Time Convex Hull with a Highway*

Teng-Kai Yu D.T. Lee†
Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan
and Institute of Information Science, Academia Sinica, Nankang, Taipei, Taiwan.
tkyu@ntu.edu.tw, dlee@iis.sinica.edu.tw

Abstract

We consider the problem of computing the time convex hull of a set of points in the presence of a straight-line highway in the plane. The traveling speed in the plane is assumed to be much slower than that along the highway. The shortest time path between two arbitrary points is either the straight-line segment connecting these two points or a path that passes through the highway. The time convex hull, $CH_t(P)$, of a set $P$ of $n$ points is the smallest set containing $P$ such that all the shortest time paths between any two points lie in $CH_t(P)$. In this paper we give a $O(n \log n)$ time algorithm for solving the time convex hull problem for a set of $n$ points in the presence of a highway.

1. Introduction

The convex hull problem for a set $P$ of points in the plane is a classic problem in computational geometry, and the first algorithm with $O(n \log n)$ optimal time complexity was presented by Graham [7, 8] in 1972, where $n$ is the number of points in $P$. Since then, there have been quite a number of research results on variations of this classical problem or related problems, e.g., the Voronoi diagrams [1, 2, 3, 4, 5, 6, 9, 10, 12, 11]. A variation of the convex hull problem is defined as follows. Consider a set $P$ of points in the presence of a straight line, which models a high speed transportation highway. It is assumed that the traveling speed in the plane is much slower than that along the highway. A shortest time path between any pair of points in $P$ can be a straight-line segment connecting these two points directly or a polygonal path that passes through a portion of the highway. The time convex hull was defined by Abellanas, et al. [2]. The time convex hull, $CH_t(P)$, of a set of points $P$ in the plane in the presence of transportation highways, is the smallest set containing $P$ such that all shortest time paths between any two points in $CH_t(P)$ lie in $CH_t(P)$.

To the best of our knowledge, there are no subquadratic time algorithms for computing the time convex hull of a set of points in the presence of straight-line transportation highways (the formal definition is given in the next section) in the plane. The only method known amounts to generating all shortest time paths of all pairs of points and computing the time convex hull. It needs $O(n^2)$ time. In this paper, we will give an optimal $O(n \log n)$ algorithm to solve the problem for the case that has one straight-line highway.

2 Preliminaries

First, we will present the classical definition of the convex hull of the points in the Euclidean $d$-dimensional space $E^d$.

Definition 2.1 A subset $S$ of $E^d$ is called convex set if and only if the line segment connecting any pair of points in $S$ is completely contained in $S$. The convex hull of a point set $P$ is the smallest convex set containing $P$.

The time convex hull was defined below by M. Abellanas et al. [2], where the shortest path is measured by time instead of distance, referred to as the shortest time path.

Definition 2.2 A subset $S$ of $E^d$ is said to be a time convex set if and only if the shortest time paths connecting any pair of points in $S$ is completely contained in $S$. The time convex hull of a point set $P$ is the smallest time convex set containing $P$.

A very interesting model of this problem is the line transportation model. In this model besides the point set in the
plane, there exist some straight lines and every line has its own traveling speed different from the speed in the plane. We can imagine that we have a map in which the points correspond to cities and straight lines are highways. We further assume that any point on the highway can be an entry to the highway or an exit from the highway.

We will consider the simplest model with only one highway in the plane, i.e., the Straight-Line Transportation Model which is formally defined as below:

- There is a big highway crossing some area which we will describe as a line in the plane.
- Travelers can enter the highway at any point and travel along it at speed \( V \) in both directions and can exit from the highway at any point.
- Out of the highway travelers can move freely and the traveling speed in any direction is \( V_0 < V \).

Without loss of generality, we let highway \( H \) be positioned as the \( x \)-axis. And for convenience, we assume the traveling speed \( V_0 \) off the highway is 1. The time convex hull problem is defined as follows.

**Problem 2.3** Given a point set \( P \) and a horizontal line highway \( H \) in the plane with speed \( V \), where the speed in the plane except \( H \) is 1, find the time convex hull of \( P \) in the presence of \( H \), denoted by \( CH_t(P) \).

Figure 2-1 shows an example of \( CH_t \) of a point set in the presence of a highway \( H \). The time convex hull includes those five convex regions and the line segment in \( H \) shown in bold face.

![Figure 2-1 The \( CH_t \) of a point set with a highway \( H \).](image)

As shown in Figure 2-1, the time convex hull for Problem 2.3 may be disjoint and each component, called cluster, is a maximal convex region containing a subset \( P_i \subseteq P \), \( i = 1, 2, \ldots \), of points and a portion of the straight line \( H \).

As shown in Figure 2-1, we have 5 clusters, three of them are above \( H \) and two are below. We note that if two points are not in the same cluster, the shortest time path between them must pass through \( H \). If we have the clusters computed, finding the time convex hull would be easy. But how to find these clusters efficiently is our main question. The following lemma is obvious.

**Lemma 2.4** In the time convex hull problem defined in Problem 2.3, if points \( u \) and \( v \) belong to different clusters, then the shortest time path between \( u \) and \( v \) passes through highway \( H \).

### 3 Infinite-Speed Highway

In this section we will solve Problem 2.3 when the traveling speed of highway is infinite. On an infinite speed highway, any two points on the highway have time distance 0. If the highway \( H \) of Figure 2-1 is an infinite-speed highway, the \( CH_t \) will be as shown in Figure 3-1.

![Figure 3-1 The \( CH_t \) with an infinite highway \( H \).](image)

We observe that if there are more than one cluster in the time convex hull \( CH_t(P) \), every maximal convex region of \( CH_t(P) \) is the convex hull constructed by those points in the respective cluster and the two orthogonal projections on \( H \) of the leftmost and rightmost points, respectively, of the cluster. The leftmost (respectively rightmost) point of a point set is the one with the smallest (respectively largest) \( x \)-coordinate.

**Lemma 3.1** If there are more than one cluster in the time convex hull, then the maximal convex region of a cluster is the convex hull constructed by those points in the cluster and the two orthogonal projections on \( H \) of the rightmost and leftmost points of the cluster, respectively.

**Proof:**

Since there are more than one cluster, by Lemma 2.4 the shortest time path between every point in this cluster and any point in other cluster must pass through \( H \). In particular, the shortest time path between the leftmost (respectively rightmost) point and some point in another cluster passes through its orthogonal projection on \( H \). That is, the orthogonal projection of the leftmost (respectively rightmost) point on \( H \) must be contained in the maximal convex region of the cluster. It completes the proof.

In this paper, constructing the time convex hull involves three major steps: (1) Preprocessing. (2) Finding clusters. (3) Constructing the time convex hull. In the following three subsections, we deal with these three steps respectively.


3.1 Preprocessing

Let us first assume the set \( P \) of points have been divided into three subsets, \( P^+ \), \( P^- \) and \( P^* \), representing the subsets of points above, below and on \( H \) respectively, and the points in each subset have been sorted in non-descending order of their \( x \)-coordinates.

3.2 Finding Clusters

We shall consider only subset \( P^+ \) of points, since the arguments for \( P^- \) are similar. Unless otherwise specified, all the points considered in the following are those in \( P^+ \). Let \( d(u,v) \) denote the Euclidean distance between points \( u \) and \( v \), and let \( d(u,H) \) denote the shortest Euclidean distance between \( u \) and highway \( H \). We use subscript \( t \) to denote the time distance. For instance, \( d_t(u,v) \) denotes the time distance of the shortest time path between points \( u \) and \( v \).

In the case of infinite-speed highway, the points considered in the following are those in \( P^+ \). Let us first assume the set \( P \) of points is divided into two subsets, \( P_1 \) and \( P_2 \), such that their \( x \)-coordinates \( u_i, x < u_j, x \), if \( i < j \). For each point \( u_i \), we will determine the cluster \( C_k \) for some \( k \), to which it belongs. The clusters will also be ordered from left to right, in the sense that if \( u_i \in C_k \) and \( u_j \in C_\ell \), where \( i < j \) and \( k \neq \ell \), then \( C_k \) is left of \( C_\ell \). Without loss of generality, we will label the clusters from left to right as well, i.e., \( C_k \) is left of \( C_\ell \), if and only if \( k < \ell \). This ordering of clusters follows from the consecutive property described below.

**Property 3.2 consecutive property of points in a cluster:** If points \( u_i \) and \( u_j \), \( i < j \), are in a cluster \( C_k \), then all the points \( u_k, i < k < j \), must belong to the same cluster.

A corollary of the above is that the subsets of points belonging to different clusters \( C_k \) and \( C_\ell \), \( k \neq \ell \), are disjoint and ordered from left to right, i.e., \( u_i \in C_k, u_j \in C_\ell \) and \( k < \ell \) implies that \( i < j \).

As the set of points will be processed from left to right, for each point \( u \), we will consider only the right half of the discrimination parabola \( P_u \) as shown in Figure 3-3, and denote it by \( p_u \).

Figure 3-3 The right half discrimination parabola \( p_u \) of \( u \).

Figure 3-4 shows the four possibilities between two points \( u \) and \( v \). Note that if \( v \) dominates \( u \) (i.e., both \( x \)- and \( y \)-coordinates of point \( v \) are no less than those of point \( u \)), then \( p_u \) and \( p_v \) do not intersect. Otherwise, they intersect at a point. If \( v \) is in the cluster-region \( R_u \), \( u \) and \( v \) belong to the same cluster (as shown in Figure 3-4(a), (b)) if \( v \) is in \( R_u \), and \( u \) and \( v \) belong to distinct clusters. (as shown in Figure 3-4(c), (d)) In general, for each cluster \( C_i \) we consider the cluster region of \( C_i \), denoted \( R(C_i) \), which is defined to be the union of the cluster-regions of the points that belong to the same cluster, i.e., \( R(C_i) = \bigcup_{u \in C_i} R_u \).

We define the envelope of the cluster \( C_i \) as \( \mathcal{E}(C_i) \), which defines the right boundary of cluster region \( R(C_i) \), composed of a sequence of parabolic segments, each of which belongs to \( p_u \), for some point \( u \in C_i \). Each envelope \( \mathcal{E}(C_i) \) has its own endpoint. Note that in case point \( u \) is dominated by point \( v \), as shown in Figure 3-4(b), the right boundary of
the cluster \( C_i \) containing \( u \) and \( v \) will not contain \( p_u \). i.e., \( p_u \) is considered dominated by \( p_v \) as far as the computation of envelope \( E(C_i) \) is concerned.

If the envelope of a cluster \( C_j \) intersects another cluster \( C_i \) and \( i < j \), we say that \( C_j \) is bounded. The endpoint, denoted \( E(C_j) \), is the intersection point with another cluster \( C_i \) and \( i < j \).

The envelope of a cluster is said to be unbounded if its last parabolic segment is unbounded. The endpoint of an unbounded envelope is assumed to be at infinity \( \infty \).

\[ \begin{align*}
\text{Figure 3-4 Four possibilities between } p_u \text{ and } p_v. \\
\text{(a)} & \quad \text{(b)} \\
\text{(c)} & \quad \text{(d)} \\
\end{align*} \]

It is not difficult to see that a point \( u \in C \) contributes at most one parabolic segment to the \( E(C) \). Therefore, the total number of parabolic segments for all clusters is at most \( n \). The following property, which follows from the definition of discrimination parabola, concerning an envelope is important.

**Property 3.3 monotonicity property of envelope:** The envelope of each cluster is monotone both in \( x \) and in \( y \). That is, for any vertical line or horizontal line, it intersects the envelope at most once.

For each new point \( w \), we know that if \( w \) lies in the cluster region \( R(C_i) \), or \( w \) lies above\(^1 \) its envelope \( E(C_i) \), then \( w \) is included in cluster \( C_i \). Due to the consecutive property of clusters, we shall find the leftmost cluster \( C_k \) such that \( w \) lies above \( C_k \), and if \( w \) is included in cluster \( C_k \), then all clusters \( C_i, i > k \), will all be dissolved and included in cluster \( C_k \), as summarized in the lemma below.

\[ ^1 w \text{ lies above an envelope } E(C_i), \text{ if } w \text{ lies above } y \text{, where } t \text{ is the point of intersection between } E(C_i) \text{ and the vertical line that passes through } w; \text{ otherwise } w \text{ lies below the envelope.} \]

**Lemma 3.4** Let the clusters be \( C_1, C_2, \ldots, C_k \) for some \( k \geq 1 \), and let \( w \) be the rightmost point. If \( w \) is included in cluster \( C_i, i \leq k \), then those points in \( C_j, i < j \) also be included in cluster \( C_i \).

An envelope is considered active with respective to a vertical line \( V \) at \( x \), if the \( x \)-coordinate of its last endpoint is greater than \( x \), or equivalently its last endpoint lies to the right of \( V \) at \( x \).

Suppose we have maintained a list \( E(C_{i_1}), E(C_{i_2}), \ldots, E(C_{i_t}) \), \( i_1 < i_2 < \cdots < i_t \), of active envelopes of clusters with respective to vertical line \( V_{u_t} \) that passes through point \( u_t \).

Thus, for the next point \( w = u_{t+1} \) to be processed, we need to do the following: (i) Find the leftmost cluster \( C_i \) such that \( w \) joins \( C_i \). (ii) Adjust clusters and update \( E(C_i) \).

For part (i) we note that \( E(C_k) \) is monotonically increasing for any \( k \). We shall examine from left to right the parabolic segments of \( E(C_k) \). The parabolic segment that is left of the vertical line \( V_w \) passing through \( w \) can be ignored, and will never be examined again. As illustrated in Figure 3-5, we need to check if \( w \) lies above the parabolic segment, shown in bold face, of \( E(C_k) \).

\[ \begin{align*}
\text{Figure 3-5 Checking if } w \text{ lies above } E(C_k) \text{ or not.} \\
\text{We consider part (ii) now. Recall that we have maintained a list of active envelopes } E(C_i) \text{ for all } i \leq k \text{ for some } k, \text{ before we process the next point } w = u_{t+1}. \\
\text{Examine } E(C_i) \text{ in decreasing order of } i \text{ and for each } E(C_i) \text{ we scan the parabolic segments from left to right. It takes amortized } O(1) \text{ time and } O(n) \text{ time overall.} \\
\text{If the next point } w \text{ results in a new cluster } C_{k+1}, \text{ } E(C_{k+1}) \text{ is the parabolic segment from } w_h \text{ to } i_h, \text{ where} 
\end{align*} \]
and denotes the orthogonal projection of \( w \) onto \( H \).

If \( w \) joins cluster \( C_i \), the points in clusters \( C_{i+1}, \ldots, C_k \) will all join \( C_i \), i.e., may lie on another envelope \( E(C_j) \). If \( j < i \), \( i_w \) is the endpoint of \( C_i \). If \( j \geq i \), then we consider endpoint \( E(C_j) \) of \( E(C_j) \) and find \( C_j' \) to which \( E(C_j) \) belongs. Repeat until \( j' < i \). The last endpoint we consider is the endpoint of \( E(C_i) \). During the process we can update the envelope \( E(C_i) \) by concatenating them. It also takes amortized \( O(1) \) time and \( O(n) \) time overall.

Except the last cluster, other clusters are unchanged. Therefore, for part (ii) it takes for each point processed an amortized \( O(1) \) time, and \( O(n) \) time overall.

To sum up, other than preprocessing of the points which takes \( O(n \log n) \) time to sort the points from left to right, handling of the clusters and maintenance of the envelopes can be done in amortized \( O(1) \) time for each point and in \( O(n) \) time overall.

### 3.3 Constructing the Time Convex Hull

After we have obtained the clusters, we shall now describe how we compute the time convex hull. If there are two or more clusters, then following Lemma 3.1 every convex region of the time convex hull can be easily found in \( O(n) \) time, since the points are sorted.

When there is only one cluster, we need to determine if there exists a shortest time path of two points passing through highway \( H \). If the answer is false, then the time convex hull is the ordinary (Euclidian) convex hull, and \( O(n) \) time suffices. Otherwise, as we argue below, the convex region also satisfies the property that the orthogonal projection \( u_h \) (respectively \( v_h \)) onto \( H \) of the leftmost point \( u \) (respectively rightmost point \( v \)) must also be included in the time convex hull (cf. Lemma 3.1). In this case the amount of time taken to compute the time convex hull is also \( O(n) \), since the points are sorted.

Suppose there is only one cluster and there exist two points, say \( p \) and \( q \), such that the shortest time path between them passes through highway \( H \) as shown in Figure 3-6(a), where \( p' \) is the orthogonal projection of \( p \) onto \( H \). Let \( u \) be the leftmost point. So the region shown in Figure 3-6(a) must be contained in the time convex hull. Consider two points in the neighborhood of \( p \), one on \( \overline{u \cdot p} \) and one on \( \overline{p' \cdot p'} \). The shortest time path of these two points must be the straight line segment connecting them. By the definition of the time convex hull, it follows that the whole triangle \( \triangle u p p' \), shown in Figure 3-6(b) must be included in the time convex hull.

Next we consider a point \( w \) on \( \overline{u', p'} \) (see Figure 3-6(c)). We know that any point on \( p, p' \) must have a shortest time path to \( q \) passing through \( H \). If the shortest time path of \( w \) and \( q \) were the line segment \( \overline{w, q} \) without passing through \( H \), it will then imply that the intersection point of \( \overline{p, q'} \) and \( \overline{w, q} \) has a shortest time path to \( q \) without passing through \( H \), which is a contradiction. Thus the shortest time path of \( w \) and \( q \) must pass through \( H \), as shown in Figure 3-6(d). Therefore, we conclude that the shortest time path between \( u \) and \( q \) must pass through \( H \) via point \( u_h \), the orthogonal projection of \( u \) onto \( H \). The case for the rightmost point \( v \) is similar.

We have the following lemma.

**Lemma 3.5** If there is only one cluster in the time convex hull, and there exist two points \( p \) and \( q \) such that their shortest time path passes through \( H \), then the maximal convex region of the cluster is the convex hull constructed by those points in the cluster and the two orthogonal projections on \( H \) of the rightmost and leftmost points of the cluster, respectively.

To test if there exist two points \( p \) and \( q \) such that their shortest time path passes through \( H \) can be done by the same scan-line algorithm described above. It is trivial if there are points that lie above and below \( H \), then the number of clusters must be at least two. So we will assume that all the points are above \( H \). We need to perform the following test as a postprocessing step. Let us define for each point \( u \) the region that lies above the \( x \)-axis and above \( \overline{p, p} \) and denote it as \( \overline{p, p} \). We shall process the points from left to right, and maintain the intersection of \( \overline{p, p} \) for each point \( u \). Suppose the intersection \( \mathcal{R} = \bigcap \overline{p, p} \), \( \overline{p, p} \) for \( i = 1, 2, \ldots, \ell \) has been constructed. Note that the intersection is a convex set. When the next point \( w = u_{\ell+1} \) is examined, we first...
check if \( w \) lies in \( R \). To do so, we shall scan the parabolic segments, which define the right boundary of \( R \) from left to right to find the one that intersects the vertical line \( V \) passing through \( w \), and with a simple test we can determine if \( w \) lies above the parabolic segment \( p_u \), for some \( u \). If not, we conclude that the shortest time path between \( u \) and \( w \) must pass through \( H \), and we stop. Otherwise, we’ll need to update \( R \) and continue. To update, we’ll scan the parabolic segments, which define the right boundary of \( R \) from right to left to find the one that intersects \( p_w \). The parabolic segments scanned during the testing will not be examined again, so the total amount of time taken to maintain \( R \) is \( O(n) \).

### 3.4 Timing Analysis

In Section 3.1 it takes \( O(n \log n) \) time to sort the points. In Section 3.2, it takes \( O(n) \) time to find all clusters. And in Section 3.3 it takes \( O(n) \) time to compute all convex regions. Thus this algorithm has time complexity \( O(n \log n) \).

If the highway does not exist, this algorithm can solve the traditional convex hull problem. Since the convex hull problem has a lower bound of \( \Omega(n \log n) \) [7], the algorithm presented in Section 3 is asymptotically optimal.

**Theorem 3.6** The time convex hull problem defined in Problem 2.3 with \( V = \infty \) can be solved in time \( \Theta(n \log n) \), which is asymptotically optimal.

### 4 Bounded Speed Highway

In this section we will solve Problem 2.3 when the highway has bounded speed \( V \). We let the moving speed in the plane be \( 1 \) and the moving speed on the highway \( H \) be \( V \) and \( V > 1 \) (if \( V \leq 1 \), the highway is useless).

First we make the following observation. If the shortest time path between two points \( u \) and \( v \) passes through \( H \), their shortest time path is as shown in Figure 4-1. Unlike infinite speed model, the segment from point to highway will form an angle \( \alpha \) with vertical line, where \( \sin \alpha = \frac{1}{V} \). We call the angle \( \alpha \) as the incident angle of the highway \( H \).

![Figure 4-1 Shortest time path of \((u, v)\) with incident angle \(\alpha\)](image)

In particular, if the shortest time path from a point \( u \) to any point that passes through \( H \), the path beginning at \( u \) has two possible points of entry to \( H \), a left entry \( u' \) and a right entry \( u'' \), as shown in Figure 4-2, with incident angle \(\alpha \). We call the segment \( u, u'' \) as left entry segment and \( u, u' \) as right entry segment.

![Figure 4-2 Projecting points \(u'\) and \(u''\) to the highway.](image)

Similar to Section 3, constructing the time convex hull in this case also involves three major steps: (1) Preprocessing points. (2) Finding clusters. (3) Constructing the time convex hull.

Most of the algorithms described in Section 3 apply, except for some modifications regarding the computation of entry points to the highway. In Section 3 the orthogonal projection of any point is the point of entry to the highway, when the traveling speed is unbounded.

We shall discuss the steps that are different and need modifications.

First, the discrimination curve \( p_u \) of a point \( u \) is not just a simple parabola. \( p_u \) will be defined as the following, where the \( x \)- and \( y \)-coordinates of \( u \) are \( a \) and \( b \) respectively.

\[
p_u = \frac{\sqrt{(x - a)^2 + (y - b)^2} - b \cos^{-1} \alpha + ((x - y \tan \alpha) - (a + b \tan \alpha))/V + y \cos^{-1} \alpha \cdot x}{\sqrt{(x - a)^2 + (y - b)^2}}
\]

If any point \( v \) to the right of \( u \) lies below \( P_u \), then the shortest time path between \( u \) and \( v \) passes through \( H \). Figure 4-3 shows an example, where point \( v \) is on \( P_u \) so that \( d(u, v) = d(u, u') + d(u', v') + d(v'', v) \).

![Figure 4-3 The parabolic discrimination curve \( p_u \).](image)

Second, the convex regions defined by the clusters have different forms. The four typical types are shown at Figure 4-4. The entry points associated with each cluster will be computed as follows. A general rule of thumb is that on the left side of the convex region if we decide to use left entry (respectively right entry) point, we will identify the leftmost left entry point (respectively leftmost right entry point), and similarly on the right side of the convex region.
We consider first the case of only one cluster $C$. Without loss of generality we assume that the points are all in $P^+$. Then we use the method given in Section 3.3 to determine if any pair of points have shortest time path passing through $H$ in $O(n)$ time. It is trivial if there is no such pair. If there exists a pair of such points, the time convex hull is as shown in Figure 4.4 (b). The proof of correctness is similar to that given in Section 3.3.

If there are more than one cluster, we first decide for each cluster if the type of Figure 4-4(a) applies with some modification.

Find the leftmost cluster $C_i$ above $H$ and the two points $p$ and $q$, $(p$ may equal $q)$ of $C_i$, where $p$ has leftmost left entry point $p^\prime$ and $q$ has leftmost right entry point $q^\prime$. (1) If there exist any points on the right of the straight line directed from $p$ to $p^\prime$, denoted $\ell(p, p^\prime)$, we do nothing to this cluster. (2) If there are no points on the right side of the directed straight line $\ell(q, q^\prime)$, we select this right entry point $q^\prime$ as part of the convex region for the left side of $C_i$. (3) If neither (1) nor (2) holds, there must be some points on the right side of $\ell(q, q^\prime)$ and on the left side of $\ell(p, p^\prime)$. Those points must be either contained in the cluster $C_j$ on $H$ or contained in some clusters below $H$, we then select the leftmost cluster $C_k$ below $H$, and do the following to find the union of the leftmost clusters above and below $H$ respectively. (i) Find the left external common tangent of $C_i$ and $C_j$, which connects points $r_j$ of $C_i$ to $s_j$ on $H$. (ii) Find the left external common tangent of $C_i$ and $C_k$, which connects points $r_k$ of $C_i$ and $s_k$ of $C_j$, crossing $H$ at $s_k$. If $s_j$ is left to $s_k$, we choose $\tau_{js}$ as the left edge of $C_i$. If $s_k$ is left to $s_j$, we have $\tau_{ks}$ as the left edge of $C_i$. The bottleneck of this part is to find the common tangent. But it is known that the common tangent can be found in $O(\log n)$ time. For the rightmost cluster above $H$ and respectively for the two leftmost and rightmost clusters below $H$ we do the same.

The correctness of the computation of the convex regions for the clusters is omitted.

It can be easily seen that the modification needed to compute the convex regions of these clusters is at most $O(n)$.

**Theorem 4.1** The time convex hull problem as defined in Problem 2.3 with bounded $V > 1$ can be solved in $\Theta(n \log n)$ time, which is asymptotically optimal.

## 5 Conclusion

In this paper, we have presented an asymptotically optimal $\Theta(n \log n)$ algorithm for solving the time convex hull problem as defined in Problem 2.3 for a set of $n$ points in the plane. This is the first optimal algorithm presented to date for solving this problem. The online version of this problem when the points are given one at a time or the dynamic version of this problem when the points can be inserted or deleted are among the problems for future study. A simple method that achieves an $O((\log n)^2)$ time per point for the online version can be obtained. Whether this online version can be solved in optimal $O(\log n)$ time per point remains to be seen. Other more sophisticated model such as more than one straight-line highways, 3- or higher-dimensions, or other time metrics will be of interest as well.

## References


