Vol. 35 No. 12 Transactions of Information Processing Society of Japan Dec. 1994

Regular Paper

The Relation between a Polygon Containment Problem and the Problem of Sorting $X+Y$

ANTONIO HERNÁNDEZ BARRERA†

The polygon containment problem is the problem of deciding whether a given polygon P which is allowed to have some kinds of motions, can be placed into another fixed Q. We show that the polygon containment problem in the case of rectilinearly convex polygons under translation, the problem of sorting $X+Y$ and the problem of sorting sums of consecutive numbers are equivalent.

1. Introduction

1.1 An Overview of Polygon Containment Problems

The polygon containment problem is the problem of deciding whether a given polygon P, which is allowed to have some kinds of motions, can be placed into another fixed Q. Here we briefly mention some of the results obtained for it.

Suppose P and Q are m-gon and n-gon respectively; P can move while Q remains fixed. Assuming only translations, an algorithm that runs in $O(mn \log m)$ when both P and Q are rectilinearly convex polygons appeared in Ref. 3). Also under translation, the problem considering P convex and Q any simple polygon was solved in Ref. 4) using $O(mn \log mn)$. For both P and Q being not necessarily convex and even not connected, Avnaim and Boinnonnat proposed in Ref. 2) an $O(m^2n^2 \log mn)$ algorithm and showed that the method can be generalized to an $O(m^3n^3 \log mn)$ procedure when rotations are also possible. All of these algorithms compute the whole feasible region. Applications where knowing the whole feasible region may be necessary, were mentioned in Ref. 3) and Ref. 4).

1.2 What This Paper Is About

If the size of the output in each specific case is taken into account, it might be said that all of the above-mentioned results are nearly optimal. For example, when both P and Q are rectilinearly convex polygons, it was proven in Ref. 3) that the boundary of the set of feasible placements of P inside Q is a rectilinearly convex polygonal region that could reach $\Omega(mn)$ edges, while the algorithm used there to obtain that region is $O(mn \log m)$.

The results we present here show that the difficulty of computing the feasible region in the PCP (we will use "PCP" as an abbreviation for "polygon containment problem") under translation when both polygons are rectilinearly convex ones is closely related to the difficulty of sorting sets of numbers of the form $X+Y$. More precisely, we prove that these problems are equivalent.

Sorting $X+Y$, where X and Y are the sets of real numbers $(x_i)_{1\le i\le n}$ and $(y_j)_{1\le j\le m}$ respectively, consists of sorting the sums $(x_i+y_j)_{1\le i\le n,\ 1\le j\le m}$. It has been studied before (Ref. 5), Ref. 6) and Ref. 7)) but the question of how much computation time is really needed is still open, i. e. an optimal $\theta(mn)$ has not been found.

That equivalence relation means that any algorithm which solves one problem in $\theta(mn)$ can be transformed into a corresponding algorithm for solving the other one within the same bound. In fact, our proof consists of giving these transformations.

By using that relation and the results in Ref. 7), we also prove here that the PCP under translation in the case of rectilinearly convex polygons and the problem of sorting the sums of consecutive numbers

$$\left\{\sum_{k=i}^{j} a_k,\ 1\le i\le j\le n\right\},$$

$(a_i)_{1\le i\le n}$ any sequence of real numbers, are equivalent when $m=n$.

2. Preliminaries

2.1 Reductions

When we say in this paper that a problem P

† Department of Mathematics, Faculty of Science, Hiroshima University

reduces to another Q, $P \longrightarrow Q$, we mean that an $O(nm)+T(n, m)$ ($n+m$ is the size of all of the problems we will discuss as can be seen in subsection 2.2) algorithm for solving P exists, where $T(n, m)$ is the time complexity of Q. P is *equivalent* to Q, $P \longleftrightarrow Q$, if $P \longrightarrow Q$ and $Q \longrightarrow P$.

2.2 General Definitions and Notations

In what follows, we denote the abscissae and the ordinate of a point p as $p.x$ and $p.y$ respectively.

A polygon P is *rectilinear* if the edges of its boundary are either vertical or horizontal. P is *rectilinearly convex* if it is rectilinear and the intersection of every horizontal or vertical line with P is a connected (possibly empty) segment. See **Fig. 1.**

Suppose P and Q are rectilinearly convex m-gon and n-gon respectively. Q is fixed in the coordinate system with origin O_q while P is in the coordinate system with origin O_p which can translate.

Problem P1 Find all placements of O_p in the fixed coordinate system so that P is contained in Q.

It was proven in Ref. 3) that the set of placements in which P is contained in Q is a rectilinearly convex polygon with at most nm bounding edges. That set of placements will be denoted here by $H(P, Q)$ and it will be always assumed that it is described by its vertices given in some order.

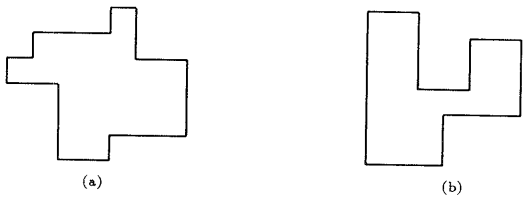


Fig. 1 (a) Rectilinearly convex.
(b) Nonrectilinearly convex.

Let us think of A and B as staircase polygonal lines as in **Fig. 2.** Suppose that A (B) refers to the origin O_A (O_B). We will consider that B and its coordinate system are fixed and A with its coordinate system can translate. We suppose that the bottommost edge of B (the topmost edge of A) is horizontal and its left (right) extreme is at some point in the infinite. On the other hand, the rightmost edge of B (the leftmost edge of A) is vertical and its upper (lower) extreme is also at some point in the infinite.

We assume that B is represented by vertices $b_1, b_2, \cdots, b_n$ and A by vertices $a_1, a_2, \cdots, a_m$ as in Fig. 2.

Problem P2 Find the vertices of the rectilinear polygonal line that O_A would describe in the fixed coordinate system as A slides along the edges of B.

In other words, we want a description of all the positions that O_A would reach if, beginning at some point in the infinite with the topmost edge of A "touching" the bottommost edge of B, A translates to the right until it is possible to go upward without intersecting B, then A translates in that direction until the distance between some pair of horizontal segments is zero, etc. Notice that in essence, what we want is to solve a polygon containment problem in which both regions are not bounded. So we can call that rectilinear polygonal line $H(A, B)$.

If O_A is placed at a position O in the system O_B, the coordinates of the vertex a_j ($1 \leq j \leq m$) in the fixed coordinate system are $(O.x + a_j.x, O.y + a_j.y)$ that we will denote as a_j^o.

Problem P3 Sort $X+Y$, where X and Y are the sets of real numbers $(x_i)_{1 \leq i \leq n}$ and $(y_j)_{1 \leq j \leq m}$ respectively, i.e. sort the sums $(x_i+y_j)_{1 \leq i \leq n,\ 1 \leq j \leq m}$.

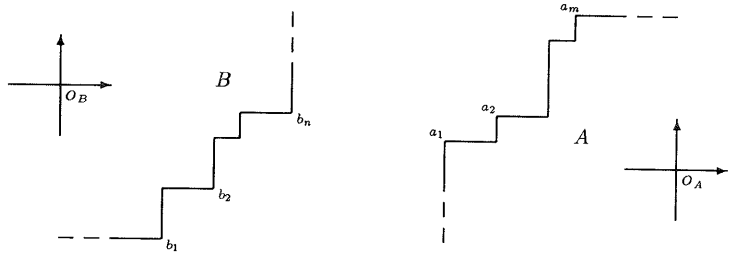


Fig. 2 The staircase polygonal lines A and B.

We will use in some parts of this paper the term $X-Y$ (meaning the set $(x_i-y_j)_{1\le i\le n,\ 1\le j\le m}$) instead of $X+Y$, because it is clearer with respect to the situation we are discussing. Obviously, sorting $X+Y$ is equivalent to sorting $X-Y$.

3. The PCP Reduces to Sort $X+Y$

First we prove P2——→P3.

A *contact* between a_j and b_i is the intersection between A and B which arises when a_j and b_i are coincident (**Fig. 3**) as A slides along the edges of B. Notice that a contact between two vertices does not always arise as A slides along the edges of B.

Initially suppose O_A is placed in a position O so that there is contact between b_1 and a_m, that is, $a_m^o=b_1$.

Let's consider the sets $X_V=\{b_i\cdot y\}_{1\le i\le n}$ and $Y_V=\{a_j^o\cdot y\}_{1\le j\le m}$. Let's denote the difference $b_i\cdot y-a_j^o\cdot y$ as $(i,j)_V$. So $X_V-Y_V=\{(i,j)_V, 1\le i\le n,\ 1\le j\le m\}$. For the sake of simplicity we assume now that no two pairs in X_V-Y_V are the same, i. e. there is not i, j, r, s, $1\le i, r\le n$, $1\le j, s\le m$, such that $(i,j)_V=(r,s)_V$.

Loosely speaking, our algorithm computes all contacts, in the order they take place, between vertices in B and vertices in A as the latter translates. Note that, if for a certain location p of O_A, a_j contacts b_i, and we know that the next contact as A slides first upward and after to the right is b_r and a_s, then the next two positions of O_A are $(p.x, p.y+(b_r.y-a_s^p.y))$ and $(p.x+(b_r.x-a_s^p.x), p.y+(b_r.y-a_s^p.y))$, where the last one is the position for O_A in which b_r contacts a_s. This observation tells us how to obtain all the positions of O_A.

So, the problem is to obtain b_r and a_s. Let $(X_V-Y_V)^*$ be the sorted set X_V-Y_V.

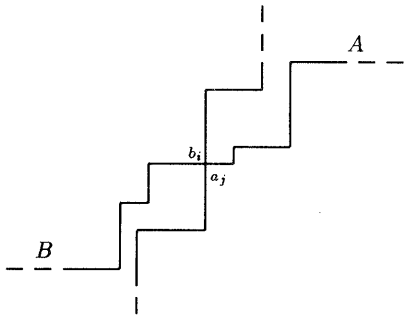


Fig. 3 A contact between a_j and b_i.

Lemma 3.1 Suppose b_i touches a_j $(1\le i\le n,\ 1\le j\le m)$ for a certain location p of O_A. Then, the next contact as A slides upward and to the right is between b_r and a_s if $(r,s)_V$ is the next pair in $(X_V-Y_V)^*$ following $(i,j)_V$ on the condition that $b_r.x-a_s^p.x>0$.

Thus, to compute the set of points which O_A can be at, it is only necessary to scan the sequence $(X_V-Y_V)^*$ asking for the pairs which satisfy the condition of the lemma.

It should be noticed that the running time of this algorithm would be dominated by sorting. The rest could be done in $O(nm)$. Here, we should merge m sorted sets $\{b_i.y-a_j^o.y\}(1\le j\le m)$. Initially, we could insert the minimum element of each set in a priority queue.[1] By deleting the minimum from the priority queue and inserting the corresponding element from the sets $\{b_i.y-a_j^o.y\}$ we could sort X_V-Y_V in $O(nm\ log\ m)$, since there would be $O(nm)$ operations, each one taking $O(log\ m)$. This equals the upper bound shown in Ref. 3). Could X_V-Y_V be sorted in less time, taking into consideration the particular structure of this set? We already said that for this problem of sorting, the question of how much computation time is really needed is still open. It was proven in Ref. 7) that, for two given sequences of numbers $(x_i)_{1\le i\le N}$ and $(y_j)_{1\le j\le N}$, there exists an algorithm to compute the N^2 sums $(x_i+y_j)_{1\le i,\ j\le N}$ in $O(N^2)$ comparisons. In fact, such an algorithm was presented there, but unfortunately, its performance was analyzed just in terms of comparisons and the existence of an algorithm with a similar bound in the case of a more general study remains still unknown.

Thus, we have proven the following theorem:

Theorem 3.2 P2——→P3.

Now we are ready to prove the main result of this section:

Theorem 3.3 P1——→P3.

Proof The algorithm described in Ref. 3) to compute the set of placements of P so that P is contained in Q consists of three parts:

(1) Divide P and Q each into four quadrant parts. The lower right quadrant for example, consists of the bottommost edge, the steps up and to the right and the rightmost edge. In a similar way we can describe the lower left, upper right and upper left quadrants.

(2) Determine the placements for matching quadrant parts (with open edges extended to rays).

(3) Intersect the four regions of step (2), giving the set of possible positions of P inside Q.

As was shown in Ref. 3), (1) can be done in $O(m+n)$ while (3) requires $O(mn)$. The algorithm designed for the second step took $O(mn \log m)$. Then, the whole algorithm complexity depends on (2). It is straightforward to recognize in the quadrant parts the staircase polygonal lines defined in the problem P2. So P1$\longrightarrow$P2. It follows from Theorem 3.2 that P1 $\longrightarrow$P3. □

4. Sorting $X+Y$ Reduces to Solve the PCP

In this chapter we prove that P3$\longrightarrow$P1.

Let's make some assumptions in relation with the problem of sorting $X-Y$. They do not result in any loss of generality. We will consider X and Y two sorted sets of numbers on the real line (it can be done in $O(max(n \log n, m \log m))$), i. e. $x_1 \leq x_2 \leq \cdots \leq x_n$, $y_1 \leq y_2 \leq \cdots \leq y_m$. We also assume $y_m = x_1$ (otherwise we can get sorted $X-Y$ by sorting $X-\{y_j-\alpha, y_j \in Y\}$, α such that $y_m-\alpha=x_1$).

Theorem 4.1 P3$\longrightarrow$P2.

Proof Using the set Y, we construct the staircase polygonal line A. Let's generate the vertices a_j, $1 \leq j \leq m$ in such a way that all of them lie on the same line L. Let's consider that the clockwise angle between this line and the abscissae axis, measured from L, is some θ, $0< \theta <90°$. We consider that, for every j, $1 \leq j \leq m$, $a_j.x = y_j$. The bottommost edge extends to a ray to the left while the rightmost edge extends to a ray upward. We construct the polygonal line B in a similar way, assuming that every vertex b_i, $1 \leq i \leq n$, lies on L. See **Fig. 4.** These constructions can be done in $O(m+n)$.

Let's suppose that an algorithm for determining $H(A,B)$ (the path O_A describes as A translates along the edges of B) is known. Let $h_0, h_1, \cdots, h_k$ be the list of its vertices, where h_0 is a vertex with its ordinate in the infinite. We can notice that $\forall v \exists i, j$ such that $b_i = a_j^{h_v}$ and that $\forall i, j \exists v$ such that $b_i = a_j^{h_v}$. These lemmas follow from here,

Lemma 4.2 $\forall v$, $1 \leq v \leq k$, $\exists i, j$, $1 \leq i \leq n$, $1 \leq j \leq m$, such that $h_v.x - h_0.x = x_i - y_j$.

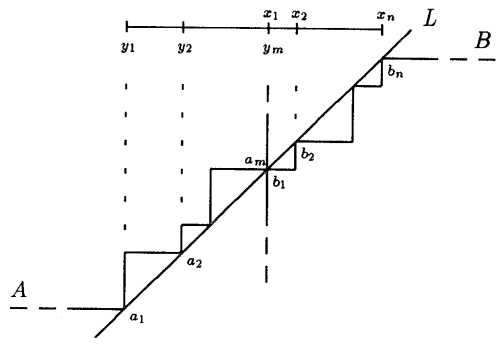


Fig. 4 Generation of the polygonal lines used in the proof of theorem 4.1.

We remark that it is possible to have $x_i - y_j = x_k - y_l$ for some i, j, k, l, $1 \leq i, k \leq n$, $1 \leq j, l \leq m$, $i \neq k$ or $j \neq l$.

Lemma 4.3 $\forall i, j$, $1 \leq i \leq n$, $1 \leq j \leq m$, $\exists v$, $1 \leq v \leq k$ such that $x_i - y_j = h_v.x - h_0.x$.

Lemma 4.2 and lemma 4.3 tell us that, except for repetitions, i. e. equal values in $X-Y$, the elements of $X-Y$ can be sorted from $H(A,B)$ immediately. Let $\overline{X-Y}$ denote $\{h_v.x - h_0.x\}_{1 \leq v \leq k}$, the set $X-Y$ with no repetitions and let $\overline{X-Y}^*$ denote the sorted set $\overline{X-Y}$. The difficulty now lies in sorting $X-Y$ from $\overline{X-Y}^*$. A naive approach using binary search would yield an $O(mn \log mn)$. We show that it is possible to sort $X-Y$ from $H(A,B)$ by using a perturbation on X and Y so that degeneracies will be eliminated.

Let δ be a real value on the condition that
$0 < \delta < min_{i,j,k,l}\{(x_k - y_l) - (x_i - y_j), (x_i - y_j) < (x_k - y_l)\}$.

It is clear that δ can be obtained in $O(mn)$ by scanning $\overline{X-Y}^*$.

We can find in $O(mn)$ sequences $\alpha = (\alpha_i)_{1 \leq i \leq n}$ and $\beta = (\beta_j)_{1 \leq j \leq m}$ such that

(i) $\alpha_i - \beta_j \neq \alpha_k - \beta_l$, $\forall i, j, k, l$, $1 \leq i, k \leq n$, $1 \leq j, l \leq m$, $i \neq k$ or $j \neq l$.

(ii) $0 \leq \alpha_i - \beta_j < \delta$, $\forall i, j$, $1 \leq i \leq n$, $1 \leq j \leq m$.

Let's define the sets X' and Y' as

$x'_i = x_i + \alpha_i \quad 1 \leq i \leq n$

$y'_j = y_j + \beta_j \quad 1 \leq j \leq m$

X' and Y' satisfy that

(i) $x_i - y_j < x_k - y_l \Rightarrow x_i - y_j \leq x'_i - y'_j < x_k - y_l$, $1 \leq i, k \leq n$, $1 \leq j, l \leq m$

(ii) $x'_i - y'_j \neq x'_k - y'_l$, $\forall i, j, k, l$, $1 \leq i, k \leq$

n, $1\leq j$, $l\leq m$, $i\neq k$ or $j\neq l$,

so $X-Y$ can be sorted from $\overline{X'-Y'}$*. □

Therefore, it is immediate that

Theorem 4.4 P3⟶P1

5. Yet Another Equivalent Problem

The results we present in this section derive from those in Ref. 7). There, the following problem was considered: Let $a_1, a_2, \cdots, a_n$ be n numbers and let's define for $1\leq i\leq j\leq n$

$$\sigma(i,j)=\sum_{k=i}^{j} a_k$$

Problem P4 Sort the set of numbers $A^+=\{\sigma(i,j),\ 1\leq i\leq j\leq n\}$

Theorem 5.1 If we consider that in the problem of sorting $X+Y$, the size of both sets X and Y is the same, i. e. $n=m$, then P3⟶P4.

Proof See the explanation in Ref. 7) about how an algorithm for sorting A^+ can be adapted to sort $(x_i+y_j)_{1\leq i,j\leq n}$. □

Theorem 5.2 P4⟶P3

Proof Suppose we are given n numbers $a_1, a_2, \cdots, a_n$. Without loss of generality we assume n is odd. Let's take $N=\frac{n+1}{2}$. Following a similar idea to that in Ref. 7), we will define the $2N$ values $x_1, x_2, \cdots, x_N, y_1, y_2, \cdots, y_N$ as follows:

$$\begin{aligned} y_N &= \text{any real number} \\ y_{N-i} &= y_{N-i+1}-a_i \qquad 1\leq i\leq N-1 \\ x_1 &= a_N - y_1 \\ x_i &= a_{N+i-1}+x_{i-1} \qquad 2\leq i\leq N \end{aligned}$$

The set A^+ can be represented as shown in **Fig. 5.**

Therefore, the set A^+ can also be represented as shown in **Fig. 6.**

If we take a look at the three regions indicated in the pyramid of Fig. 6, we can understand that actually, the central one is $X+Y$, the region on the left is the set$\{y_i-y_j, 1\leq j<i\leq N\}$ and the

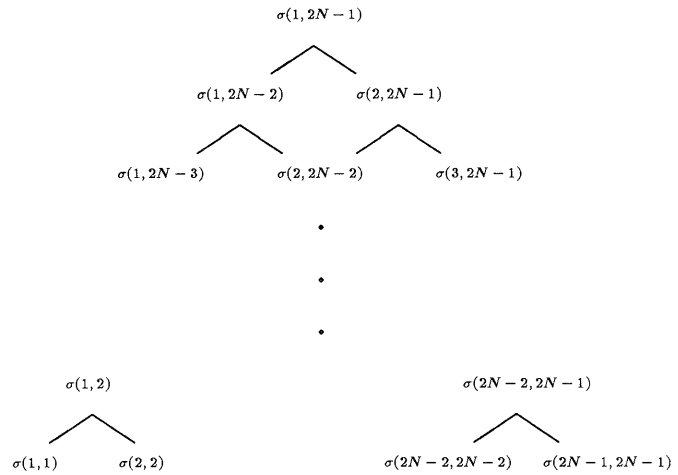


Fig. 5 Representation of A^+.

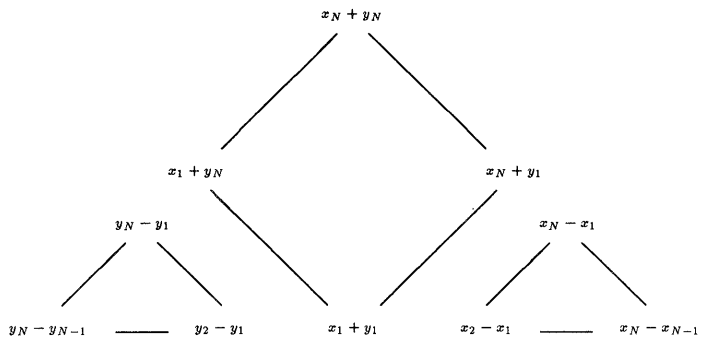


Fig. 6 Another way of seeing A^+.

region on the right is the set $\{x_i - x_j, 1 \leq j < i \leq N\}$.

The last two sets are subsequences of $Y - Y$ and $X - X$ respectively. Hence, an algorithm for sorting $X + Y$ may be used to sort each region. By merging the three sorted sets, set A^+ will be finally sorted. □

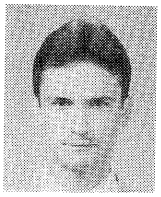
6. Conclusions

The main result of this paper is a proof of the equivalence between the polygon containment problem in the case of rectilinearly convex polygons under translation, the problem of sorting $X + Y$ and the problem of sorting sums of consecutive numbers.

References

1) Aho, A. V., Hopcroft, J. E. and Ulman, J. D.: *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, Mass. (1975).
2) Avnaim, F. and Boissonnat, J. D.: Polygon Placement under Translation and Rotation, *Proceedings of the STACS 1988*, Lecture Notes in Computer Science 294, pp. 322-333, Springer-Verlag (1988).
3) Baker, B. S., Fortune, S. J. and Mahaney, S. R.: Polygon Containment under Translation, *J. Algorithms*, Vol. 7, pp. 532-548 (1986).
4) Fortune, S. J.: A Fast Algorithm for Polygon Containment by Translation, *Proceedings of the 12th ICALP*, Lecture Notes in Computer Science 194, pp. 189-198, Springer-Verlag (1985).
5) Fredman, M. L.: How Good Is the Information Theory Bound in Sorting ?, *Theoretical Computer Science*, Vol. 1, pp. 355-361 (1976).
6) Harper, L. H., Payne, T. H., Savage, J. E. and Straus, E.: Sorting X+Y, *Comm. ACM*, Vol. 18, No. 6, pp. 347-349 (1975).
7) Lambert, J. L.: Sorting the Sums $(x_i + y_j)$ in $O(n^2)$ Comparisons, *Proceedings of the STACS 1990*, Lecture Notes in Computer Science 415, pp. 195-206, Springer-Verlag (1990).

(Received May 6, 1994)
(Accepted July 14, 1994)



A. Hernández Barrera was born in Habana City, Cuba, on June 11, 1964. He received the B. S. degree in computer science from Habana University, Cuba, in 1987. In the same year he joined the Computer Science Department, Faculty of Mathematics-Cybernetics, Habana University. He received the M. S. degree in mathematics from Hiroshima University, Japan, in 1994. He is currently working toward the Dr. Sci. at Hiroshima University. His research interests include computational geometry and the design and analysis of algorithms. He is a member of the IPSJ.