Modeling & Analyzing Massive Terrain Data Sets (STREAM Project)

Pankaj K. Agarwal

Workshop on Algorithms for Modern Massive Data Sets
Diverse elevation point data: density, distribution, accuracy

**Photogrammetry** 0.76m v. accuracy (5ft contours)

**Lidar** 0.15m v. accuracy; altitude 700m and 2300m

**RTK-GPS** 0.10m v. accuracy
Constructing Digital Elevation Models

Grid DEM: Elevation stored at uniform grid points

TIN: Triangulation; elevation stored at vertices

Contours Maps: Iso-contour lines at regular intervals
Natural feature extraction

Maps of topo parameters: computed simultaneously with interpolation using partial derivatives of the RST function, terrain features: combined thresholds of topo parameters

Extracted foredunes 1999-2004 using profile curvature and elevation threshold
how to automatically track features that change some of their properties over time?
Terrain Analysis

- Flow Analysis
- Watershed hierarchy
- Visibility
- Navigation
Challenge: Massive Data Sets

• **LIDAR**
  – NC Coastline: 200 million points – over 7 GB
  – Neuse River basin (NC): 500 million points – over 17 GB
  – Appalachian Range: 50GB-5TB

• **Output is also large**
  – 10ft grid: 10GB
  – 5ft grid: 40GB
Approximation Algorithms

• *Exact computation expensive*
  
  Many practical problems are intractable
  
  Multiple & often conflicting optimization criteria
  
  Suffices to find a near-optimal solution

• *Tunable Approximation algorithms*
  
  – Tradeoff between accuracy and efficiency
  
  – User specifies tolerance
I/O-Bottleneck

• Data resides in secondary memory
• Disk access is $10^6$ times slower than main memory access
  – Maximize useful data transferred with each access
  – Amortize large access time by transferring large contiguous blocks of data
• I/O – not CPU – is often the bottleneck for processing massive data
I/O-efficient Algorithms [AV88]

- Traditional algorithms optimize CPU computation
  - Not sensitive to penalty of disk access
- I/O model
  - Memory is finite
  - Data is transferred in blocks (B: block size)
  - Complexity: #disk blocks transferred (#I/Os)

\[
\text{Scan}(N) = O(N/B) \\
\text{Sort}(N) = O\left(\frac{N}{B} \log \frac{N}{M/B}\right)
\]

\( B \sim 2K-8K \)

External Memory Algorithms [Vitter]
The TerraStream Modules

(Classification) (DEM Construction) Conditioning Flow Routing

Flow Accumulation Watershed Hierarchies Quality Metrics Contour Lines
TIN DEM

Given a plane triangulation $\Sigma$ with a height $f(v)$ for each vertex $v$, one can linearly interpolate $f$ in the interior of every face of $\Sigma$. 
Level Sets, Contours

The level-set at height $l$ is $f^{-1}(l)$. Each connected component of a level set is called a contour.
Given levels $L = \{l_1, \ldots, l_k\}$, the contour map is $f^{-1}(L)$. 
Contour Maps

Given a set of levels \( L = \{l_1, \ldots, l_k\} \), compute the contour map \( f^{-1}(L) \) such that each contour is reported separately and in sorted (circular) order.
Contour Maps

- Usage of contour lines (also called iso-contours, isogons, etc) goes back to at least 17\textsuperscript{th} century
Computing Contour Map: Internal Memory Algorithm

Find a seed point on each contour and traverse the triangulation to trace each contour.

Use a simple data structure (e.g., contour trees) to compute seed points.

Query time: \(O(\log N + T)\) \(T:\) #contour edges

Contour map: \(O(N\log N + T)\) \(T:\) #contour map edges

For massive terrains

I/O efficiency is bad: \(O(N+T)\) instead of \(O((N+T)/B)\)
Storing a TIN on a disk so that it can be traversed with as few I/Os possible

Is there a good ordering of the triangles?
Our results

**Ordering Theorem:** A total ordering, called **C-ordering**, of triangles can be computed in $O(\text{Sort}(N))$ I/Os s.t. the subsequence of triangles intersecting a contour appears along the contour and contours in a level set are broken in nested order.

$a_1a_2 - -b_1 - - - c_1c_2c_3 - - - - b_2b_3 - d_1d_2 - -b_4 - -a_3a_4a_5$

Individual contours can be retrieved in $O(T/B)$ I/Os from this ordering

- Computing contour maps: $O(\text{Sort}(N)+T/B)$ I/Os
- Answering contour queries
  - Preprocessing Time: $O(\text{Sort}(N))$ I/Os
  - Space: $O(N/B)$ disk blocks
  - Query: $O(\log_B N+T/B)$
Height Graph

\[ H = (V(\Sigma), \{ u \rightarrow v : uv \in E(\Sigma), f(u) < f(v) \}) \]

\( H^* \): Dual graph of \( H \)

\[ t \prec t' \iff t^* \rightarrow t'^* \text{ in } H^*. \]
Critical Points

maximum

minimum

saddle

regular
Simple Terrains

A simple terrain is one with no saddles; thus 1 max and 1 min (boundary). Take a min (bd) to max path $P$ and delete its dual from $H^*$. 

Once the dual graph becomes acyclic, the induced relation \( \prec \) becomes a partial order which can then be topologically sorted into a total order.
Positive & Negative Saddles

A saddle is negative if it joins two disjoint connected components of its sublevel-set and positive otherwise.

If we replace $f$ with $-f$, the two types switch roles.
**Cut Trees**

**Positive Cut-Tree:** follow an ascending path in every connected component of the upper link of every *positive* saddle, joining paths when they collide.

**Lemma.** The result is a tree (not just forest) that reaches every maximum.
**Lemma.** Doing this removes all positive saddles and maxima and adds a new maximum.
Surgery & Contours

The simplified terrain $T'$ has all the triangles of the original terrain $T$ plus a number of auxiliary triangles.

All contours of a level set of $T$ are combined in a single contour in $T'$.
Nesting of Contours

**Theorem.** In any contour of $T'$, contours of $T$ are broken (by segments from auxiliary triangles) in a nested (parenthesized) manner.

\[
\begin{align*}
&\{ a_1a_2\} - \{ b_1 - - \} \{ c_1c_2c_3 \} - - - b_2b_3 \{ d_1d_2 \} - b_4 \} - a_3a_4a_5
\end{align*}
\]

This results a simple stack algorithm that separates tracks belonging to individual contours in $O(T/B)$ I/Os.
Ongoing Work

- Compute C-ordering for general manifolds
- Maintain C-ordering for hierarchical representation of terrains
- Denoising contours
Collaborators

Lars Arge
Helena Mitasova
Andy Danner
Thomas Molhave

- Sponsored by
  - Army Research Office
    W911NF-04-1-0278

Bardia Sadri
Ke Yi
I/O-Efficient Algorithms

Scan the triangles (in the order laid out on the disk) and generate all segments. Then sort the output.

I/O Complexity:

\[ O(\text{Sort}(N) + \text{Sort}(T)). \]

Answering a contour query:

Preprocessing: \( O(N \log_B N) \), Space: \( O(N/B) \) blocks

Query: \( O(\log_B N + 1/B) \)
Front-ends

ArcGIS Extension
GRASS Extension
Command Line Tools

Front-End GUI
MapInfo