ParaStream: A parallel streaming Delaunay triangulation algorithm for LiDAR points on multicore architectures

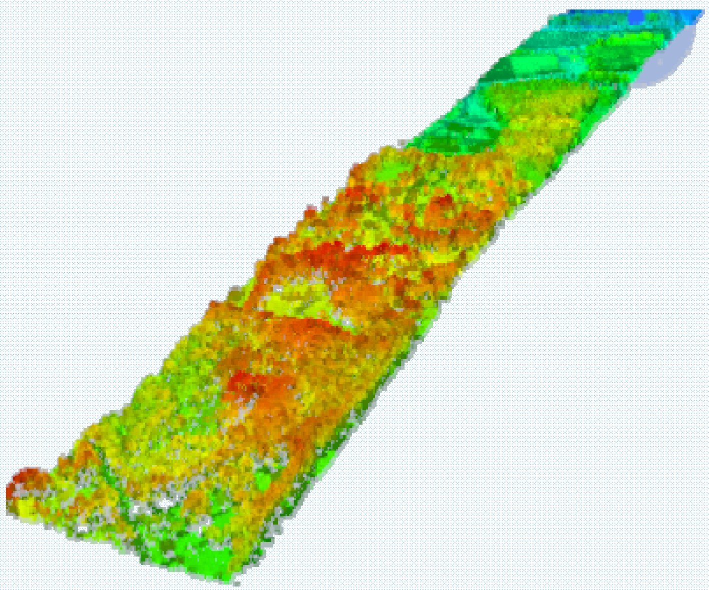
Xuefeng Guan

Wuhan University & TU Delft

Management of massive point cloud, December 8, 2015

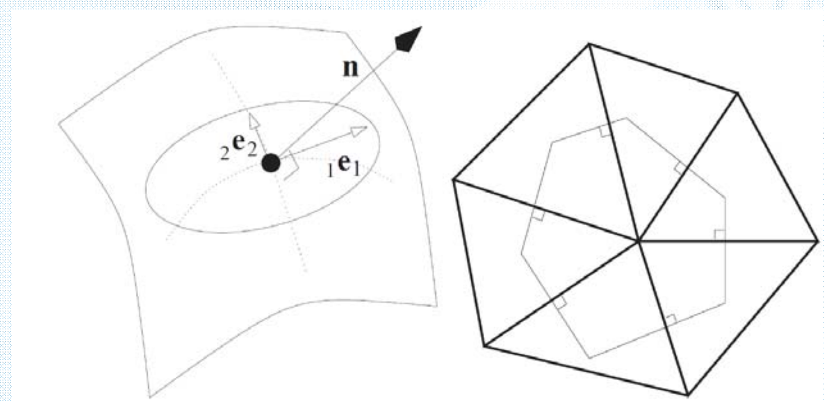
Contents

Background



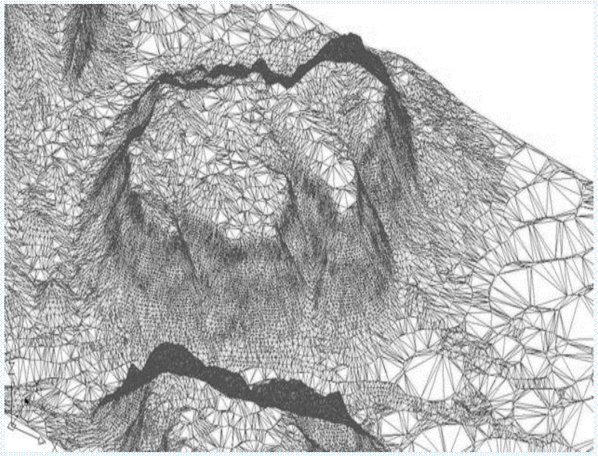
LiDAR Point Cloud

DT and its dual Voronoi can be used to derive point geometry attributes for the further continuous LOD research.



Normal, Gradient, Curvature, …

Challenges



Challenge 1:

During the process massive point cloud can not be fitting in the main memory of commodity computers.

Challenge 2:

Processing such massive point cloud, e.g. Delaunay Triangulation, are usually time-consuming.

Research Objective

To address these challenges, a parallel Delaunay triangulation algorithm will be proposed for processing billions of points from discrete LiDAR LAS files.

- Integrate the idea of streaming computation
- Exploit the parallelism of the multi-core platform

Contents

- Background & Research Objective
- Parallel Streaming Delaunay Triangulation

  ---Transformation from sequential D&C DT to parallel DT

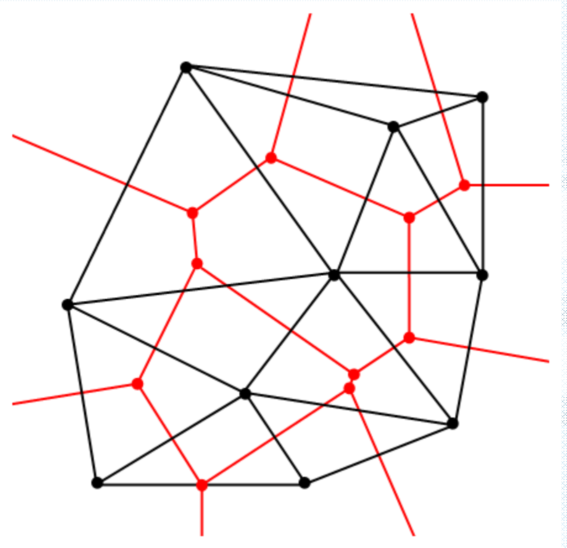
  ---Transformation from batch execution to streaming execution
- Performance Evaluation
- Conclusion & Future work

Delaunay Triangulation

A 2D Delaunay triangulation for a set ***P*** of points in a plane is a triangulation DT(***P***) such that no point in ***P*** is inside the circumcircle of any triangle in DT(***P***).

- DT & Voronoi diagram

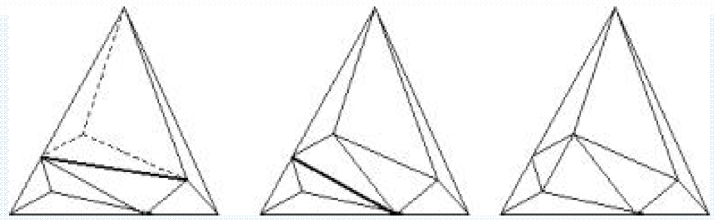
  dual graph



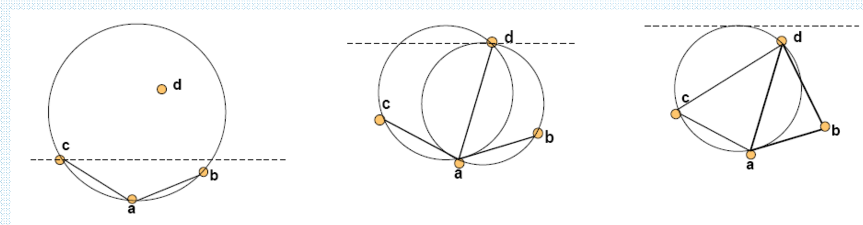
$$V(p) = \{x \in R^d : \|x - p\| \le \|x - q\|, \forall q \in P, p \ne q\}$$

Delaunay Triangulation Methods

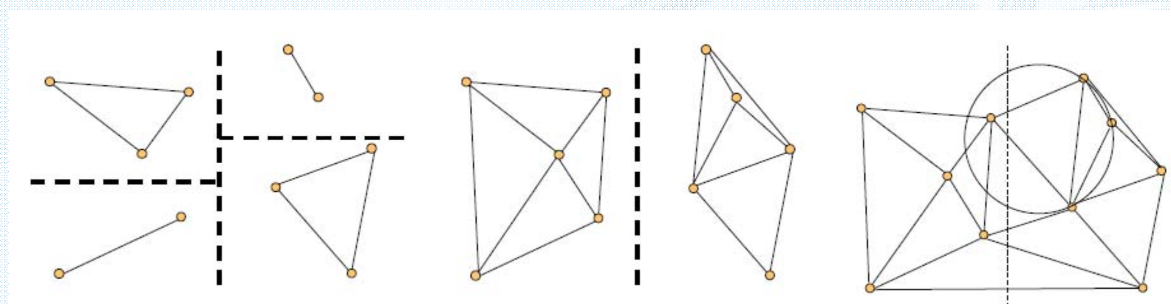
- Incremental Insertion



- Sweepline



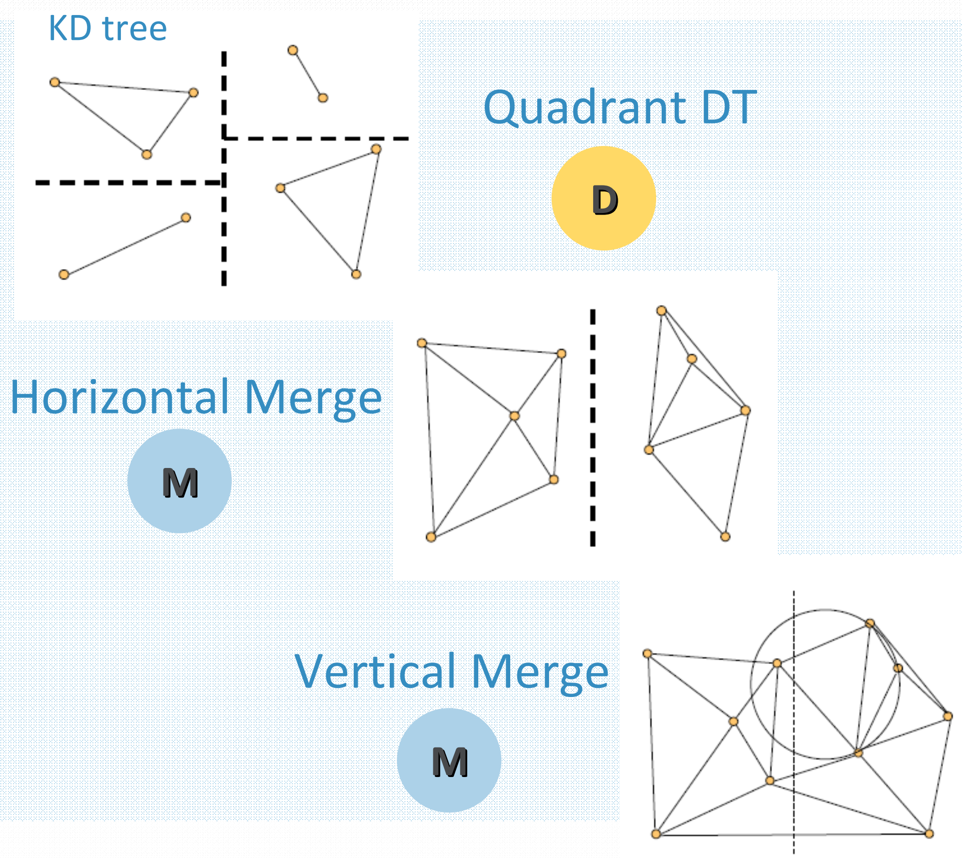
- Divide & Conquer (D&C)

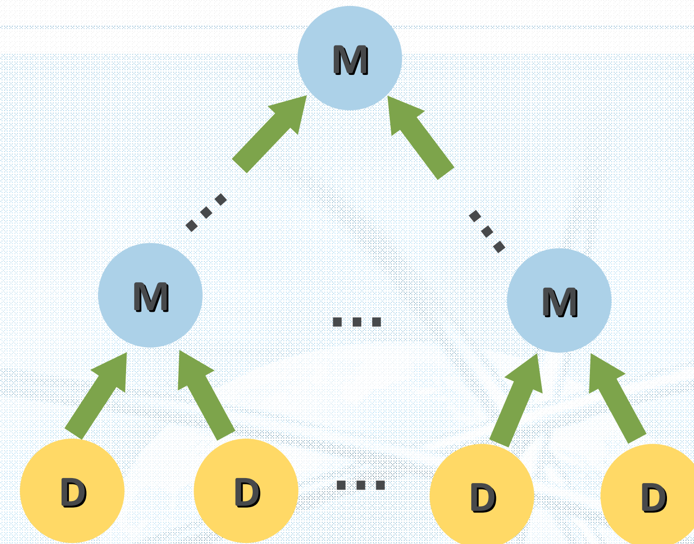


(a) (b) (c)

Triangle : Jonathan Shewchuk, UC Berkley
http://www.cs.cmu.edu/~quake/triangle.html

D&C Delaunay Triangulation

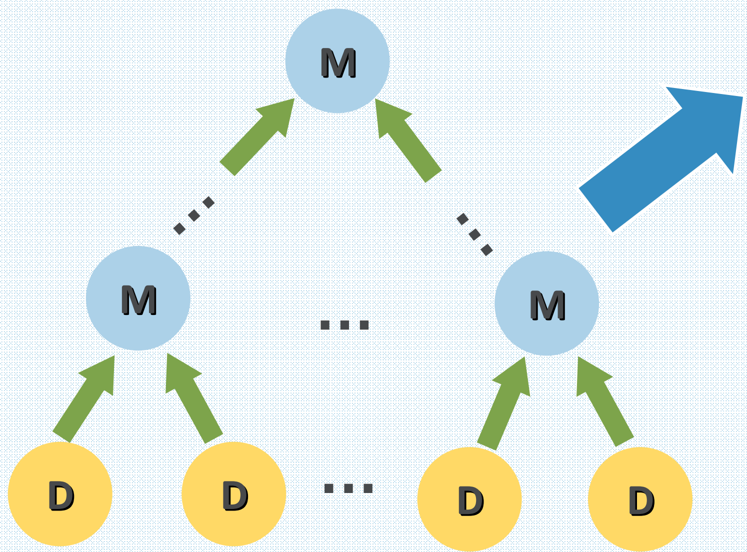




D can contain 2~3 points here, and also can be generalized to represent a rectangle block

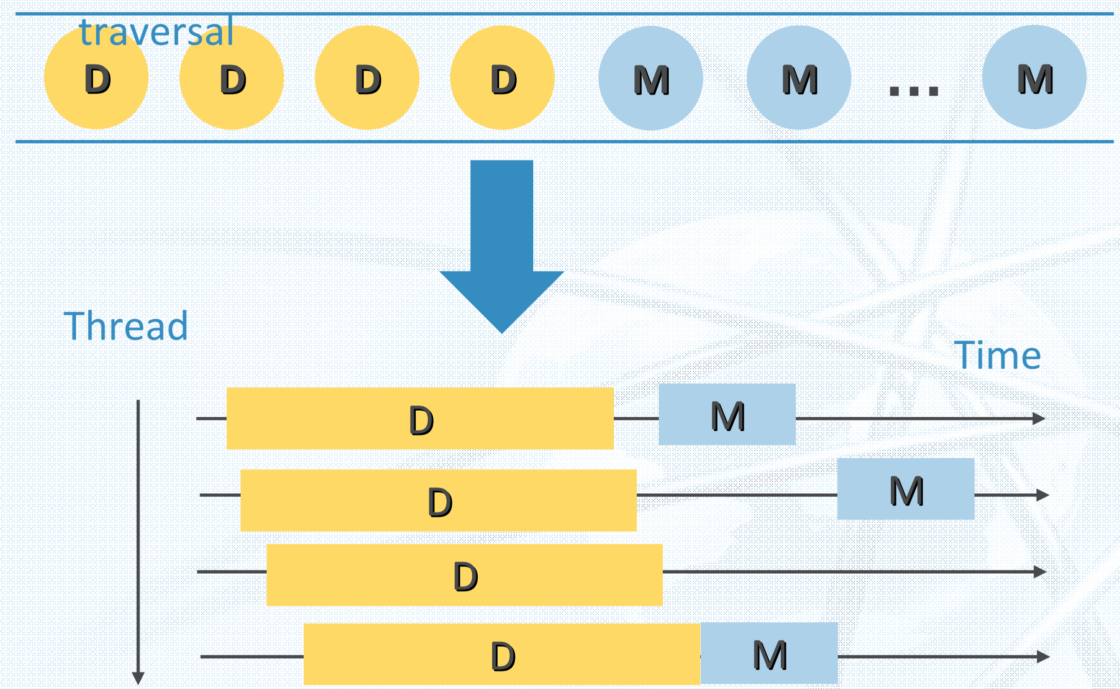
Different job granularity &
Different partition support
(kd tree, quad tree, grid, strips,...)

Parallel Streaming DT----Parallel pipelines



The job kd tree of a D&C DT

The job queue created by a breadth-first tree traversal



Parallel pipelines mapping

Streaming Computation

Streaming computation has **four** requirements:

- **Sequential data access**: in the streaming model, the input data that are operated upon should not be randomly accessed from disk or memory
- **Linear execution:** the operations on elements of the input streams are orderly chained and work together as a pipeline.
- **Data locality:** when discrete elements of input streams pass through the pipeline of operations, each kernel operation on a given element does not involve too far away elements of streams.

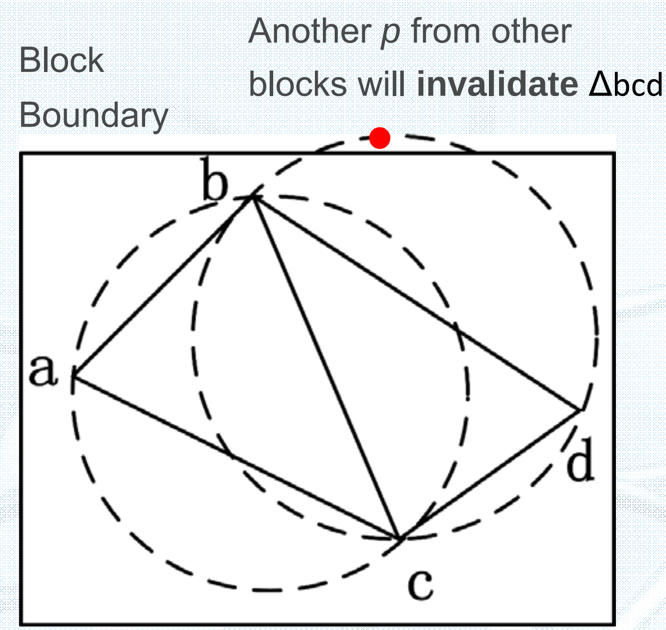
Spatial Sort

- **Memory recycling:** streaming computation loads the input data sequentially, processes elements of input using a small memory buffer, outputs results immediately and frees partial or entire occupied memory space for later use.

Parallel Streaming DT---- Finalized triangles

At any given time during the 2D Delaunay triangulation, the triangles in the mesh can be divided into two categories: *finalized* and *unfinalized*. The triangle, which has already been guaranteed to be part of the final output, is called the finalized triangle. Any triangle that will be eliminated and thus not appear in the final output, is called an unfinalized triangle (Isenburg et al. 2006)

Roughly determined by ○ and □ Intersection

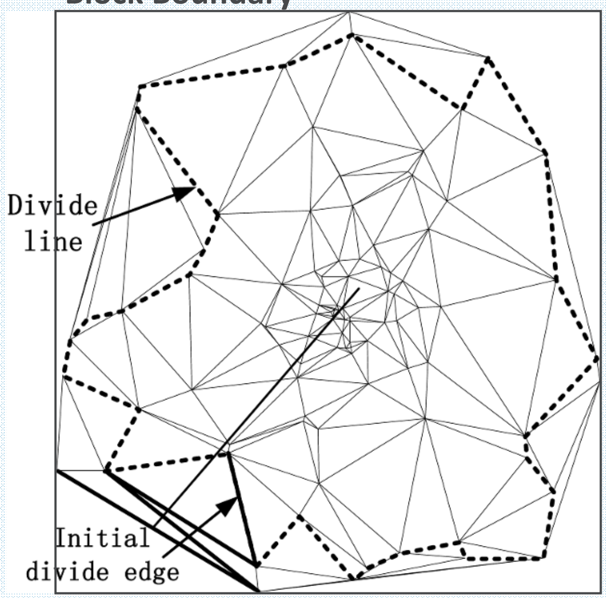


A finalized triangle(Δabc) &
An unfinalized triangle(Δbcd)

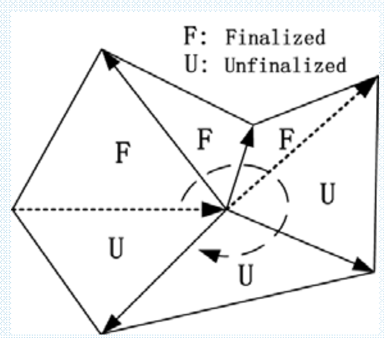
Parallel Streaming DT---- Erase of finalized triangles

Erase finalized triangles after the DT & Merge steps to realize memory reuse.

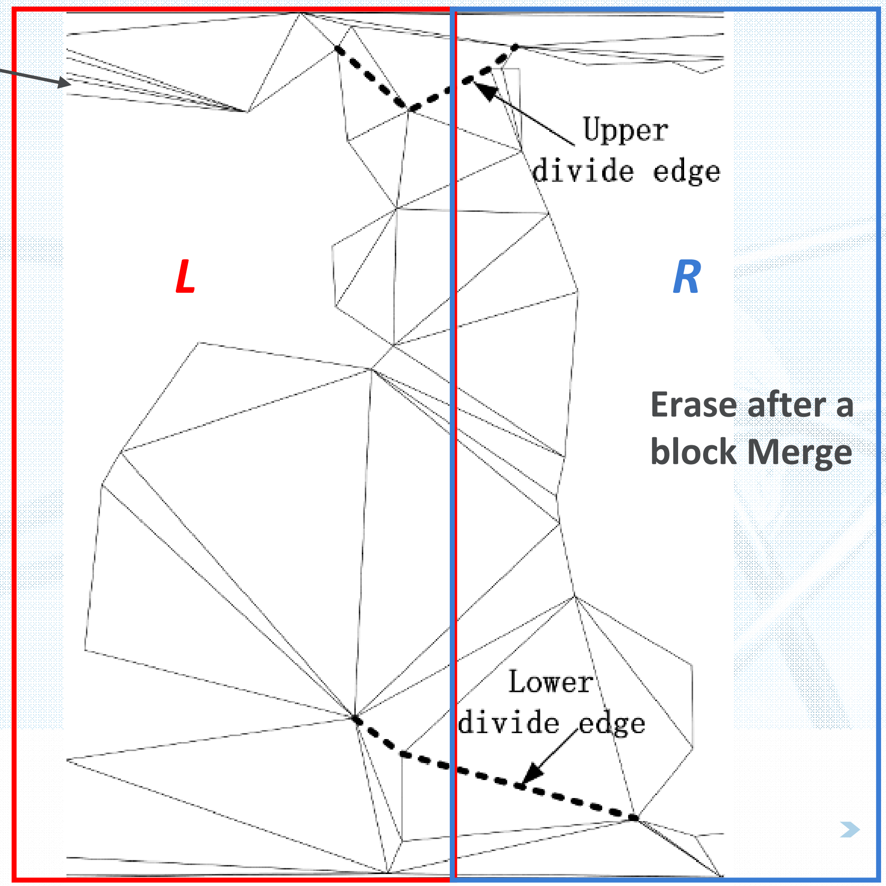
Block Boundary



Erase after each block DT

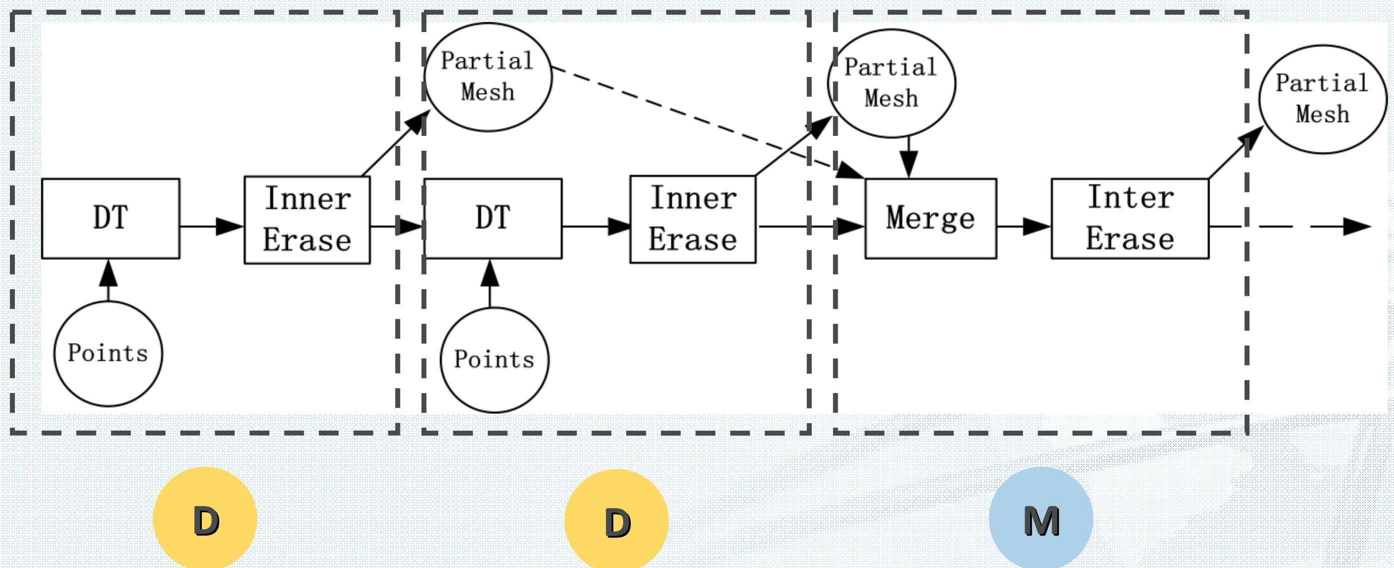


Subset after Δ erased



Erase after a block Merge

Parallel Streaming DT---- The final version of one DT pipeline



Contents

- Background & Research Objective
- Parallel Streaming Delaunay Triangulation
- <u>Performance Evaluation</u>
- Conclusion & Future work

Experimental Configuration

- Three performance evaluations

  ---*InnerErase* performance in the block DT

  ---Sequential performance compared with other open-source DT methods

  ---Parallel performance and scalability

- Experimental Data

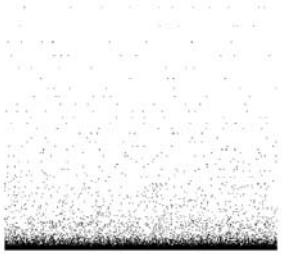
  ---Synthetic point datasets with different distribution patterns (*for test1*)

  ---real-world LAS files(0.883 billion points; 16.4GB; 2173 LAS files) (*for test2 and 3*)

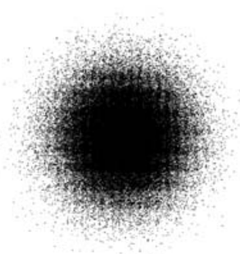
- Hardware

  ---Two quad-core Intel Xeon E5405(2GHz each core), 8GB DDR2-667 ECC SDRAM, and 1TB SATA hard disk(7200rpm, 32MB cache)

Evaluation of finalized triangle erase



Line



Normal

Uniform

Points ($\times 10^6$)	Line Distribution			Normal Distribution			Uniform Distribution		
	T_{dt}	T_e	Pct.	T_{dt}	T_e	Pct.	T_{dt}	T_e	Pct.
0.1	0.241	0.005	2.20%	0.175	5.91E-05	0.034%	0.177	0.001	0.49%
0.5	1.336	0.009	0.71%	1.122	9.21E-05	0.008%	1.109	0.003	0.23%
1	2.684	0.011	0.42%	2.476	1.02E-04	0.004%	2.421	0.004	0.16%
2	5.171	0.014	0.27%	5.384	1.18E-04	0.002%	5.284	0.006	0.11%
4	9.629	0.016	0.17%	11.685	1.20E-04	0.001%	11.292	0.009	0.08%

More

Less

T_{dt} is the time spent on Delaunay triangulation; T_e is the time spent on the *Inner Erase* step. *Pct.* is the ratio of T_e to T_{dt}. (*Time in seconds*)

Trivial and nearly negligible

Performance comparison between different DT methods

Blocks (LAS files)	Points (million)	ParaStream		Triangle(D&C)		Streaming TIN		
		Time(s.)	Memory	Time	Memory	Level	Time	Memory (MB)
1	0.53	1.30	44.08	1.36	39.94	4	1.53	7.42
2	0.87	2.09	36.85	2.07	64.46	4	2.48	7.85
4	1.79	4.37	43.87	4.39	131.94	6	5.56	7.88
16	6.85	16.36	50.45	15.97	499.60	6	20.08	13.19
32	15.25	36.50	57.39	35.74	1110.29	8	51.70	26.13

Execution Time:
1. Triangle
2. ParaStream
3. Streaming TIN

Memory Usage:
1. Streaming TIN
2. ParaStream
3. Triangle

Streaming TIN uses an internal multilevel quadtree in each block. This means a much **smaller** granularity.

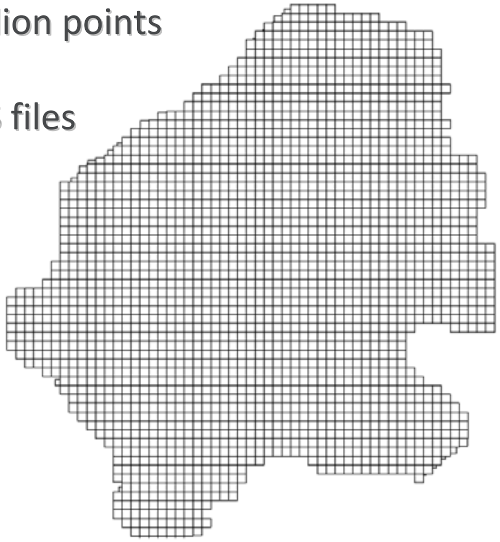
Triangle : Jonathan Shewchuk, UC Berkley, 1996
Streaming TIN(LAStools): Martin Isenburg, UNC, 2006 (The owner of Rapidlasso GmbH)

Evaluation of parallel streaming DT

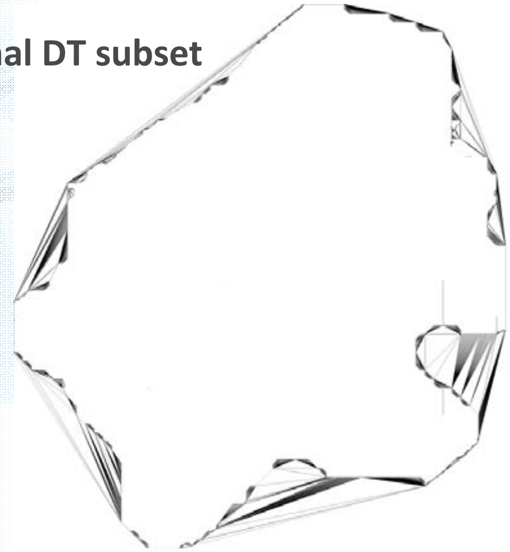
Threads	Direct output		Compression on output		
	Time(s.)	Speedup	Time	Speedup	Memory(MB)
1	2584.01		6195.94		313.17
2	1623.03	1.59	3207.84	1.93	365.41
4	1269.79	2.03	1624.04	3.82	456.80
6	1255.62	2.06	1144.17	5.42	506.16
8	1280.79	2.02	965.17	6.42	590.49

Less I/O, Faster DT, Better Speedup

0.883 billion points
16.4GB
2173 LAS files



Final DT subset



Contents

Conclusion

A robust parallel D&C Delaunay triangulation scheme called ***ParaStream*** is proposed particularly for shared-memory multicore architectures to process billions of LiDAR points.

- Scalable for billions of points
- Scalable for tens of CPU cores
- Very efficient memory usage

Future work

- 2D TIN → 3D Delaunay Triangulation;
- Evaluation with much larger datasets (e.g. ahn2, 640 billion);
- Implementation on top of database(e.g. MongoDB);
- Hybrid computing support (CPU plus GPU tasks);
- Applied in the discrete/continuous LoD point clouds.

Thanks for your attention!

guanxuefeng@whu.edu.cn