Necklaces, Convolutions, and $X + Y$

David Bremner[*]   Timothy M. Chan[†]   Erik D. Demaine[‡]

Jeff Erickson[§]   Ferran Hurtado[¶]   John Iacono[‖]

Stefan Langerman[**]   Ileana Streinu[††]   Perouz Taslakian[‡‡]

Abstract

We give subquadratic algorithms that, given two necklaces each with n beads at arbitrary positions, compute the optimal rotation of the necklaces to best align the beads. Here alignment is measured according to the ℓ_p norm of the vector of distances between pairs of beads from opposite necklaces in the best perfect matching. We show surprisingly different results for $p = 1$, $p = 2$, and $p = \infty$. For $p = 2$, we reduce the problem to standard convolution, while for $p = \infty$ and $p = 1$, we reduce the problem to (min, +) convolution and (median, +) convolution. Then we solve the latter two convolution problems in subquadratic time, which are interesting results in their own right. These results shed some light on the classic sorting $X + Y$ problem, because the convolutions can be viewed as computing order statistics on the antidiagonals of the $X + Y$ matrix. All of our algorithms run in $o(n^2)$ time, whereas the obvious algorithms for these problems run in $\Theta(n^2)$ time.

1 Introduction

How should we rotate two necklaces, each with n beads at different locations, to best align the beads? More precisely, each necklace is represented by a set of n points on the unit-circumference circle, and the goal is to find rotations of the necklaces, and a perfect matching between the beads of the two necklaces, that minimizes some norm of the circular distances between matched beads. In particular, the ℓ_1 norm minimizes the average absolute circular distance between matched beads, the ℓ_2 norm minimizes the average squared circular distance between matched beads, and the ℓ_∞ norm
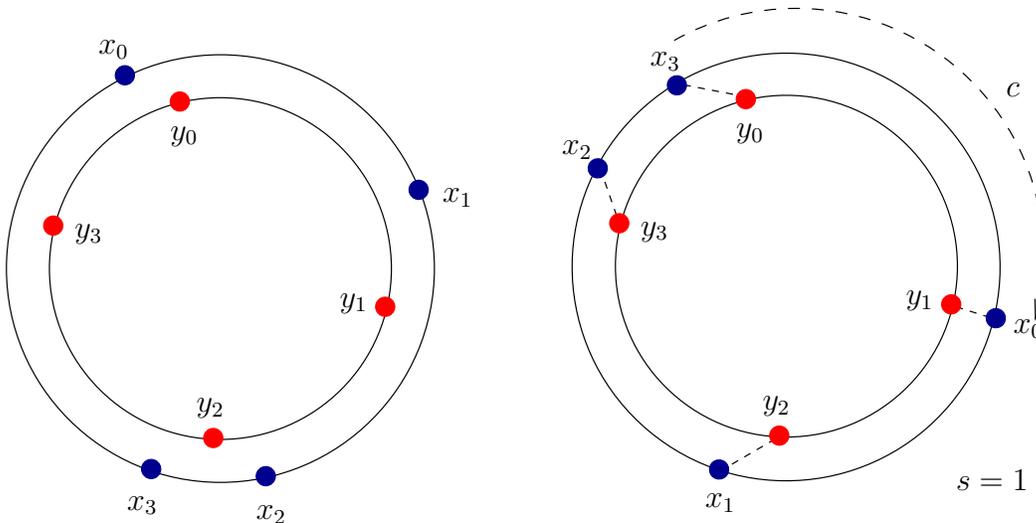
Figure 1: An example of necklace alignment: the input (left) and one possible output (right).

minimizes the maximum circular distance between matched beads. The ℓ_1 version of this necklace alignment problem was introduced by Toussaint [Tou04a] in the context of comparing rhythms in computational music theory, with applications to rhythm phylogeny [DBFG$^+$04, Tou04b].

Toussaint [Tou04a] gave a simple $O(n^2)$-time algorithm for ℓ_1 necklace alignment, and highlighted as an interesting open question whether the problem could be solved in $o(n^2)$ time. In this paper, we solve this open problem by giving $o(n^2)$-time algorithms for ℓ_1, ℓ_2, and ℓ_∞ necklace alignment, in both the standard real RAM model of computation and the less realistic nonuniform linear decision tree model of computation.

Our approach is based on reducing the necklace alignment problem to another important problem, convolution, for which we also obtain improved algorithms. The $(+, \cdot)$ convolution of two vectors $\mathbf{x} = \langle x_0, x_1, \ldots, x_{n-1} \rangle$ and $\mathbf{y} = \langle y_0, y_1, \ldots, y_{n-1} \rangle$, is the vector $\mathbf{x} * \mathbf{y} = \langle z_0, z_1, \ldots, z_{n-1} \rangle$ where $z_k = \sum_{i=0}^{k} x_i \cdot y_{k-i}$. While any $(\oplus, \odot)$ convolution can be computed in $O(n^2)$ time, the $(+, \cdot)$ convolution can be computed in $O(n \lg n)$ time using the Fast Fourier Transform [CT65, Gau66, HJB85], because the Fourier transform converts convolution into elementwise multiplication. Indeed, fast $(+, \cdot)$ convolution was one of the early breakthroughs in algorithms, with applications to polynomial and integer multiplication [Ber], batch polynomial evaluation [CLRS01, Problem 30-5], 3SUM [BDP05], string matching [FP74, Ind98, Kal02], matrix multiplication [CKSU05], and even juggling [CKL].

As we show in Theorems 1, 3, and 11, respectively, ℓ_2 necklace alignment reduces to standard $(+, \cdot)$ convolution, ℓ_∞ necklace alignment reduces to $(\min, +)$ [and $(\max, +)$] convolution, and ℓ_1 necklace alignment reduces to $(\mathrm{median}, +)$ convolution (whose kth entry is $\mathrm{median}_{i=0}^{k} (x_i - y_{k-i})$). The $(\min, +)$ convolution problem has appeared frequently in the literature, already appearing in Bellman's early work on dynamic programming in the early 1960s [BK62, FH04, Mar00, Mor70, Roc70, Str96]. Its name varies among "minimum convolution", "min-sum convolution", "inf-convolution", "infimal convolution", and "epigraphical sum".[1] To date, however, no worst-case $o(n^2)$-time algorithms for this convolution, or the more complex $(\mathrm{median}, +)$ convolution, has been obtained. In

---

[1] "Tropical convolution" would also make sense, by direct analogy with tropical geometry, but we have never seen this terminology used in print.

2

this paper, we develop worst-case $o(n^2)$-time algorithms for $(\min, +)$ and $(\text{median}, +)$ convolution, in the real RAM and the nonuniform linear decision tree.

Necklace alignment problem. More formally, in the *necklace alignment problem*, the input is a number p representing the ℓ_p norm, and two sorted vectors of n real numbers, $\mathbf{x} = \langle x_0, x_1, \ldots, x_{n-1} \rangle$ and $\mathbf{y} = \langle y_0, y_1, \ldots, y_{n-1} \rangle$, representing the two necklaces. See Figure 1. Canonically, we assume that each number x_i and y_i is in the range $[0, 1)$, representing a point on the unit-circumference circle (parameterized clockwise from some fixed point).

The optimization problem involves two parameters. The first parameter, the *offset* $c \in [0, 1)$, is the clockwise rotation angle of the first necklace relative to the second necklace. The second parameter, the *shift* $s \in \{0, 1, \ldots, n\}$, defines the perfect matching between beads: bead i of the first necklace matches with bead $(i + s) \bmod n$ of the second necklace. (Here we use the property that an optimal perfect matching between the beads does not cross itself.)

The goal of the ℓ_p necklace alignment problem is to find the offset $c \in [0, 1)$ and the shift $s \in \{0, 1, \ldots, n\}$ that minimize

$$\sum_{i=0}^{n-1} \left| x_i - y_{(i+s) \bmod n} + c \right|^p$$

or, in the case $p = \infty$, that minimize

$$\max_{i=0}^{n-1} \left| x_i - y_{(i+s) \bmod n} + c \right|.$$

Although not obvious from the definition, the ℓ_1, ℓ_2, and ℓ_∞ necklace alignment problems all have trivial $O(n^2)$ solutions. In each case, as we show, the optimal offset c can be computed in linear time for a given shift value s (sometimes even independent of s). The optimization problem is thus effectively over just $s \in \{0, 1, \ldots, n\}$, and the objective costs $O(n)$ time to compute for each s, giving an $O(n^2)$-time algorithm.

Related work. Although necklaces are studied throughout mathematics, mainly in combinatorial settings, we are not aware of any work on the necklace alignment problem before Toussaint [Tou04a]. He introduced ℓ_1 necklace alignment, calling it the *cyclic swap-distance* or *necklace swap-distance* problem, with a restriction that the beads lie at integer coordinates. Colannino et al. [CDH$^+$] consider some different distance measures between two sets of points on the real line in which the matching does not have to match every point. They do not, however, consider alignment under such distance measures.

The only subquadratic results for $(\min, +)$ convolution concern two special cases. First, the $(\min, +)$ convolution of two convex sequences or functions can be trivially computed in $O(n)$ time by a simple merge, which is the same as computing the Minkowski sum of two convex polygons. This special case is already used in image processing and computer vision. Second, Bussieck et al. [BHWZ94] proved that the $(\min, +)$ convolution of two *randomly permuted* sequences can be computed in $O(n \lg n)$ expected time. Our results are the first to improve the worst-case running time for $(\min, +)$ convolution.

Connections to $X + Y$. The necklace alignment problems, and their corresponding convolution problems, are also intrinsically connected to problems on $X + Y$ matrices. Given two lists of n numbers, $X = \langle x_0, x_1, \ldots, x_{n-1} \rangle$ and $Y = \langle y_0, y_1, \ldots, y_{n-1} \rangle$, $X + Y$ is the matrix of all pairwise sums, whose (i, j)th entry is $x_i + y_j$. A classic unsolved problem [DMO] is whether the entries of

$X + Y$ can be sorted in $o(n^2 \lg n)$ time. Fredman [Fre76a] showed that $O(n^2)$ comparisons suffice in the nonuniform linear decision tree model, but it remains open whether this can be converted into an $O(n^2)$-time algorithm in the real RAM model. Steiger and Streinu [SS95] gave a simple algorithm that takes $O(n^2 \log n)$ time while using only $O(n^2)$ comparisons.

The $(\min, +)$ convolution is equivalent to finding the minimum element in each antidiagonal of the $X + Y$ matrix, and similarly the $(\max, +)$ convolution finds the maximum element in each antidiagonal. We show that ℓ_∞ necklace alignment is equivalent to finding the antidiagonal of $X + Y$ with the smallest *range* (the maximum element minus the minimum element). The $(\text{median}, +)$ convolution is equivalent to finding the median element in each antidiagonal of the $X + Y$ matrix. We show that ℓ_1 necklace alignment is equivalent to finding the antidiagonal of $X + Y$ with the smallest *median cost* (the total distance between each element and the median of the elements). Given the apparent difficulty in sorting $X + Y$, it seems natural to believe that the minimum, maximum, and median elements of every antidiagonal cannot be found, and that the corresponding objectives cannot be minimized, any faster than $O(n^2)$ total time. Figure 2 shows a sample $X + Y$ matrix with the maximum element in each antidiagonal marked, with no apparent structure. Nonetheless, we show that $o(n^2)$ algorithms are possible.

Our results. In the standard real RAM model, we give subquadratic algorithms for the ℓ_1, ℓ_2, and ℓ_∞ necklace alignment problems, and for the $(\min, +)$ and $(\text{median}, +)$ convolution problems, using techniques of Chan [Cha05]. Despite the roughly logarithmic factor improvements for ℓ_1 and ℓ_∞, these results do not use word-level bit tricks of word-RAM fame.

1. $O(n \lg n)$-time algorithm on the real RAM for ℓ_2 necklace alignment (Section 2).

2. $O(n^2 / \lg n)$-time algorithm on the real RAM for ℓ_∞ necklace alignment and $(\min, +)$ convolution (Section 3).

3. $O(n^2 (\lg \lg n)^2 / \lg n)$-time algorithm on the real RAM for ℓ_1 necklace alignment and $(\text{median}, +)$ convolution (Section 4).

In the nonuniform linear decision tree model, we give particularly fast algorithms for the ℓ_1 and ℓ_∞ necklace alignment problems, using techniques of Fredman [Fre76a, Fre76b]:

4. $O(n \sqrt{n})$-time algorithm in the nonuniform linear decision tree model for ℓ_∞ necklace alignment and $(\min, +)$ convolution (Section 3).

5. $O(n \sqrt{n \lg n})$-time algorithm in the nonuniform linear decision tree model for ℓ_1 necklace alignment and $(\text{median}, +)$ convolution (Section 4).

(Although we state our results here in terms of $(\min, +)$ and $(\text{median}, +)$ convolution, the results below use $-$ instead of $+$ because of the synergy with necklace alignment.)

2 ℓ_2 Necklace Alignment

In this section, we show how ℓ_2 necklace alignment reduces to standard convolution, leading to an $O(n \lg n)$-time algorithm.

Theorem 1 *The ℓ_2 necklace alignment problem can be solved in $O(n \lg n)$ time on a real RAM.*

4
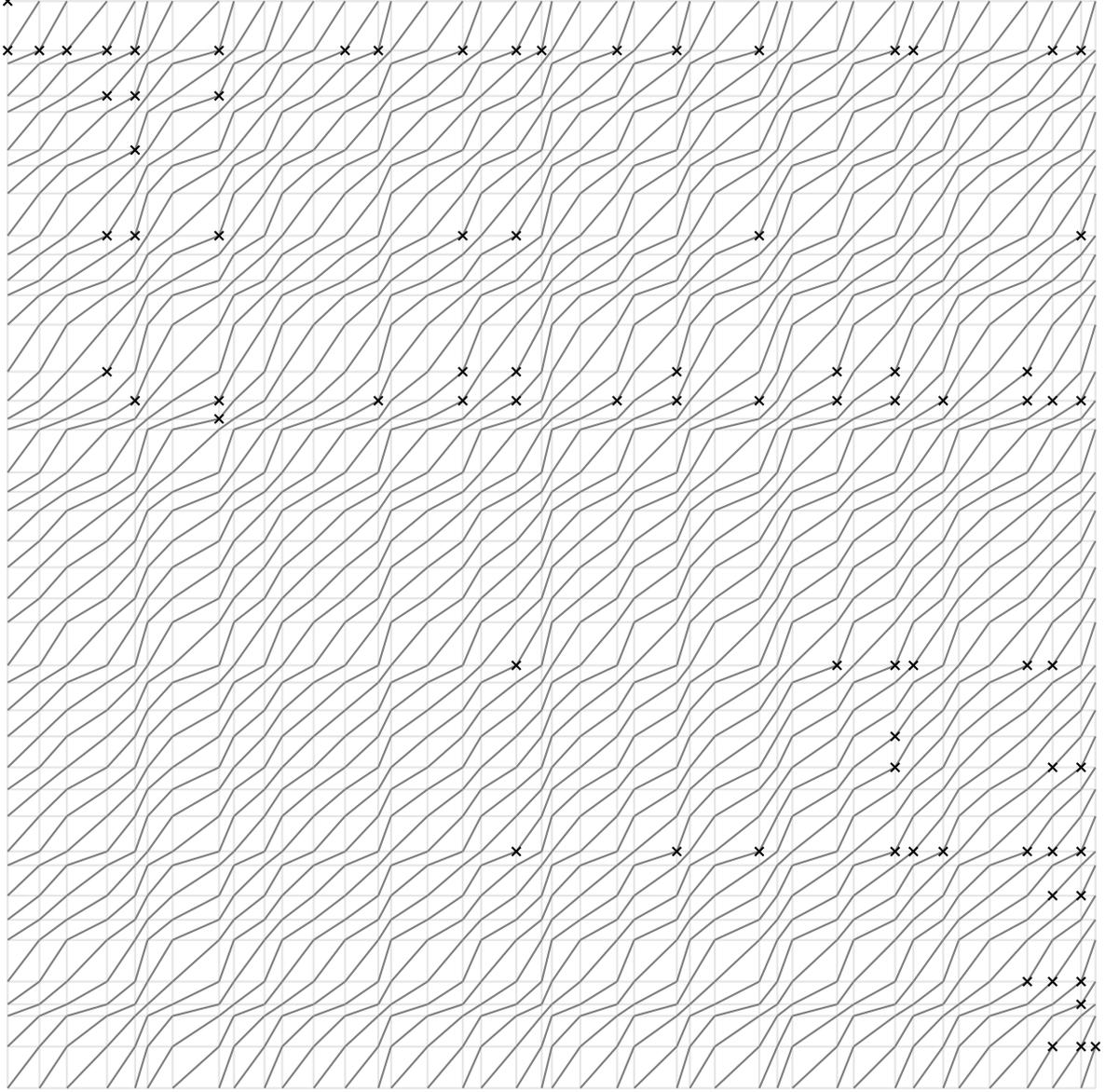
Figure 2: An $X + Y$ matrix with the maximum element in each antidiagonal marked with an $\times$.

Proof: The objective expands algebraically to

$$\sum_{i=0}^{n-1} \left(x_i - y_{(i+s) \bmod n} + c\right)^2 = \sum_{i=0}^{n-1} \left(x_i^2 + y_{(i+s) \bmod n}^2 + 2cx_i - 2cy_{(i+s) \bmod n} + c^2\right) - 2\sum_{i=0}^{n-1} x_i y_{(i+s) \bmod n}$$

$$= \sum_{i=0}^{n-1} \left(x_i^2 + y_i^2 + 2cx_i - 2cy_i + c^2\right) - 2\sum_{i=0}^{n-1} x_i y_{(i+s) \bmod n}$$

$$= \left[\sum_{i=0}^{n-1} \left(x_i^2 + y_i^2\right) + 2c\sum_{i=0}^{n-1} \left(x_i - y_i\right) + c^2\right] - 2\sum_{i=0}^{n-1} x_i y_{(i+s) \bmod n}.$$

5

The first term depends solely on the inputs and the variable c, while the second term depends solely on the inputs and the variable s. Thus the two terms can be optimized separately. The first term can be optimized in $O(n)$ time using the quadratic formula. The second term can be computed, for each $s \in \{0, 1, \ldots, n-1\}$, in $O(n \lg n)$ time using $(+, \cdot)$ convolution (and therefore optimized in the same time). Specifically, define the vectors

$$\mathbf{x}' = \langle x_0, x_1, \ldots, x_{n-1}, \underbrace{0, 0, \ldots, 0}_{n} \rangle,$$

$$\mathbf{y}' = \langle y_{n-1}, y_{n-2}, \ldots, y_0, y_{n-1}, y_{n-2}, \ldots, y_0 \rangle.$$

Then, for $s' \in \{0, 1, \ldots, n-1\}$, the $(n+s')$th entry of the convolution $\mathbf{x}' * \mathbf{y}'$ is

$$\sum_{i=0}^{n+s'} x_i' y_{n+s'-i}' = \sum_{i=0}^{n-1} x_i y_{n+s'-i}' = \sum_{i=0}^{n-1} x_i y_{[n-(n+s'-i)] \bmod n} = \sum_{i=0}^{n-1} x_i y_{(i-s') \bmod n},$$

which is the desired entry where $s' = n - s$. We can compute the entire convolution in $O(n \lg n)$ time using the Fast Fourier Transform. □

3 ℓ_∞ Necklace Alignment

First we show the relation between ℓ_∞ necklace alignment and $(\min, -)$ convolution. We need the following basic fact:

Fact 2 *For any vector $\mathbf{z} = \langle z_0, z_1, \ldots, z_{n-1} \rangle$, the minimum value of $\max_{i=0}^{n-1} |z_i + c|$ is*

$$\frac{1}{2} \left(\max_{i=0}^{n-1} z_i - \min_{i=0}^{n-1} z_i \right),$$

which is achieved when $c = \frac{1}{2} \left(\min_{i=0}^{n-1} z_i + \max_{i=0}^{n-1} z_i \right)$.

Theorem 3 *The ℓ_∞ necklace alignment problem can be reduced in $O(n)$ time to one $(\min, -)$ convolution and one $(\max, -)$ convolution.*

Proof: For two necklaces $\mathbf{x}$ and $\mathbf{y}$, we apply the $(\min, -)$ convolution to the following vectors:

$$\mathbf{x}' = \langle x_0, x_1, \ldots, x_{n-1}, \underbrace{0, 0, \ldots, 0}_{n} \rangle,$$

$$\mathbf{y}' = \langle y_{n-1}, y_{n-2}, \ldots, y_0, y_{n-1}, y_{n-2}, \ldots, y_0 \rangle.$$

Then, for $s' \in \{0, 1, \ldots, n-1\}$, the $(n+s')$th entry of $\mathbf{x}' \overset{-}{\underset{\min}{*}} \mathbf{y}'$ is

$$\min_{i=0}^{n+s'}(x_i' - y_{n+s'-i}') = \min_{i=0}^{n-1}(x_i - y_{n+s'-i}') = \min_{i=0}^{n-1}(x_i - y_{[n-(n+s'-i)] \bmod n}) = \min_{i=0}^{n-1}(x_i - y_{(i-s') \bmod n}),$$

which is $\min_{i=0}^{n-1}(x_i - y_{(i+s) \bmod n})$ if we let $s' = n - s$. By symmetry, we can compute the $(\max, -)$ convolution $\mathbf{x}' \overset{-}{\underset{\max}{*}} \mathbf{y}'$ and use it to compute $\max_{i=0}^{n-1}(x_i - y_{(i+s) \bmod n})$ for each $s \in \{0, 1, \ldots, n-1\}$. Applying Fact 2, we can therefore minimize $\max_{i=0}^{n-1} |x_i - y_{(i+s) \bmod n} + c|$ over c, for each $s \in \{0, 1, \ldots, n-1\}$. By brute force, we can minimize over s as well using $O(n)$ additional comparisons and time. □

For our nonuniform linear decision tree results, we use the main theorem of Fredman's work on sorting $X + Y$:

Theorem 4 [Fre76a] *For any fixed set Γ of permutations of N elements, there is a comparison tree of depth $O(N + \lg |\Gamma|)$ that sorts any sequence whose rank permutation belongs to Γ.*

Theorem 5 *The $(\min, -)$ convolution of two vectors of length n can be computed in $O(n\sqrt{n})$ time in the nonuniform linear decision tree model.*

Proof: Let $\mathbf{x}$ and $\mathbf{y}$ denote the two vectors of length n, and let $\mathbf{x} \mathbin{\overset{-}{\underset{\min}{*}}} \mathbf{y}$ denote their $(\min, -)$ convolution, whose kth entry is $\min_{i=0}^{k} (x_i - y_{k-i})$.

First we sort the set $D = \{x_i - x_j, y_i - y_j : |i - j| \leq d\}$ of pairwise differences between nearby x_i's and nearby y_i's, where $d \leq n$ is a value to be determined later. This set D has $N = O(nd)$ elements. The possible sorted orders of D correspond to cells in the arrangement of hyperplanes in $\mathbb{R}^{2n}$ induced by all N^2 possible comparisons between elements in the set, and this hyperplane arrangement has $O(N^{4n})$ cells. By Theorem 4, there is a comparison tree sorting D of depth $O(N + n \lg N) = O(nd + n \lg n)$.

The comparisons we make to sort D enable us to compare $x_i - y_{k-i}$ versus $x_j - y_{k-j}$ for free, provided $|i - j| \leq d$, because $x_i - y_{k-i} < x_j - y_{k-j}$ precisely if $x_i - x_j < y_{k-i} - y_{k-j}$. Thus, in particular, we can compute

$$M_k(\lambda) = \min \left\{ x_i - y_{k-i} \ \middle| \ i = \lambda, \lambda + 1, \ldots, \min\{\lambda + d, n\} - 1 \right\}$$

for free (using the outcomes of the comparisons we have already made).

We can rewrite the kth entry $\min_{i=0}^{k}(x_i - y_{k-i})$ of $\mathbf{x} \mathbin{\overset{-}{\underset{\min}{*}}} \mathbf{y}$ as $\min\{M_k(0), M_k(d), M_k(2d), \ldots, M_k(\lceil k/d \rceil d)\}$, and thus we can compute it in $O(k/d) = O(n/d)$ comparisons between differences. Therefore all n entries can be computed in $O(nd + n \lg n + n^2/d)$ total time.

This asymptotic running time is minimized when $nd = \Theta(n^2/d)$, i.e., when $d^2 = \Theta(n)$. Substituting $d = \sqrt{n}$, we obtain a running time of $O(n\sqrt{n})$ in the nonuniform linear decision tree model. $\qquad\square$

Combining Theorems 3 and 5, we obtain the following result:

Corollary 6 *The ℓ_∞ necklace alignment problem can be solved in $O(n\sqrt{n})$ time in the nonuniform linear decision tree model.*

Our results on the real RAM use the following geometric lemma from Chan's work on all-pairs shortest paths:

Lemma 7 [Cha05, Lemma 2.1] *Given n points $p_1, p_2, \ldots, p_n$ in d dimensions, each colored either red or blue, we can find the P pairs (p_i, p_j) for which p_i is red, p_j is blue, and p_i dominates p_j (i.e., for all k, the kth coordinate of p_i is at least the kth coordinate of p_j), in $2^{O(d)} n^{1+\varepsilon} + O(P)$ time.*

Theorem 8 *The $(\min, -)$ convolution of two vectors of length n can be computed in $O(n^2/\lg n)$ time on a real RAM.*

Proof: Let $\mathbf{x}$ and $\mathbf{y}$ denote the two vectors of length n, and let $\mathbf{x} \mathbin{\overset{-}{\underset{\min}{*}}} \mathbf{y}$ denote their $(\min, -)$ convolution. For each $i \in \{0, 1, \ldots, n\}$, we define the d-dimensional points

$$\begin{aligned}
p_i &= (x_i - x_{i+1}, \ x_i - x_{i+2}, \ \ldots, \ x_i - x_{\min\{i+d,n\}-1}), \\
q_i &= (y_i - y_{i+1}, \ y_i - y_{i+2}, \ \ldots, \ y_i - y_{\min\{i+d,n\}-1}),
\end{aligned}$$

7

We apply Lemma 7 to this set of points, where the p_i's are defined to be red and the q_j's are defined to be blue, to obtain all dominating pairs (p_i, q_j).

Point p_i dominates $q_{j'}$ precisely if $x_i - x_{i+\delta} \geq y_{j'} - y_{j'+\delta}$ for all $\delta \in \{1, 2, \ldots, d-1\}$ (ignoring the indices beyond $n-1$). By re-arranging terms, this condition is equivalent to $x_i - y_{j'} \geq x_{i+\delta} - y_{j'+\delta}$ for all $\delta \in \{1, 2, \ldots, d-1\}$, i.e., $x_i - y_{j'} = \max_{\delta=1}^{d-1}(x_{i+\delta} - y_{j'+\delta})$. If we substitute $j' = k - j$, we obtain that (p_i, q_{k-j}) is a dominating pair precisely if $x_i - y_{k-j} = \max_{\delta=1}^{d-1}(x_{i+\delta} - y_{k-j+\delta})$. Thus, there can be at most $O(n^2/d)$ such pairs, because if (x_i, y_{k-j}) is a dominating pair, then $(x_{i+\delta}, y_{k-j+\delta})$ cannot be a dominating pair for $\delta \in \{1, \ldots, d-1\}$. (Here we assume that the max is achieved uniquely, which can be arranged by standard perturbation techniques or by breaking ties consistently [Cha05].) Hence, the running time of Lemma 7 is $2^{O(d)}n^{1+\varepsilon} + O(n^2/d)$ time, which is $O(n^2/\lg n)$ if we choose $d = \varepsilon \lg n$ for a sufficiently small constant $\varepsilon > 0$. Also, the dominating pairs allow us to compute

$$M_k(\lambda) = \max \left\{ x_i - y_{k-i} \; \middle| \; i = \lambda, \lambda + 1, \ldots, \min\{\lambda + d, n\} - 1 \right\}$$

for each λ. We can rewrite the kth entry $\max_{i=0}^{k}(x'_i - y'_{k-i})$ of $\mathbf{x} \underset{\max}{*} \mathbf{y}$ as $\max\{M_k(0), M_k(d), M_k(2d), \ldots, M_k(\lceil k/d \rceil d)\}$, and thus we can compute it in $O(k/d) = O(n/d)$ time. Therefore all n entries can be computed in $O(n^2/d) = O(n^2/\lg n)$ time on a real RAM. $\qquad\square$

Combining Theorems 3 and 8, we obtain the following result:

Corollary 9 *The ℓ_∞ necklace alignment problem can be solved in $O(n^2/\lg n)$ time on a real RAM.*

This approach likely cannot be improved beyond $O(n^2/\lg n)$. Such an improvement would require an improvement to Lemma 7, which would in turn improve the fastest known algorithm for all-pairs shortest paths in dense graphs, the $O(n^3/\lg n)$-time algorithm of [Cha05].

4 ℓ_1 Necklace Alignment

First we show the relation between ℓ_∞ necklace alignment and (median, $-$) convolution. We need the following basic fact:

Fact 10 *For any vector $\mathbf{z} = \langle z_0, z_1, \ldots, z_{n-1} \rangle$, $\sum_{i=0}^{n-1} |z_i + c|$ is minimized when $c = \operatorname*{median}\limits_{i=0}^{n-1} z_i$.*

Theorem 11 *The ℓ_1 necklace alignment problem can be reduced in $O(n)$ time to one (median, $-$) convolution.*

Proof: For two necklaces $\mathbf{x}$ and $\mathbf{y}$, we apply the (median, $-$) convolution to the following vectors, as in the proof of Theorem 3:

$$\mathbf{x}' = \langle x_0, x_1, \ldots, x_{n-1}, \underbrace{0, 0, \ldots, 0}_{n} \rangle,$$

$$\mathbf{y}' = \langle y_{n-1}, y_{n-2}, \ldots, y_0, y_{n-1}, y_{n-2}, \ldots, y_0 \rangle.$$

Then, for $s' \in \{0, 1, \ldots, n-1\}$, the $(n+s')$th entry of $\mathbf{x}' \underset{\text{med}}{\overline{*}} \mathbf{y}'$ is

$$\operatorname*{med}_{i=0}^{n+s'}(x'_i - y'_{n+s'-i}) = \operatorname*{med}_{i=0}^{n-1}(x_i - y'_{n+s'-i}) = \operatorname*{med}_{i=0}^{n-1}(x_i - y_{[n-(n+s'-i)] \bmod n}) = \operatorname*{med}_{i=0}^{n-1}(x_i - y_{(i-s') \bmod n}),$$

8

which is $\mathrm{median}_{i=0}^{n-1}(x_i - y_{(i+s)\bmod n})$ if we let $s' = n - s$. Applying Fact 10, we can therefore minimize $\mathrm{median}_{i=0}^{n-1}|x_i - y_{(i+s)\bmod n} + c|$ over c, for each $s \in \{0, 1, \ldots, n-1\}$. By brute force, we can minimize over s as well using $O(n)$ additional comparisons and time. $\qquad\square$

Our results for (median, $-$) convolution use the following result of Frederickson and Johnson:

Theorem 12 [FJ82] *The median element of the union of k sorted lists, each of length n, can be computed in $O(k \lg n)$ time and comparisons.*

We begin with our results for the nonuniform linear decision tree model:

Theorem 13 *The* (median, $-$) *convolution of two vectors of length n can be computed in $O(n\sqrt{n \lg n})$ time in the nonuniform linear decision tree model.*

Proof: As in the proof of Theorem 6, we sort the set $D = \{x_i - x_j, y_i - y_j : |i - j| \le d\}$ of pairwise differences between nearby x_i's and nearby y_i's, where $d \le n$ is a value to be determined later. By Theorem 4, this step requires $O(nd + n \lg n)$ comparisons between differences. These comparisons enable us to compare $x_i - y_{k-i}$ versus $x_j - y_{k-j}$ for free, provided $|i - j| \le d$, because $x_i - y_{k-i} < x_j - y_{k-j}$ precisely if $x_i - x_j < y_{k-i} - y_{k-j}$. In particular, we can sort each list

$$L_k(\lambda) = \Big\langle x_i - y_{k-i} \;\Big|\; i = \lambda, \lambda + 1, \ldots, \min\{\lambda + d, n\} - 1 \Big\rangle$$

for free. By Theorem 12, we can compute the median of $L_k(0) \cup L_k(d) \cup L_k(2d) \cup \cdots \cup L_k(\lceil k/d \rceil d)$, i.e., $\mathrm{median}_{i=0}^{k}(x_i - y_{k-i})$, in $O((k/d) \lg d) = O((n/d) \lg d)$ comparisons. Also, in the same asymptotic number of comparisons, we can binary search to find where the median fits in each of the $L_k(\lambda)$ lists, and therefore which differences are smaller and which differences are larger than the median. This median is the kth entry of $\mathbf{x} \overset{-}{\underset{\mathrm{med}}{*}} \mathbf{y}$. Therefore, we can compute all n entries of $\mathbf{x} \overset{-}{\underset{\mathrm{med}}{*}} \mathbf{y}$ in $O(nd + n \lg n + (n^2/d) \lg d)$ comparisons. This asymptotic running time is minimized when $nd = \Theta((n^2/d) \lg d)$, i.e., when $d^2/\lg d = \Theta(n)$. Substituting $d = \sqrt{n \lg n}$, we obtain a running time of $O(n\sqrt{n \lg n})$ in the nonuniform linear decision tree model. $\qquad\square$

Combining Theorems 11 and 13, we obtain the following result:

Corollary 14 *The ℓ_1 necklace alignment problem can be solved in $O(n\sqrt{n \lg n})$ time in the nonuniform linear decision tree model.*

Now we turn to the analogous results for the real RAM:

Theorem 15 *The* (median, $-$) *convolution of two vectors of length n can be computed in $O(n^2 (\lg \lg n)^2 / \lg n)$ time on a real RAM.*

Proof: Let $\mathbf{x}$ and $\mathbf{y}$ denote the two vectors of length n, and let $\mathbf{x} \overset{-}{\underset{\mathrm{med}}{*}} \mathbf{y}$ denote their (median, $-$) convolution. For each permutation π on the set $\{0, 1, \ldots, d-1\}$, and for each $i \in \{0, d, 2d, \ldots, \lfloor n/d \rfloor d\}$, we define the d-dimensional points

$$
\begin{aligned}
p_{\pi,i} &= \big(x_{i+\pi(0)} - x_{i+\pi(1)}, \; x_{i+\pi(1)} - x_{i+\pi(2)}, \; \ldots, \; x_{i+\pi(\min\{d,n-i\}-2)} - x_{i+\pi(\min\{d,n-i\}-1)}\big), \\
q_{\pi,i} &= \big(y_{i-\pi(0)} - y_{i-\pi(1)}, \; y_{i-\pi(1)} - y_{i-\pi(2)}, \; \ldots, \; y_{i-\pi(\min\{d,n-i\}-2)} - y_{i-\pi(\min\{d,n-i\}-1)}\big),
\end{aligned}
$$

9

For each permutation π, we apply Lemma 7 to the set of red points $\{p_{\pi,i} : i\}$ and the set of blue points $\{q_{\pi,j} : j\}$, to obtain all dominating pairs $(p_{\pi,i}, q_{\pi,j})$.

Point $p_{\pi,i}$ dominates $q_{\pi,j'}$ precisely if $x_{i+\pi(\delta)} - x_{i+\pi(\delta+1)} \geq y_{j'-\pi(\delta)} - y_{j'-\pi(\delta+1))}$ for all $\delta \in \{0, 1, \ldots, d-2\}$ (ignoring the indices beyond $n-1$). By re-arranging terms, this condition is equivalent to $x_{i+\pi(\delta)} - y_{j'-\pi(\delta)} \geq x_{i+\pi(\delta+1)} - y_{j'-\pi(\delta+1)}$ for all $\delta \in \{0, 1, \ldots, d-2\}$, i.e., π is a sorting permutation of $\langle x_i - y_{j'}, x_{i+1} - y_{j'-1}, \ldots, x_{i+d-1} - y_{j'-d+1}\rangle$. If we substitute $j' = k - j$, we obtain that $(p_{\pi,i}, q_{\pi,k-j})$ is a dominating pair precisely if π is a sorting permutation of the list $L_k(i) = \langle x_i - y_{k-j}, x_{i+1} - y_{k-j+1}, \ldots, x_{\min\{i+d,n\}-1} - y_{\min\{k-j+d,n\}-1}\rangle$. Thus, there can be at most $O(n^2/d^2)$ total dominating pairs over all i, j, π, because there are $O(n/d)$ choices for each of i and j, and if $(x_{\pi,i}, y_{\pi,k-j})$ is a dominating pair, then $(x_{\pi',i}, y_{\pi',k-j})$ cannot be a dominating pair for any permutation $\pi' \neq \pi$. (Here we assume that the sorted order is unique, which can be arranged by standard perturbation techniques or by breaking ties consistently [Cha05].) Hence, the running time of the $d!$ executions of Lemma 7 is $d!\, 2^{O(d)} n^{1+\varepsilon} + O(n^2/d^2)$ time, which is $O(n^2 (\lg \lg n)^2 / \lg^2 n)$ if we choose $d = \varepsilon \lg n / \lg \lg n$ for a sufficiently small constant $\varepsilon > 0$. By Theorem 12, we can compute the median of $L_k(0) \cup L_k(d) \cup L_k(2d) \cup \cdots \cup L_k(\lceil k/d \rceil d)$, i.e., $\operatorname*{median}\limits_{i=0}^{k}(x_i - y_{k-i})$, in $O((k/d) \lg d) = O((n/d) \lg d)$ comparisons. Also, in the same asymptotic number of comparisons, we can binary search to find where the median fits in each of the $L_k(\lambda)$ lists, and therefore which differences are smaller and which differences are larger than the median. This median is the kth entry of $\mathbf{x} \overline{\underset{\mathrm{med}}{*}} \mathbf{y}$. Therefore all n entries can be computed in $O(n^2 (\lg d)/d) = O(n^2 (\lg \lg n)^2 / \lg n)$ time on a real RAM. $\qquad\square$

Combining Theorems 11 and 15, we obtain the following result:

Corollary 16 *The ℓ_1 necklace alignment problem can be solved in $O(n^2 (\lg \lg n)^2 / \lg n)$ time on a real RAM.*

As before, this approach likely cannot be improved beyond $O(n^2/\lg n)$, because such an improvement would require an improvement to Lemma 7, which would in turn improve the fastest known algorithm for all-pairs shortest paths in dense graphs [Cha05].

5 Conclusion

It would be interesting to explore whether the structural information extracted by our algorithms about the $X + Y$ problem could be used to devise a $o(n^2 \lg n)$ algorithm for $X + Y$. We believe that any further improvements to our algorithms for necklace problems would require progress and/or have interesting implications on all-pairs shortest paths [Cha05].

Acknowledgments

References

[BDP05]   Ilya Baran, Erik D. Demaine, and Mihai Pătraşcu. Subquadratic algorithms for 3SUM. In *Proceedings of the 9th Workshop on Algorithms and Data Structures (WADS 2005)*, volume 3608 of *Lecture Notes in Computer Science*, pages 409–421, Waterloo, Ontario, Canada, August 15–17 2005.

[Ber]   Daniel J. Bernstein. Fast multiplication and its applications. In J. Buhler and P. Stevenhagen, editors, *Algorithmic Number Theory*. Cambridge University Press. To appear.

[BHWZ94]   Michael Bussieck, Hannes Hassler, Gerhard J. Woeginger, and Uwe T. Zimmermann. Fast algorithms for the maximum convolution problem. *Operations Research Letters*, 15(3):133–141, 1994.

[BK62]   Richard Bellman and William Karush. Mathematical programming and the maximum transform. *Journal of the Society for Industrial and Applied Mathematics*, 10(3):550–567, 1962.

[CDH+]   Justin Colannino, Mirela Damian, Ferran Hurtado, John Iacono, Henk Meijer, Suneeta Ramaswami, and Godfried Toussaint. An $O(n \log n)$-time algorithm for the restriction scaffold assignment. *Journal of Computational Biology*. To appear.

[Cha05]   Timothy M. Chan. All-pairs shortest paths with real weights in $O(n^3/\log n)$ time. In *Proceedings of the 9th International Workshop on Algorithms and Data Structures*, volume 3608 of *Lecture Notes in Computer Science*, pages 318–324, Waterloo, Canada, August 2005.

[CKL]   Jean Cardinal, Steve Kremer, and Stefan Langerman. Juggling with pattern matching. *Theory of Computing Systems*. To appear.

[CKSU05]   Henry Cohn, Robert Kleinberg, Balazs Szegedy, and Christopher Umans. Group-theoretic algorithms for matrix multiplication. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 379–388, 2005.

[CLRS01]   Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, second edition, 2001.

[CT65]   James W. Cooley and John W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19:297–301, 1965.

[DBFG+04]   J. Miguel Díaz-Báñez, Giovanna Farigu, Francisco Gómez, David Rappaport, and Godfried T. Toussaint. El compás flamenco: A phylogenetic analysis. In *Proceedings of BRIDGES: Mathematical Connections in Art, Music, and Science*, pages 61–70, Winfield, Kansas, July–August 2004.

[DMO]   Erik D. Demaine, Joseph S. B. Mitchell, and Joseph O'Rourke. Problem 41: Sorting $X + Y$ (partial sums). In *The Open Problems Project*. http://cs.smith.edu/~orourke/TOPP/P41.html.

[FH04]   Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Distance transforms of sampled functions. Technical Report TR2004-1963, Cornell Computing and Information Science, September 2004.

[FJ82]   Greg N. Frederickson and Donald B. Johnson. The complexity of selection and ranking in $x + y$ and matrices with sorted columns. *Journal of Computer and System Sciences*, 24(2):197–208, 1982.

[FP74]   Michael J. Fischer and Michael S. Paterson. String-matching and other products. In *Complexity of computation*, volume VII of *SIAM-AMS Proceedings*, pages 113–125. American Mathematical Society, 1974. Proceedings of the SIAM-AMS Applied Mathematics Symposium, New York, 1973.

[Fre76a]   Michael L. Fredman. How good is the information theory bound in sorting? *Theoretical Computer Science*, 1(4):355–361, 1976.

[Fre76b]   Michael L. Fredman. New bounds on the complexity of the shortest path problem. *SIAM Journal on Computing*, 5(1):83–89, 1976.

[Gau66]   Carl F. Gauss.   *Werke*, volume 3.   Königlichen Gesellschaft der Wissenschaften, Göttingen, 1866.

[HJB85]   Michael T. Heideman, Don H. Johnson, and C. Sidney Burrus. Gauss and the history of the fast Fourier transform. *Archive for History of Exact Sciences*, 34(3):265–277, 1985.

[Ind98]   Piotr Indyk. Faster algorithms for string matching problems: Matching the convolution bound. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, pages 166–173, Palo Alto, California, November 1998.

[Kal02]   Adam Kalai. Efficient pattern-matching with don't cares. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete algorithms*, pages 655–656, San Francisco, California, 2002.

[Mar00]   Petros Maragos. Differential morphology. In S. Mitra and G. Sicuranza, editors, *Nonlinear Image Processing*, chapter 10, pages 289–329. Academic Press, 2000.

[Mor70]   Jean-Jacques Moreau.   Inf-convolution, sous-additivité, convexité des fonctions numériques. *Journal de Mathématiques Pures et Appliquées, Neuvième Série*, 49:109–154, 1970.

[Roc70]   R. Tyrrell Rockafellar. *Convex analysis*. Number 28 in Princeton Mathematical Series. Princeton University Press, 1970.

[SS95]   William L. Steiger and Ileana Streinu. A pseudo-algorithmic separation of lines from pseudo-lines. *Information Processing Letters*, 53(5):295–299, 1995.

[Str96]   Thomas Strömberg. The operation of infimal convolution. *Dissertationes Mathematicae (Rozprawy Matematyczne)*, 352:58, 1996.

[Tou04a]   Godfried Toussaint. The geometry of musical rhythm. In *Revised Papers from the Japan Conference on Discrete and Computational Geometry*, volume 3742 of *Lecture Notes in Computer Science*, pages 198–212, Tokyo, Japan, 2004.

[Tou04b]   Godfried T. Toussaint. A comparison of rhythmic similarity measures. In *Proceedings of the 5th International Conference on Music Information Retrieval*, pages 242–245, Barcelona, Spain, October 2004. A longer version appears as Technical Report SOCS-TR-2004.6, School of Computer Science, McGill University, August 2004.