アルゴリズム 38－5
（1994．3．18）

高次元点集合の合同性判定について

阿久津 達也
機械技術研究所

本稿では、d次元（ユークリッド）空間の二つの点集合Aと$B(|A| = |B| = n)$が与えられた時、それらが合同であるかないかを判定する問題について考察する。まず、dを固定した時に、この問題に対する$O(n^{(d-1)/2}(\log n)^2)$時間の確率的アルゴリズムを示す。このアルゴリズムは birthday paradox という良く知られた性質に基づいて構成されている。この結果は既知の結果($O(n^{d-2}\log n)$時間)を大きく改善するものである。そして、dを固定しない場合、この問題がグラフの同型性判定問題と同程度以上に困難であることを示し、また、関連する他の問題についても議論する。

ON DETERMINING THE CONGRUITY OF POINT SETS IN HIGHER DIMENSIONS

Tatsuya AKUTSU

Mechanical Engineering Laboratory,
1-2 Namiki, Tsukuba, Ibaraki, 305 Japan.
e-mail: akutsu@mel.go.jp

This paper considers the following problem: given two point sets A and B ($|A| = |B| = n$) in d-dimensional Euclidean space, determine whether or not A is congruent to B. First, this paper presents a randomized algorithm which works in $O(n^{(d-1)/2}(\log n)^2)$ time. This improves the previous result (an $O(n^{d-2}\log n)$ time deterministic algorithm). The birthday paradox, which is a well-known property in combinatorics, is used effectively in our algorithm. Next, this paper shows that if d is not bounded, the problem is at least as hard as the graph isomorphism problem in the sense of the polynomiality. Several related results are described too.

1 Introduction

Recently, geometric pattern matching problems have been studied extensively in computational geometry [3, 4, 10]. Most of such studies have been done for approximate matchings in two or three dimensions. Few studies for exact matchings in higher dimensions have been done. This paper studies the problem of determining the exact congruity in higher dimensions.

Several studies have been done for exact matchings. $O(n \log n)$ time algorithms for determining the congruity of various objects in two dimensions were developed by Manacher [12], Atallah [5] and Highnam [11]. Sugihara developed an $O(n \log n)$ time algorithm for determining the congruity of two polyhedra in three-dimensions [13]. Atkinson developed an $O(n \log n)$ time algorithm for determining the congruity of two point sets in three-dimensions. Alt et al. developed an $O(n^{d-2} \log n)$ time algorithm for determining the congruity of two point sets in d-dimensions [4]. However, to my knowledge, no improvement on their result has been done.

In this paper, we present an $O(n^{\frac{d-1}{2}}(\log n)^2)$ time randomized algorithm for determining the congruity of two point sets A and B ($|A| = |B| = n$) in d-dimensional Euclidean space. This improves the previous result [4] considerably while our algorithm is a randomized one. Moreover, we show that if d is not bounded, the congruity problem is at least as hard as the graph isomorphism problem in the sense of the polynomiality. Several related results are described too.

2 Preliminaries

Let E^d denotes the d-dimensional Euclidean space. For a point $\boldsymbol{p}$, $\boldsymbol{p}(i)$ denotes the i'th coordinate value of $\boldsymbol{p}$. For a point set $P = \{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_n\}$, the centroid of P is the point given by $\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{p}_i$, and $dim(P)$ denotes the number of dimensions of the affine hull of P.

A mapping T of E^d onto itself is said to be *isometric* if $\overline{\boldsymbol{pq}} = \overline{T(\boldsymbol{p})T(\boldsymbol{q})}$ for all two points $\boldsymbol{p}$ and $\boldsymbol{q}$. Let $A = \{\boldsymbol{a}_1, \cdots, \boldsymbol{a}_n\}$ and $B = \{\boldsymbol{b}_1, \cdots, \boldsymbol{b}_n\}$ denote point sets in E^d respectively. For A, we define $T(A) = \{T(\boldsymbol{a}_i) | \boldsymbol{a}_i \in A\}$. If there exists an isomorphic mapping which satisfies $B = T(A)$, A and B are said to be *congruent*. If A and B are congruent, we write $A \cong B$. $\boldsymbol{a}_i \in A$ and $\boldsymbol{b}_j \in B$ are called *equivalent* if $\boldsymbol{b}_j = T(\boldsymbol{a}_i)$ holds for some isometric mapping T such that $B = T(A)$. An isometric mapping T can be written in the form $T : \boldsymbol{p} \mapsto M\boldsymbol{p} + \boldsymbol{a}$ where M is a $d \times d$ orthonormal real matrix, i.e., $M^T = M^{-1}$, and $\boldsymbol{a}$ is any d-vector. T is called the *first* (resp. *second*) *kind* if $det(M) = +1$ (resp. -1). Since any T of the second kind can be written as $\boldsymbol{p} \mapsto MJ\boldsymbol{p} + \boldsymbol{a}$ where $J(x_1, \cdots, x_d) = (-x_1, x_2, \cdots, x_d)$ and $det(A) = +1$ [4], we only consider isometric mappings of the first kind.

In this paper, points may be labeled with integers. In such cases, equivalent points must have the same label. A data structure $C(A)$ representing a point set A is called a *canonical form* of A if the size of $C(A)$ is $O(n)$ and it satisfies the following condition: $C(A) = C(B)$ if and only if $A \cong B$. Note that, once canonical forms are computed, the congruity can be determined by comparing them.

3 A randomized algorithm for congruity

In this section, we present a randomized algorithm for determining the congruity of point sets in d-dimensions, where we assume that d ($d \leq 3$) is a fixed constant.

3.1 Birthday paradox

The birthday paradox is a well-known property in combinatorics [8]. It states that on the average, 24 persons are needed for at least two of them having the same birthday, assuming all birth dates to be equally distributed over the days in the year. If there were n days in a year, $\Theta(\sqrt{n})$ persons would be needed. The birthday paradox has been applied to several algorithms [8, 9]. For applying the birthday paradox to the congruity problem, the following observation is useful: if $A \cong B$ and a set of $O(\sqrt{n})$ points A' (resp. B') is chosen randomly from A (resp. B), there is at least a pair of equivalent points $(a_i, b_j) \in A' \times B'$ with high probability. Once an equivalent point pair is given, the congruity problem in d-dimensions can be reduced to the congruity problem in $(d-1)$-dimensions using a similar reduction as in Ref.[4]. Thus, reducing the problem recursively, we can solve the congruity problem.

3.2 Algorithm

The following procedure $CheckCongruity(\{A_1, \cdots, A_h\}, \{B_1, \cdots, B_k\})$ determines whether or not their is at least a pair (A_i, B_j) such that $A_i \cong B_j$, where A_i's and B_j's are point sets.

Procedure $CheckCongruity(\{A_1, \cdots, A_h\}, \{B_1, \cdots, B_k\})$
begin
  if there is a pair (A_i, B_j) such that $dim(A_i) = dim(B_j) \leq 3$ and $A_i \cong B_j$ **then** -(#1)
    begin output 'YES'; stop **end**;
  Remove A_i's (resp. B_i's) such that $dim(A_i) \leq 3$ (resp. $dim(B_i) \leq 3$);
  (Let the remaining set be $AA = \{A_1, \cdots, A_{h'}\}$ (resp. $BB = \{B_1, \cdots, B_{k'}\}$))
  if $AA = \{\}$ or $BB = \{\}$ **then begin** output 'NO'; stop **end**; -(#2)
  for all $A_i \in AA$ **do**
    Choose a point set $A'_i \subset A_i$ randomly such that $|A'_i| = \min(|A_i| - 1, \lceil K\sqrt{n} \rceil)$
    and A'_i does not contain the centroid of A_i;
  for all $B_i \in BB$ **do**
    Choose a point set B'_i randomly in the same way;
  for all $A_i \in AA$ **do for all** $a_j \in A'_i$ **do** $A_{ij} \leftarrow proj(A_i, a_j)$; -(#3)
  for all $B_i \in BB$ **do for all** $b_j \in B'_i$ **do** $B_{ij} \leftarrow proj(B_i, b_j)$; -(#4)
  $CheckCongruity(\{A_{11}, A_{12}, \cdots, A_{21}, \cdots\}, \{B_{11}, B_{12}, \cdots, B_{21}, \cdots\})$
end

Note that K is a constant to be determined later. In the above procedure, a d-dimensional point set A_i (resp. B_i) is reduced to the $(d-1)$-dimensional point set $proj(A_i, \boldsymbol{p})$ (resp. $proj(B_i, \boldsymbol{p})$) where $\boldsymbol{p}$ is not the centroid of A_i (resp. B_i). $proj(A_i, \boldsymbol{p})$ is computed by the following procedure (see Fig.1). Note that a similar procedure is used in Ref. [4].

Procedure $Project(A_i, \boldsymbol{p})$
begin
  Let $\boldsymbol{c}$ be the centroid of A_i;
  Let H be the hyperplane such that $\boldsymbol{c} \in H$ holds and $\overline{\boldsymbol{pc}}$ is perpendicular to H;
  Let H' be the hyperplane such that $\boldsymbol{p} \in H'$ holds and H' is parallel to H;
  for all $\boldsymbol{a} \in A_i \cap H'$ **do**
    Replace $\boldsymbol{a}$ with $\boldsymbol{a} + \delta \overrightarrow{\boldsymbol{pc}}$ where δ is a sufficiently small constant;
  $proj(A_i, \boldsymbol{p}) \leftarrow \{\, \boldsymbol{q} \mid (\exists \boldsymbol{a} \in A_i)(\overline{\boldsymbol{pq}} \parallel \overline{\boldsymbol{pa}}) \wedge \boldsymbol{q} \in H \,\}$;
  Label each point $\boldsymbol{q} \in proj(A_i, \boldsymbol{p})$; - ($)
  return $proj(A_i, \boldsymbol{p})$
end

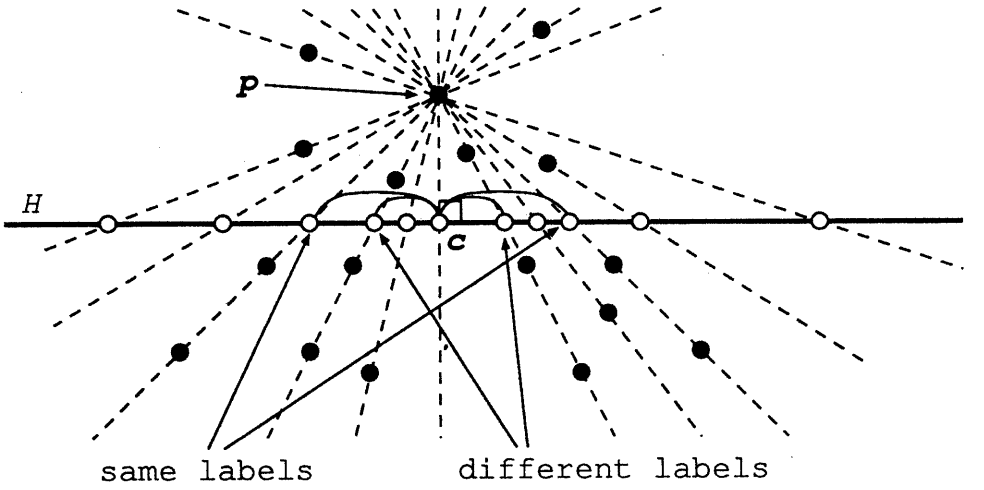

Figure 1: Projection from d-dimensions to $(d-1)$-dimensions

In step (\$), each point is labeled so that q and r are labeled with the same integer number if and only if $Q \cup \{p, c\} \cong R \cup \{p, c\}$ holds by an isometric mapping not moving p or c, where $Q = \{a \in A_i | a$ is projected to $q\}$ and $R = \{a \in A_i | a$ is projected to $r\}$. This labeling procedure is done simultaneously for all projected points in the same depth of the recursion. Note that it can be done in $O(Ln\log(Ln))$ time using an appropriate sorting algorithm, where L is the total number of A_{ij}'s and B_{ij}'s.

3.3 Analysis

First, we analyze the time complexity and the space complexity of procedure *CheckCongruity*. Note that *CheckCongruity*($\{A\}, \{B\}$) is executed for determining the congruity of A and B.

[Lemma 2.1] *CheckCongruity*($\{A\}, \{B\}$) works in $O(n^{\frac{d-1}{2}} \log n)$ time using $O(n^{\frac{d-1}{2}})$ space. *(Proof)* *CheckCongruity* is executed at most $d-2$ times recursively. Note that the number of point sets contained in the arguments of the recursive execution of k'th depth is $O((\sqrt{n})^{k-1}) = O(n^{\frac{k-1}{2}})$.

Next, we analyze the time complexity for each recursive step where we assume that m is the number of point sets contained in the arguments. To analyze the time complexity, we only consider the parts (#1) (#3) (#4) since these parts are crucial.

To execute (#1), we do not directly compare every pair (A_i, B_j). Instead, we first compute the canonical form of each A_i (resp. B_i) such that $dim(A_i) \leq 3$ (resp. $dim(B_i) \leq 3$), and then we sort all the canonical forms. Once the sorted sequence of canonical forms is computed, it is easy to test whether or not there is a pair (A_i, B_j) such that $A_i \cong B_j$, since the test can be done by comparison of strings. The canonical form of a (labeled) point set of size N in three-dimensions can be computed in $O(N \log N)$ time [1]. Thus, the canonical forms can be computed in $(mn\log n)$ time. Since the size of each canonical form is $O(n)$, sorting can be done in $O(mn\log m)$ time. Thus, part (#1) can be executed in $O(mn\log(mn))$ time.

The time required for (#3) and (#4) is $O(mn^{\frac{3}{2}}\log(mn^{\frac{3}{2}}))$ since $O(m\sqrt{n})$ point sets are projected. However, note that the parts after (#2) are not executed in the last step ($(d-2)$'th step) of the recursion.

Thus, the total computation time for (#1) is

$$O((d-2)\times n^{\frac{d-3}{2}}\times n\times\log(n^{\frac{d-3}{2}}\times n)) \;=\; O(n^{\frac{d-1}{2}}\log n)\,,$$

since d is assumed to be a constant. The computation time for (#3) and (#4) is

$$O((d-3)\times n^{\frac{d-4}{2}}\times n^{\frac{3}{2}}\times\log(n^{\frac{d-4}{2}}\times n^{\frac{3}{2}})) \;=\; O(n^{\frac{d-1}{2}}\log n)\;.$$

Thus, the time complexity is $O(n^{\frac{d-1}{2}}\log n)$ in total.

Since $O((d-2)n^{\frac{d-3}{2}})$ sets are constructed in total, the space complexity is $O(n^{\frac{d-1}{2}})$. □

Next, we analyze the probability that the procedure succeeds. The following proposition is proved in a straight-forward way.

[Lemma 2.2] If $CheckCongruity(\{A\},\{B\})$ outputs 'YES', then $A\cong B$.

The following lemma is a variant of the birthday paradox.

[Lemma 2.3] If two subsets $S_1\subset S$ and $S_2\subset S$, each of which consists of at least $\sqrt{\ln(\frac{1}{1-q})\,n}$ elements, are chosen randomly from S ($|S|=n$), then $|S_1\cap S_2|\neq\{\}$ holds with probability at least q.

(Proof) Let $P(n,m)$ be the probability that $|S_1\cap S_2|\neq\{\}$ holds if S_1 and S_2 such that $|S_1|=|S_2|=m$ are chosen randomly from S. Then, the following inequality holds:

$$\begin{aligned}P(n,m) &= 1-(\frac{n-m}{n})(\frac{n-m-1}{n-1})(\frac{n-m-2}{n-2})\cdots(\frac{n-2m+1}{n-m+1})\\ &\geq 1-(\frac{n-m}{n})^m\,.\end{aligned}$$

Thus, it is sufficient that $1-(\frac{n-m}{n})^m\geq q$ holds. Using the following inequalities:

$$\begin{aligned}m\ln(\frac{n-m}{n}) &\leq \ln(1-q)\;,\\ \ln(1-\frac{m}{n}) &\leq \frac{\ln(1-q)}{m}\;,\\ \frac{1}{m}\ln(\frac{1}{1-q}) &\leq (\frac{m}{n})+(\frac{1}{2})(\frac{m}{n})^2+(\frac{1}{3})(\frac{m}{n})^3+\cdots\;,\\ &\qquad(\text{from } \ln(1-x)=-x-\frac{x^2}{2}-\frac{x^3}{3}-\cdots)\end{aligned}$$

it is sufficient that $\frac{m}{n}\geq\frac{1}{m}\ln(\frac{1}{1-q})$ holds. Thus, the lemma holds. □

[Theorem 2.4] The congruity of two point sets in d-dimensions can be tested in $O(n^{\frac{d-1}{2}}(\log n)^2)$ time and $O(n^{\frac{d-1}{2}})$ space with probability at least $1-\frac{1}{n^h}$ where h ($h>1$) is any fixed constant.

(Proof) Let $K=\sqrt{\ln(\frac{1}{1-(\frac{1}{2})^{\frac{1}{d-3}}})}$ in procedure $CheckCongruity$. Then, if there is at least a pair (A_i,B_j) such that $A_i\cong B_j$, $CheckCongruity(\{A_1,\cdots\},\{B_1,\cdots\})$ finds at least a pair of equivalent points $(a,b)\subset A_i'\times B_j'$ (and produces at least a pair $(A_{ij},B_{i'j'})$ such that $A_{ij}\cong B_{i'j'}$) with probability at least $(\frac{1}{2})^{\frac{1}{d-3}}$ (from Lemma 2.3). Thus, $CheckCongruity(\{A\},\{B\})$ outputs 'YES' with probability at least $\frac{1}{2}$ if $A\cong B$.

If $CheckCongruity$ is executed $h\log n$ times, it will output 'YES' at least once with probability at least $1-\frac{1}{n^h}$ if $A\cong B$. Thus, the theorem holds. □

Note that *CheckCongruity* can be modified for determining the congruity of such objects as polyhedra. Moreover, we can make a parallel version (an NC algorithm) of *CheckCongruity* using a parallel sorting algorithm [7] and a parallel algorithm for computing canonical forms of three-dimensional point sets [2]. However, details are omitted here.

3.4 Application to subset matching

The birthday paradox can be applied to the following exact matching problem: given point sets $P = \{p_1, \cdots, p_m\}$ and $Q = \{q_1, \cdots, q_n\}$ such that $m \leq n$ in E^1, determine whether or not there is a subset $S \subset Q$ such that $P \cong S$. Of course, this problem can be solved in $O(mn)$ time using a simple algorithm, where we assume that P and Q are given as sorted sequences. However, it seems very difficult to develop an $o(mn)$ time algorithm. Here, we show a randomized algorithm which works in $o(mn)$ time in a very special case. For P, we define $m(P) = \max_c |\{(p_i, p_j) | i \leq j \wedge |\overline{p_i p_j}| = c\}|$. We consider the special case where $m(P) \leq M$ holds for some constant M. Then, the following procedure determines whether or not there is such a subset S in $O((\frac{n^2}{m} + m^2)\text{polylog}(m))$ time with high probability. Note that it is $o(mn)$ if $n^{\frac{1}{2}+\varepsilon} < m < n^{1-\varepsilon}$ holds for any small constant $\varepsilon > 0$.

Procedure *CheckSubCongruity*(P, Q)
begin
  Choose randomly $U \subset Q$ such that $|U| = \frac{Kn}{m}$;
  for all $q \in U$ **do**
    begin
      for all $1 \leq i \leq m$ **do** $A[i] \leftarrow 0$;
      for all $1 \leq j \leq n$ **do**
        for all i such that $(\exists k)(p_k - p_i + q = q_j)$ **do** $A[i] \leftarrow A[i] + 1$; -($)
      if $(\exists i)(A[i] = m)$ **then begin** output 'YES'; stop **end**
    end;
  output 'NO'
end

In this procedure, the following variant of the birthday paradox is used. Let S be a subset of Q such that $|S| = m$. Then, if we choose randomly a point set $U \subset Q$ such that $|U| = \frac{Kn}{m}$, $(\exists q \in U)(q \in S)$ holds with high probability, where K is an appropriate constant.

Here, we briefly discuss on the time complexity while details are omitted here. Part ($) is the crucial part for analyzing the time complexity. Since we assume that $m(P) \leq M$ holds for some constant M, part ($) can be done in $O(\log m + M)$ time per execution if $O(m^2)$ points constructed from P are preprocessed appropriately and the binary search technique is used. Thus, the above procedure solves the problem in $O((\frac{n^2}{m} + m^2)\text{polylog}(m))$ time with high probability if $m(P) < M$ holds for some constant M.

4 Hardness results

Although we have presented an improved algorithm for the congruity problem, it is not a polynomial time one if d is not bounded. Thus, it is natural to ask whether or not there is a polynomial time algorithm for the congruity problem even if d is not bounded. The following theorem suggests that the answer is 'NO' since no polynomial time deterministic or randomized algorithm has been developed for the graph isomorphism problem.

[**Theorem 4.1**] Assume that there exists a polynomial time deterministic (resp. randomized) algorithm for the congruity problem even if d is not bounded. Then, there exists a polynomial time deterministic (resp. randomized) algorithm for the graph isomorphism problem.
(Proof) We prove it showing a polynomial time reduction from the graph isomorphism problem to the congruity problem.

Let a pair of undirected graphs $G_1(V,E)$ and $G_2(W,F)$ be an instance of the graph isomorphism problem. We can assume without loss of generality that $|V| = |W| = n$. Let $V = \{v_1, \cdots, v_n\}$ and $W = \{w_1, \cdots, w_n\}$. From G_1, we construct a point set A in E^n as follows. First, a point set A_1 such that:

$$A_1 = \{ \boldsymbol{a}_i \mid v_i \in V \wedge \boldsymbol{a}_i(i) = 1 \wedge (\forall j \neq i)(\boldsymbol{a}_i(j) = 0) \}$$

is constructed. Next, a point set A_2 such that:

$$A_2 = \{ \boldsymbol{a} \mid \{v_i, v_j\} \in E \wedge \boldsymbol{a} = \frac{\boldsymbol{a}_i + \boldsymbol{a}_j}{2} \}$$

is constructed. Then, A is defined by $A = A_1 \cup A_2 \cup \{\mathbf{o}\}$ where $\mathbf{o}$ denotes the origin. B is constructed from G_2 in the same way.

Then, it is easy to see that A is congruent to B if and only if G_1 is isomorphic to G_2. Since the construction of A and B can be done in polynomial time, the theorem holds. □

Conversely, the congruity problem may be reduced to the graph isomorphism problem in the following way, while we have not yet proved the correctness of the reduction. From a point set A, we construct an undirected complete graph $G_1(V,E)$ such that $V = \{v_i | \boldsymbol{a}_i \in A\}$. From a point set B, we construct $G_2(W,F)$ in the same way. Then, each edge $e \in E \cup F$ is labeled with an integer number so that the following condition is satisfied:

$$(\forall e, f \in E \cup F)(\mathit{label}(e) = \mathit{label}(f) \Longrightarrow \mathit{dist}(e) = \mathit{dist}(f)) ,$$

where $\mathit{dist}(\{v_i, v_j\}) = |\overline{\boldsymbol{a}_i \boldsymbol{a}_j}|$ and $\mathit{dist}(\{w_i, w_j\}) = |\overline{\boldsymbol{b}_i \boldsymbol{b}_j}|$. This reduction can be done in polynomial time. Thus, if the following statement is correct (we believe that it is a known result), the congruity problem can be reduced to the graph isomorphism problem in polynomial time: $A \cong B$ if and only if there is a permutation π such that $(\forall i, j)(|\overline{\boldsymbol{a}_i \boldsymbol{a}_j}| = |\overline{\boldsymbol{b}_{\pi(i)} \boldsymbol{b}_{\pi(j)}}|)$.

Note that, using a similar reduction (a reduction to the subgraph isomorphism problem) as in Theorem 4.1, we can show that the following problem is NP-complete if d is not bounded: given point sets $P = \{\boldsymbol{p}_1, \cdots, \boldsymbol{p}_m\}$ and $Q = \{\boldsymbol{q}_1, \cdots, \boldsymbol{q}_n\}$ such that $m \leq n$, determine whether or not there is a subset $S \subset Q$ such that $P \cong S$.

5 Conclusion

In this paper, we have presented a randomized algorithm for determining the congruity of point sets in d-dimensions, which improves the previous result. However, our algorithm is not necessarily optimal. Thus, it would be interesting to develop more efficient algorithms. In our algorithm, the birthday paradox is used effectively. It would be also interesting to apply the birthday paradox to other pattern matching problems.

On the other hands, we have shown that the congruity problem is hard if the number of dimensions is not bounded. This hardness result suggests that approximate matching problems in higher dimensions are hard since they seem to be harder than exact matching problems. However, it does not mean that we can not develop practical pattern matching algorithms

in higher dimensions. It seems that the congruity problem can be solved efficiently in most cases, since the d-dimensional congruity problem can be reduced to the $(d-1)$-dimensional problem efficiently by computing a special point (except the centroid) invariant with isometric mappings, and such a special point seems to be computed efficiently in most cases. For the graph isomorphism problem, several algorithms which work in polynomial time in most cases have been developed [6]. Thus, it would be intersting to develop pattern matching algorithms in higher dimensions which work in polynomial time in most cases.

References

[1] T. Akutsu. "Algorithms for determining geometrical congruity in two and three dimensions". In *Proceeding of 3rd International Symposium on Algorithms and Computation*, pp. 279–288, 1992.

[2] T. Akutsu. "A parallel algorithm for determining the congruity of point sets in three dimensions". Technical Report AL31-3, Information Processing Society of Japan, 1993.

[3] H. Alt and M. Godau. "Measuring the resemblance of polygonal curves". In *Proceedings of ACM Symposium on Computational Geometry*, pp. 102–109, 1992.

[4] H. Alt, K. Mehlhorn, H. Wagener, and E. Welzl. "Congruence, similarity, and symmetries of geometric objects". *Discrete and Computational Geometry*, Vol. 3, pp. 237–256, 1988.

[5] M. J. Atallah. "On symmetry detection". *IEEE Transactions on Computers*, Vol. C-34, pp. 663–666, 1985.

[6] L. Babai and L. Kučera. "Canonial labeling of graphs in linear average time". In *Proceedings of IEEE Symposium on Foundations of Computer Science*, pp. 39–46, 1979.

[7] R. Cole. "Parallel merge sort". In *Proceedings of IEEE Symposium on Foundations of Computer Science*, pp. 511–516, 1986.

[8] P. Flajolet, D. Gardy, and L. Thimonier. "Birthday paradox, coupon collectors, cashing algorithms and self-organizing search". *Discrete Applied Mathematics*, Vol. 39, pp. 207–229, 1992.

[9] H. Gazit and J. Reif. "A randomized parallel algorithm for planar graph isomorphism". In *Proceeding of ACM Symposium on Parallel Algorithms and Architectures*, pp. 210–219, 1990.

[10] P. J. Heffernan and S. Schirra. "Approximate decision algorithm for point sets congruence". In *Proceedings of ACM Symposium on Computational Geometry*, pp. 93–101, 1992.

[11] P. T. Highnam. "Optimal algorithms for finding the symmetries of a planar point set". *Information Processing Letters*, Vol. 22, pp. 219–222, 1986.

[12] G. Manacher. "An application of pattern matching to a problem in geometrical complexity". *Information Processing Letters*, Vol. 5, pp. 6–7, 1976.

[13] K. Sugihara. "An $n \log n$ algorithm for determining the congruity of polyhedra". *Journal of Computer and System Sciences*, Vol. 29, pp. 36–47, 1984.