Finding All Maximal Subsequences with Hereditary Properties

Drago Bokal, Sergio Cabello, and **David Eppstein**

31st International Symposium on Computational Geometry
Eindhoven, Netherlands, June 2015

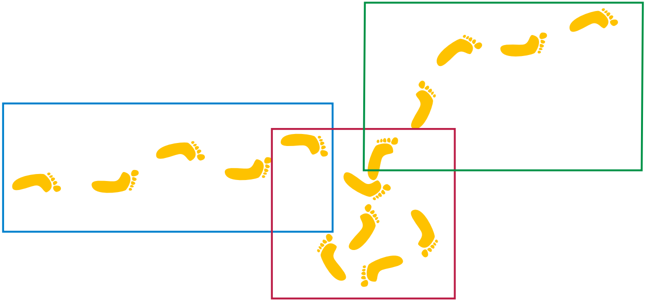
Trajectory analysis

CC-BY-SA image "my way"
by Michal Osmenda
from Wikimedia commons

Data: sequence of points from one or more trajectories, in two or three dimensions, possibly also with timestamps

Key problems include disambiguation of overlapping trajectories; clustering and finding representative paths for clusters; decomposition into pathlets; prediction and intentionality analysis

Windowed queries into trajectories



Goal: Build a data structure that can quickly answer qualitative queries about contiguous subsequences of a trajectory

Could be used for exploratory data analysis, or as a subroutine (e.g. to decompose paths into subsequences with uniform motion)

Previous work on windowed queries

Bannister, Eppstein, DuBois & Smith, SODA'13:

Data = two-party communication events

Query = graph-theoretic properties of the events within a time window

Bannister, Devanny, Goodrich & Simons, CCCG'14:

Data = timestamped points (same as here)

Query = extreme points of convex hull, approximate nearest neighbors, etc.

We handle simpler queries (only Boolean answers) more quickly

Our focus is less on query time and more on preprocessing

Our queries

- Does the subtrajectory have unit diameter? (Is the subject not moving much?)
- Does the convex hull of the subtrajectory have unit area? (Is the subject moving along an unobstructed path?)
- Is there a direction for which the subtrajectory is monotone? (Is subject moving in one direction but avoiding obstacles?)

The (trivial) data structure

Query: does subsequence (i, j) have a (Boolean) property $\mathcal{P}$?

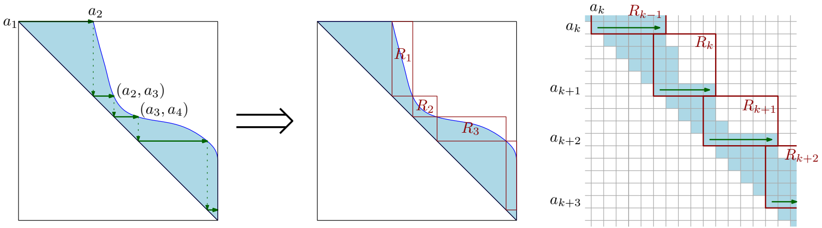
We consider only *hereditary* properties:
if a subsequence has $\mathcal{P}$, so do all its sub-subsequences

Store, for each i, the *horizon* $j^*(i)$ s.t. (i, j^*) is maximal with $\mathcal{P}$.

To handle query (i, j), compare j with $j^*(i)$

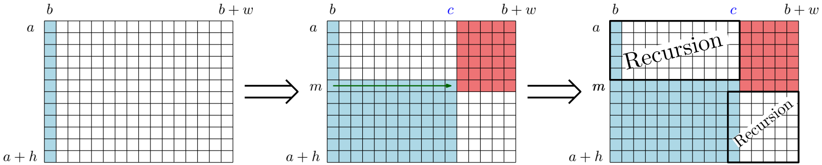
But how do we find all of the horizons, efficiently?

Key ideas (1)



- Greedily partition grid of potential queries (i, j) into *frontier rectangles* in which top right and bottom left corners are maximal yes-instances in their rows
- Partition is based on solving a collection of single-horizon search problems whose sizes sum to $O(n)$
- Use sequential or binary search for each single-horizon search

Key ideas (2)



Recursive divide and conquer into frontier subrectangles

Split point = single horizon in median row

Complication: subproblem size ≠ rectangle size

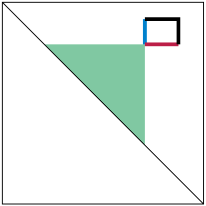
So subproblems do not shrink quickly enough for divide and conquer to be efficient

Key ideas (3)

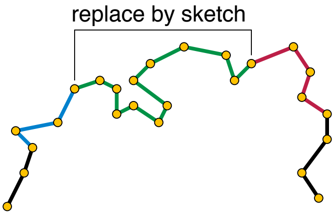
The subtrajectories for a rectangular subproblem have three parts:

- Prefix of variable length given by row number in the rectangle
- Middle part of fixed length
- Suffix of variable length given by column number

Replacing the middle part by a small *sketch* allows the subproblem sizes to shrink more quickly in the divide and conquer



matrix of queries



trajectory

Example sketch for testing monotonicity: the range of angles for which the subtrajectory is monotonic

Results

We can find all j-maximal subsequences of the trajectory that have property $\mathcal{P}$ ...

- ... in time $O(n)$, when $\mathcal{P}$ is monotonicity
- ... in time $O(n \log n \log\log n)$, when $\mathcal{P}$ is unit area
- ... in time $O(n \log^2 n)$, when $\mathcal{P}$ is unit diameter

Open: What other similar problems on trajectories fit into this framework? What about non-Boolean properties?