows that its distance to the decoded center is no greater than its encoded distance to the original center. This works for every input and every center simultaneously, proving equality of optimal radii. □

For example, when $q = 3$ the three letters are represented by $z_i a_{i,1} a_{i,2}$, $a_{i,1} z_i a_{i,2}$, and $a_{i,1} a_{i,2} z_i$. The marker moves while the two anchors retain their order. A noncanonical order such as $a_{i,2} z_i a_{i,1}$ may be decoded as zero: it differs from every canonical order, so the possible unit Hamming cost is always paid for by its Ulam distance.

Proof of Theorem 3. Lokshtanov, Marx, and Saurabh prove that Closest String has no $2^{o(d \log d)}|I|^{O(1)}$-time algorithm under ETH [8, Theorem 3.1]. In their construction from $h \times h$ Hitting Set, the strings have length $L = h$, alphabet size $q = 2h + 1$, and radius $d = h - 1$. Applying Lemma 8 yields permutations of

$$n = Lq = (d + 1)(2d + 3)$$

symbols, with the same number of inputs and the same radius. The expansion is polynomial, and the lemma gives equivalence even when the permutation center is unrestricted. A Ulam algorithm in the excluded time would therefore solve these Closest String instances in the time excluded by the cited theorem. □

6 Concluding remarks

For one center, the upper and lower bounds determine the radius dependence up to constant factors in the exponent under ETH, with a linear factor in the explicit input size for the upper bound. For several centers, the remaining question is the best joint dependence on k and d. The present argument bounds the number of introduced references and moved symbols separately; the one-center lower bound alone does not settle the cost of coordinating many clusters.

References

[1] Tian Bai, Fedor V. Fomin, Petr A. Golovach, Yash Hiren More, and Simon Wietheger. Clustering permutations under the Ulam metric: A parameterized complexity study. In Sayan Bhattacharya, Danupon Nanongkai, Michael Benedikt, and Gabriele Puppis, editors, *53rd International Colloquium on Automata, Languages, and Programming (ICALP 2026)*, volume 374 of *Leibniz International Proceedings in Informatics (LIPIcs)*, Article 19, pages 19:1–19:23. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2026. https://doi.org/10.4230/LIPIcs.ICALP.2026.19.

[2] Tian Bai, Fedor V. Fomin, Petr A. Golovach, Yash Hiren More, and Simon Wietheger. Clustering permutations under the Ulam metric: A parameterized complexity study. Full version of the ICALP 2026 paper, *arXiv:2604.25734v1 [cs.DS]*, 2026. https://arxiv.org/abs/2604.25734.

[3] Christian Bachmaier, Franz J. Brandenburg, Andreas Gleißner, and Andreas Hofmeier. On the hardness of maximum rank aggregation problems. *Journal of Discrete Algorithms*, 31:2–13, 2015. https://doi.org/10.1016/j.jda.2014.10.002.

[4] Diptarka Chakraborty, Kshitij Gajjar, and Agastya Vibhuti Jha. Approximating the center ranking under Ulam. In *41st IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2021)*, volume 213 of *LIPIcs*, Article 12, pages 12:1–12:21, 2021. https://doi.org/10.4230/LIPIcs.FSTTCS.2021.12.

[5] David Aldous and Persi Diaconis. Longest increasing subsequences: From patience sorting to the Baik–Deift–Johansson theorem. *Bulletin of the American Mathematical Society*, 36(4):413–432, 1999. https://doi.org/10.1090/S0273-0979-99-00796-X.

[6] Nick Fischer and Mursalin Habib. An optimal algorithm for binary closest string. *arXiv:2605.31417v1 [cs.DS]*, 2026. https://arxiv.org/abs/2605.31417.

[7] Jens Gramm, Rolf Niedermeier, and Peter Rossmanith. Fixed-parameter algorithms for Closest String and related problems. *Algorithmica*, 37(1):25–42, 2003. https://doi.org/10.1007/s00453-003-1028-3.

[8] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Slightly superexponential parameterized problems. *SIAM Journal on Computing*, 47(3):675–702, 2018. https://doi.org/10.1137/16M1104834.

[9] Amir Abboud, MohammadHossein Bateni, Vincent Cohen-Addad, Karthik C. S., and Saeed Seddighin. On complexity of 1-center in various metrics. In *APPROX/RANDOM 2023*, volume 275 of *LIPIcs*, Article 1, 2023. https://doi.org/10.4230/LIPIcs.APPROX/RANDOM.2023.1.

[10] Nick Fischer, Elazar Goldenberg, Mursalin Habib, and Karthik C. S. Hardness of median and center in the Ulam metric. In *ESA 2025*, volume 351 of *LIPIcs*, pages 111:1–111:17, 2025. https://doi.org/10.4230/LIPIcs.ESA.2025.111.

[11] Sk Ruhul Azgor, Diptarka Chakraborty, Le Van Cuong, Debarati Das, Mursalin Habib, and Tien Long Nguyen. Ulam rank aggregation is hard to approximate for four rankings. *arXiv:2609.05874v1 [cs.CC]*, September 2026. https://arxiv.org/abs/2609.05874v1.