Output-Sensitive Construction of the Union of Triangles

Eti Ezra and Micha Sharir

Problem

Given a collection $T = \{\Delta_1, \ldots, \Delta_n\}$ of n triangles in the plane, such that there exists a subset $S \subseteq T$ (unknown to us),

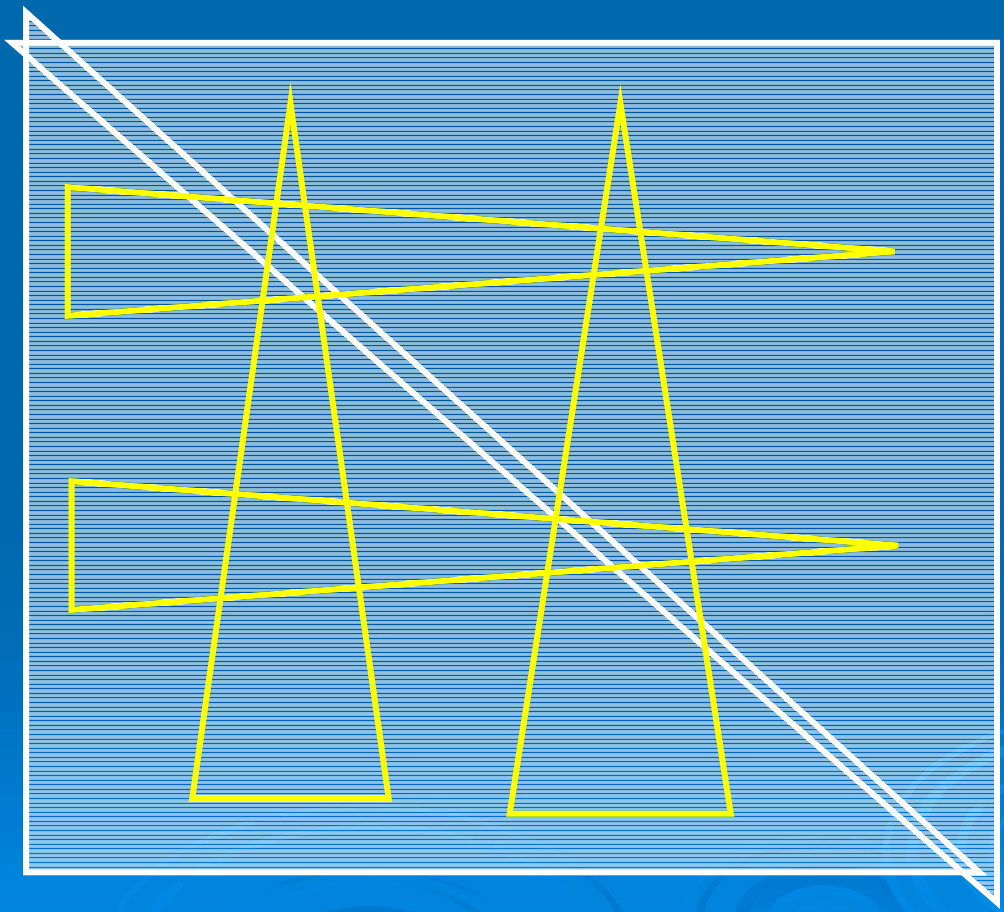
of $\xi << n$ triangles, such that

$$\bigcup_{\Delta \in S} \Delta = \bigcup_{\Delta \in T} \Delta ,$$

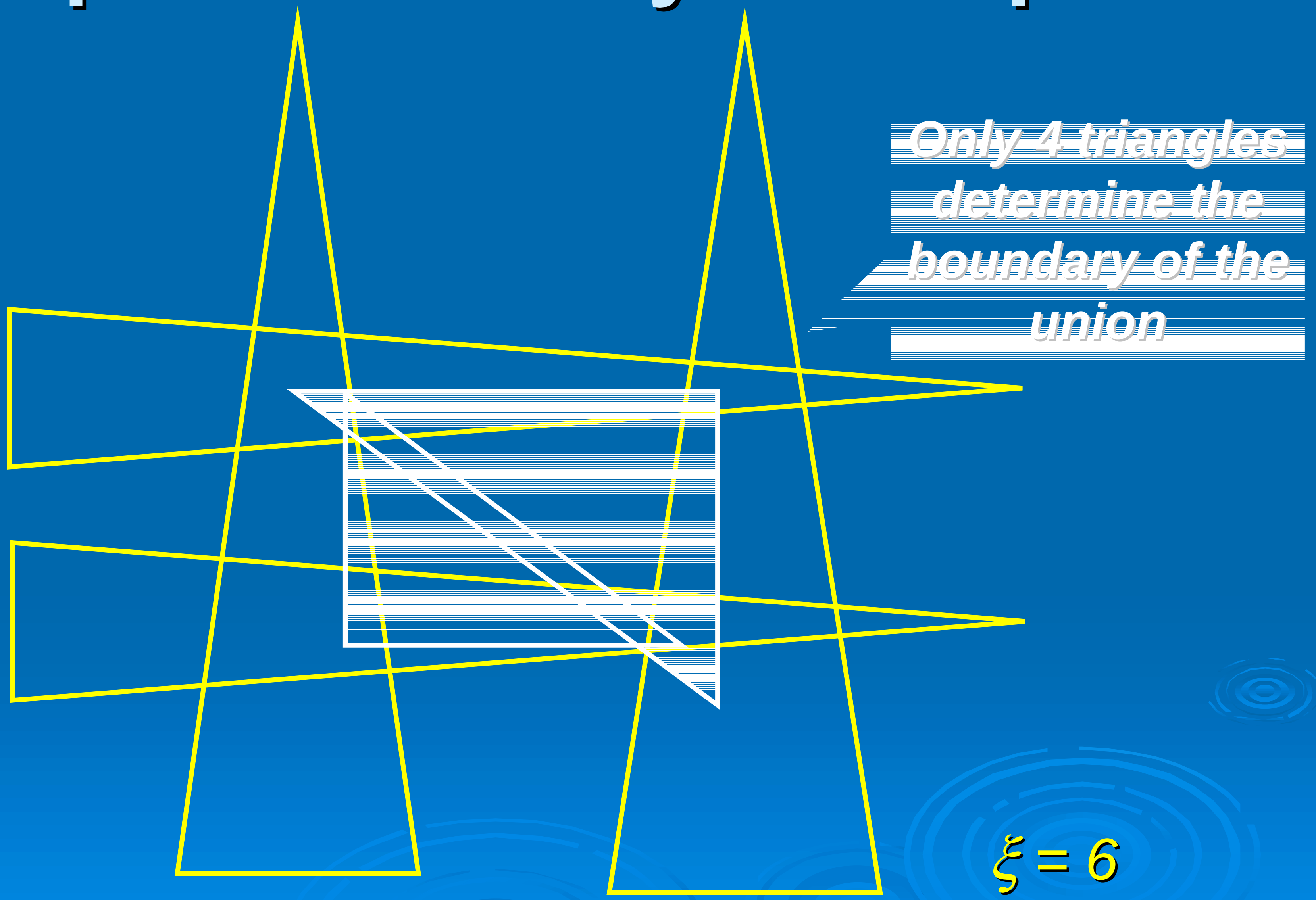
construct efficiently the union of the triangles in T.

Output Sensitivity: Example I

$\xi = 2$



Output Sensitivity: Example II



Computing the union

Constructing the arrangement of the triangles – too slow! $O(n^2)$

Output-sensitive algorithm?

unlikely to exist!

3SUM === HOLE-IN-UNION

⇓

The best known solutions to problems from the 3SUM-hard family require $\Theta(n^2)$ time in the worst case.

Our Result

We show that when there exists a subset $S \subseteq T$ of $\xi << n$ triangles, such that

$$\bigcup_{\Delta \in S} \Delta = \bigcup_{\Delta \in T} \Delta \,,$$

the union can be constructed in subquadratic time.

(for a reasonable range of ξ)

We use the Disjoint-Cover (DC) algorithm [Ezra, Halperin, Sharir 2002]

The DC algorithm – General idea

- Suppose we are given the set V^+ of all vertices of the arrangement *A(T)*, that are contained in the interior of the union.

- Run a greedy algorithm that inserts the triangles into the union by their weights.

  The weight of a triangle $\Delta \in T$:
  The number of uncovered vertices inside the triangle.

Disjoint-Cover (DC) algorithm

- Suppose we have inserted $(\Delta_1,\ldots,\Delta_j)$.
- For each remaining triangle Δ, we temporarily set S_Δ to be the set of all vertices of V^+ in the interior of Δ that are not covered by $\cup_{i<=j} \Delta_i$.
- Set $W(\Delta) = | S_\Delta |$ for each remaining triangle.
- Set Δ_{j+1} to be the triangle with the <u>maximum</u> weight.
- Update the union to include Δ_{j+1} .
- Proceed until all triangles have been chosen.

Preprocessing stage : Example step1

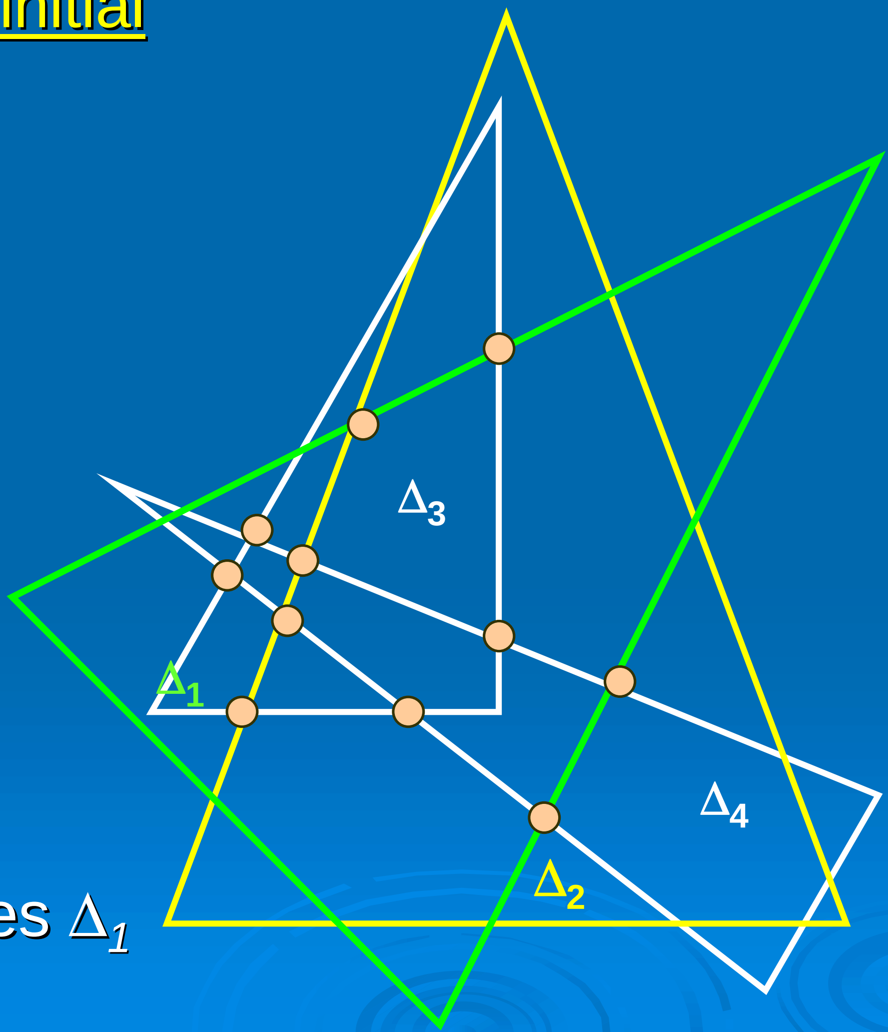
<u>Temporary initial weights</u>

W(Δ_1)=7

W(Δ_2)=5

W(Δ_3)= 3

W(Δ_4)=0



DC Chooses Δ_1

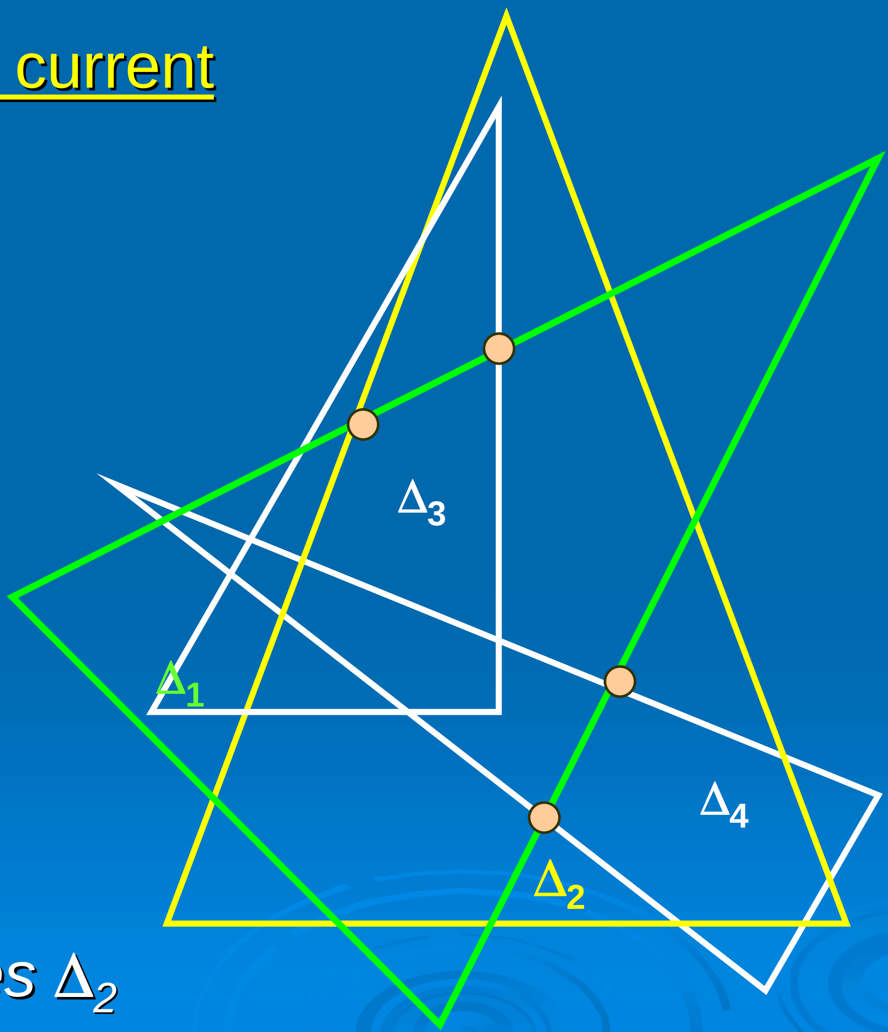
Preprocessing stage: Example step2

Temporary current weights:

$W(\Delta_2)=3$

$W(\Delta_3)=1$

$w(\Delta_4)=0$



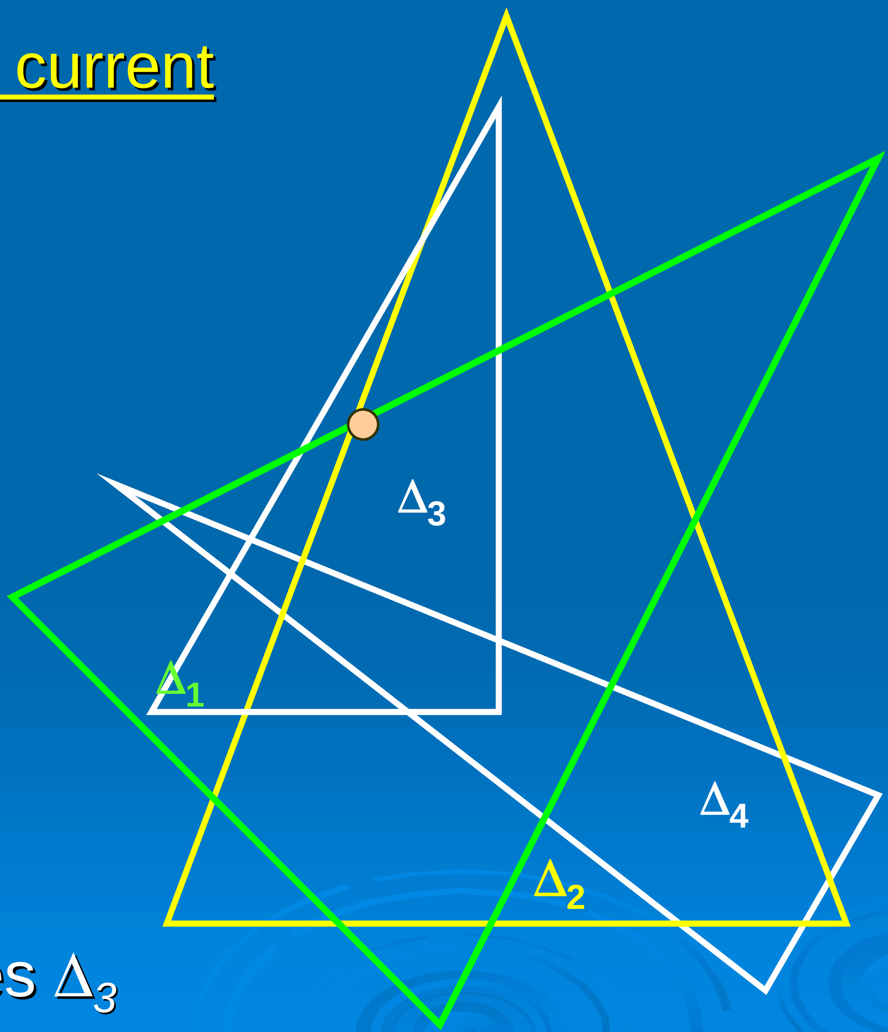
DC chooses Δ_2

Preprocessing stage: Example step3

Temporary current weights:

$W(\Delta_3)=1$

$w(\Delta_4)=0$

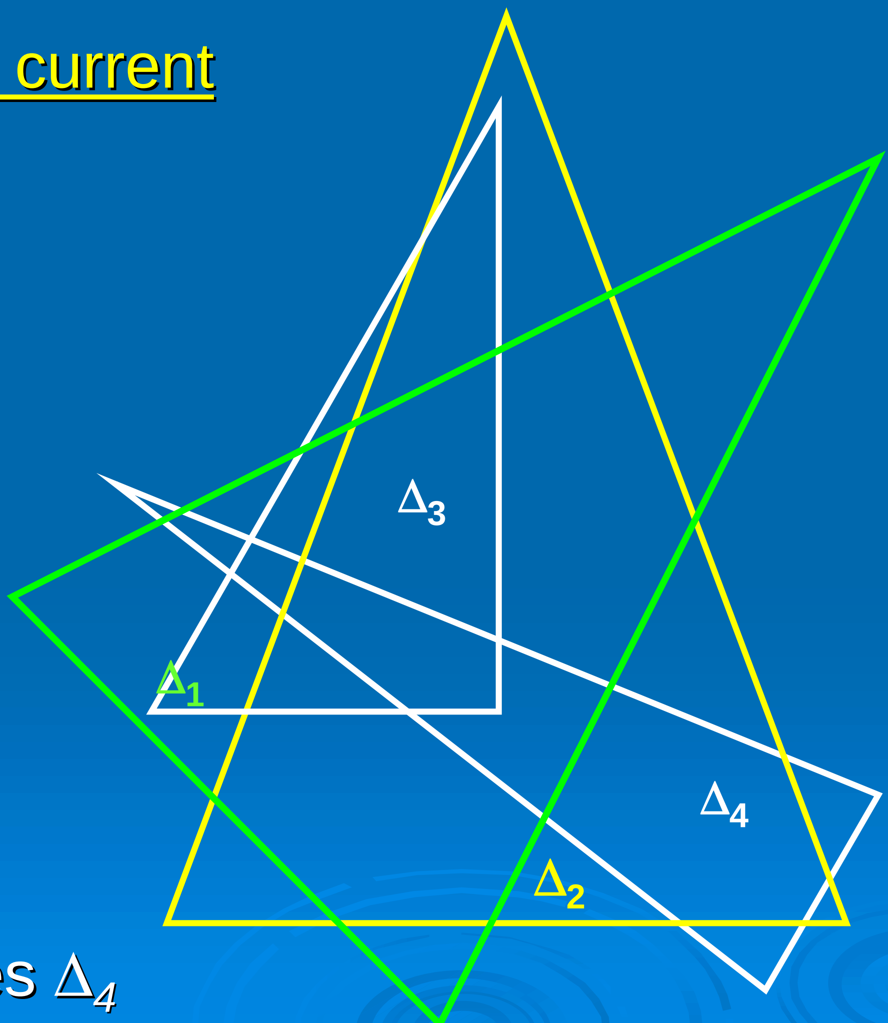


DC chooses Δ_3

Preprocessing stage: Example step4 (final)

<u>Temporary current weights:</u>

$w(\Delta_4)=0$



<u>Final weights:</u>

$W(\Delta_1)=7$

$W(\Delta_2)=3$

$W(\Delta_3)=1$

$w(\Delta_4)=0$

$(\Delta_1, \Delta_2, \Delta_3, \Delta_4)$

DC chooses Δ_4

The residual cost of the DC algorithm

How many vertices at positive-depth are constructed by the DC algorithm?

Set Cover in Hypergraphs

$H(V,E)$: $V = V^+$; $E = T$

Since there are ξ triangles that cover V^+, a greedy algorithm will find a cover of size $O(\xi \log n)$.

⇓

The residual cost of the DC algorithm is

$O(\xi^2 \log^2 n)$.

The approximate DC algorithm

We replace V^+ with a (small) random subset $R \subset V^+$.

$$|R| = r = \Omega(\xi t \log n)$$

t is a parameter of the quality of the sampling

We resample R on every iteration of the DC algorithm.

Approximate weights

Approximate weight of Δ : number of uncovered vertices of *R* inside Δ.

The algorithm proceeds as above, using approximate weights to determine the insertion order.

Analysis: General idea

The approximate weights of the heavy triangles ($W(\Delta) > \frac{\kappa}{t\xi}$, $\kappa = |V^+|$) reflect their actual weights.

$\Downarrow$

The approximate DC algorithm chooses at the j-th iteration a triangle whose real weight does not differ significantly from the largest real weight at this step.

The residual cost of the approximate DC algorithm

<u>Theorem</u>

Let $T = \{\Delta_1,\ldots,\Delta_n\}$ be a given collection of n triangles in the plane, such that ξ of them determine the union of the triangles in T.

Let $\kappa = |V^+|$, and $r = \Omega(\xi t \log n)$.

Then the residual cost of the approximate DC algorithm is $O(\xi^2 \log^2 t + \kappa/t)$, with high probability.

The first $q = O(\xi \log t)$ heavy triangles.

Implementation

- Sampling R
- Computing the insertion order
- The actual construction

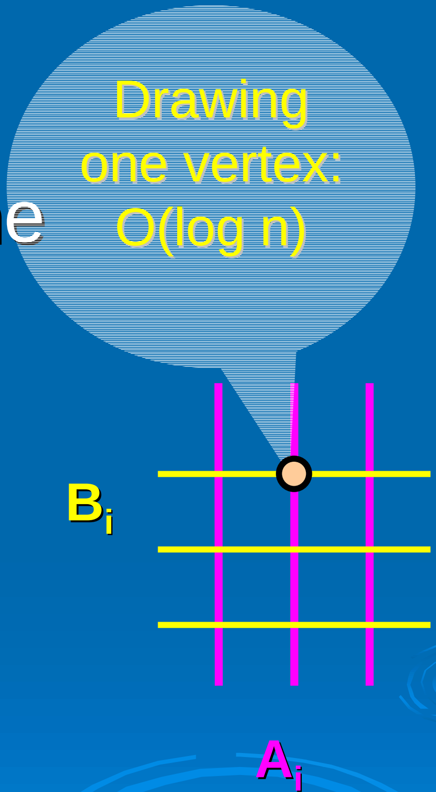
q times

$q = O(\xi \log t)$ is the number of the first triangles leaving $< \kappa/t$ uncovered vertices.

Sampling R

$\Phi = \{ e_i \mid e_i \text{ is an edge of } \Delta \in T\}$

Obtain a compact representation of the intersection graph of Φ as
$\cup\{A_i \times B_i\}_i$, where
$\Sigma_i(|A_i| + |B_i|) = O(n^{4/3+\varepsilon}), \forall\ \varepsilon > 0$.
[Agarwal 91]



The construction takes $O(n^{4/3+\varepsilon})$ time, for any $\varepsilon > 0$.

Overall running time

- Preprocessing : $O(n^{4/3+\varepsilon})$
- Drawing R: $O(r \log n)$ time
- Sample at each of the first $q = O(\xi \log t)$ iterations of the DC algorithm.
  Overall running time : $O(n^{4/3+\varepsilon} + qr \log n)$.

Computing the insertion order

Use (standard) range-searching machinery.

- Exclude all vertices of R in the interior of $\cup_{i<j} \Delta_i$: $O(j^{2/3+\varepsilon} r^{2/3+\varepsilon})$
- Perform a range-searching query on the remainder of R with $\Delta_{j, \dots,} \Delta_n$: $O(n^{2/3+\varepsilon} r^{2/3+\varepsilon})$

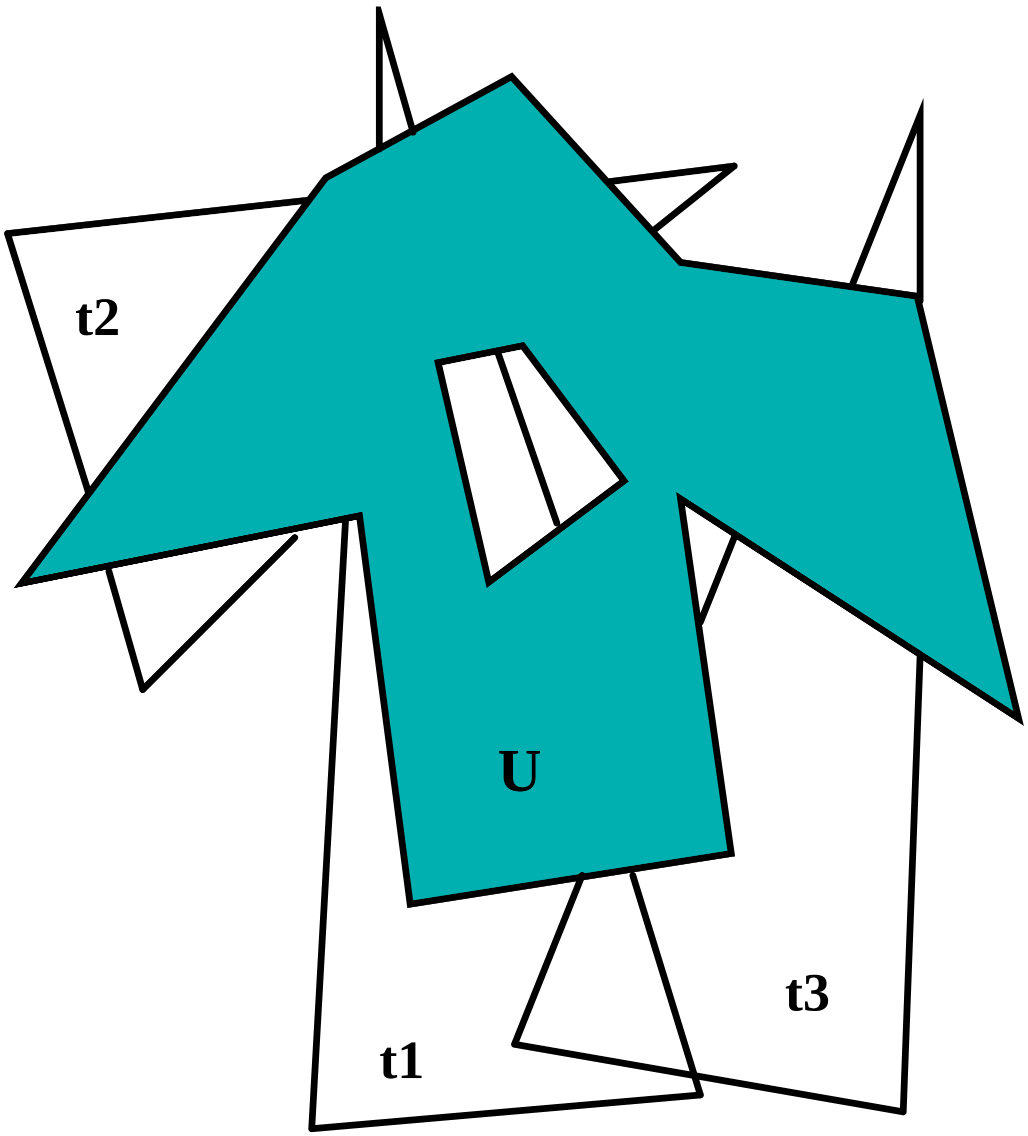
Repeating the procedure for q steps: $O(qn^{2/3+\varepsilon} r^{2/3+\varepsilon})$

The actual construction of the union

Divide the process into two stages:

- Construct the union of all the first q ('heavy') triangles.
- Insert all the remaining triangles (covering $\leq \kappa/t$ positive-depth vertices).

U is the union of the first q triangles.



t1, t2, t3 are the remaining triangles.

Overall running time

Constructing U

Reporting the intersections between U and the remaining triangles

$$O(q^2) + O(n^{2/3+\varepsilon} q^{4/3+\varepsilon} + \kappa/t) +$$

$$O(q^2 + (\xi^2 + \kappa/t) \log n)$$

Sweep-line

Overall running time of the algorithm

$O(n^{4/3+\varepsilon} + qr \log n + qn^{2/3+\varepsilon} r^{2/3+\varepsilon} + n^{2/3+\varepsilon} q^{4/3+\varepsilon} + q^2 + (\kappa/t+\xi^2)\log^2 n)$

Choose $t = \max\{\kappa^{5/3}/\xi n^{2/5}, 1\}$

⇓

$O(n^{4/3+\varepsilon} + n^{2/5+\varepsilon} \kappa^{2/5+\varepsilon} \xi^{1+\varepsilon})$, for any $\varepsilon > 0$.

The algorithm is subquadratic for $\xi \ll n^{4/5}$.

Extensions

Combinatorial part – unchanged.

Modify the implementation.

Axis-parallel ellipses in R^2.

Subquadratic, for $\xi \ll n^{8/11}$

Simplices in R^3.

Subcubic, for $\xi \ll n$

Concluding remarks

Do we have to redraw R at each iteration of the DC algorithm?

Determining the insertion order is the bottleneck of the algorithm.

Thank You!

Perform the process for j = 1,2,4, …, 2^h, …

- Construct the union of the first j triangles: $O(j^2)$.
- Count the number of intersections between U and the remaining triangles in $O(n^{2/3+\varepsilon} j^{4/3+\varepsilon})$ time, *for any* $\varepsilon > 0$.
- If this number $< \kappa/t$
  $j \leftarrow 2j$, goto 1

- Construct these intersections explicitly, and trim the edges of the remaining triangles to their portions outside U: $O(n^{2/3+\varepsilon} q^{4/3+\varepsilon} + \kappa/t)$.

- Run a sweep-line procedure on the edges of U and the trimmed portions.
  If the number of intersections $> \kappa/t$,
  $j \leftarrow 2j$, goto 1;
  else
  the union is reported : $O((\kappa/t + \xi^2)\log n)$