アルゴリズム 28−10
（1992. 7. 18）

Optimal Parallel Algorithm for Computing the Prefix Convex Hulls of A Sorted Points Set

Wei Chen, Koji Nakano,Toshimitsu Masuzawa, and Nobuki Tokura
Faculty of Engineering Science,Osaka University,Osaka,560 Japan

The optimal parallel algorithm for computing the convex hull of a sorted set S of n points in the plane is known to be executed in $O(\log n)$ time using $n/\log n$ processors on the CREW PRAM. In this paper we present a parallel algorithm for computing the convex hulls of all prefix sets of S. The algorithm runs in $O(\log n)$ time using $n/\log n$ processors on the CREW PRAM, and hence is asymptotically optimal.

ソートされた点集合の接頭部凸包を求める最適並列アルゴリズム

陳慰 中野浩嗣 増澤利光 都倉信樹
大阪大学基礎工学部情報工学科

ソートされた平面点集合が与えられたとき，その点集合の凸包をPRAM CREWの上，$O(\log)$時間，$n/\log n$プロセッサで求める最適並列アルゴリズムが既に知られている．本研究で提案する並列アルゴリズムは，その点集合の全ての接頭部の凸包をCREW PRAMの上，$O(\log n)$時間，$n/\log n$プロセッサで求める．このアルゴリズムは最適である．

1 Introduction

Given a set S of n points in the plane, the convex hull of S is the smallest convex polygon containing all the points of S. The problem of computing the convex hull has been studied extensively. It has been shown the problem has a lower bound $\Omega(n \log n)$ of sequential time complexity (in the quadratic decision-tree model), and several algorithms achieve this lower bound[12]. If the set of points are sorted, say, sorted by x coordinates, the problem can be solved in $O(n)$ time by Graham scan[12].

Convex hull algorithms have also been developed in various parallel models; especially on the CREW PRAM (i.e., the synchronous shared-memory parallel model where concurrent reads are allowed but no two processors can simultaneously write to the same memory cell). A PRAM algorithm is *cost optimal* (or speed-up optimal) if the product of the time and the number of processors is of the same order as the running time of the fastest known sequential algorithm for the same problem. It is *time optimal* if it is the fastest possible algorithm using a polynomial number of processors. A PRAM algorithm is *optimal* if it is both cost optimal and time optimal. Several optimal convex hull algorithms have been proposed on the CREW PRAM[1],[3],[4],[6]. They use $O(\log n)$ time and $O(n)$ processors. For a sorted set of points, Goodrich[10] presented an optimal parallel algorithm which runs in $O(\log n)$ time on CREW PRAM using $O(n/\log n)$ processors. The optimal parallel algorithms for computing the convex hull of a sorted point set on the other PRAM models have also been well studied[5],[8].

In this paper, we consider the problem of computing *the prefix convex hulls of a sorted set of points in the plane.* Let $S=\{s_1, s_2, \ldots, s_n\}$ be a sorted set of points in the plane, say, sorted by order of increasing x coordinates. That is, $x(s_i) \leq x(s_{i+1})$ holds for each i $(1 \leq i \leq n-1)$, where $x(s)$ stands for the x-coordinate of s. The subset $S_i = \{s_1, s_2, \ldots, s_i\}$ of the first i elements of S is called as *the ith prefix of* S. The problem of computing prefix convex hulls of S is to compute the convex hull of S_i for all $i (1 \leq i \leq n)$. The problem is expected to have applications to several probelms such as the the convex rope problems and the visibility problems.

Since the convex hull of S_i may be i-gon $(1 \leq i \leq n)$, the problem of computing prefix convex hulls has a lower bound $\Omega(n^2)$ on the cost of the parallel algorithms if each convex hull is required to be represented independently as a list of its vertices. To avoid this lower bound on the cost, we use a tree of size n to represent all these n convex hulls as follows. For any i, s_1 and s_i have the smallest and the largest x-coordinates in S_i, respectively, therefore, they are the vertices of the convex hull of S_i. The line segment $\overline{s_1 s_i}$ divides the convex hull of S_i into two convex polygonal curves: the upper convex hull above the line segment, and the lower convex hull below the line segment. The union of these upper (resp. lower) convex hulls forms a tree, called *the upper (resp. lower) convex hulls tree of* S (we give the formal definition later) (Fig.1). In this paper, the upper and lower convex hull trees of S are adopted to represent the all prefix convex hulls of S. We propose an algorithm for constructing these two trees in $O(\log n)$ time using $O(n/\log n)$ processors on the CREW PRAM. The algorithm is shown to be optimal. Since the convex hull of S is one of the prefix convex hulls of S, our algorithm is regarded as a generalization of Goodrich's algorithm in literature [10].

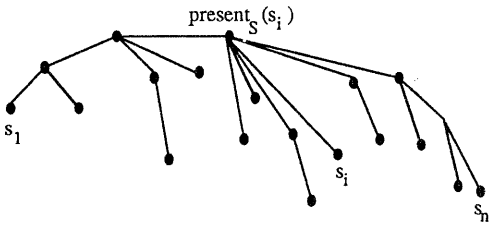

Figure 1: The upper convex hull tree of a sorted points set

2 Preliminaries

For any point s in the palne, denote x and y coordinates of s as $x(s)$ and $y(s)$ respectively. A set $S = \{s_1, s_2, \ldots, s_n\}$ of points in the plane is x-sorted if the points of S are listed by order of increasing x coordinate, i.e., $x(s_i) \leq x(s_{i+1})$ $(1 \leq i \leq n-1)$.

Consider an x-sorted set $S = \{s_1, s_2, \ldots, s_n\}$ of points in the plane and the ith $(1 \leq i \leq n)$ prefix of S, $S_i = \{s_1, s_2, \ldots, s_i\}$. Let $ch(S_i)$ denote the convex hull of S_i which is represented by the clockwise sequence of its vertices begining at s_1. The upper (resp. lower) convex hull of S_i, denoted by $uh(S_i)$ (resp. $lh(S_i)$), is the portion of $ch(S_i)$ above (resp. below) the line segment $\overline{s_1 s_i}$, and represented by the clockwise (resp. counterclockwise) sequence of its vertices, that is, the clockwise sequences of vertices from s_1 to s_i (resp. counterclockwise sequence of vertices from s_1 to s_i) in $ch(S_i)$. For any vertex s of $uh(S_i)$ (resp. $lh(S_i)$), we define the immediate predecessor of s in $uh(S_i)$ (resp. $lh(S_i)$) to be the vertex immediately before s in $uh(S_i)$ (resp.$lh(S_i)$). We give the formal definition for the upper convex hull tree of S.

Definition 1 *(The upper convex hull tree of S)*
Let S be an x-sorted set of points in the plane. Let p_i be the s_1-s_i path defined by the vertex sequence of $uh(S_i)$. The upper convex hull tree of S, denoted as $uhT(S)$, is the union of p_i over i $(1 \leq i \leq n)$. That is, $uhT(S)$ is a graph whose vertex set is S and edge set is $E = \{(s_i, s_j) \mid s_i$ is the immediate predecessor of s_j in $uh(S_j)\}$ (Fig. 1). ∎

We can give the similar definition and property for the lower convex hull tree S. In this paper, we solve the prefix convex hulls problem of S by constructing the upper and lower convex hull trees of S. Because of

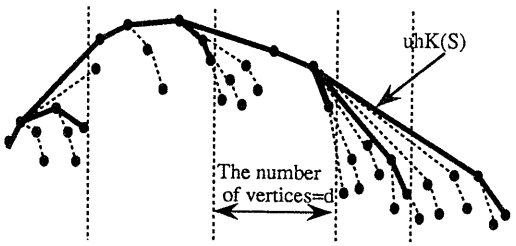

Figure 2: The skeleton of $uhT(S)$

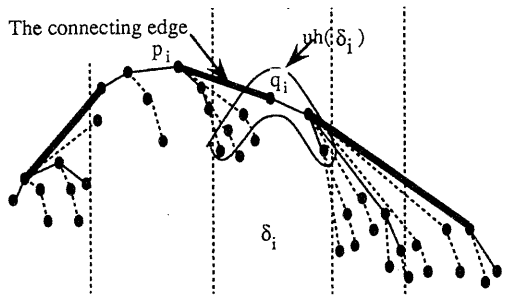

Figure 3: The connecting edges on $uhK(S)$

symmetry, we consider the upper convex hull tree only in the rest of the paper. The lower convex hull tree can be handled in the similar way.

Throughout this paper, for simpler presentation of algorithms, we omit the floor and the ceiling operators ensuring that all values are integers.

3 An overview of the algorithm

Now we give an overview of algorithm Make$uhT(S)$ which constructs the upper convex hull tree, $uhT(S)$, in $O(\log n)$ time using $n/\log n$ processors. In the algorithm, $uhT(S)$ is regarded as a rooted tree with the root s_1, and represented by the function $parent_S$ such that

$$parent_S(s_i) = \begin{cases} \text{the parent of } s_i \text{ in } uhT(S) & 2 \le i \le n \\ s_1 & i = 1 \end{cases}$$

Note that for any point s_i $(2 \le i \le n)$ of S, the parent of s_i in $uhT(S)$ is the immediate predecessor of s_i in $uh(S_i)$.

{*Algorithm*} Make$uhT(S)$

(*Input*) An x-sorted set $S = \{s_1, s_2, \ldots, s_n\}$ of points in plane.

(*Output*) The upper convex hull tree of S, $uhT(S)$, represented by function $parent_S$.

(*Method*) Our algorithm consists of two steps.

Step 1: Construct a special subgraph of $uhT(S)$, called as *the skeleton of* $uhT(S)$, in $O(\log n)$ time using $n/\log n$ processors. The definition of the skeleton of $uhT(S)$ is given in Definiton 2.

Step 2: Extend the skeleton of $uhT(S)$ to $uhT(S)$ in $O(\log n)$ time using $n/\log n$ processors. ∎

Definition 2 *(The skeleton of $uhT(S)$)*

Let $d = \log n$ and $m = n/d$. The skeleton of $uhT(S)$, denoted as $uhK(S)$, is a subgraph of $uhT(S)$ formed by the union of $p_{k \cdot d}$ (the s_1-$s_{k \cdot d}$ path defined by the vertex sequence of $uh(S_{k \cdot d})$) over k $(1 \le k \le m)$ (Fig. 2). That is, $uhK(S)$ is a graph whose vertex set is $SK = \{s_i \mid s_i$ is the vertex of $uh(S_{k \cdot d}), 1 \le k \le m\}$ and edge set is $EK = \{(s_i, s_j) \mid s_i$ is the immediate predecessor of s_j in $uh(S_{k \cdot d})$, $1 \le k \le m\}$

We describe Step 1 in section 4 and Step 2 in section 5 respectively.

4 Constructing $uhK(S)$ (Step 1)

4.1 The outline of the algorithm for constructing $uhK(S)$

Let $d = \log n$ and $m = n/d$. This section presents an algorithm which constructs $uhK(S)$ in $O(d + \log m + \log\log m \cdot \log d)$ time using m processors. Therefore, $uhK(S)$ can be constructed in $O(\log n)$ time using $n/\log n$ processors. In the algorithm, $uhK(S)$ is regarded as a rooted tree with the root s_1.

Partition $S = \{s_1, s_2, \ldots, s_n\}$ into m sets $\delta_1, \delta_2, \ldots, \delta_m$ of equal size, where $\delta_i = \{s_{(i-1) \cdot d+1}, s_{(i-1) \cdot d+2}, \ldots, s_{i \cdot d}\}$. It is clear that for any k $(1 \le k \le m)$, the vertices appearing in $uh(S_{k \cdot d})$ are those of $uh(\delta_i)$ $(1 \le i \le k)$. Therefore, we construct $uhK(S)$ by divide-and-conquer technique as follows.

(A) Partition S into $\delta_1, \delta_2, \ldots, \delta_m$, and compute $uh(\delta_i)$ in $O(d)$ time sequentially[2]. When the computation completes, the sequence of vertices of $uh(\delta_i)$ $(1 \le i \le m)$ is stored in array $uh\delta_i$: $[1..d]$ of point.

(B) Construct $uhK(S)$ from $uh(\delta_i)$ $(1 \le i \le m)$ in $O(\log m + \log\log m \cdot \log d)$ time using m processors. ∎

In Sections 4.2-4.4 we give the algorithm for (B).

4.2 A concise description of $uhK(S)$

For constructing $uhK(S)$ efficiently, we introduce a concise description for $uhK(S)$. Given $uh(\delta_i)$ $(1 \le i \le m)$, we show that $uhK(S)$ is determined by m edges of $uhK(S)$. We give the definition of these edges.

Definition 3 *(The connecting edge of $uh(\delta_i)$ on $uhK(S)$)*

For any i $(1 \le i \le m)$, let $l_i = (p'_i, p_i)$ be the upper common tangent of $uh(\delta_i)$ and $uh(S_{(i-1) \cdot d})$, where p_i is the t_ith vertex of $uh(\delta_i)$ and p'_i is the t'_ith vertex of $uh(\delta_{w_i})$ $(1 \le w_i \le i-1)$ (Fig.3). We call (p'_i, p_i) as the connecting edge of $uh(\delta_i)$ on $uhK(S)$, define it with a function, $cedge_S$, such that $cedge_S(i) = (t'_i, t_i, w_i)$. For convenience, let $cedge_S(1) = (1, 1, 1)$. ∎

By Definition 3, there are $m - 1$ connecting edges on $uhK(S)$ defined with function $cedge_S$. On $uhK(S)$, the following properties hold.

Property 1 *(1) If some vertex of $uh(\delta_i)$ belongs to $uh(S_{k \cdot d})$, then all the vertices of $uh(\delta_i)$ belonging to $uh(S_{k \cdot d})$ constitute a contiguous subsequence of $uh(\delta_i)$.*

That is, if the vertices s_a and s_b of $uh(\delta_i)$ $((i-1)\cdot d+1 \le a < b \le i\cdot d)$ are the vertices of $uh(S_{k\cdot d})$, vertex s_c $(a<c<b)$ of $uh(\delta_i)$ is also the vertex of $uh(S_{k\cdot d})$.

(2) Upper convex hull $uh(S_{i\cdot d})$ is determined by at most $i-1$ connecting edges on $uhK(S)$ as follows. Let $cedge_S(i_k) = (t'_k, t_k, i_{k-1})$, $cedge_S(i_{k-1}) = (t'_{k-1}, t_{k-1}, i_{k-2})$, $\ldots$, $cedge_S(i_2) = (t'_2, t_2, i_1)$, where $i_k = i$ and $i_1 = 1$. The vertex sequence of $uh(S_{i\cdot d})$ is the concatenation of the contiguous subseguence of $uh(\delta_{i_j})$ over j $(1 \le j \le k)$ (note the vertex sequence of $uh(\delta_j)$ is stored in array $uh\delta_j$):

$$
\begin{array}{llll}
uh\delta_{i_1}[t_1](t_1=1), & uh\delta_{i_1}[t_1+1], & \ldots, & uh\delta_{i_1}[t'_2], \\
uh\delta_{i_2}[t_2], & uh\delta_{i_2}[t_2+1], & \ldots, & uh\delta_{i_2}[t'_3], \\
 & \vdots & & \\
uh\delta_{i_k}[t_k], & uh\delta_{i_k}[t_k+1], & \ldots, & uh\delta_{i_k}[d]
\end{array}
$$

■

We call $cedge_S(i_1)$,$cedge_S(i_2)$,$\ldots$,$cedge_S(i_k)$ as the connecting edge sequence of $uh(S_{i\cdot d})$.

Given $uh(\delta_i)$ $(1 \le i \le m)$, we adopt m connecting edges on $uhK(S)$ as the concise description of $uhK(S)$. Therefore, in the algorithm for (B), we compute m connecting edges of $uhK(S)$.

4.3 Algorithm for (B)

Now we show the algorithm for (B), that is, to construct $uhK(S)$ (i.e., compute the function $cedge_S$) from $uh(\delta_i)$ $(1 \le i \le m)$ in $O(\log m + \log\log m \cdot \log d)$ time using m processors.

{*Algorithm*} Make$uhK(S)$

(*Input*) Arrays $uh\delta_1, uh\delta_2 \ldots, uh\delta_m$, where $uh\delta_i$ stores the sequence of vertices of upper convex hull $uh(\delta_i)$.

(*Output*) Array C: $[1..m][1..3]$ of integer. $C[i][1..3]$ stores $cedge_S(i)$ for each i $(1 \le i \le m)$. That is, if $cedge_S(i) = (t', t, j)$, $C[i][1]$,$c[i][2]$ and $C[i][3]$ store t', t and j respectively.

(0) If the number of points in S is at most d, then S is equal to δ_1 and $uhK(S)$ is equal to $uh(\delta_1)$. This completes the computation for this case.

(1) Let $m = n/d$. Divide $S(= \delta_1 \cup \delta_2 \cup \cdots \cup \delta_m)$ into $m^{1/3}$ equally-sized subsets, $S^1, S^2, \ldots, S^{m^{1/3}}$, where $S^i = \delta_{(i-1)\cdot m^{2/3}+1} \cup \delta_{(i-1)\cdot m^{2/3}+2} \cup \cdots \cup \delta_{i\cdot m^{2/3}}$.
Let
$S^i_{k\cdot d} = \delta_{(i-1)\cdot m^{2/3}+1} \cup \delta_{(i-1)\cdot m^{2/3}+2} \cup \cdots \cup \delta_{(i-1)\cdot m^{2/3}+k}$
$(1 \le k \le m^{2/3})$. Note $S^i = S^i_{m^{2/3}\cdot d}$ and $uhK(S^i)$ is the union of $uh(S^i_{k\cdot d})$ over k.

(2) Recursively
construct $uhK(S^i)$ (i.e., compute $cedge_{S^i}$ which is stored in $C[(i-1)\cdot m^{2/3}+1 \,..\, i\cdot m^{2/3}]$) for every $i = 1, 2, \ldots, m^{1/3}$ in parallel.

(3) Construct $uhK(S)$ by combining $uhK(S^1)$,$uhK(S^2)$,$\ldots$,$uhK(S^{m^{1/3}})$.

In the following, we explain the process of the combination, that is, to compute $cedge_S$ from $cedge_{S^i}$ $(1 \le i \le m^{1/3})$.

(3.1) Compute $uh(S^1 \cup S^2 \cup \cdots \cup S^j)$ for all j $(1 \le j \le m^{1/3})$.

(i) For each i $(1 \le i \le m^{1/3})$, pick out $uh(S^i)$ as follows.

Consider the connecting edge sequence of $uh(S^i)$ $(= uh(S^i_{m^{2/3}\cdot d}))$, that is, the sequence $C[i_1]$, $C[i_2]$, $\ldots$, $C[i_k]$, where $i_1 = (i-1)\cdot m^{2/3}+1$, $i_k = i\cdot m^{2/3}$, and $i_r = C[i_{r+1}][3]$ $(1 \le r \le k-1)$. Let $C[i_j][1] = t'_j$ and $C[i_j][2] = t_j$. Store the sequence $i_1, i_2, \ldots, i_k$ to array L_i: $[1..k]$ of integer such that $L_i[j] = i_j$. The sequence of vertices of $uh(S^i)$ is stored in k arrays: $uh\delta_{i_1}[t_1..t'_2]$,$uh\delta_{i_2}[t_2..t'_3]$,$\ldots$,$uh\delta_{i_k}[t_k..d]$.

(ii) For each i, j $(1 \le i < j \le m^{1/3})$, compute T^{ij}, the upper common tangent of $uh(S^i)$ and $uh(S^j)$ by Lemma 5 of Appendix.

(iii) For each j $(2 \le j \le m^{1/3})$, find T^j, the tangent with the smallest slope among T^{ij} $(1 \le i \le j-1)$.

(iv) For each j $(1 \le j \le m^{1/3})$, find $uh(S^1 \cup S^2 \cup \cdots \cup S^j)$ as follows.

Let $T^{j_k} = T^{j_{k-1}, j_k}, T^{j_{k-1}} = T^{j_{k-2}, j_{k-1}}, \ldots, T^{j_2} = T^{j_1, j_2}$, where $j_k = j$ and $j_1 = 1$.

(a) Store $j_1, j_2, \ldots, j_k$ to array I_j: $[1..m^{1/3}]$ of integer, such that $I_j[i] = j_i$.

Let $\widehat{uh(S^{j_i})}^j$ $(1 \le i \le k)$ be the maximal contiguous subsequence $uh(S^{j_i})$ such that all the vertices of $\widehat{uh(S^{j_i})}^j$ belong to $uh(S^1 \cup S^2 \cup \cdots \cup S^j)$. It can be seen that the sequence of vertices of $uh(S^1 \cup S^2 \cup \cdots \cup S^j)$ is the concatenation of $\widehat{uh(S^{j_1})}^j$, $\widehat{uh(S^{j_2})}^j$,$\ldots$,$\widehat{uh(S^{j_k})}^j$ (Fig.4). We consider how to pick $\widehat{uh(S^{j_i})}^j$ out of $uh(S^{j_i})$.

By (3.1)(i) of this algorithm, $uh(S^{j_i})$ is stored in arrays: $uh\delta_{l_1}[t_1..t'_2]$,$uh\delta_{l_2}[t_2..t'_3]$,$\ldots$, where l_r is stored in $I_{j_i}[r]$, and t'_r and t_r are stored in $C[l_r][1]$ and $C[l_r][2]$ respectively. Let p and q be the contact points of the upper common tangent T^{j_i} and $T^{j_{i+1}}$ on $uh(S^{j_i})$, respectively, where p is the xth vertex of $uh(\delta_{l_f})$ and q is the x'th vertex of $uh(\delta_{l_g})$ $(\delta_{l_f}, \delta_{l_g} \subseteq S^{j_i}$ and $f \le g)$. The the vertex sequence of $\widehat{uh(S^{j_i})}^j$ is stored in a list of arrays as follows: $uh\delta_{l_f}[x..t'_{f+1}]$,$uh\delta_{l_{f+1}}[t_{f+1}..t'_{f+2}]$,$\ldots$, $uh\delta_{l_g}[t_g..x']$.

(b) Let X, X', F and G be the arrays: $[1..m^{1/3}][1..m^{1/3}]$ of integer. Store x, x', f and g such that $X[j][i] = x$, $X'[j][i] = x'$, $F[j][i] = f$ and $G[j][i] = g$.

(3.2) For each $uh(\delta_k)$, where $\delta_k \subset S^j$, determine the connecting edge of $uh(\delta_k)$ on $uhK(S)$, $cedge_S(k)$, as follows.

(i) Find the upper common tangent of $uh(\delta_k)$ and $uh(S^1 \cup S^2 \cup \cdots \cup S^{j-1})$ by Lemma 5 of Appendix. Let q' and q be the contact points of the tangent, and q' be the t'th vertex of $uh(\delta_r)$ $(\delta_r \subset S^h, 1 \le h \le j-1)$ and q be the tth vertex of $uh(\delta_k)$.

(ii) Let the connecting edge of $uh(\delta_k)$ on $uhK(S^j)$, which is represented by $C[k][1..3]$, is (p', p). Compare the slope of tangent $\overline{q'q}$ with that of line seg-

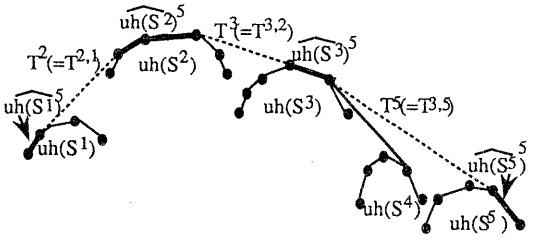

Figure 4: The constitution of $uh(S^1 \cup S^2 \cup \cdots \cup S^5)$

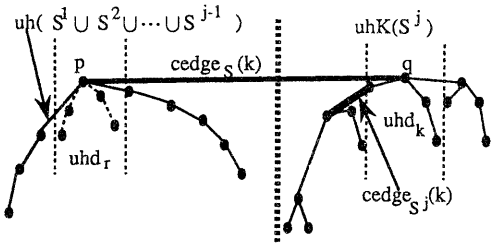

Figure 5: Determining the connecting edge of $uh(\delta_k)$ on $uhK(S)$

ment $\overline{p',p}$. If the slope of the tangent is smaller, then (q',q) becomes the connecting edge of $uh(\delta_k)$ on $uhK(S)$ (Fig. 5). In this case, set $C[k][1..3] := (t',t,r)$. Otherwise, (p',p) is the connecting edge of $uh(\delta_k)$ on $uhK(S)$. That is, $C[k]$ remains as before. ∎

Theorem 1 *The skeleton $uhK(S)$ of $uhT(S)$ can be constructed in $O(\log m + \log\log m \cdot \log d)$ time using m processors.*

Proof : We prove the theorem by analysing each step of (3).

In (3.1)(i), we store a list of size $O(m^{2/3})$ to array L_i by using the list ranking algorithm. For each i, this can be performed in $O(\log m)$ time using $m^{2/3}/\log m$ processors[7]. Therefore (3.1)(i) can be executed in $O(\log m)$ time using $m/\log m$ processors. In (3.1)(ii), for each i,j, common tangent T^{ij} can be computed in $O(\log(md))$ time using a single processor by lemma 5 of Appendix. Therefore, (3.1)(ii) can be performed in $O(\log md)$ time using $m^{2/3}$ processors. In (3.1)(iii), for each j, T^j can be computed in $O(\log m)$ time using $m^{1/3}/\log m$ processors by using the maximum finding algorithm[9]. Therefore (3.1)(iii) can be implemented in $O(\log m)$ time using $m^{2/3}/\log m$ processors. In (3.1)(iv)(a), for each j, array I_j can be gotten in $O(\log m)$ time using $m^{1/3}/\log m$ processors by list ranking similar to Step 1. Therefore (3.1)(iv)(a) can be executed in $O(\log m)$ time using $m^{2/3}/\log n$ processors. In (3.1)(iv)(b), arrays F, G, X, X' can be gotten in $O(1)$ time using $m^{2/3}$ processor. Therefore, (3.1)(iv)(b) can be executed in $O(\log m)$ time using $m^{2/3}/\log m$ processors. In (3.2)(i), the common tangent of $uh(\delta^k)$ and $uh(S^1 \cup S^2 \cup \ldots \cup S^{j-1})$ can be found in $O(\log(md))$ time using a single processor by Lemma 6 of Appendix. Since for each j, there are $m^{2/3}$ upper common tangents to be computed, (3.2)(i) can be executed in $O(\log(md))$ time using m processors. Step (3.2)(ii) can obviously be executed in $O(1)$ time using m processors.

From the above analysis, we get the following recurrence relation for the running time $T(m)$ of Make$uhK(S)$:

$T(m) = T(m^{2/3}) + O(\log(md))$

From the recurrence relation, we obtain $T(m)$=$O(\log m + \log\log m \cdot \log d)$. The number of proccesors used is m. ∎

We have constructed the skeleton of $uhT(S)$, $uhK(S)$, which is the subgraph of $uhT(S)$. In the nxet section, we show how to extend $uhK(S)$ to $uhT(S)$.

5 Constructing $uhT(S)$ (Step 2)

This section presents the algorithm which extends $uhK(S)$ to $uhT(S)$ in $O(\log n)$ time using $n/\log n$ processors. Let $d = \log n$. The algorithm consists of the following two steps.

(C) For every i $(1 \le i \le m)$, construct the upper convex hull tree of δ_i, $uhT(\delta_i)$ in $O(d)$ time seqentially by Graham scan[2].

(D) Construct $uhT(S)$ by using $uhK(S)$ and $uhT(\delta_i)$ $(1 \le i \le m)$ in $O(d)$ time using m processors. ∎

Therefore, $uhT(S)$ can be constructed in $O(\log n)$ time using $n/\log n$ processors. In what follows we explain the algorithm for (D). We first give the following lemma.

Lemma 1 *Let $U = \{v_1, v_2, \ldots, v_h, q_1, q_2, \ldots, q_h\}$ be an x-sorted set of points in the plane, and let $U_1 = \{v_1, v_2, \ldots, v_h\}$, $U_2 = \{q_1, q_2, \ldots, q_h\}$. When $uhT(U_1)$ and $uhT(U_2)$ are given, $uhT(U)$ can be constructed in $O(h)$ time using a single processor.*

Proof : We only need to determine the parent for each vertex of U_2, since for any i $(2 \le i \le h)$ $parent_U(v_i)$ is the same as $parent_{U_1}(v_i)$.

It can be seen that for any point q_i $(1 \le i \le h)$ of U_2, $parent_U(q_i)$ is either $parent_{U_2}(q_i)$ or the contact point of the tangent from q_i to $uh(U_1)$. Assume that U_2 is divided into two points sets: A={$q \in U_2$ — $parent_U(q)$=$parent_{U_2}(q)$} and B={$q \in U_2$ — $parent_U(q)$=the contact point of the tangent from q to $uh(U_1)$ }, where the points of both A and B are listed in order of increasing x-coordinate. We can make the following observations (Fig. 6).

Observation 1: Traverse the points of B from left to right . Whenever we arrive a new point we draw the tangent from this point to $uh(U_1)$. From the property of upper convex hulls, during traversing the contact point of the tangent on $uh(U_1)$ moves to the left only.

Observation 2: For any point q_j in U_2, q_j is a point of B iff there exists a point q_k $(k < j)$, called as *the milestone of* q_j, such that

(1) q_k is the point of B,

(2) $q_{k+1}, q_{k+2}, \ldots, q_{j-1}$ are all the points of A, and

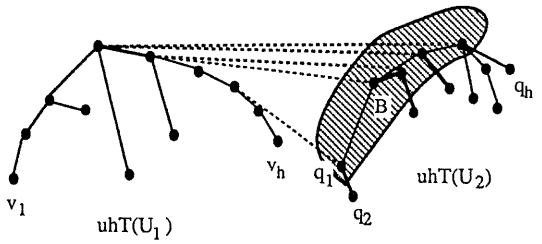

Figure 6: The proof of Lemma 1

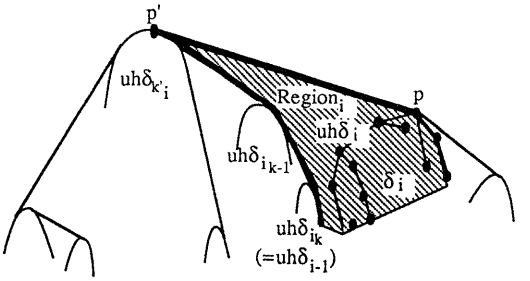

Figure 7: The proof of Lemma 2

(3) suppose that the contact point of the tangent between q_k and $uh(U_1)$ is $v_{k'}$, the slope of $\overline{v_{k'}, q_j}$ is smaller than the slope of $\overline{v_{k'}, q_k}$.

From above observations, we can compute the parents for all vertices of U_2 in $O(h)$ time by sweeping all these vertices from q_1 to q_h as follows. First, since q_1 must be the point of B, to compute the parent of q_1 we draw the tangent from q_1 to $uh(U_1)$ by sweeping from v_h towards v_1. That is, we check if line segment $\overline{q_1, v_h}$ is the tangent. This can be done in $O(1)$ time. If it is not the tangent, check line segment $\overline{q_1, v_{h-1}}$. We repeat this process, untill for some j, line segment $\overline{q_1, v_j}$ is the tangent between q_1 and $uh(U_1)$. Thus the parent of q_1 is v_j. Then judge if q_2 is in B (this can be done in $O(1)$ time by Observation 2 . Note that the milestone of q_2 is q_1 if q_2 is in B). If it is not in B, the parent of q_2 is $parent_{U_2}(q_2)$. Then judge if q_3 is in B. We repeat this process, untill we find a point, say q_i ($i \geq 2$), in B (then the milestones becomes q_i. We always keep the new milestone during processing). By Observation 1, to compute the parent of q_i we only need to draw the tangent from q_i to $v_j, v_{j-1}, \ldots, v_1$. Continuing in the same way, all parents of U_2 will be computed in $O(h)$ time. ∎

Lemma 2 *Given $uhK(S)$ and $uhT(\delta_i)$ $(1 \leq i \leq m)$,$uhT(S)$ can be constructed in $O(d)$ time using m processors.*

Proof : We show how to determine $parent_S(s_r)$ for any point s_r of δ_i $(1 \leq i \leq m)$ by utilizing $uhK(S)$ and $uhT(\delta_i)$. For any j $(1 \leq j \leq i-1)$, let $\widehat{uh(\delta_j)}^{i-1}$ be the maximal contiguous subsequence (possibly empty) of $uh(\delta_j)$ such that all vertices of $\widehat{uh(\delta_j)}^{i-1}$ belong to $uh(S_{(i-1)\cdot d})$. Let $cedge_S(i_1)$,$cedge_S(i_2)$, ...,$cedge_S(i_k)$ $(i_k = i-1, i_1 = 1)$ be the sequence of connecting edges of $uh(S_{(i-1)\cdot d})$ and let $cedge_S(i_j) = (t'_j, t_j, f_j)$, where $f_j = i_{j-1}$. Upper convex hull $uh(S_{(i-1)\cdot d})$ is formed by concatenating $\widehat{uh(\delta_{i_1})}^{i-1}$, $\widehat{uh(\delta_{i_2})}^{i-1}$, ..., $\widehat{uh(\delta_{i_k})}^{i-1}$, where $\widehat{uh(\delta_{i_j})}^{i-1}$ is the contiguous subsequence of vertice of $uh(\delta_{i_j})$ from the t_jth vertex to the t'_{j+1}th vertex.

Let the connecting edge of $uh(\delta_i)$ on $uh(S_{i\cdot d})$ be (p', p), where p' belongs to $uh(\widehat{\delta_{i_{k'}}})^{i-1}$ $(1 \leq k' \leq k)$. For the last point of δ_{i-1} and the last point of δ_i, p' is the nearest common ancestor in tree $uhT(S)$ (Fig. 7). If $parent_S(s_r) \neq parent_{S'}(s_r)$, $parent_S(s_r)$ must be the vertex of $uh(\widehat{\delta_{i_j}})^{i-1}$ $(k' \leq j \leq k)$. That is, if let $Region_i$ be the region enclosed by two paths of $uhT(S)$: one is the path from the last point of δ_{i-1} to p' and other is the path from the last point of δ_i to p' (Fig. 7), $parent_S(s_r)$ can only be contained in $Region_i$. Therefore, for each j $(h \leq j \leq k)$, We use one processor for $\widehat{uh(\delta_{i_j})}^{i-1}$, to check if $parent_S(s_r)$ belongs to $\widehat{uh(\delta_{j_i})}^{i-1}$ for each point s_r of δ_i. This can be done by using Lemma 1 in $O(d)$ time. Since the areas $Region_2$,$Region_3$,...,$Region_m$ are mutually disjoint, the number of processors required is m. ∎

Since $d = \log n$, we obtain the following theorem from Lemma 2.

Theorem 2 *The upper convex hull tree of S can be constructed in $O(\log n)$ time using $n/\log n$ processors on CREW.* ∎

6 Optimality

In this section, we prove the optimality of algorithm Make$uhT(S)$. Obviously, algorithm Make$uhT(S)$ is cost optimal. We prove it is also time optimal by showing that it requires $\Omega(\log n)$ time to construct the upper convex hull tree of a sorted set of n points in plane on CREW PRAM with a polynomial number of processors. We give the optimality by reducing the maximum finding problem to the problem of constructing the upper convex hull tree. We first give a lemma as follows.

Lemma 3 *Let $S = \{s_1, s_2, \ldots, s_n\}$ be a set of points in plane sorted in order of increasing x coordinate and $S' = \{s_n, s_{n-1}, \ldots, s_1\}$. Let $uhT(S)$ and $uhT(S')$ be the upper convex hull trees of S and S' respectively ($S'$is the set of points in plane sorted in order of decreasing x coordinate. $uhT(S')$ can be defined and constructed similarly to $uhT(S)$). Point s_k has the largest y coordinate in S if and only if $y(s_k) \geq y(parent_S(s_k))$ and $y(s_k) \geq y(parent_{S'}(s_k))$ hold.*

Proof : Obviously, if s_k has the largest y coordinate in S, $y(s_k) \geq y(parent_S(s_k))$ and $y(s_k) \geq y(parent_{S'}(s_k))$ hold. Assume $y(s_k) \geq y(parent_S(s_k))$ and $y(s_k) \geq parent_{S'}(s_k)$ hold. The condition $y(s_k) \geq y(parent_S(s_k))$ implies that s_k has the largest y coordinate in $S_k = \{s_1, s_2, \ldots, s_k\}$. That is because (1) for any s_j $(1 \leq j \leq k)$ of S_k s_j is contained in the region closed by $uh(S_k)$, and (2) s_k is of the largest

y coornate in $uh(S_k)$ by the property of upper convex hull. From the condition $y(s_k) \geq parent_{S'}(s_k)$, it follows that s_k is of the largest y coordinate in $S'_{n-k+1}=\{s_n, s_{n-1}, \ldots, s_{n-k+1}\}$ similarly. Therefore, s_k is of the largest y coordinate in S. ∎

Theorem 3 *It requires $\Omega(\log n)$ time to construct the upper convex hull tree of a sorted set of n points in plane on CREW PRAM with a polynomial number of processors.*

Proof: The problem of finding the maximum of integers can be reduced to the problem of constructing the upper convex hull tree of a sorted set of points in the plane as follows. Let $I=\{i_1, i_2, \ldots, i_n\}$ be a set of integers. Construct a set of points in plane sorted in increasing x coordinate $S=\{(1,i_1),(2,i_2),\ldots,(n,i_n)\}$ and compute $uhT(S)$ and $uhT(S')$ respectively, where $S'=\{s_n, s_{n-1}, \ldots, s_1\}$. Find point s_k such that $y(s_k) \geq y(parent_S(s_k))$ and $y(s_k) \geq parent_{S'}(s_k)$ hold. The largest integer of I is $y(s_k)$.

All of the operations except constructing $uhT(S)$ and $uhT(S')$ can be done in $O(1)$ time using n processors. $uhT(S')$ can be constructed in the same way as $uhT(S)$. Therefore, construction of $uhT(S)$ is at least as hard as the maximum finding problem for I. The lower bound for constructing $uhT(S)$ then is gotten from the $\Omega(\log n)$ lower bound for computing the maximum of I on CREW PRAM[2]. ∎

Conclusion

We have shown an optimal algorithm for solving the prefix convex hulls problem of a sorted n-point set in $O(\log n)$ time using $n/\log n$ processors in CREW PRAM. This, of course, immediately implies that the prefix convex hulls problem for a monotone polygon can be solved in the same complexity. The data structure we used is very simple. The prefix convex hulls problem has many application. This is not only because the convex hull problem has many applications, but also because we suspect that the prefix convex hulls problem itself may have the applications to the problems such as the convex rope problem and the visibility problem.

Appendix

We discuss the sequential methods for finding the upper common tangent of two upper convex hulls, which appear in algorihtm Make$uhK(S)$ of Section 4.3. For any two convex polygons, the following lemma holds.

Lemma 4 [11] *Let U and V be two convex polygon with the sizes h_1 and h_2 respectively. If each of U and V is given in an array, the upper common tangent of U and V can be found in $O(\log(h_1+h_2))$ time.* ∎

Let $S=\{s_1, s_2, \ldots, s_n\}$ be a sorted point set in the plane, d be the integer and $m = n/d$. Let $\delta_i = \{s_{(i-1)\cdot d+1}, s_{(i-1)\cdot d+2}, \ldots, s_{i\cdot d}\}$ $(1 \leq i \leq m)$ and $S^j = \{\delta_{(j-1)\cdot m^{2/3}+1} \cup \delta_{(j-1)\cdot m^{2/3}+2} \cup \cdots \cup \delta_{j\cdot m^{2/3}}\}$ $(1 \leq j \leq m^{1/3})$.

In algorithm Make$uhK(S)$, step (3.1)(ii) computes the upper common tangent of two upper convex hulls, $uh(S^i)$ and $uh(S^j)$, where

(1) the vertex sequence of $uh(S^j)$ is given in a list of f arrays: $uh\delta_{i_1}[t_1..t'_2], uh\delta_{i_2}[t_2..t'_3], \ldots, uh\delta_{i_f}[t_f..d]$, where $uh\delta_\tau$ is array: $[1..d]$ of point and $f \leq m^{2/3}$.

(2) $i_1, i_2 \ldots, i_f$ are given in array $I_i[1..f]$ such that $I_i[r] = i_\tau$.

(3) $t_1, t_2, \ldots, t_f$ are given in array $C[i_1..i_f][1]$ and $t'_1, t'_2, \ldots, t'_f$ are given in array $C[i_1..i_f][2]$, where C is array: $[1..m][1..3]$ of integer.

The vertex sequence of $uh(S^j)$ is given in the same way, if we change i, t and z and f above to j, z and g respectively.

Lemma 5 *The upper common tangent of $uh(S^i)$ and $uh(S^j)$ can be found in $O(\log(md))$ time.*

Proof :

Let U' and V' be two convex polygons formed by the sequences of vertices: $uh\delta_{i_1}[t_1], uh\delta_{i_2}[t_2], \ldots, uh\delta_{i_f}[t_f]$ and $uh\delta_{j_1}[z_1], uh\delta_{j_2}[z_2], \ldots, uh\delta_{j_f}[z_f]$, respectively. Since i_τ is stored in array $I[r]$ and t_τ is stored in array $C[i_\tau][1]$, U' can be considered to be given in an array. Similarly, V' can also be considered to be given in an array. Find the upper common tangent of U' and V' by Lemma 4 in $O(\log m)$ time. Let the contact points be $uh\delta_{i_x}[t_x]$ and $uh\delta_{j_y}[t_y]$. Let U'' be the convex polygon formed by the vertices given in $uh\delta_{i_{x-1}}[t_{x-1}..t'_x], uh\delta_{i_x}[t_x..t'_{x+1}]$ and V'' be the convex polygon formed by the vertices given in $uh\delta_{j_{y-1}}[t_{y-1}..t'_y]$, $uh\delta_{j_y}[t_y..t'_{y+1}]$. The upper common tangent of $uh(S^i)$ and $uh(S^j)$ must be one of the following three tangents[4]:

(1)the upper common tangent of $uh(S^i)$ and V'',

(2) the upper common tangent of U'' and $uh(S^j)$, and

(3) the upper common tangent of U'' and V''.

We can find which is the upper common tangent of $uh(S^i)$ and $uh(S^j)$ by determining which is above $uh(S^i)$ and $uh(S^j)$. It can be determined in $O(1)$ time just by checking whether the adjacents of the contact points of the tangent are below that tangent. In the following, we see how to find these three tangents.

The upper common tangent of U'' and V'' can be found in $O(\log d)$ time by Lemma 4. The upper common tangent of $uh(S^i)$ and V'' (the upper common tangent of U'' and $uh(S^j)$ can be found similarly) is found as follows. Find the upper common tangent of U' and V'' in $O(\log(m+d))$ time by Lemma 4. Let the contact points in U' be $uh\delta_{i_p}[t_p]$. Let U''' be the convex polygon formed by the vertices given in $uh\delta_{i_{p-1}}[t_{p-1}..t'_p]$ and $uh\delta_{i_p}[t_p..t'_{p+1}]$. The upper common tangent of $uh(S^i)$ and V'' is the same as that of V'' and U'''[4]. Find the upper common tangent of V'' and U''' by Lemma 4 in $O(\log d)$ time.

Therefore, the upper common tangent of $uh(S^i)$ and $uh(S^j)$ can be found in $O(\log(md))$ time. ∎

In algorithm Make$uhK(S)$, step (3.2)(i) computes the upper common tangent of two convex hulls, $uh(\delta_k)$

and $uh(S^1 \cup S^2 \cup \cdots \cup S^{j-1})$, where $uh(\delta_k)$ is given in an array $uh\delta_h$: $[1..d]$ of point, and $uh(S^1 \cup S^2 \cup \cdots \cup S^{j-1})$ is given in a list of arrays as follows:

$$\begin{array}{llll} uh\delta_{l_{f_1}}[x_1..t'_{f_1+1}], & uh\delta_{l_{f_1+1}}[t_{f_1+1}..t'_{f_1+2}], & \ldots, & uh\delta_{l_{g_1}}[t_{g_1}..x'_1] \\ uh\delta_{l_{f_2}}[x_2..t'_{f_2+1}], & uh\delta_{l_{f_2+1}}[t_{f_2+1}..t'_{f_2+2}], & \ldots, & uh\delta_{l_{g_2}}[t_{g_2}..x'_2] \\ \vdots & & & \\ uh\delta_{l_{f_k}}[x_k..t'_{f_k+1}], & uh\delta_{l_{f_k+1}}[t_{f_k+1}..t'_{f_k+2}], & \ldots, & uh\delta_{l_{g_k}}[t_{g_k}..x'_k], \end{array} \quad (2)$$

where $k \leq j$ and $g_r - f_r \leq m^{2/3}$. In addition,

(1) $f_1, f_2, \ldots, f_k$ is given in array $F[j][1..k]$ such that $F[j][i] = f_i$.

(2) $g_1, g_2, \ldots, g_k$ is given in array $G[j][1..k]$, such that $G[j][i] = g_i$.

(3) $x_1, x_2, \ldots, x_k$ is given in array $X[j][1..k]$, such that $X[j][i] = x_i$.

(4) $x'_1, x'_2, \ldots, x'_k$ is given in array $X'[j][1..k]$, where $X'[j][i] = x_i$.

(5) $l_{f_1}, l_{f_2}, \ldots, l_{f_k}$ is given is array $L_j[f_1..f_k]$.

Lemma 6 *The upper common tangent of $uh(\delta_h)$ and $uh(S^1 \cup S^2 \cup \cdots \cup S^{j-1})$ can be found in $O(\log(md))$ time.*

Proof : Let convex polygon U be formed by the sequence of vertices: $uh\delta_{l_{f_1}}[x_1], uh\delta_{l_{f_2}}[x_2], \ldots, uh\delta_{l_{f_k}}[x_k]$. Since f_i, l_{f_i} and x_i $(1 \leq i \leq k)$ are given in array F, L_j and X respectively, U can be considered to be given in an array. Find the upper common tangent of U and $uh\delta_h$ in $O(log(m+d))$ by Lemma 4. Let the contact point of the tangent on U be $uh\delta_{l_{f_i}}[x_i]$ and U' be the convex polygon formed by the sequence of vertices given in a list of arrays:

$$\begin{array}{l} A_1 = uh\delta_{l_{f_{i-1}}}[x_{i-1}..t'_{f_{i-1}+1}], \\ A_2 = uh\delta_{l_{f_{i-1}+1}}[t_{f_{i-1}+1}..t'_{f_{i-1}+2}], \\ \vdots \\ A_a = uh\delta_{l_{g_{i-1}}}[t_{g_{i-1}}..x'_{i-1}], \\ A_{a+1} = uh\delta_{l_{f_i}}[x_i..t'_{f_i+1}], \\ A_{a+2} = uh\delta_{l_{f_i+1}}[t_{f_i+1}..t'_{f_i+2}], \\ \vdots \\ A_{a+b} = uh\delta_{l_{g_i}}[t_{g_i}..x'_i], \end{array}$$

where $a = g_{i-1} - f_{i-1} + 1$ and $b = g_i - f_i + 1$. The upper common tangent of $uh(\delta_h)$ and $uh(S^1 \cup S^2 \cup \cdots \cup S^{j-1})$ is the same as that of U' and $uh\delta_h$[4]. Let convex polygon U'' be formed by the first vertex of A_r for all r:

$uh\delta_{l_{f_{i-1}}}[x_{i-1}], uh\delta_{l_{f_{i-1}+1}}[t_{f_{i-1}+1}], \ldots, uh\delta_{l_{g_{i-1}}}[t_{g_{i-1}}]$
$uh\delta_{l_{f_i}}[x_i], uh\delta_{l_{f_i+1}}[t_{f_i+1}], \ldots, uh\delta_{l_{g_i}}[t_{g_i}]$

For the same reason as U', U'' can be consider to be given in an array. Find the upper common tangent of U'' and $uh\delta_h$ in $O(log(m+d))$ by Lemma 4. Let the contact point of the tangent on U'' be the first vertex of A_w and U''' be the convex polygon formed by the sequence of vertices given in A_{w-1} and A_w. Then, the upper common tangent of U' and $uh\delta_h$ is the same as that of $uh\delta_h$ and U'''. Find the the upper common tangent of U''' and $uh\delta_h$ in $O(\log d)$ time by Lemma 4. Therefore, the upper common tangent of $uh(\delta_h)$ and $uh(S^1 \cup S^2 \cup \cdots \cup S^{j-1})$ is found in $O(\log(md))$ time. ∎

References

(1) A.Aggarwal, B.Chazelle, L.Guibas, C.O'Dunlaing, and C.Yap: Parallel computational geometry. Algorithmica, 3,293-327,1988.

(2) M.J.Atallah and D.C.Chen: An optimal parallel algorithm for the visibility of a simple polygon from a point. Proc. of the Fifth Annual ACM Symposium on Computational Geometry,114-123,1989.

(3) M.J.Atallah and M.T.Goodrich: Efficient parallel solutions to some geometric problems. Journal of Parallel and Distributed Computing, 3,492-507,1986.

(4) M.J.Atallah and M.T.Goodrich: Parallel algorithms for some functions of two convex polygons. Algorithmica, 3,535-548,1988.

(5) W.Chen,K.Nakano, T.Masuzawa, and N.Tokura: Optimal Parallel Algorithms for Finding the Convex Hull of a Sorted Point Set. Tech.Rep.Trans.IPSJ, AL22-4,1991.

(6) R.Cole and M.T.Goodrich: Optimal parallel algorithms for polygon and point set problems. In Proc. of the Fourth Annual ACM Symposium on Computational Geometry,1988.

(7) R.Cole: Faster optimal paralell prifix sums and list ranking. Information and Control,81,334-352,1989.

(8) P-O.Fjällström,J.Katajainen,C.Levcopoulos and O.Petersson: A sublogarithmic convex hull algorithm. Bit,30,378-384,1990.

(9) A.Gibbons and W.Rytter: Efficient parallel algorithms. Cambridge University Press,1988.

(10) M.T.Goodrich: Finding the convex hull of a sorted point set in paralle . Information Processing Letters, 26,173-179,1987.

(11) M.H.Overmars and J.V.Leeuwen: Maintenance of configurations in the plane. J.Comput. System Sci. 23,166-204,1981.

(12) F.P.Preparata and M.L.Shamos; Computational geometry: an introduction. Springer-Verlag,1985.