Streaming Algorithms

Meng-Tsung Tsai

06/12/2018

Reminder

Final Project Presentation. Jun 22 10:10 - 12:40

Written Assignment # 3 Jun 24 23:55

The final grade will be announced on e3new by Jun 26. If you have any question regarding the grade, let me know no later than Jun 28.

D&C-based Streaming Algorithms

There are many divide-and-conquer algorithms that can be canonically "translated" into multiple-pass streaming algorithms.

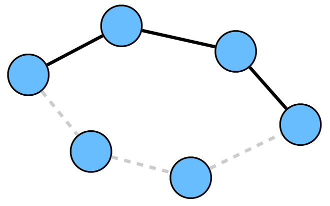
We will see an example problem, computing the convex hull, in today's lecture.

The upper hull

We only consider how to compute the upper hull in the following slides.

If one knows how to compute the upper hull, then he can compute the lower hull analogously.

Example.



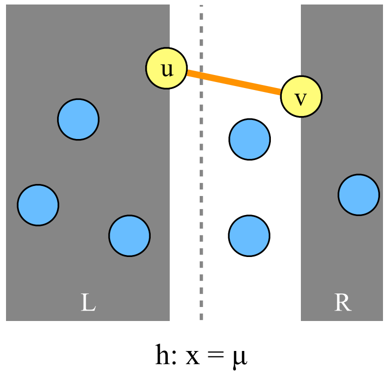
The upper hull is the set of hull edges so that all input points lie below them.

On a RAM

O(nlogh)-time algorithms

	Remove **r** hull edges at a time	Remove **r** extreme points at a time
r = 1	Kirkpatrick and Seidel 1986	Chan 1992
any **r** ≥ 1	Chan and Chen 2007	Farach-Colton et al. 2018

Kirkpatrik-Seidel algorithm

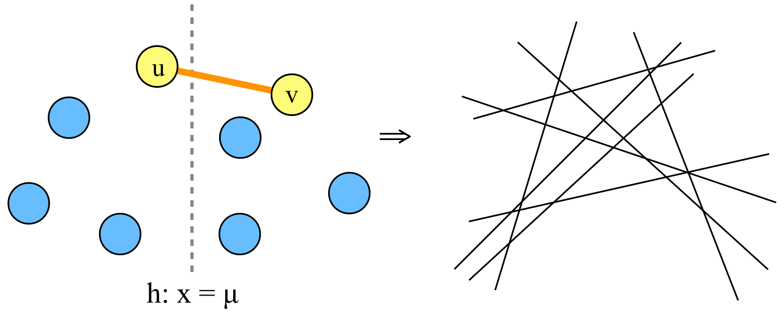


```
UH(P){
    Step 1. Find a line h: $x = \mu$ that
    partitions the points in P evenly.

    Step 2. Find the hull edge (u, v)
    that crosses h.

    Step 3. Let
    L = {p in P: $x(p) < x(u)$} and
    R = {p in P: $x(p) > x(v)$}.
    Recurse on UH(L) and UH(R).
}
```

Finding the hull edge by an LP

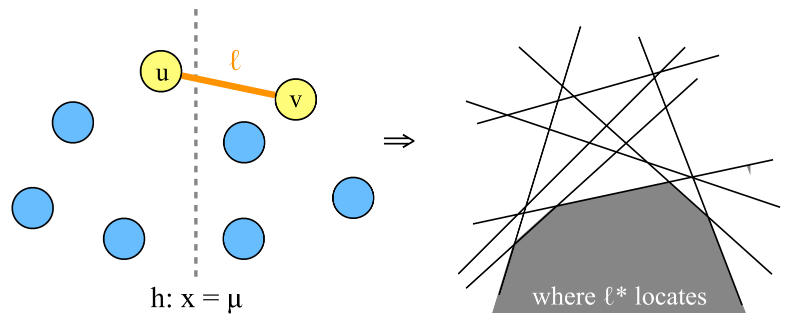
Point-Line Duality:

Map each point $p = (a, b)$ into a line p^*: $y = ax - b$, and map each line ℓ: $y = mx - c$ into a point $\ell^* = (m, c)$.



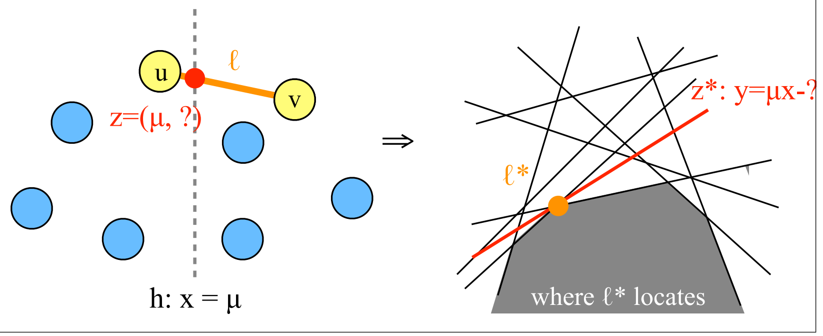
Finding the hull edge by an LP

Property: If p is below ℓ in the primal plane, then ℓ^* is below p^* in the dual plane. Since all points p in P lies below the line ℓ that passes through u and v, ℓ^* is below all p^*.



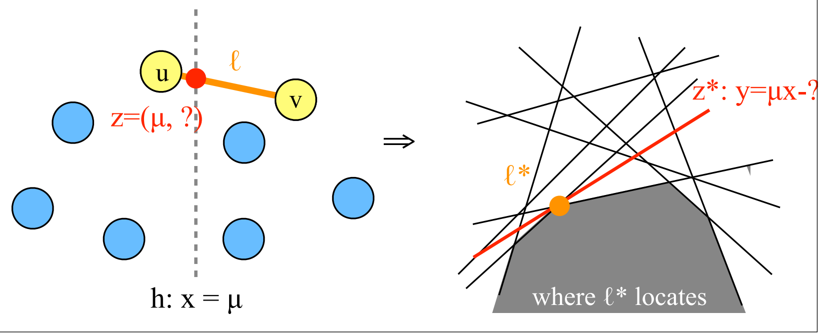
Finding the hull edge by an LP

Property: If p is on ℓ in the primal plane, then ℓ^* is on p^* in the dual plane. Let ℓ and h intersect at $z = (\mu, ?)$. Then, z^*: $y = \mu x - ?$ that touches ℓ^* and ℓ^* is the only point in the feasible region that can touch with z^*. (Why?)

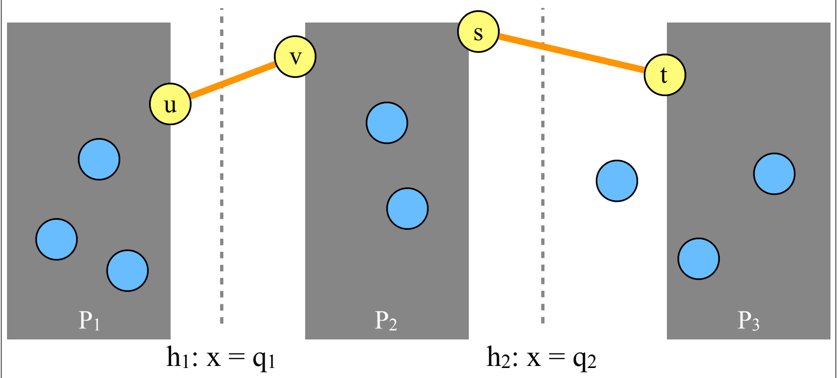


Finding the hull edge by an LP

As a result, finding the hull edge in the UH that crosses a line is equivalent to solving a linear program.



Chan and Chen's algorithm



CC's algorithm is a generalization of KS's by partitioning P into $r \geq 1$ smaller subsets and recursing on subproblems.

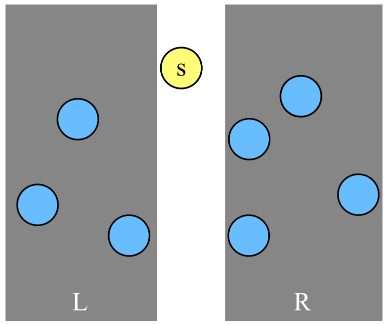
The advantages of CC's algorithm

[1] The computation tree of the recursion algorithm has a smaller height. In particular, if $r = n^\delta$, then the height is some constant.

[2] The size of working space that a computation node needs is $O(n^\delta)$, which is relatively small compared to n. We will see why this is true in a moment.

CC's algorithm can be adapted to the streaming model. It needs $O(hn^\delta)$ space and $O(1/\delta^2)$ passes.

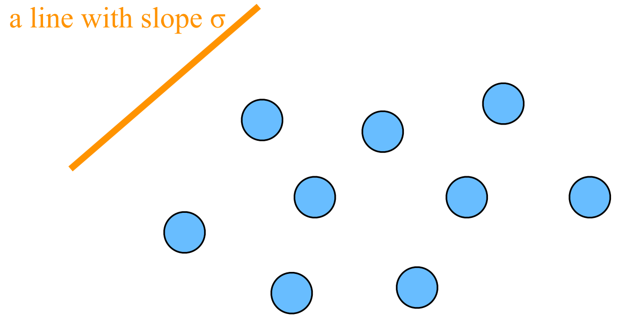
Chan's algorithm



```
UH(P){
    Step 1. Find an extreme point s
    that partitions the points in P
    evenly.
    Step 2. Let
    L = {p in P: x(p) < x(s)} and
    R = {p in P: x(p) > x(s)}.
    Recurse on UH(L) and UH(R).
}
```

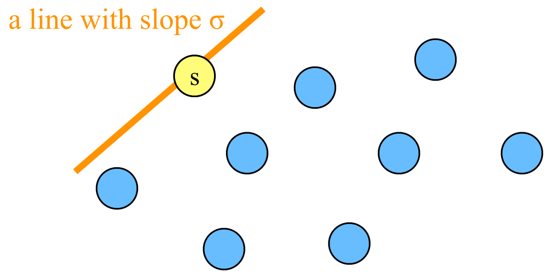
Finding the extreme point by selection

Pick a slope σ. Let s be the extreme point that supports σ.



Finding the extreme point by selection

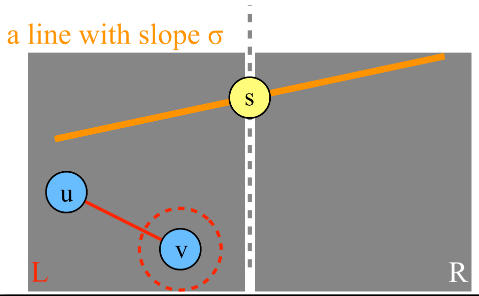
Pick a slope σ. Let s be the extreme point that supports σ.



Does the above procedure need to solve an LP?

Finding the extreme point by selection

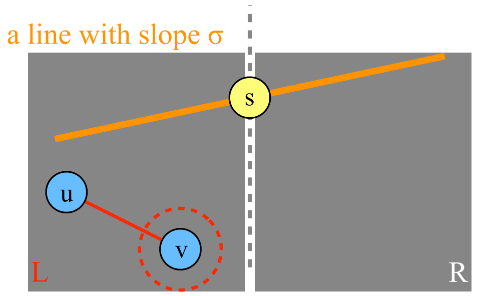
Group the points in P into n/2 pairs. The slope σ is picked as the median slope of the n/2 point pairs.



If the point pair (u, v) has slope at most σ, then point v cannot be an extreme point in L, i.e. can be pruned.

Finding the extreme point by selection

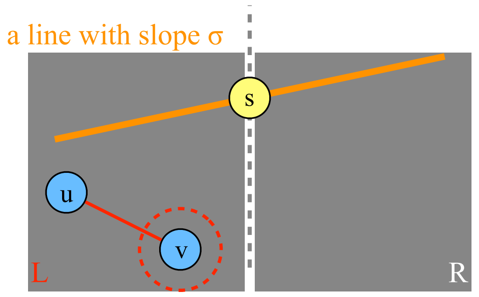
Group the points in P into n/2 pairs. The slope σ is picked as the median slope of the n/2 point pairs.



Since σ is the median slope of all point pairs, after the pruning, there are at most 3n/4 points in L.

Finding the extreme point by selection

Group the points in P into n/2 pairs. The slope σ is set as the median slope of the n/2 point pairs.



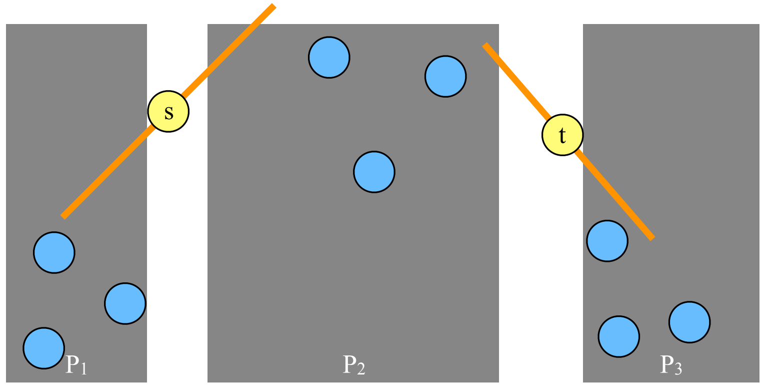
In this way, we find an extreme point that partitions the point set P "evenly".

Challenges

Suppose that we pick more extreme points to partition P into finer subsets.

It might be the case that a major portion of point pairs fall within the same slab, but at most one point can be pruned in each pair. Thus, the partition could be "uneven".

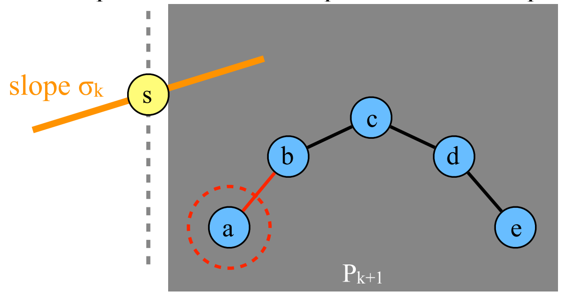
Our algorithm



Our algorithm is a generalization of Chan's by partitioning P into $r \geq 1$ smaller subsets and recursing on subproblems.

Finding the extreme points by selection

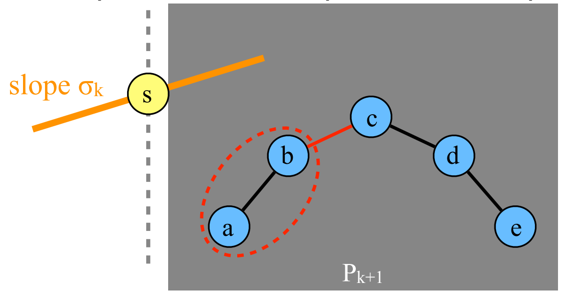
Group the points in P into n/(r+1) groups.
Find the hull edges in the upper hull of each group.
The slope σ's are set as the quantiles of the slope of the hull edges.



If the point pair (a, b) has slope at most σ, then point a cannot be an extreme point in P_{k+1}, i.e. can be pruned.

Finding the extreme points by selection

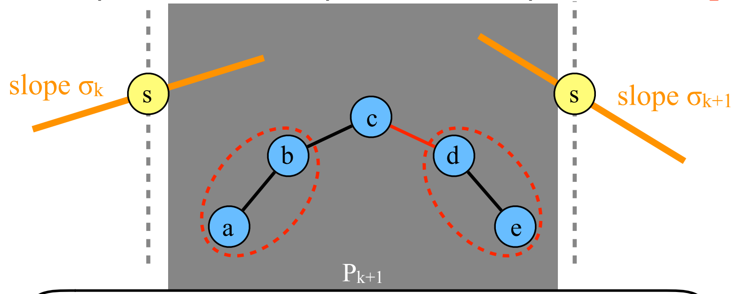
Group the points in P into n/(r+1) groups.
Find the hull edges in the upper hull of each group.
The slope σ's are set as the quantiles of the slope of the hull edges.



If the point pair (b, c) has slope at most σ, then point b cannot be an extreme point in P_{k+1}, i.e. can be pruned.

Finding the extreme points by selection

Group the points in P into n/(r+1) groups.
Find the hull edges in the upper hull of each group.
The slope σ's are set as the quantiles of the slope of the hull edges.



Prune similarly on the other side. Since σ_k and σ_{k+1} are quantiles, there are at most 2n/(r+1) points in each P_{k+1} after pruning.

Running time of these four algorithms

Given the computation tree of a recursive algorithm, if:

(1) the total running time contributed by each level is $O(L)$,

(2) the running time needed by a child node is a constant fraction of the parent node,

(3) there are $O(h)$ internal computation nodes,

then the running time of the entire computation is $O(L \log h)$.

By appealing to the above lemma, the running time of the introduced four algorithms are all $O(n \log h)$.

In the Streaming Model

Streaming algorithm

```
UH(P){
    Pass 1. Find r quantile slopes from the slopes
    of the hull edges of each small convex hull

    Pass 2. Find the extreme points that supports
    these quantile slopes.

    Pass 3. Use the found extreme points to prune
    the point set P, and on the fly output the point set
    $P_1, P_2, ..., P_{r+1}$. // the same pass as the first pass of
    its child subproblems

    Recurse on UH($P_1$), UH($P_2$), ..., UH($P_{r+1}$).
}
```

Streaming algorithm

We pick $r = n^\delta$, and the computation tree has height $2/\delta = O(1/\delta)$. As a result, the pass complexity is $O(1/\delta)$.

We have at most h computation nodes, each of which needs to memoize n^δ quantiles. The approximate quantile of n values can be found using $O(\log n)$ space. In total, the space complexity is $O(h\, n^\delta \log n)$.