MADALGO – CENTER FOR MASSIVE DATA ALGORITHMICS


Morten Revsbæk
Aarhus University

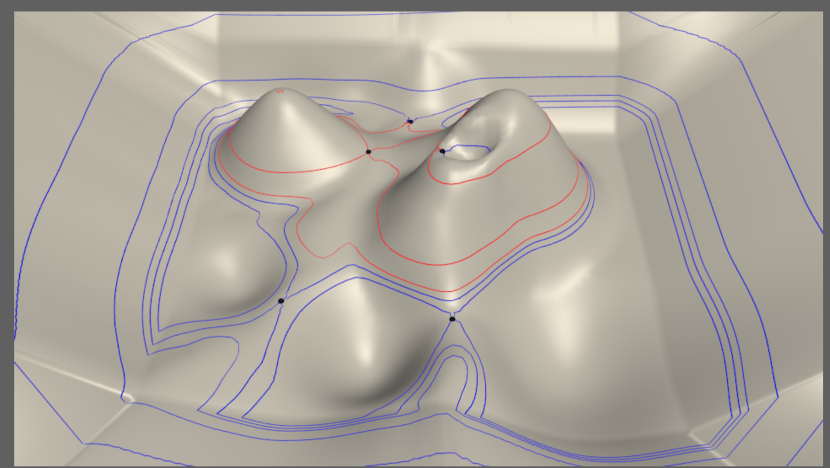

Jungwoo Yang
Aarhus University

Danmarks Grundforskningsfond
Danish National Research Foundation

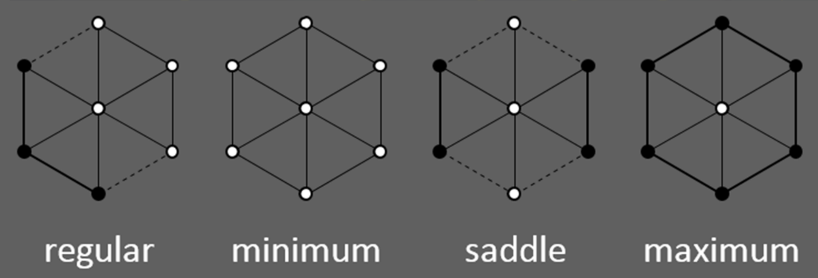
Maintaining Contour Trees of Dynamic Terrains

Introduction

Terrain

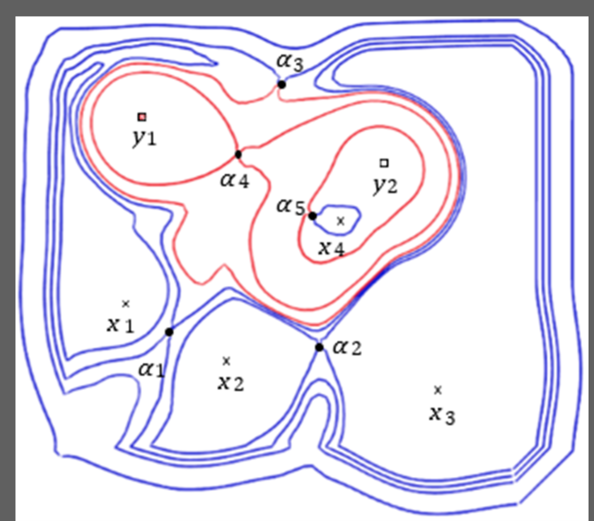


- Terrain is often represented as a planar triangulation (TIN) ***M*** with a continuous height function $h : \mathbf{R}^2 \to \mathbf{R}$
- Vertex Types
  - Determined by the number of down (or up) components in neighborhood (see fig.).
  - A non-regular vertex is called a ***critical vertex***
- Assume each vertex has unique height and there are only simple saddle (# comp. = 2)*

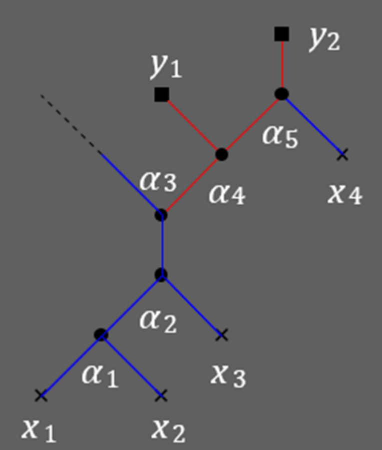


Filled (hollow) vertices are lower (higher) neighbors

Level set and Contour tree



- ***l-level*** set of a terrain is a subset $\boldsymbol{M_l}$ of ***M*** such that $h(v) = l$ for all v in $\boldsymbol{M_l}$.
- A ***contour C*** is a connected component in $\boldsymbol{M_l}$
- ***C*** is ***red/blue*** if inside is *higher/lower*.
- ***Contour tree*** is a topological abstraction of the terrain. It captures the topological changes of contours in the terrain.
  - The topological changes occurs only at critical vertices.
  - Minimum – Create, Maximum – Destroy.
  - Saddle – Merge or Split.
    - Call it ***merge saddle*** or ***split saddle***
  - Can be computed in $O(n \log n)$ time. [1]



Problem

Maintaining the contour tree of a terrain under the following operation:
ChangeHeight*(*v*, *r*)** : Change the height of a vertex *v* in ***M to *r*.

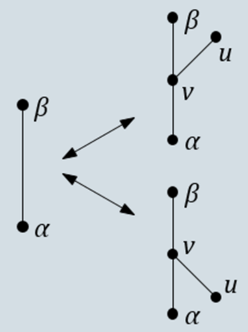
References

[1] H. Carr, J. Snoeyink, and U. Axen. *Computing contour trees in all dimensions*. Computational Geometry, 2003.
[2] D. D. Sleator, and R. E. Tarjan. *A Data Structure for Dynamic Trees*. STOC, 1981.
[3] P. K. Agarwal, L. Arge, and K. Yi. *I/O-efficient batched union-find and its applications to terrain analysis*. SoCG, 2006.

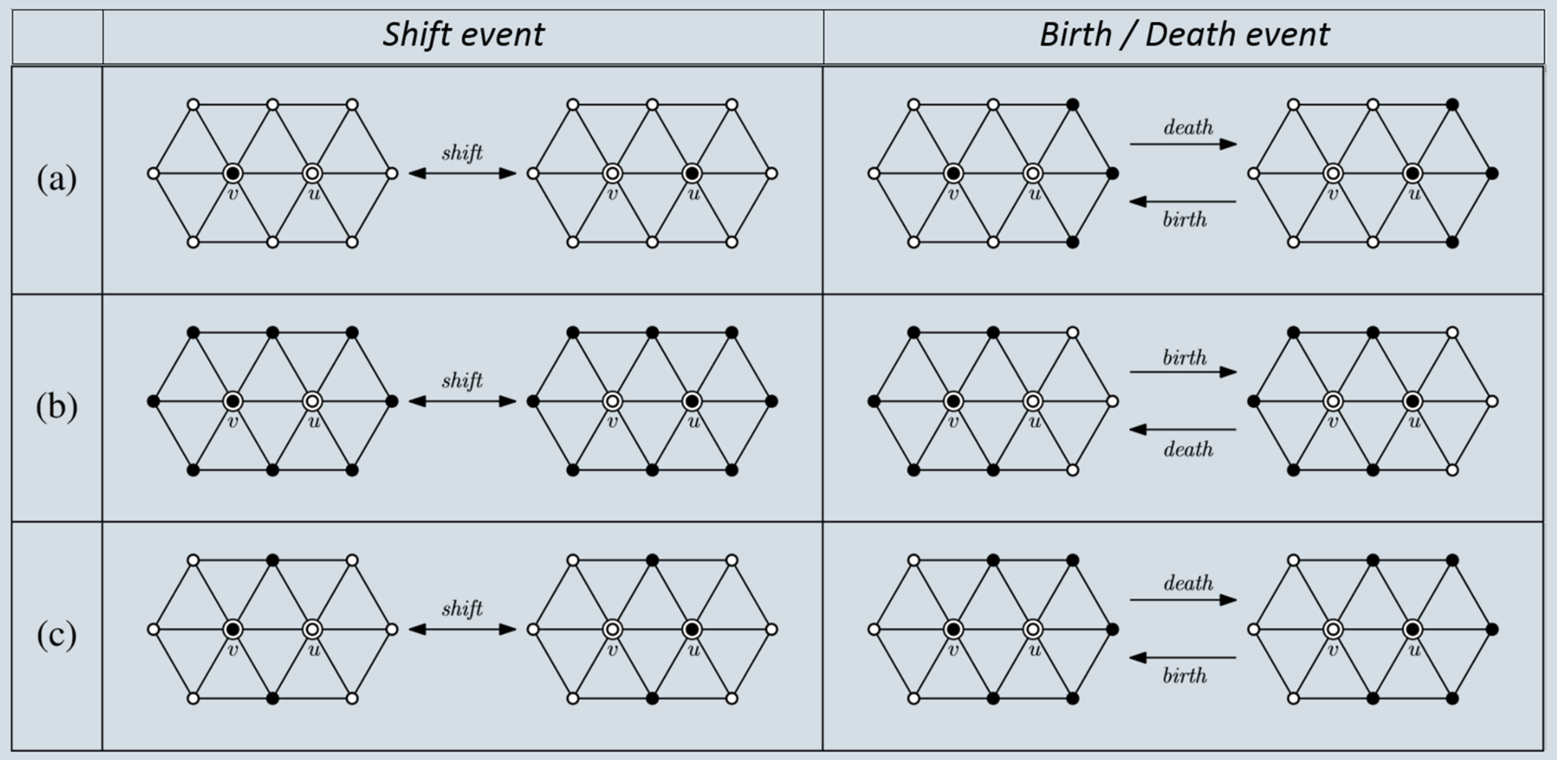
* Non-simple saddle can be split into simple saddles, e.g. [2].

Events

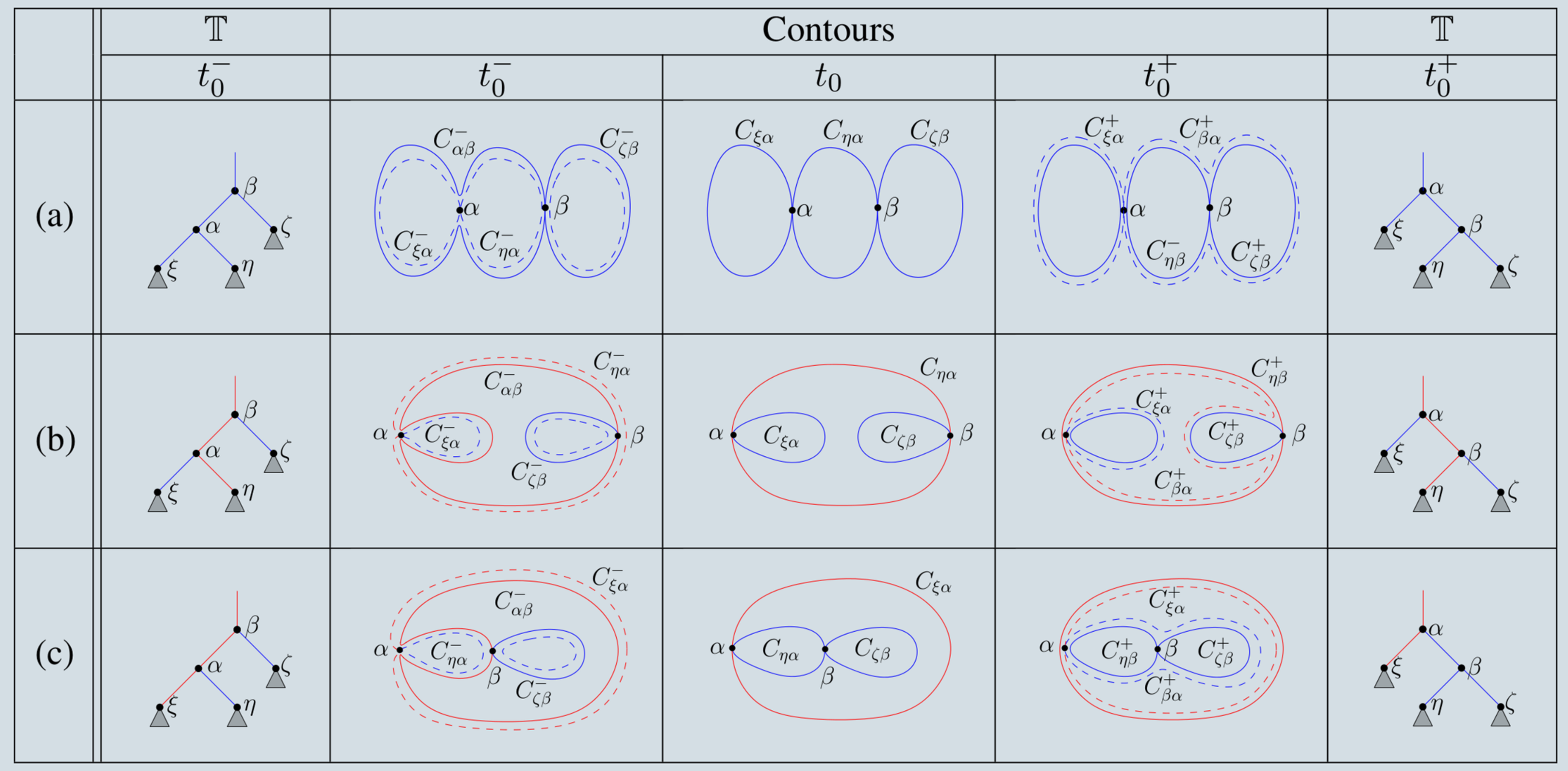
ChangeHeight*(*v*,*r*)** operation is processed as a continuous deformation for the terrain over time. During this continuous deformation the combinatorial structure of the contour tree ***T changes only at discrete time instance, called ***events***. More precisely, a event happens when $h(v) = h(u)$ for a vertex *u* in ***M***. We characterize the possible events.



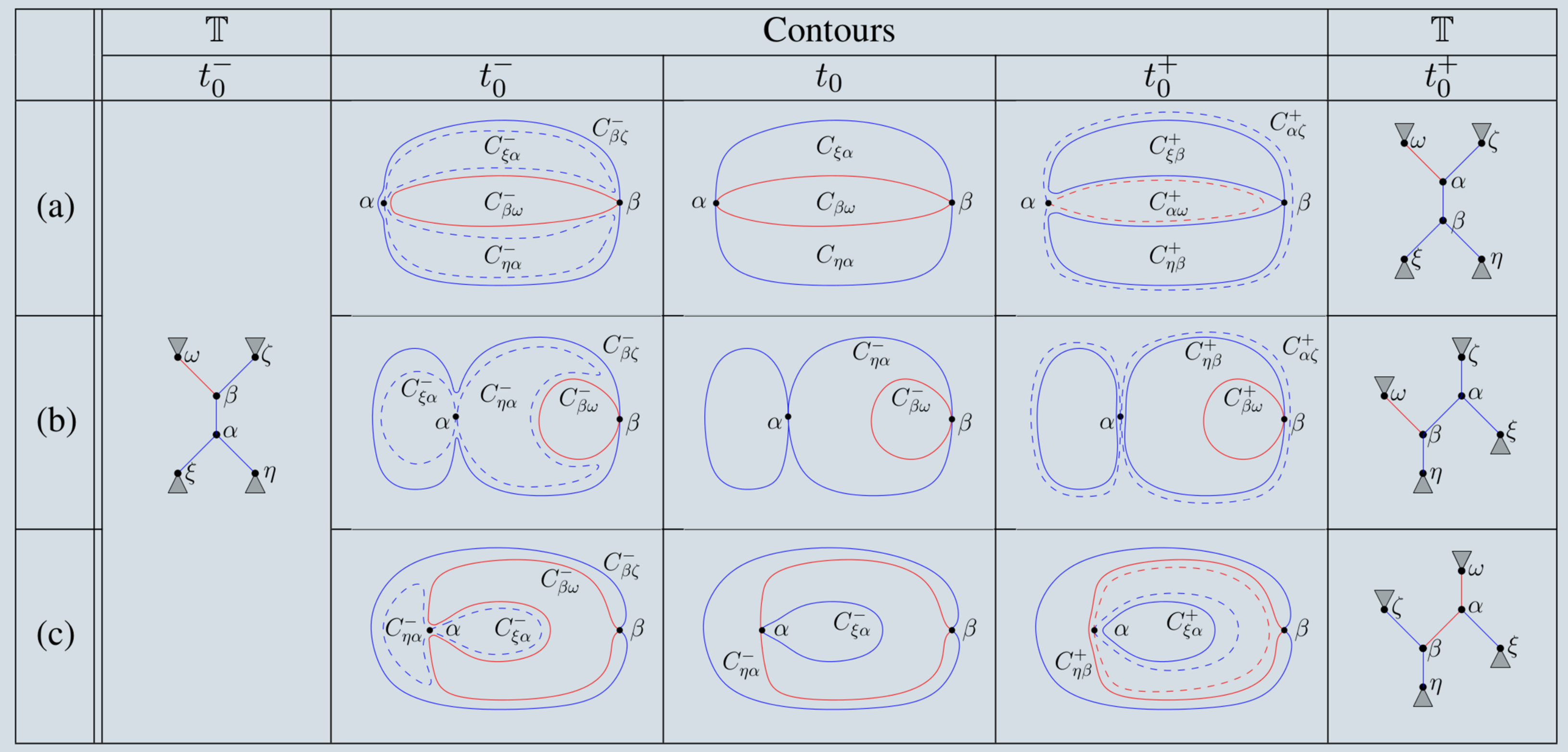
- ***Local Event*** – When *v* and *u* are adjacent in *M*.
  - ***Shift event*** – Change label of node.
  - ***Birth/Death event*** – Create/destroy a pair of nodes.



- ***Interchange Event*** – When *u* and *v* are saddle vertices and they are lying on the same contour. Let *α* (*β*) be the node in ***T*** corresponding to *v* (*u* resp.), and let t_0 the moment the event occurs. Let t_0^- (t_0^+) also be the moment before (after resp.) the event. (Think it as raising *α* up)
  - When the two saddles are both merge saddles. (For split, upside down)



  - When one saddle is merge saddle and the other is split saddle. (mixed)



Data Structure

For the dynamic contour tree, we maintain two additional data structures called ***ascent*** and ***descent trees***. Ascent trees are defined by the vertices of ***M*** and a set of oriented edges such that each vertex has an edge pointing to a lower neighbor. Descent trees are defined in the same way but with edges pointing to a higher neighbor. For a vertex *v* in ***M***, let min_v (max_v) be the minimum (maximum) on the root of the descent tree (ascent tree resp.) containing *v*.

Handling Local Event

- Update ascent tree and descent tree if any edges in trees are flipped.
- Event type can be determined by scanning the neighbor vertices and comparing their height.
- *Shift event*: Simply update the label corresponding node in ***T***.
  - (*α*,*β*) is always on the path in ***T*** between min_v and max_v.
- *Birth event*: Find the edge (*α*,*β*) in ***T*** that related to the contour containing *v*, and create new nodes for *v* and *u*.
- *Death event*: Remove the node *v* and *u* and connect *α* and *β* in ***T***.

Time complexity

All operations for a event can be implemented in $O(d + \log n)$ time by Link-Cut tree [3], where *d* is the maximum degree of *v* and *u*.

Handling Interchange Event

- Non-mixed event: Determine which one of the children of *α* contains min_u (or $amin_u$**). Then, the child switches its parent to *β* and *β* becomes a child of *α*.
- Mixed event:
  - To determine whether it is case (a), we check if the lowest common ancestor of min_u and $amin_u$ is *α*. If it is the case, the sign change event occurs so just the labels get swapped.
  - For case (b) (c), we find *η* by finding the child of *α* contains min_u and change the parent of *η* to *β*. Similarly we find the parent of *β* on the way to max_v and make the node *v*'s parent. Finally, *α* becomes a parent of *β*.

** A saddle has two lower components in its neighborhood. $amin_u$ is the minimum on the root of the descent tree containing a lower neighbor of *u* not equal to min_u.