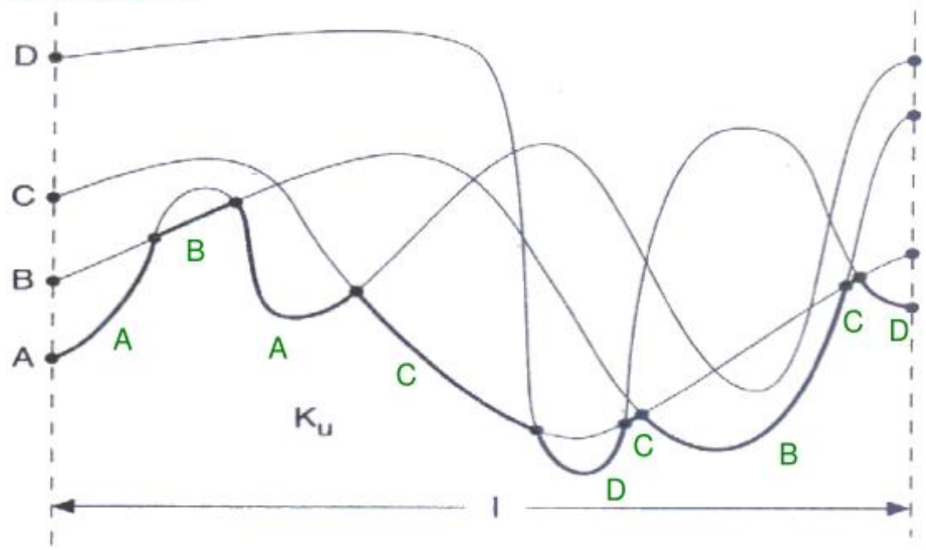
Computational Geometry



Lower Envelopes

Michael T. Goodrich

with some slides by Thomas Ottmann

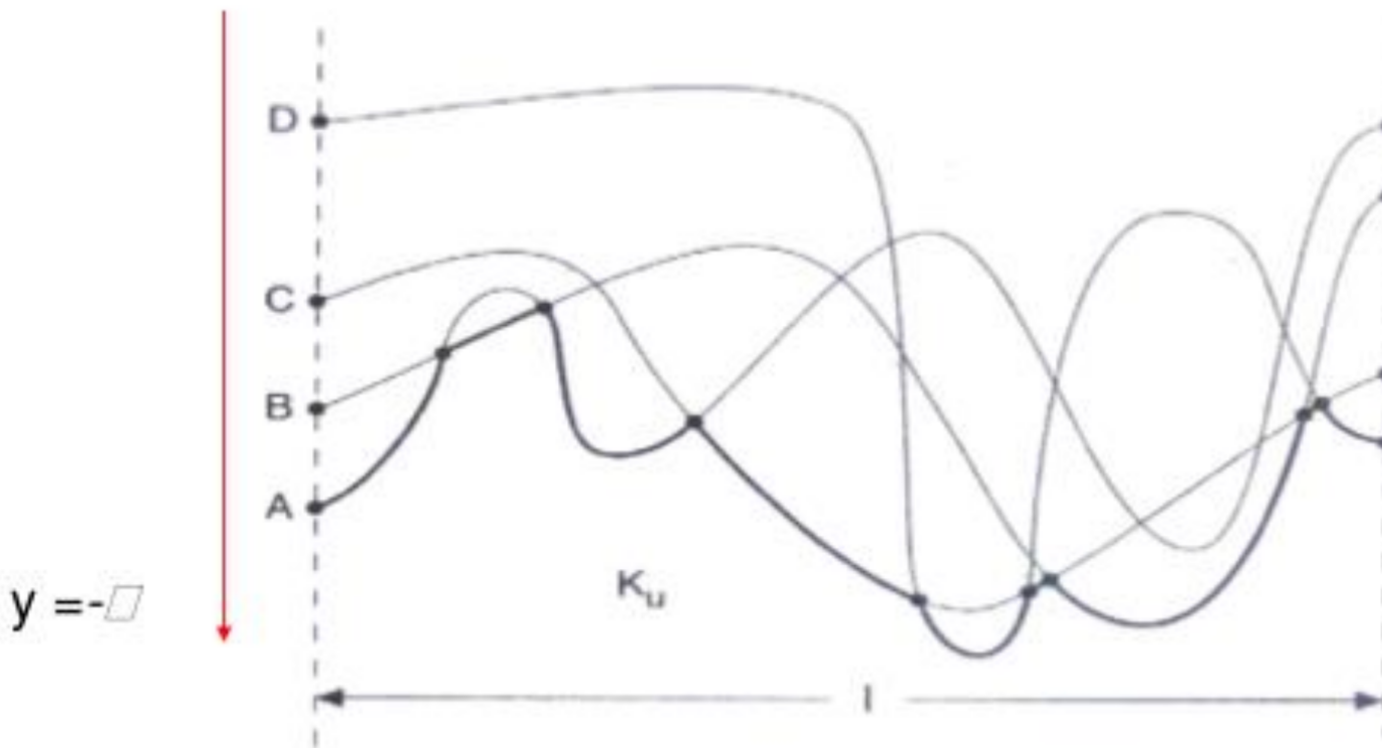
Definition

Definition of the Lower Envelope (untere Kontur) of a set of functions:

Given n real-valued functions, all defined on a common interval I, then the minimum is :

$$f(x) = min_{1 \leq i \leq n} f_i (x)$$

The graph of $f(x)$ is called the lower envelope of the f_i's.



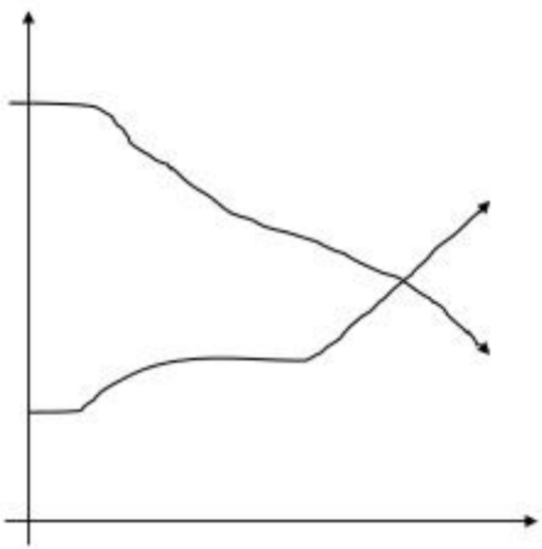
x-Monotone Curves

Definition :
A curve c is x-monotone if any vertical line either does not intersect c, or it intersects c at a single point.

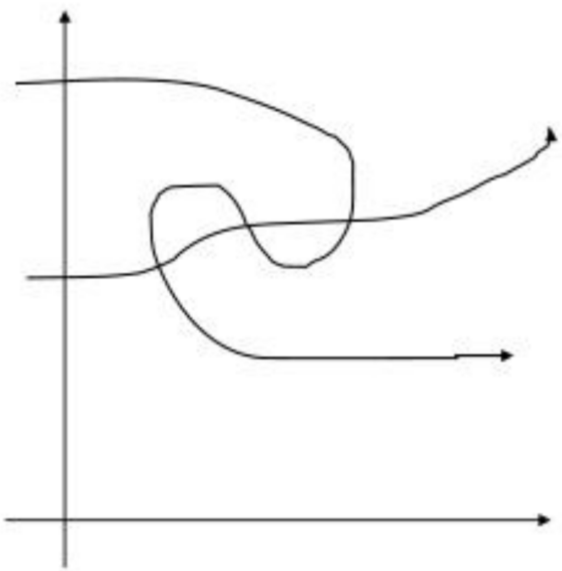
Assumptions

- All functions are x-monotone.
- Function evaluation and determination of intersection points take time *O(1)*.
- The space complexity of the description of a function f_i is also constant.

x-monotone:

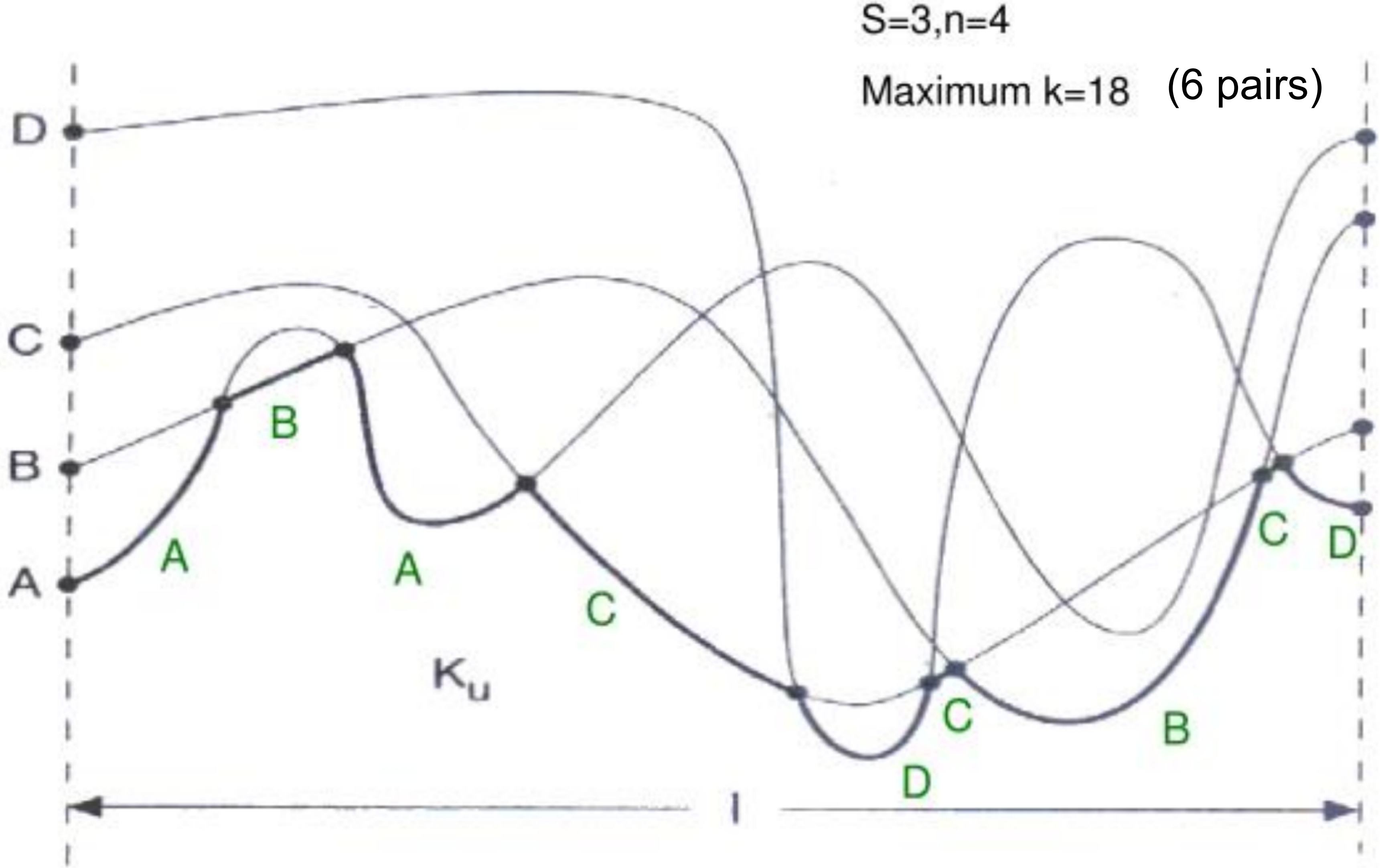


Not *x-monotone:*



Representing a lower envelope

s = # times any pair of curves intersect

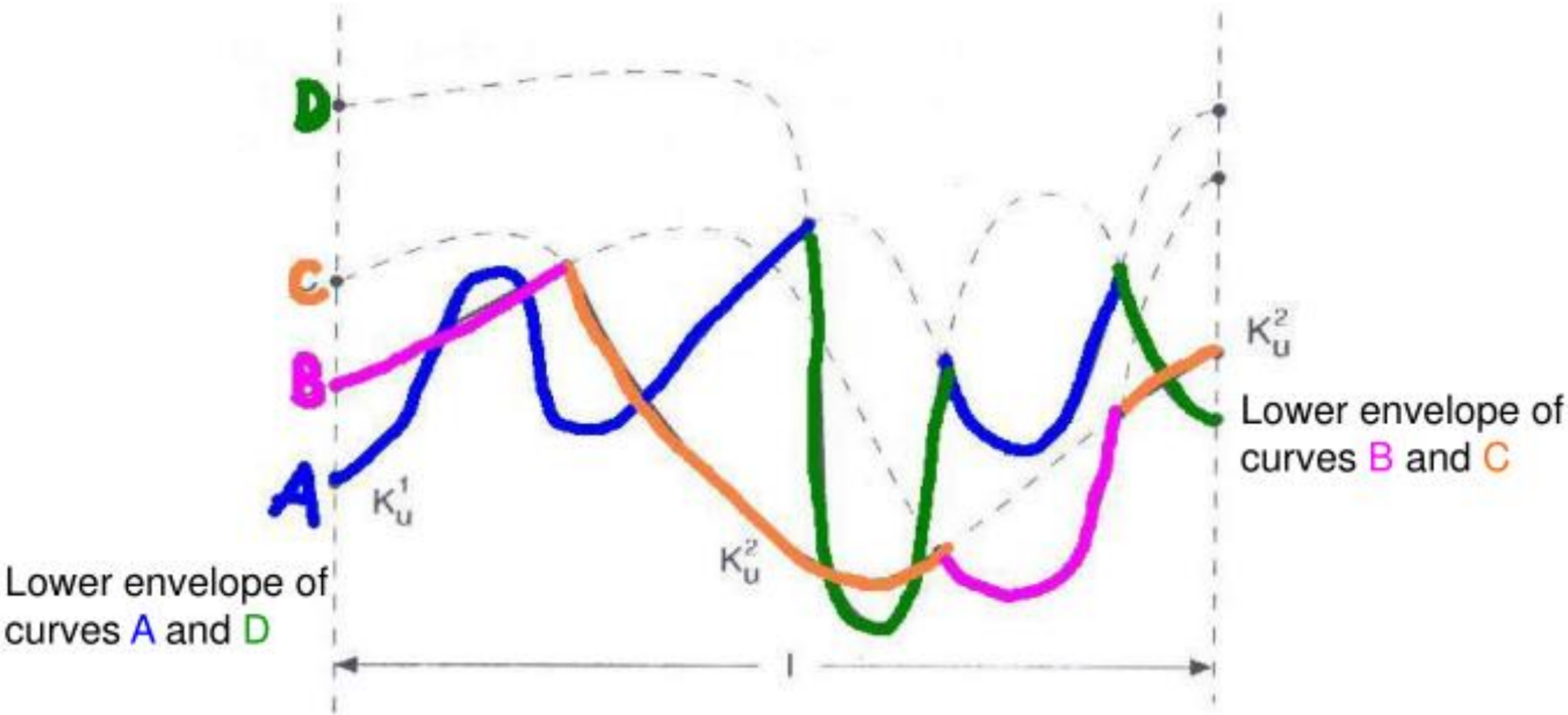


Divide-and-Conquer + Plane-sweep

1. Divide: the set S of n functions into two disjoint sets S_1 and S_2 of size n/2.
2. Conquer: Compute the lower envelopes L_1 and L_2 for the two sets S_1 and S_2 of smaller size.
3. Merge: Use a sweep-line algorithm for merging the lower envelopes L_1 and L_2 of S_1 and S_2 into the lower envelope L of the set S.

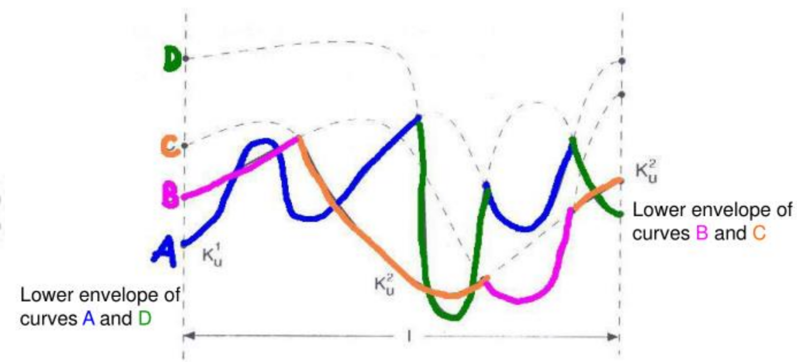
The Merge Step (plane-sweep)

The lower envelopes of curves A,D and B,C



The Merge Step (plane-sweep)

The lower envelopes of curves A,D and B,C



Sweep over L_1 and L_2 from left to right:

Event points: All vertices of L_1 and L_2,
all intersection points of L_1 and L_2

At each instance of time, the event queue contains only 3 points:

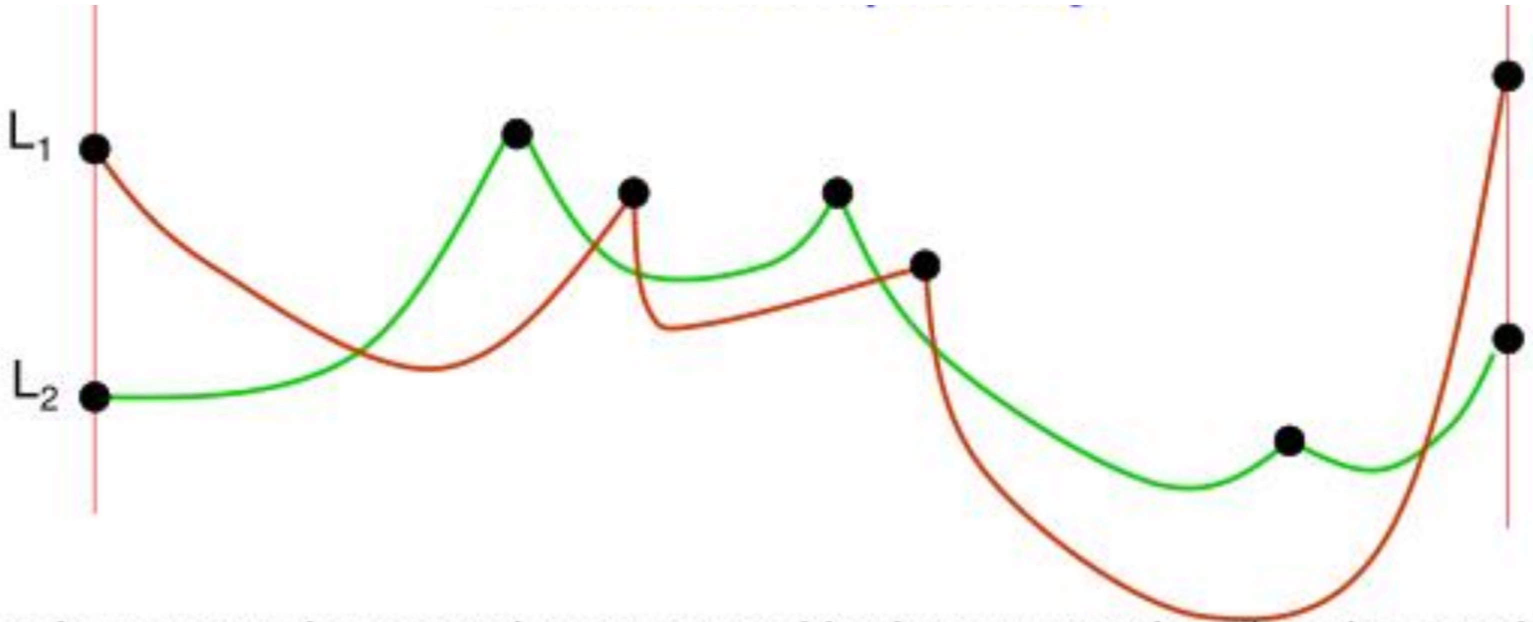
1 (the next) right endpoint of a segment of L_1

1 (the next) right endpoint of a segment of L_2

The next intersection point of L_1 and L_2, if it exists.

Sweep status structure: Contains two segments in y-order

Time Complexity



The lower envelope can be computed in time proportional to the number of events (halting points of the sweep line).

At each event point, a constant amount of work is sufficient to update the SSS and to output the result.

Total runtime of the merge step: O(#events).

How large is this number?

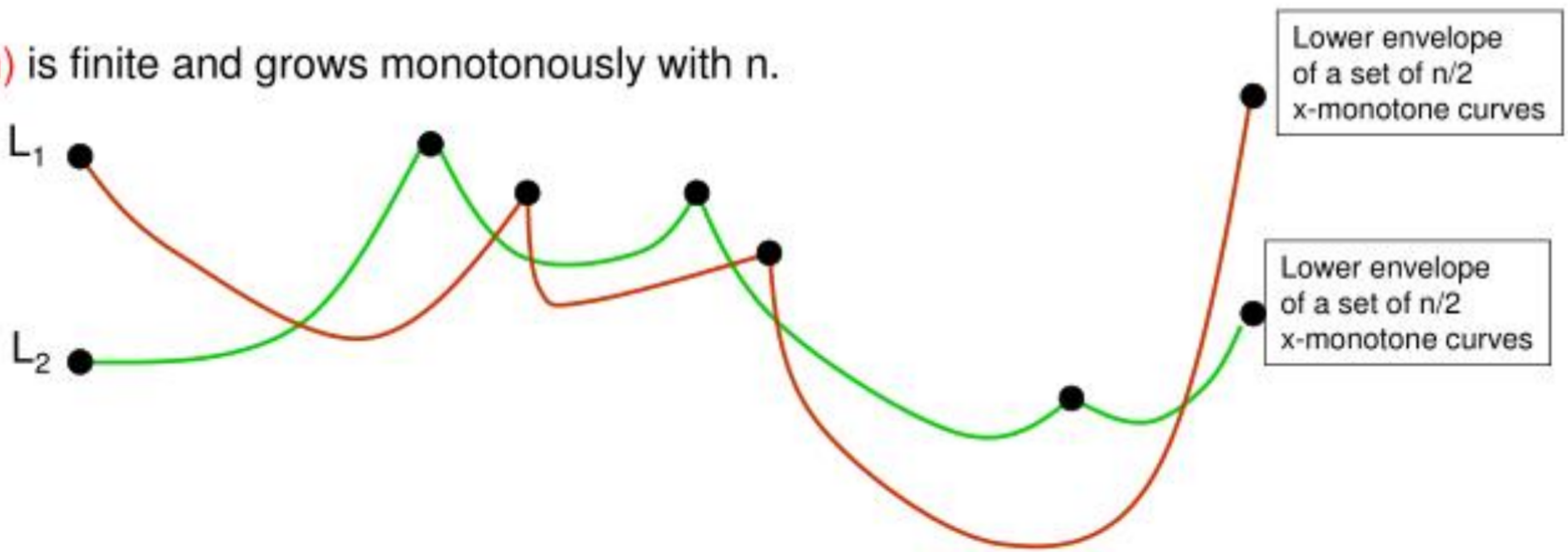
A Complexity Measure

Define $\lambda_s(n)$:

the maximum number of segments of the lower envelope of an arrangement of

- n different x-monotone curves over a common interval
- such that every two curves have at most s intersection points

$\lambda_s(n)$ is finite and grows monotonously with n.



$2\lambda_s(n/2) \leq 2\ \lambda_s(n)$

Analyzing Divide-and-Conquer

If n =1, do nothing, otherwise:

1. Divide: the set S of n functions into two disjoint sets S_1 and S_2 of size n/2.
2. Conquer: Compute the lower envelopes L_1 and L_2 for the two sets S_1 and S_2 of smaller size.
3. Merge: Use a sweep-line algorithm for merging the lower envelopes L_1 and L_2 of S_1 and S_2 into the lower envelope L of the set S.

Time complexity T(n) of the D&C/Sweep algorithm for a set of n x-monotone curves, s.t. each pair of curves intersects in at most s points:

$T(1) = C$
$T(n) \leq 2\,T(n/2) + C\,\lambda_s(n)$

Time Complexity

Using the Lemma : For all s, $n \geq 1$, $2\lambda_s(n) \leq \lambda_s(2n)$,

and the recurrence relation $T(1) = C$, $T(n) \leq 2\,T(n/2) + C\,\lambda_s(n)$ yields:

Theorem: To calculate the lower envelope of n different x-monotone curves on the same interval, with the property that any two curves intersect in at most s points can be computed in time $O(\lambda_s(n) \log n)$.

So we just need a good upper bound for $\lambda_s(n)$…

Davenport-Schinzel Sequences (DSS)

Consider words (strings) over an alphabet $\{A, B, C, \ldots\}$ of n letters.

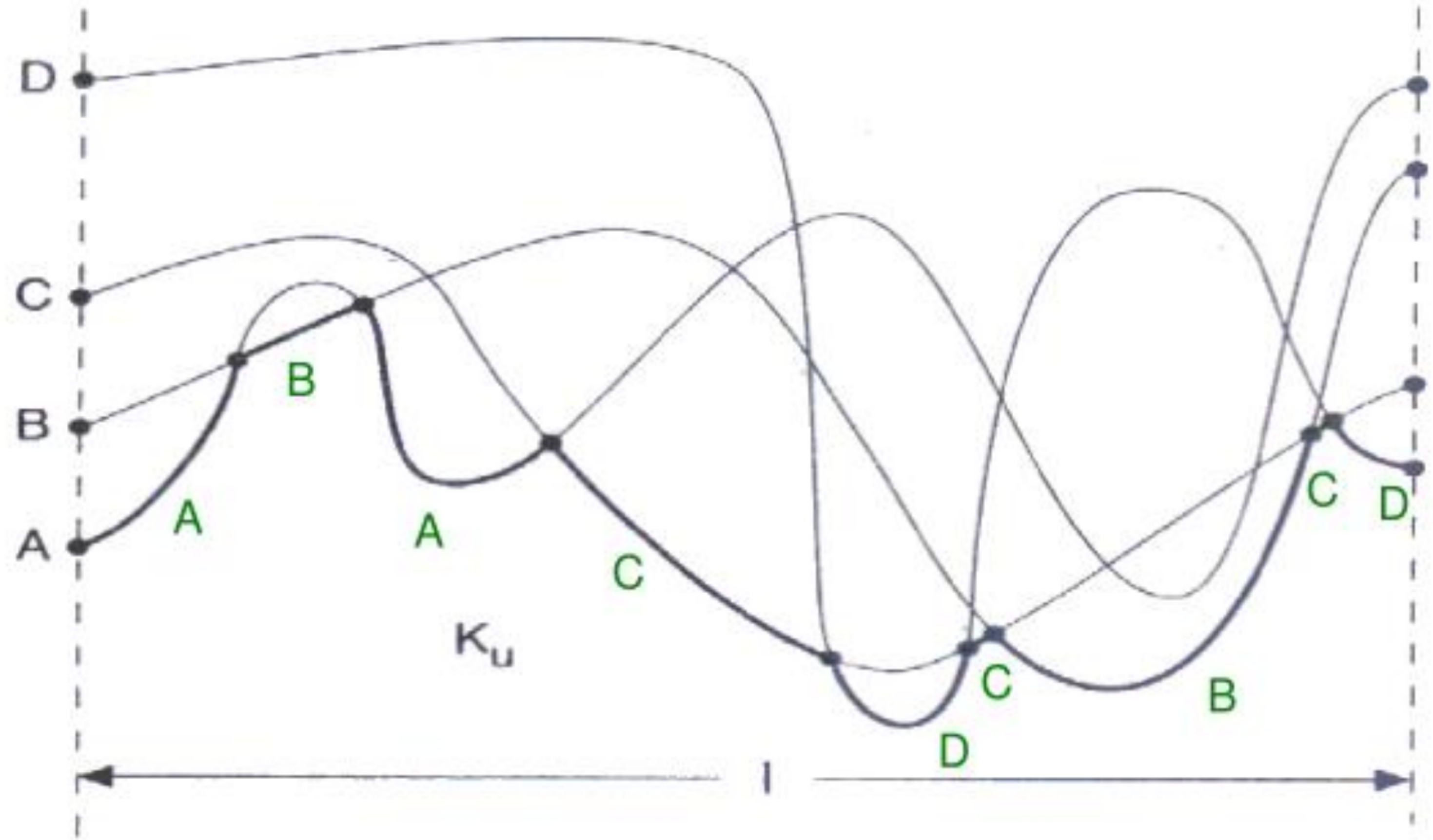
A DSS of order s is a word such that

- no letter occurs more than once on any two consecutive positions
- the order in which any two letters occur in the word changes at most s times.

Examples: ABBA is no DSS, ABDCAEBAC is DSS of order 4,
(it contains A..B..A..B..A)

Theorem:
The maximal length of a DSS of order s over an alphabet of n letters is $\lambda_s(n)$.

Relationship to Lower Envelopes



Lower envelope contains the segments ABACDCBCD in this order.

Because s = 3, no pair of letters alternates more than 3 times.
Thus, this is a DSS of order 3.

“Almost” Linear Size

Properties of $\lambda_s(n)$

1. $\lambda_1(n) = n$
2. $\lambda_2(n) = 2n - 1$
3. $\lambda_s(n) \leq s\,(n-1)\,n/2 + 1$
4. $\lambda_s(n) \in O(n \log^* n)$, where $\log^* n$ is the smallest integer m, s.t. the m-th iteration of the logarithm of n

$$\log_2(\log_2(\ldots(\log_2(n))\ldots))$$

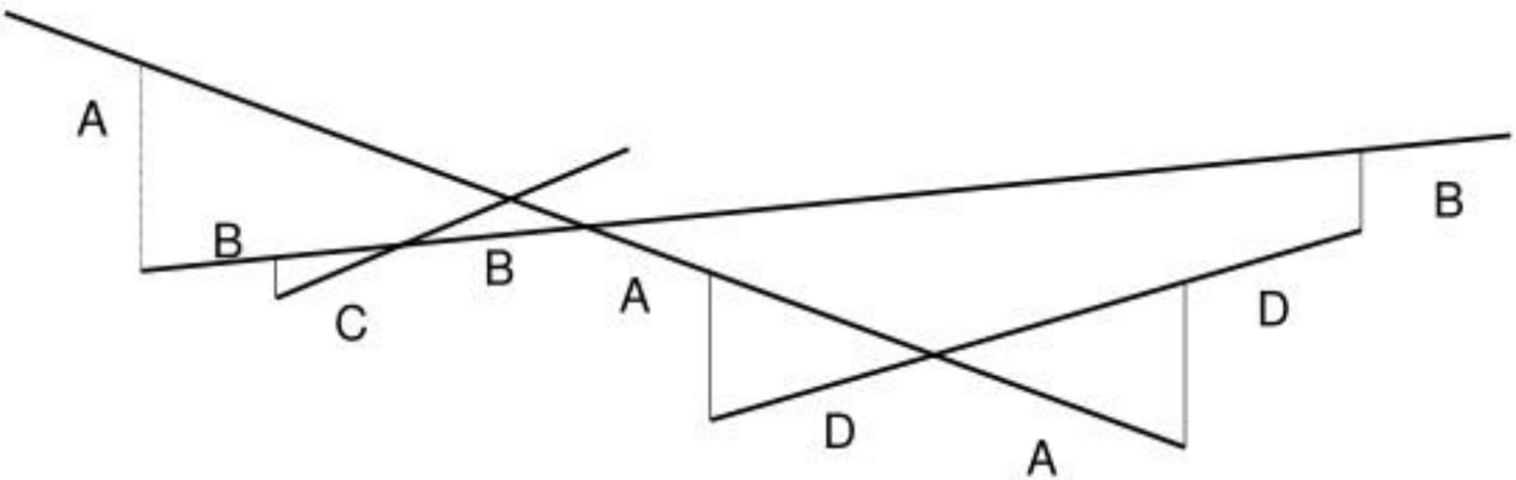
yields a value ≤ 1:

Note: For realistic values of n, the value $\log^* n$ can be considered as constant!

Example: For all $n \leq 10^{20000}$, $\log^* n \leq 5$

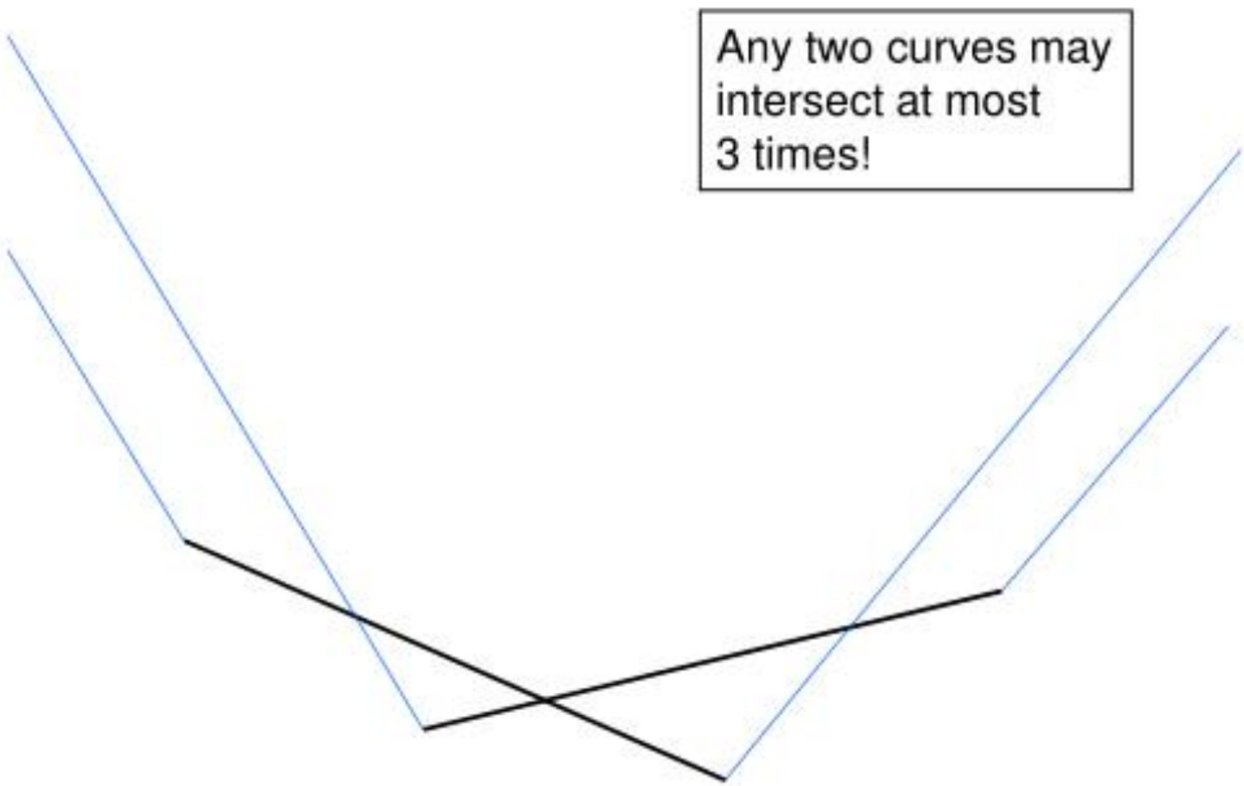
Line Segments

Lower envelope of an arrangement of line segments in general position



Theorem: The lower envelope of n linesegments in general position has $O(\lambda_3(n))$many segments. It can be computed in time $O(\lambda_3(n) \log n)$.

Line Segments



The lower envelope of line segments has size O(n log* n),
so it is almost linear, but not quite linear.