Using a Space Filling Curve for the Management of Dynamic Point Cloud Data in a Relational DBMS

Stella Psomadaki
P5 Presentation

Theo Tijssen
Peter van Oosterom
Roderik Lindenbergh
Ulf Hackauf
External: Fedor Baart

TU Delft

Deltares

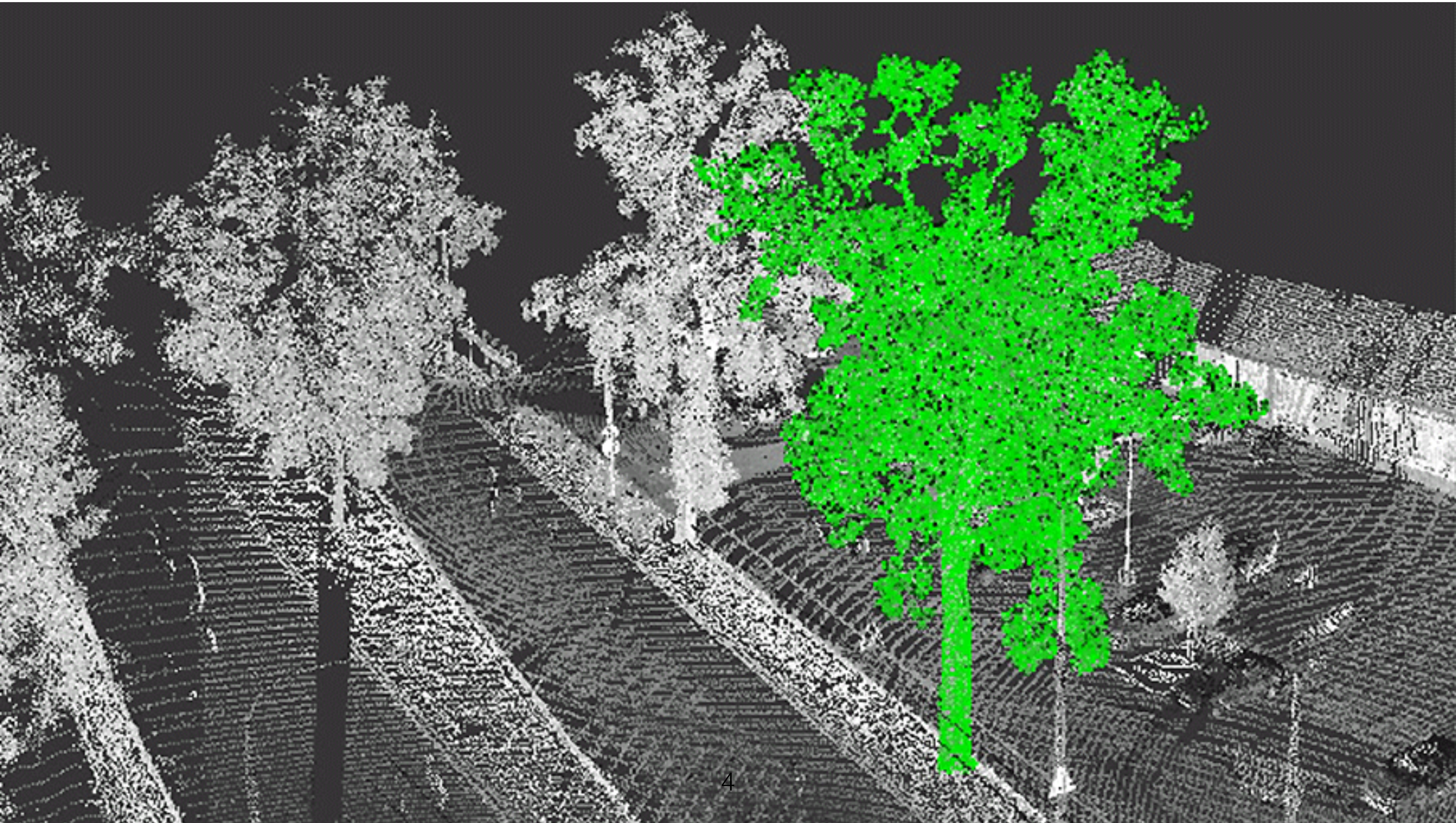
Contents

- Introduction
- Methodology
- Results
- Conclusions & Future work

Introduction

What is a Point cloud?

Source: gisuser



Point clouds

- Rapid growth in point cloud usage
- The management of point clouds is challenging
- Typically managed using files (e.g. LAS, LAZ)
- …But, DBMSs provide point cloud management solutions.

ahn2.pointclouds.nl/

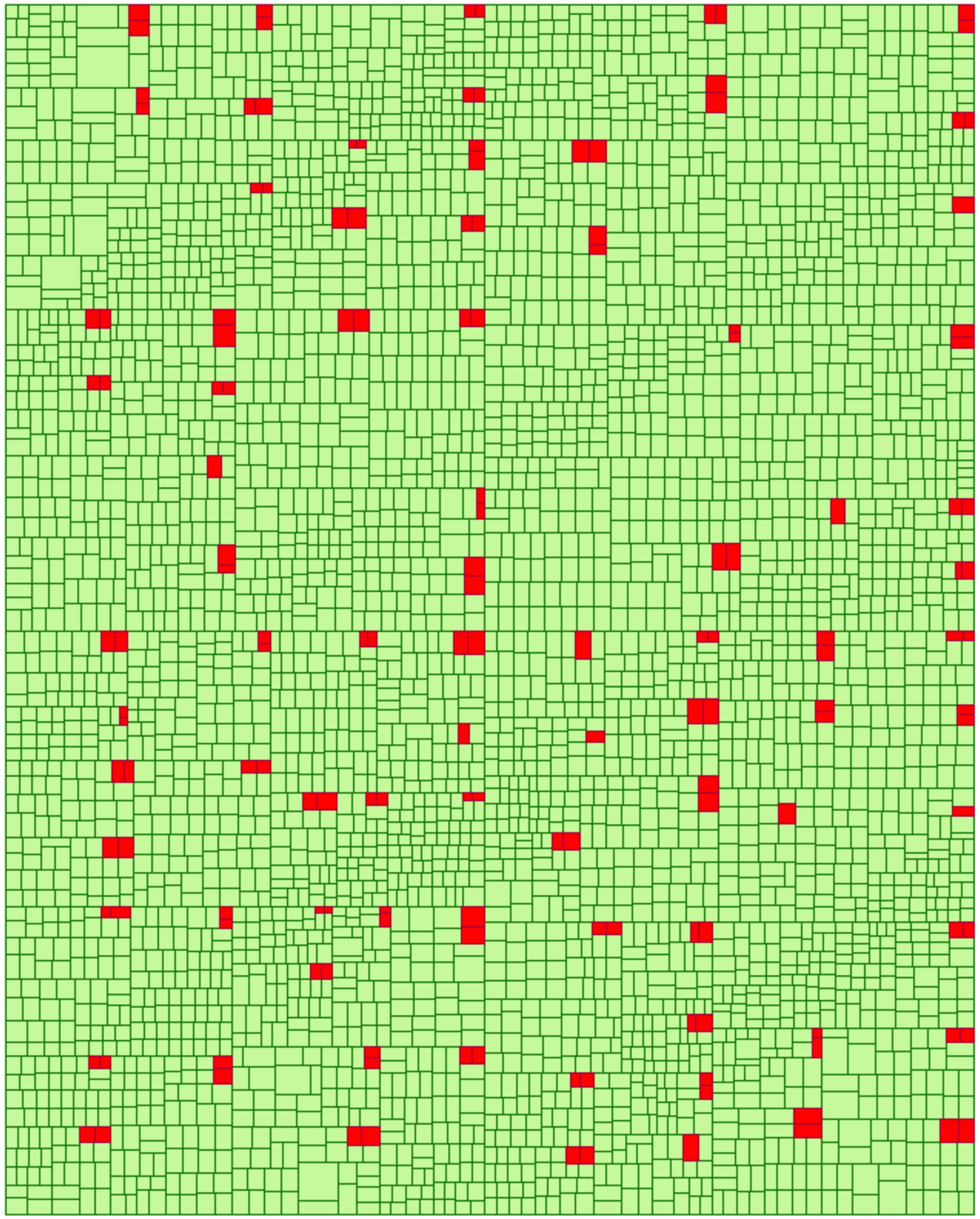
Management of PC in DBMS

Current approaches:

- Oracle *SDO_PC*
- PostgreSQL *pgpointcloud*

Organise points in **blocks**, meaning groups of spatially close points

...or use a normal **flat table**

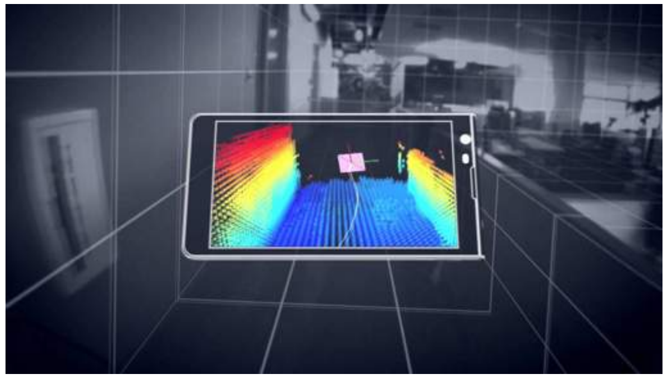


Source: Massive point clouds for eSciences
http://www.gdmc.nl:8080/mpc/

Dynamic point clouds

- Today, developments in point cloud acquisition devices allow repeated scans of the same area
- Dynamic point clouds
  - growing datasets
  - *time is an additional dimension*

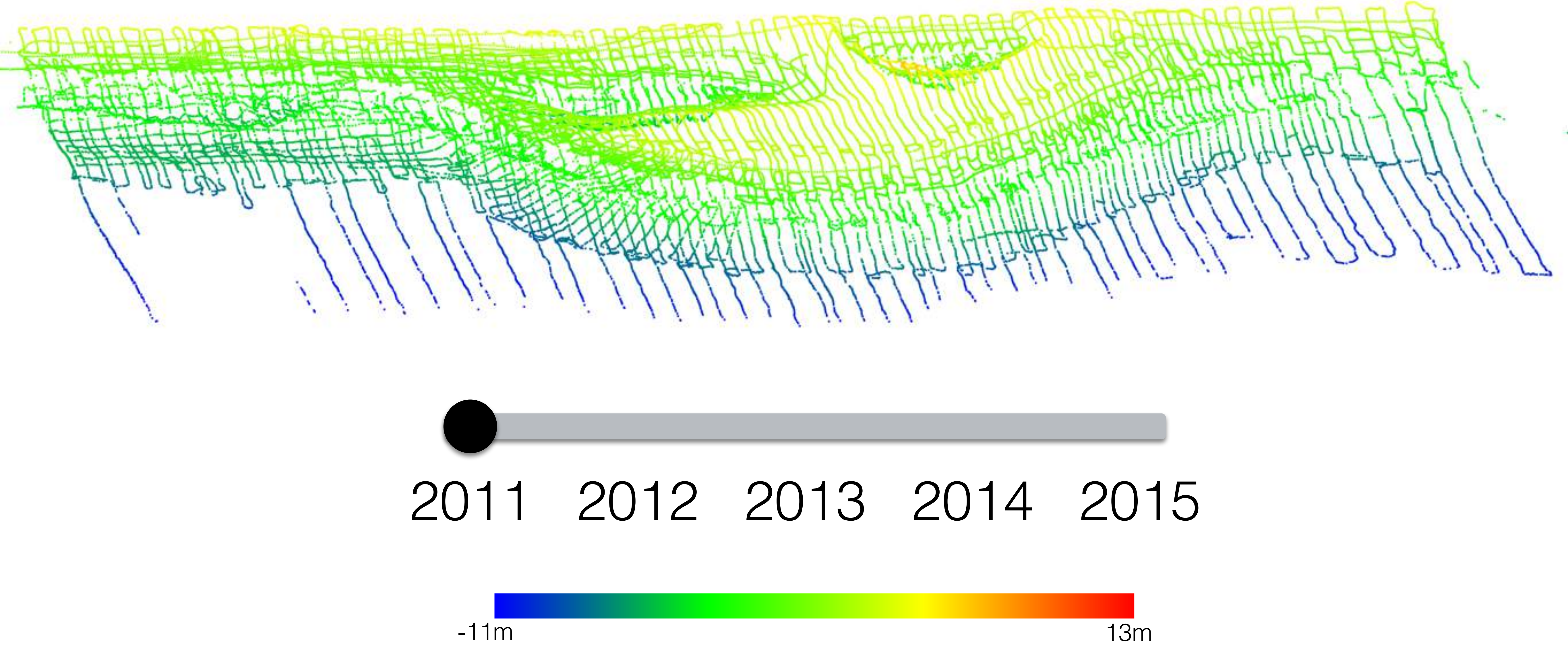
Source: youtube.com



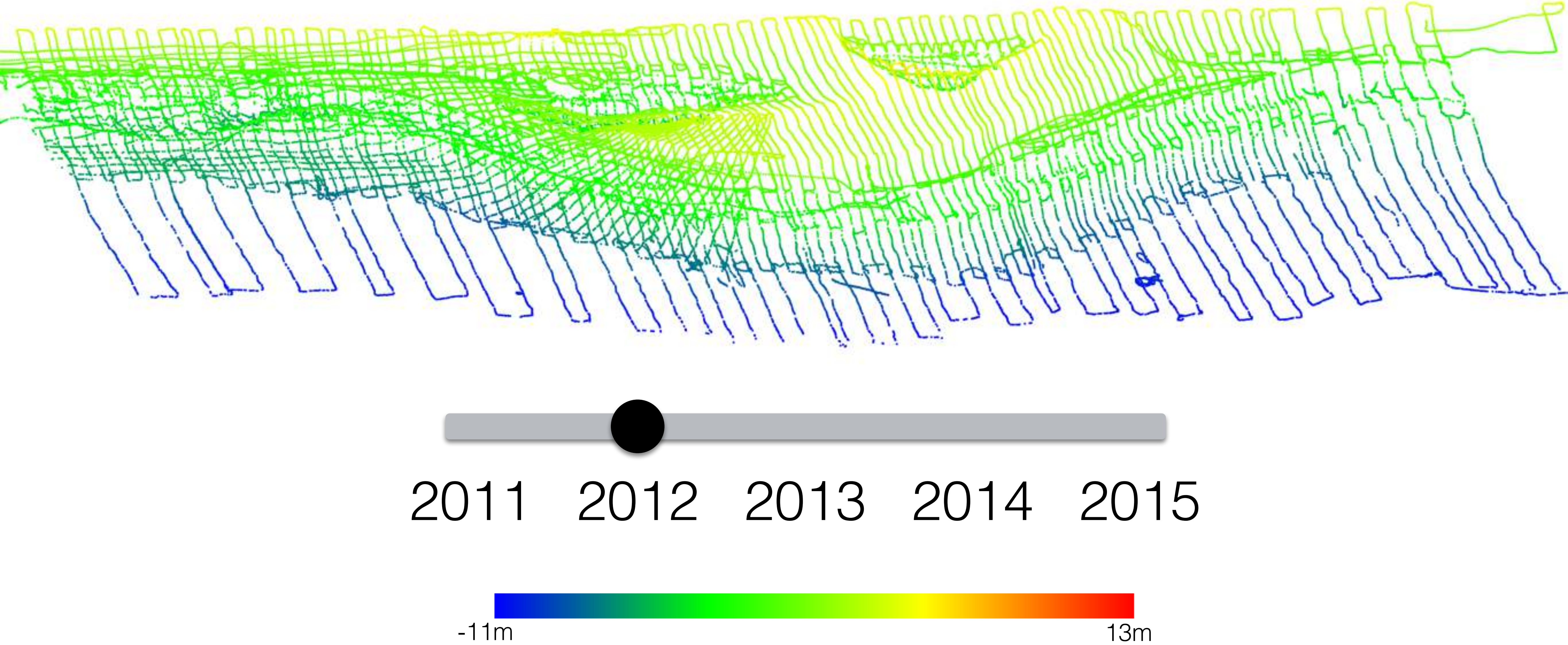
Source: Flickr

Source: TU Delft

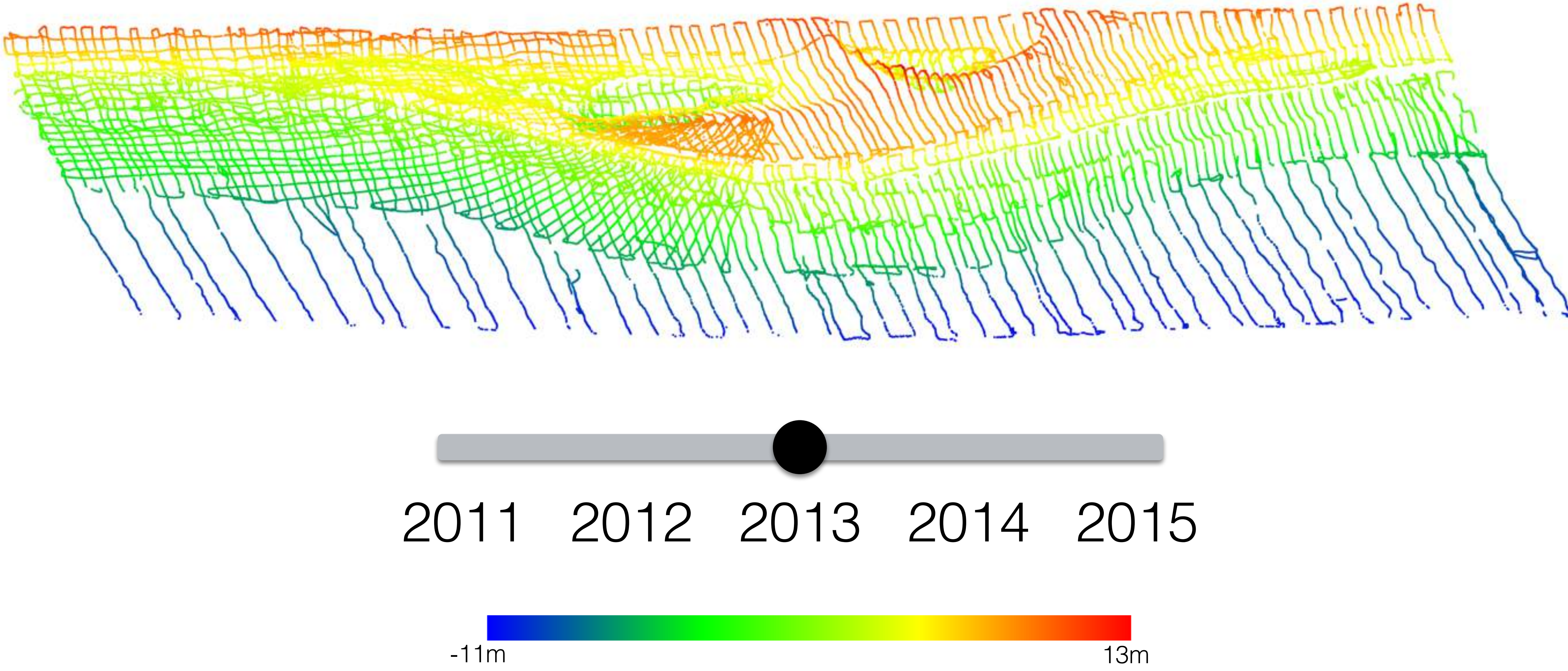
Source: Wikimedia Commons



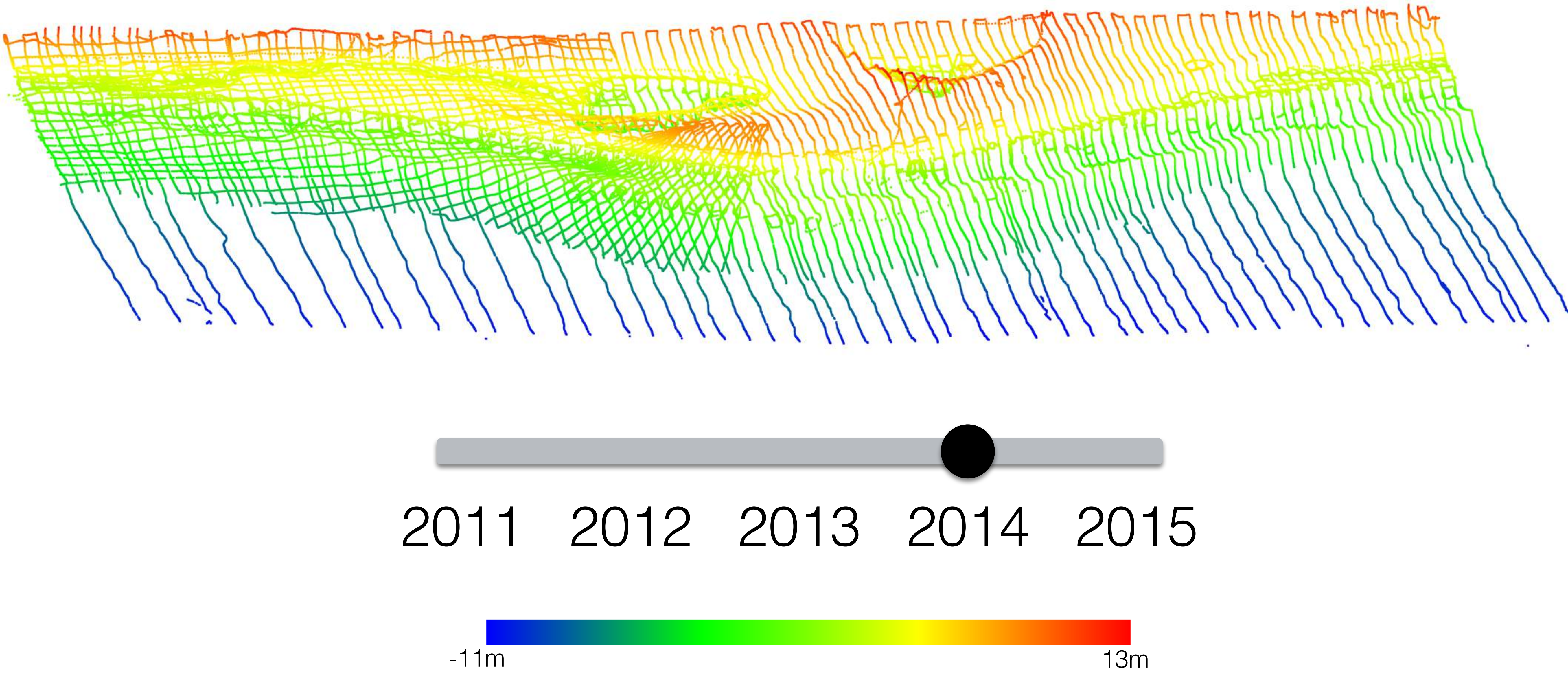
Data courtesy of Deltares



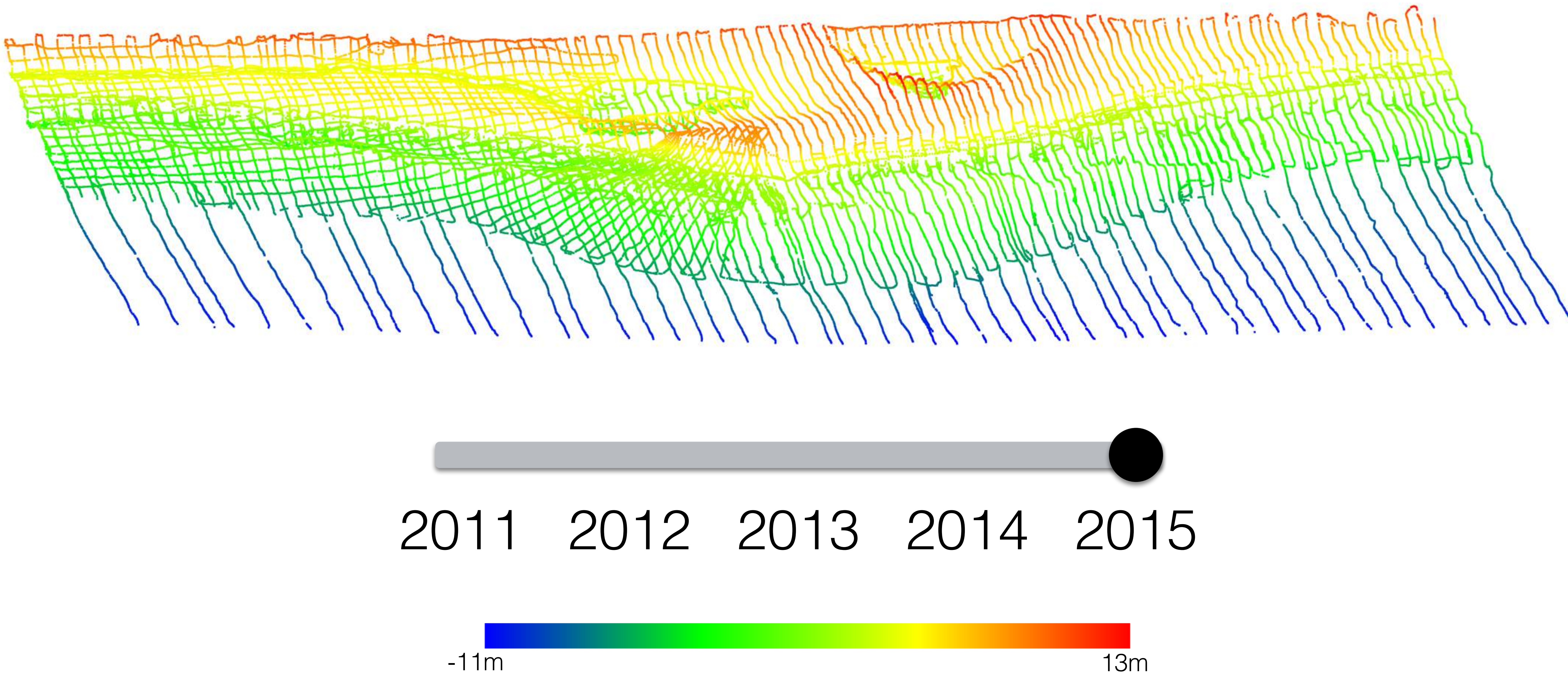
Data courtesy of Deltares



Data courtesy of Deltares



Data courtesy of Deltares



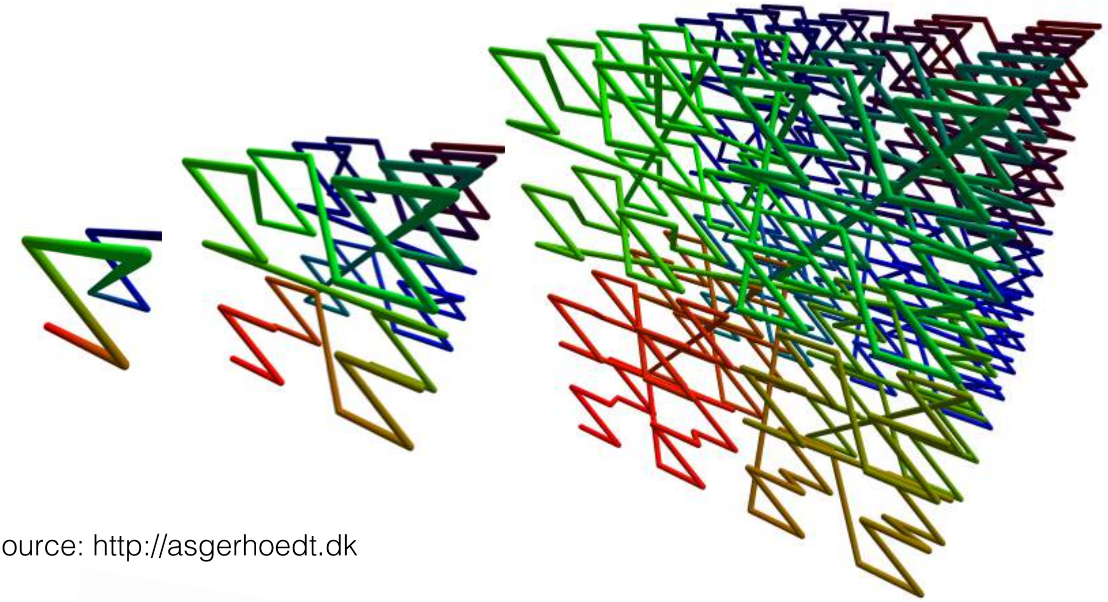
Data courtesy of Deltares

Managing dynamic PC?

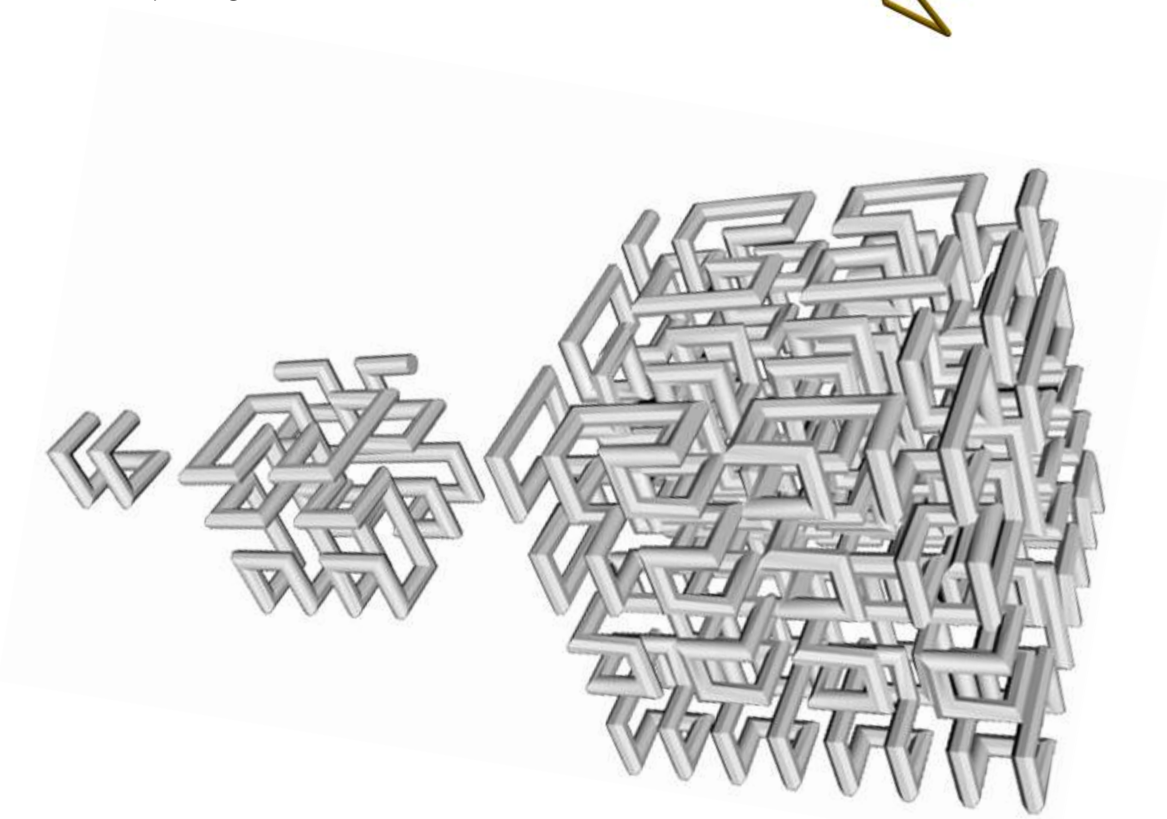
- Blocks
  - ☑ compact storage with better scalability, less overhead, better compression
  - □ overlapping blocks, adding new data not trivial
- Flat
  - ☑ flexible, insertions trivial, Use a SFC to improve the organisation (van Oosterom et. al., 2015)
  - □ large storage requirements, overhead

Space Filling Curves

- Apply a linear ordering to a multidimensional domain
- Why?
  - Dimensionality reduction
  - Full resolution curve
  - Clustering of points

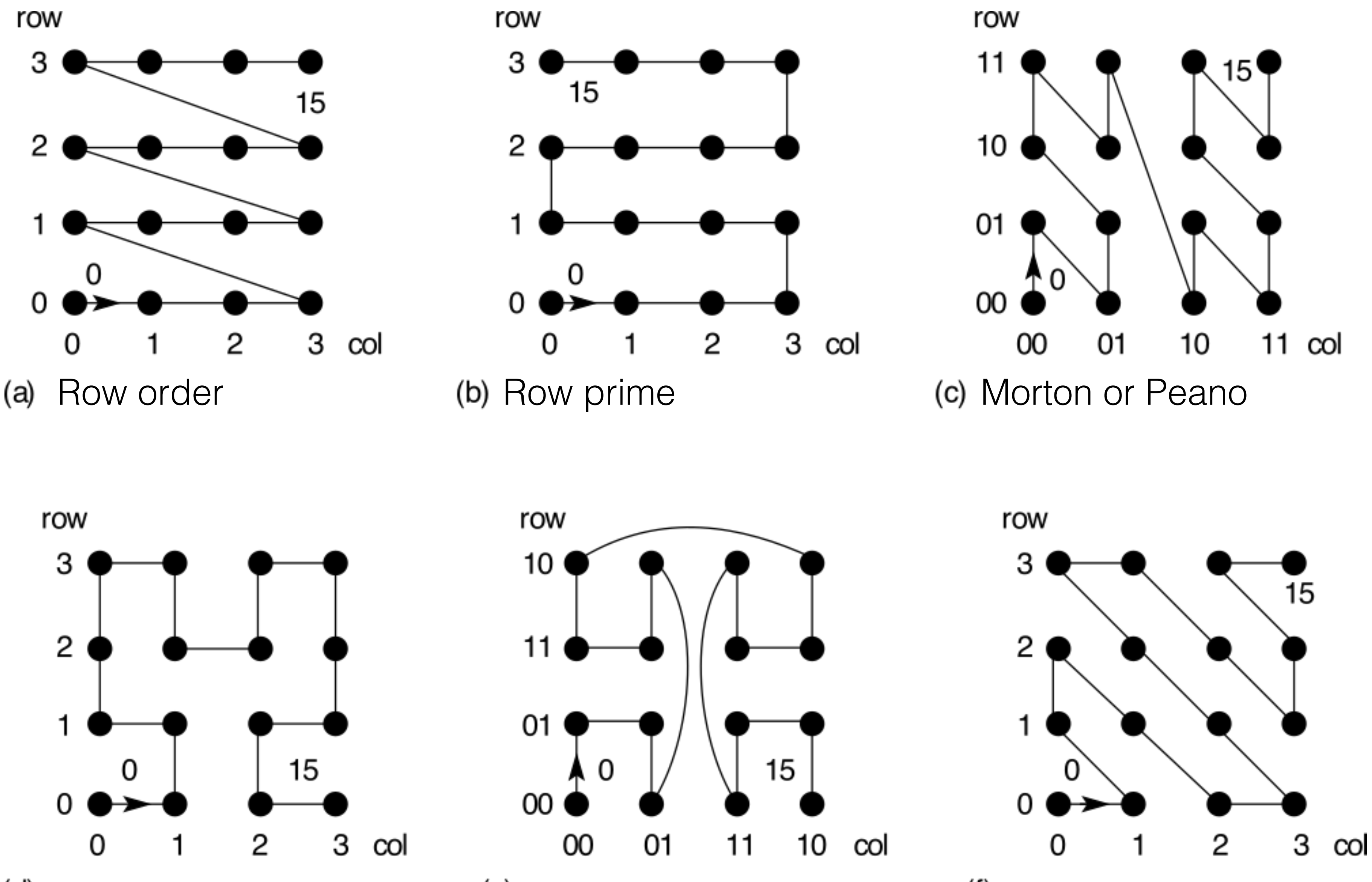


Source: http://asgerhoedt.dk



Sánchez-Cruz H, Bribiesca E;
Study of compression efficiency for three-dimensional discrete curves.
Opt. Eng. 0001;47(7):077206-077206-9.

Space Filling Curves



(a) Row order

(b) Row prime

(c) Morton or Peano

(d) Hilbert

(e) Grey

(f) Cantor - diagonal

Source: van Oosterom, Peter. "Spatial access methods." *Geographical information systems* 1 (1999): 385-400.

Space Filling Curves

- Morton Curve

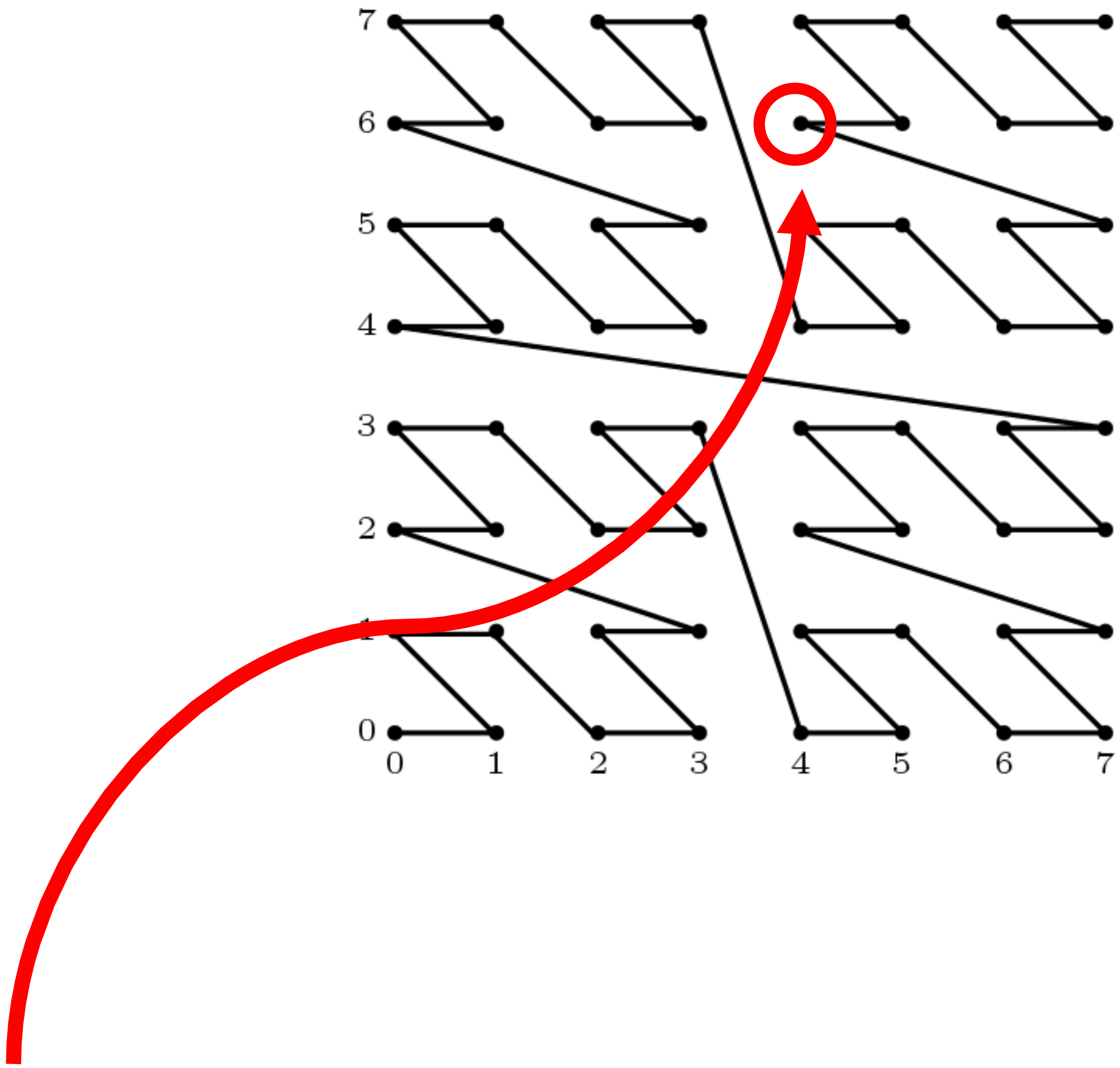
- Bitwise interleaving

Example:

x = 4 or 0100 in binary

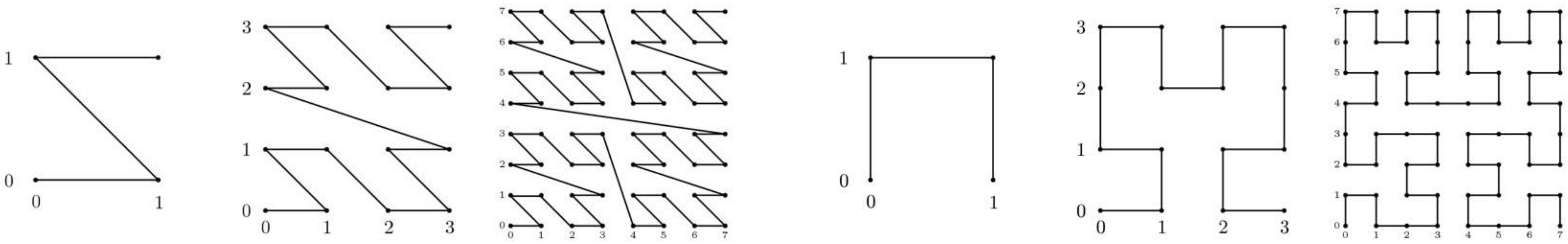
y = 6 or 0110 in binary

morton = 00111000 or 56



Research Question

Is a Space Filling Curve (SFC) approach an appropriate method for integrating the space and time components of point clouds in order to support efficient management and querying (use) in a DBMS?



Methodology:
A Space Filling Curve approach

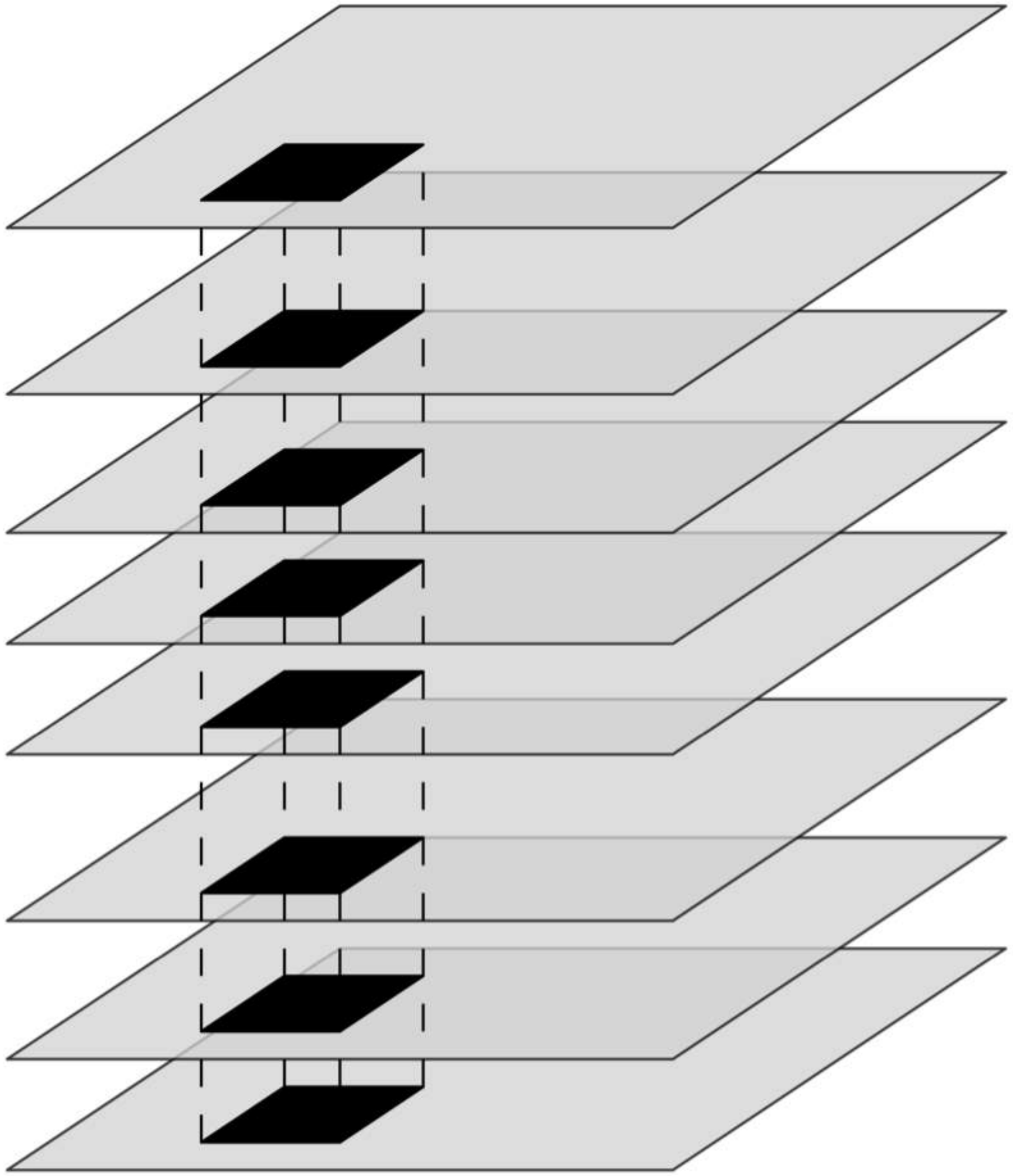
Requirements

Requirements for spatio-temporal data management [Adapted from Gaede and Gunther, 1998]:

- Should support *operations* other than just retrieval of the data.
- Should be *dynamic*: support insertions
- Should be *scalable*: adapt to growing database.
- Should be *efficient* in terms of time (and space): minimise as much as possible the number of disk accesses

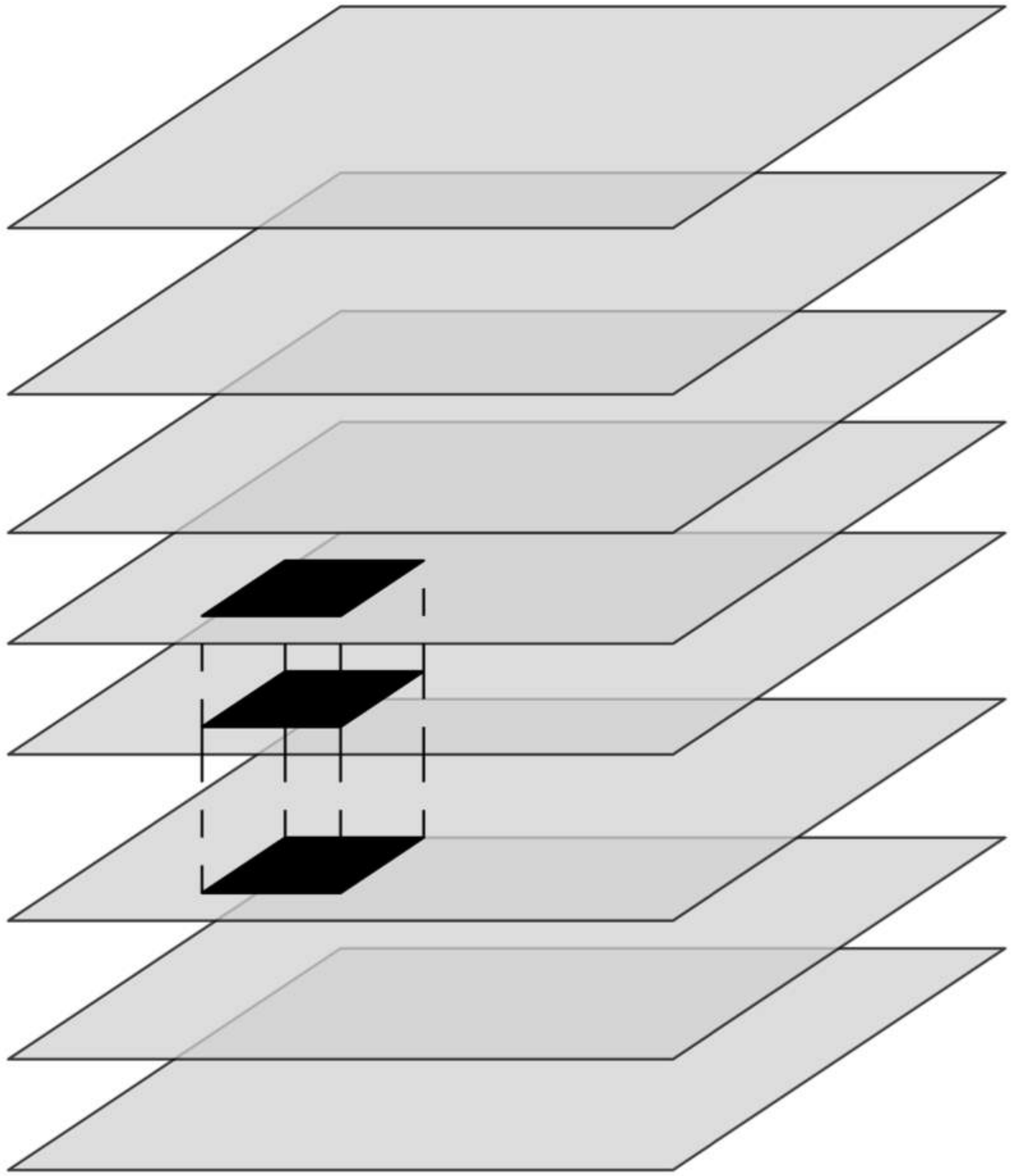
Important queries

- **Space** queries: all points located in a specific area over the complete time range



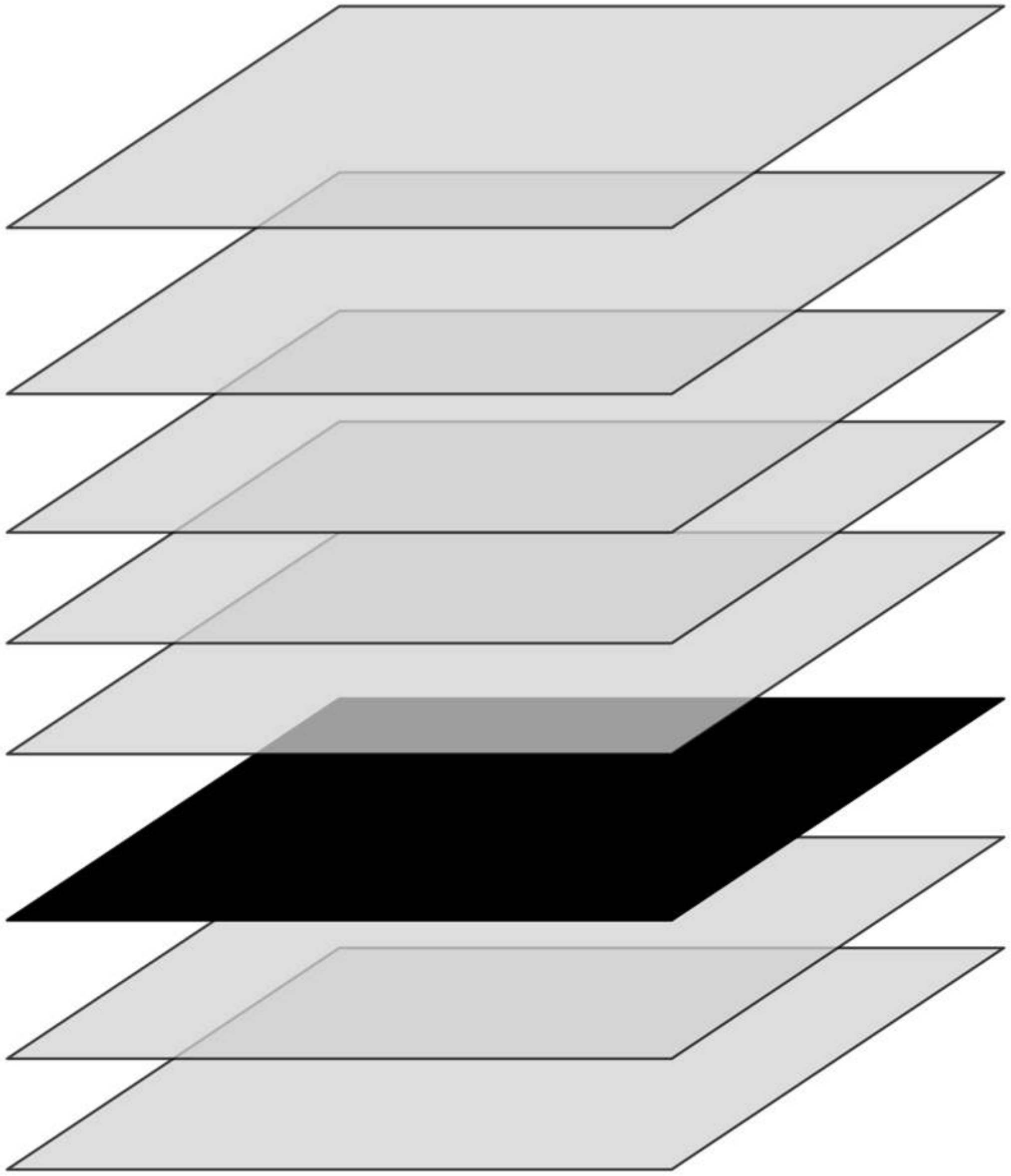
Important queries

- **Space - time** queries: all points located in a specific area during a specific time range



Important queries

- **Time** queries: all points of a specific time moment or range, for the whole spatial domain



A SFC approach

Structuring space and time is not a trivial problem.
Contradiction:

- Points close in space and time should be stored (up to a certain extent) in contiguous blocks in disk, for *fast spatio-temporal retrieval*.

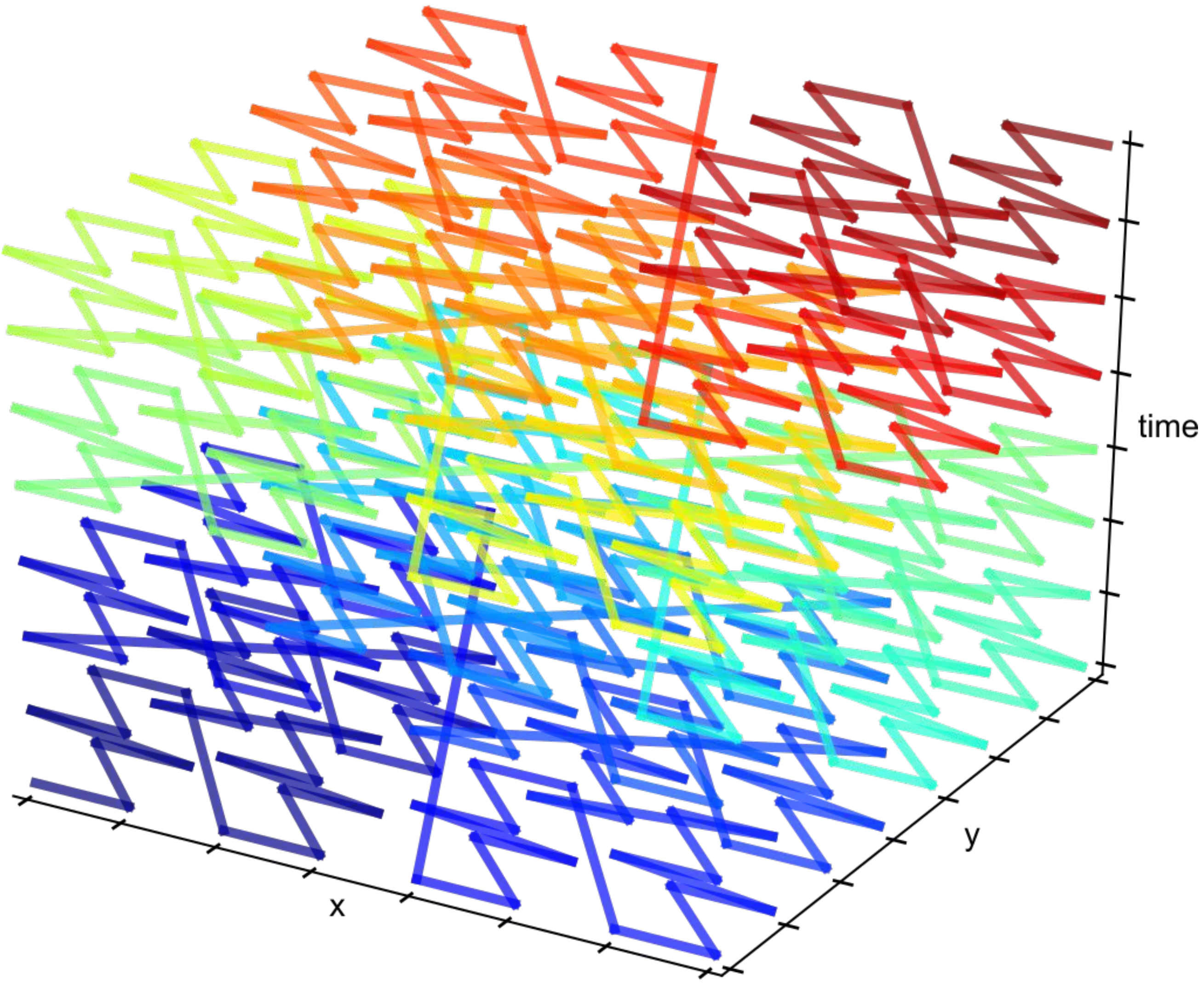
- Already organised points should not be reorganised when inserting new data, *for fast loading.*

A SFC approach

Integrated space and time approach: all dimensions have equal part in SFC.

Two treatments of z:

1. as an attribute.
2. as part of the SFC key.

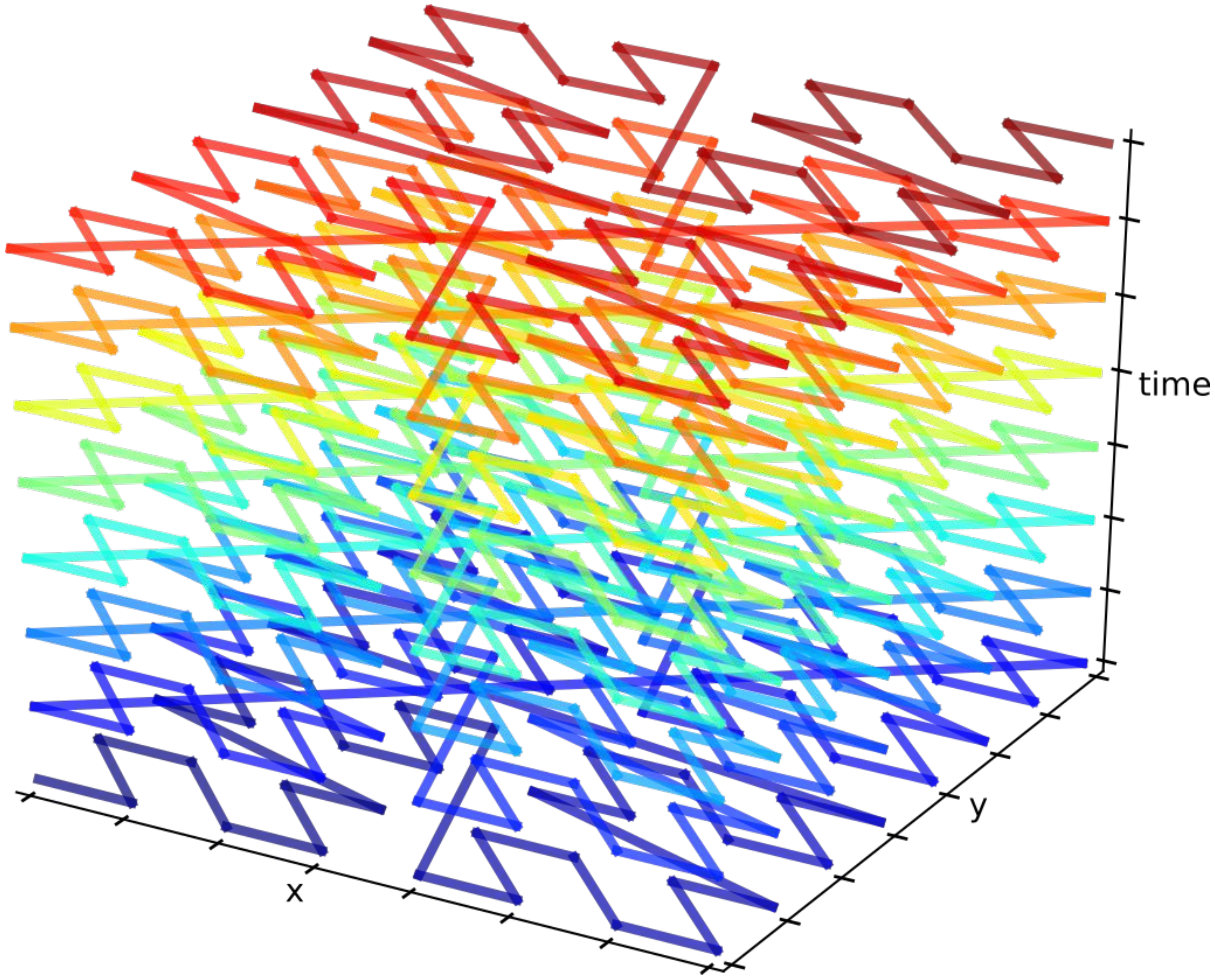


A SFC approach

Non-integrated space and time approach: time dominates over space.

Two treatments of z:

1. as an attribute.
2. as part of the SFC key.



A SFC approach - Loading

Two step approach:

- **Preparation**: Read files and convert to SFC key, according to
  - o integration of space and time,
  - o treatment of z and
  - o scaling of time

  The data are bulk loaded into a normal heap table

- **Loading**: Sort the data based on the key into an Index Organised Table (data stored in the B-Tree index)

A SFC approach - Query

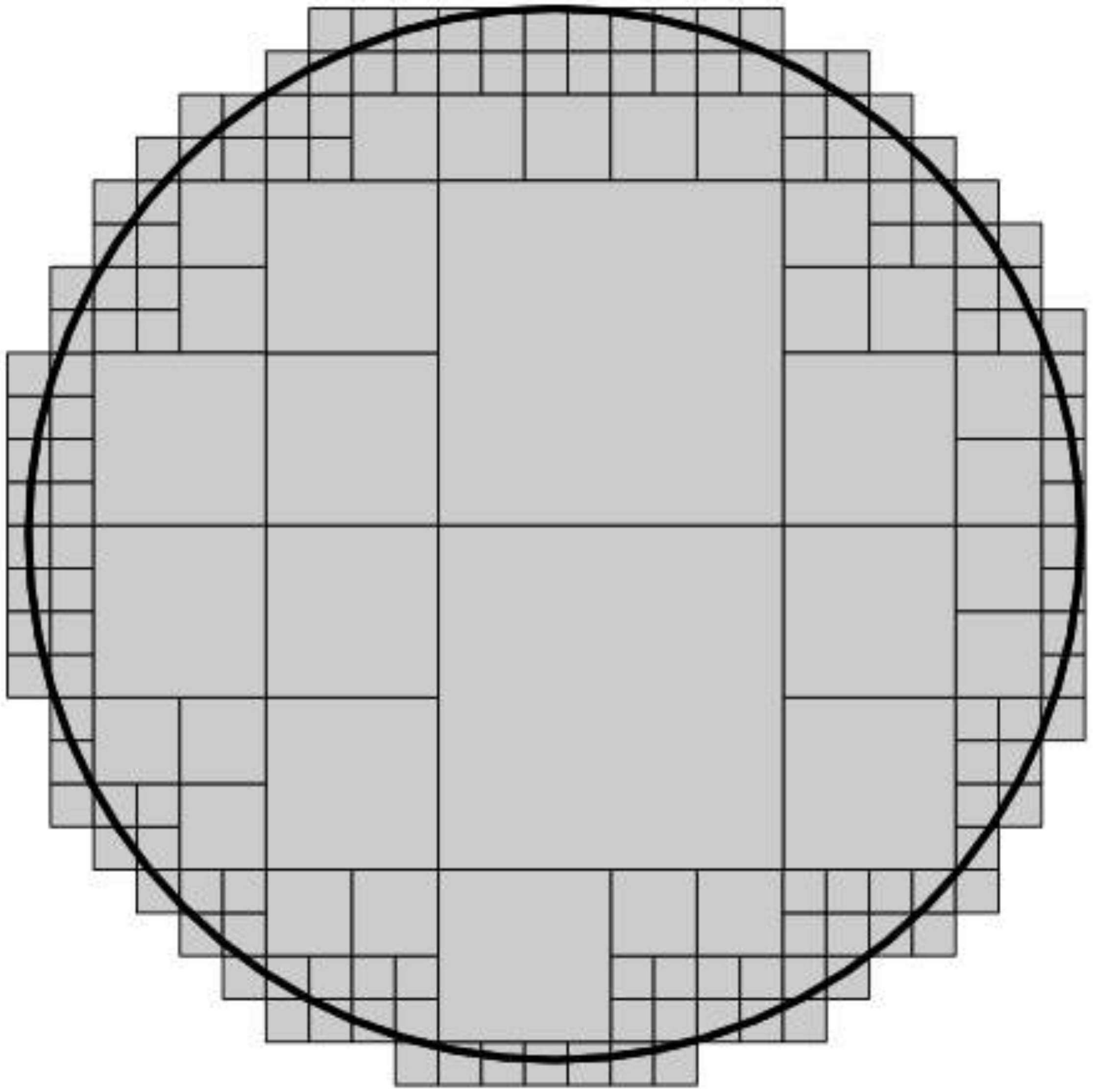
- Translation of the n-D query geometry into a number of continuous runs on the curve.
- Take advantage of the quadrant recursive characteristic of Morton curve: Use a Quadtree/ Octree/ 2^n-tree
- The maximum depth of the tree affects:
    - the number of ranges
    - the approximation of the query geometry

A SFC approach - Query

Multi-step query procedure

- Filter step: approximate query geometry using the 2^n-tree
- Fetch the approximated data and decode back to the original dimensions
- Refinement step: Detect the false hits using a Point in Polygon operation, or time and z refinement.

A SFC approach - Query



Identify Tree Cells

Direct neighbour merging

Reduce the number of ranges without affecting the approximation, by merging neighbouring ranges.

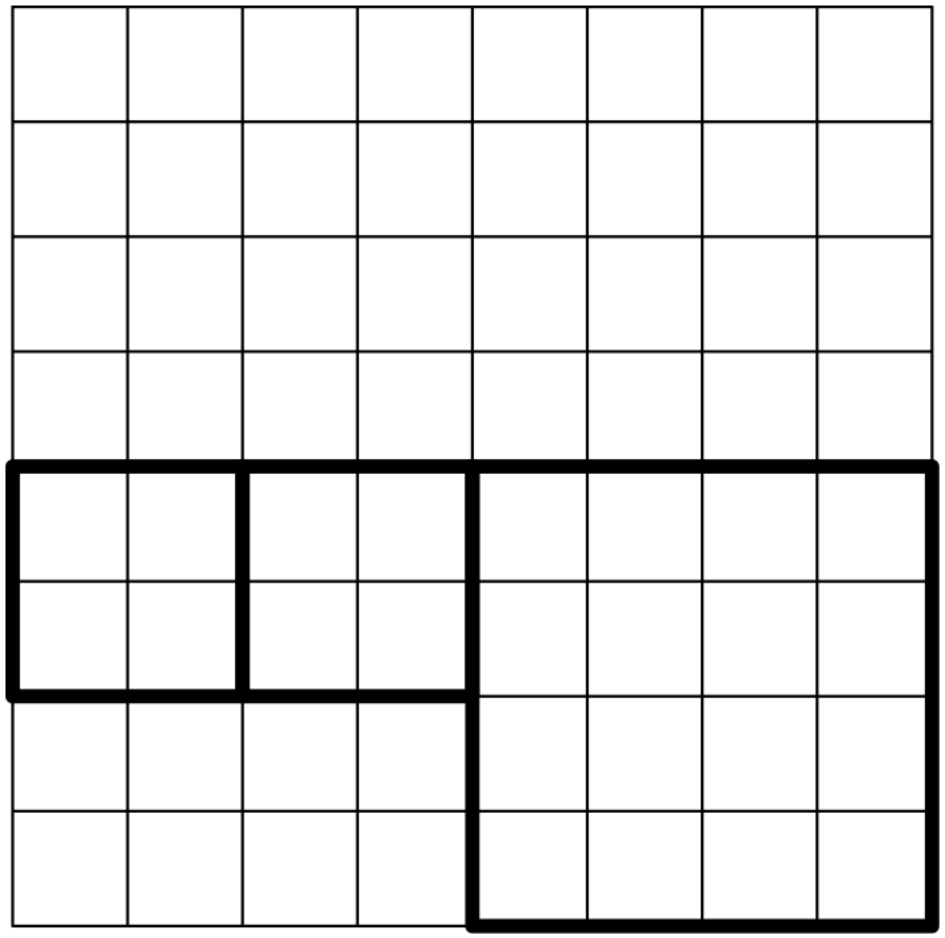


Figure a: Original 3 ranges

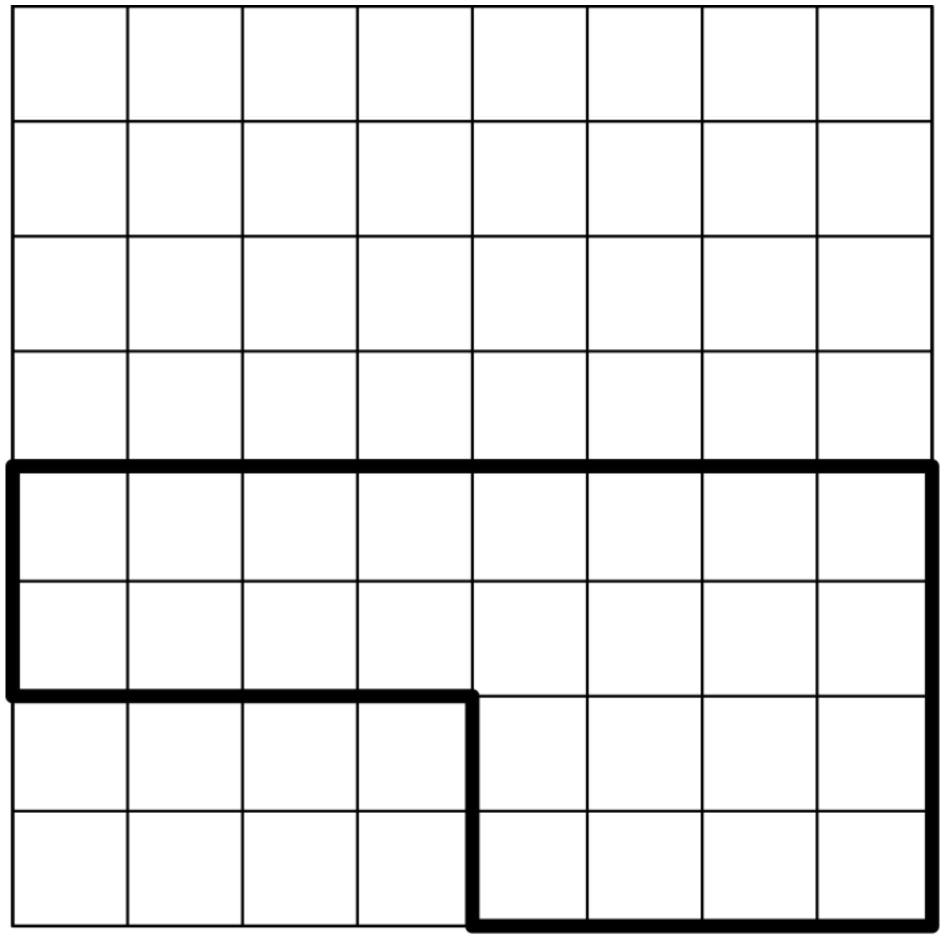
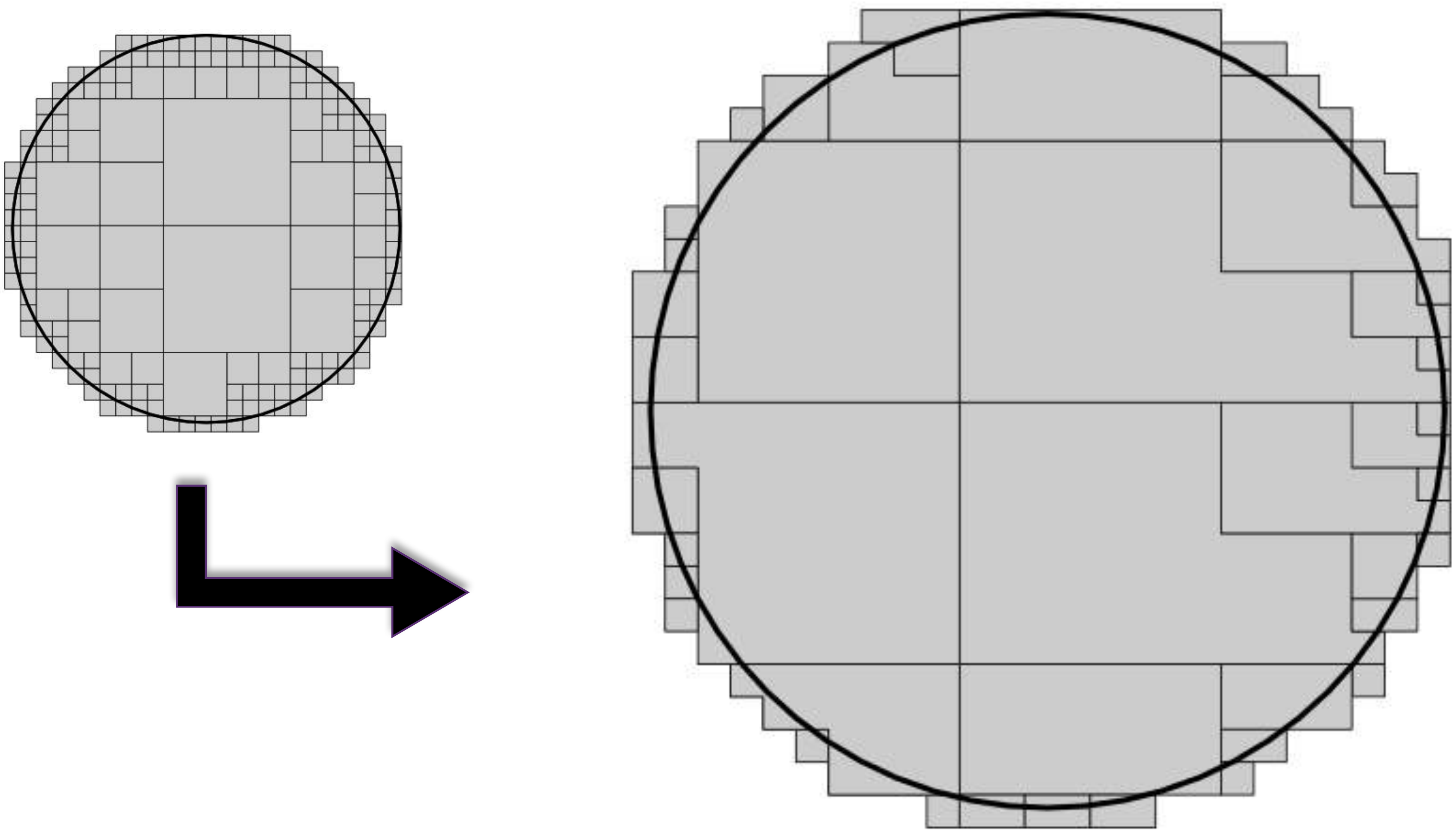


Figure b: Direct neighbour merging (1 range)

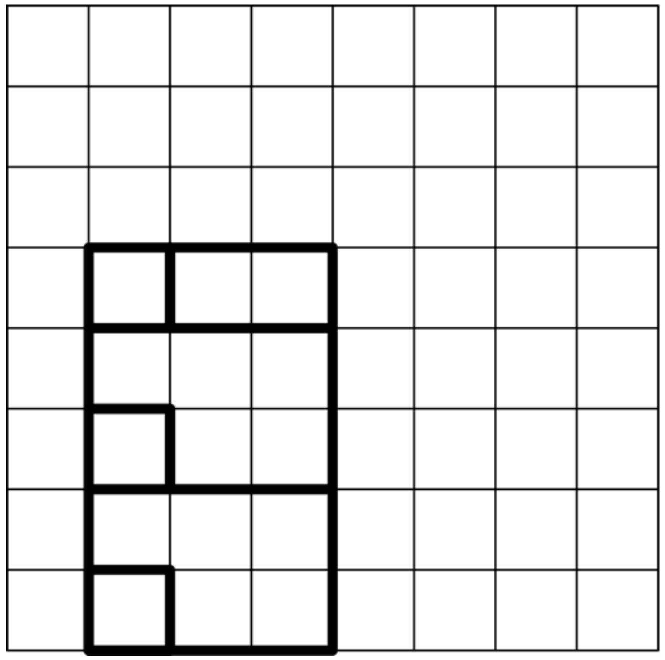
Direct neighbour merging



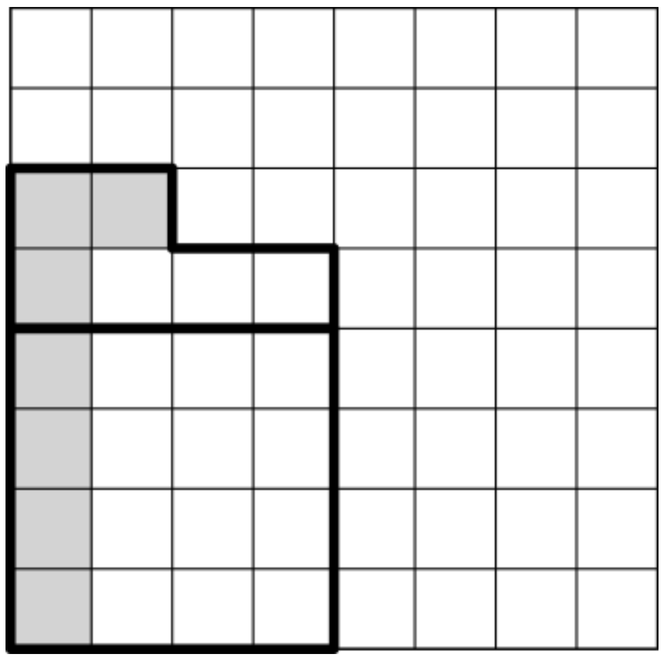
Merge of direct neighbours

Merging to maximum number

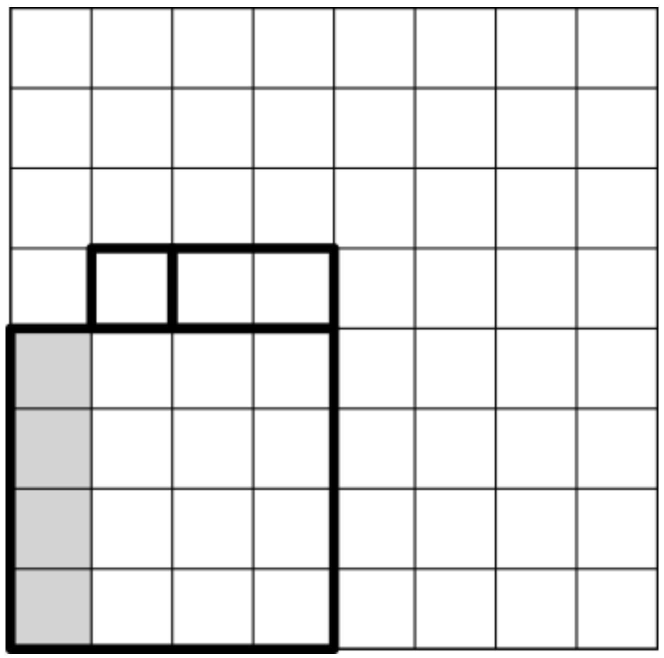
- Impose upper limit to the number of ranges
- Approximation gets slightly worse
- More false hits fetched during the filter step



Original 6 ranges

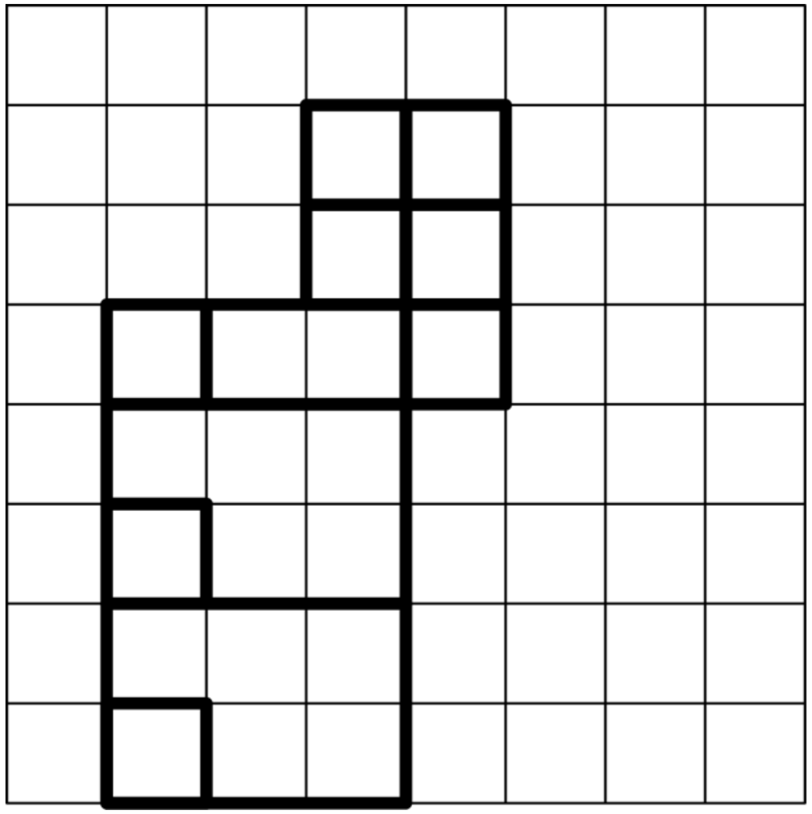


maximum 2 ranges

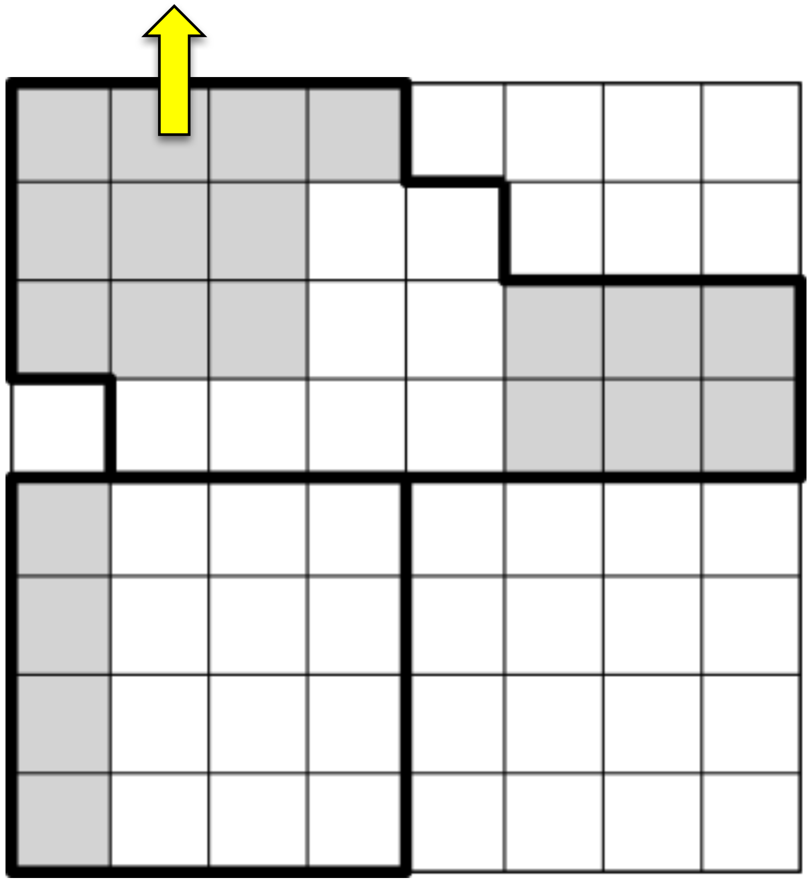


maximum 3 ranges

Merging to maximum number

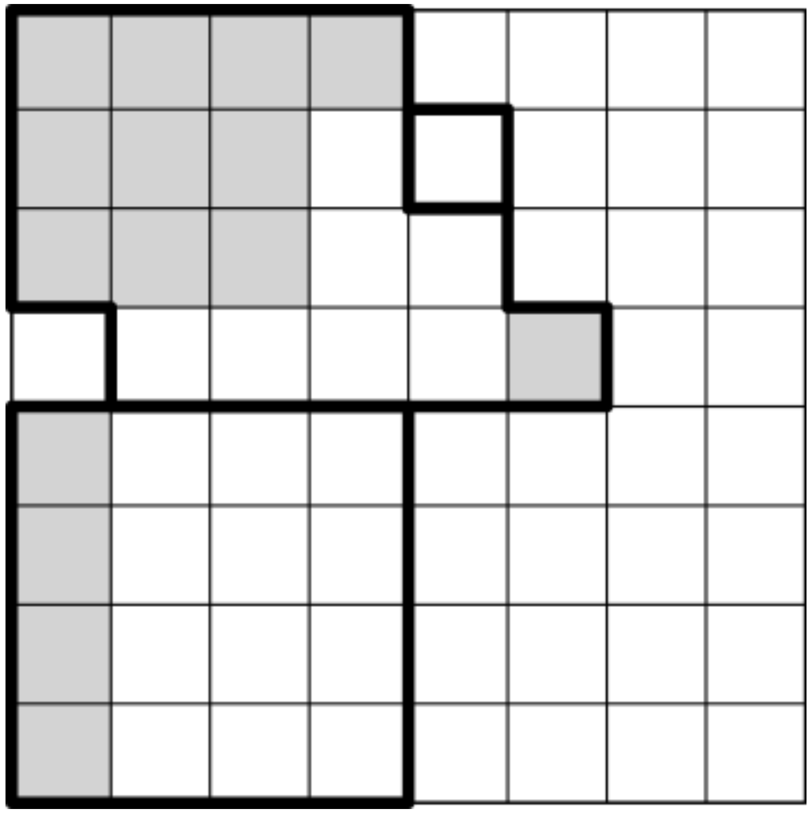


Original 11 ranges



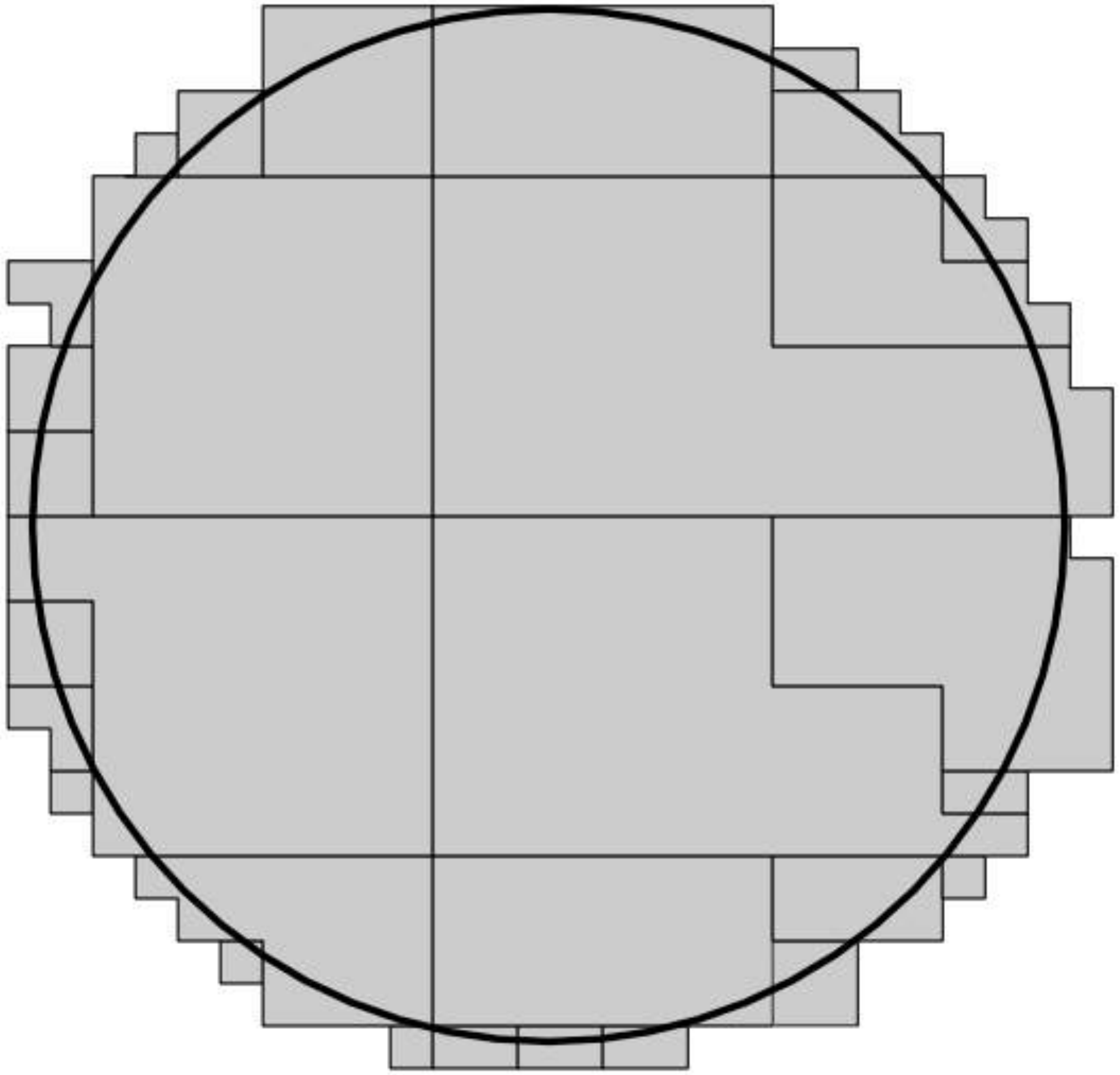
Additional space

maximum 2 ranges

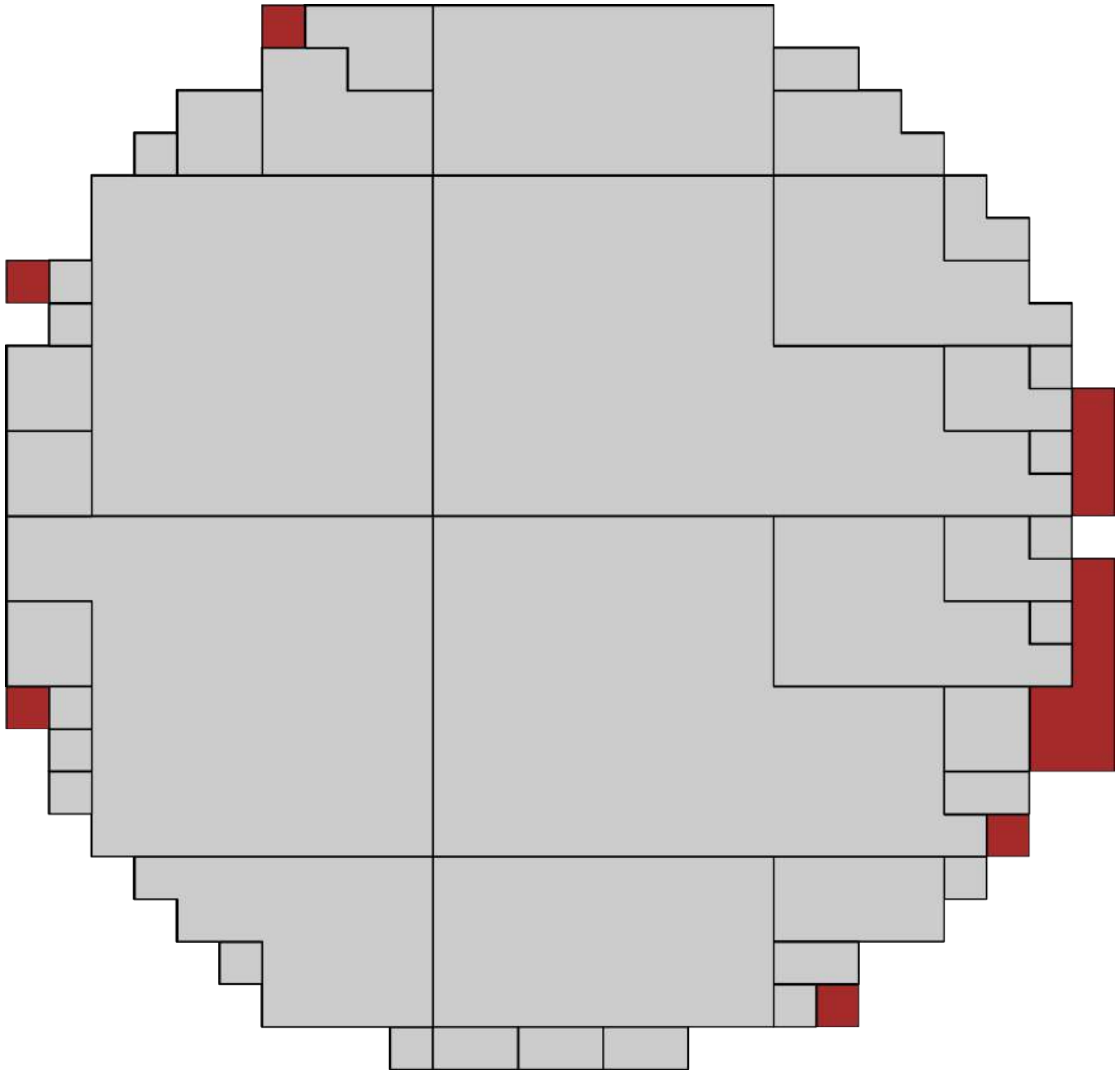


maximum 3 ranges

Merging to maximum number



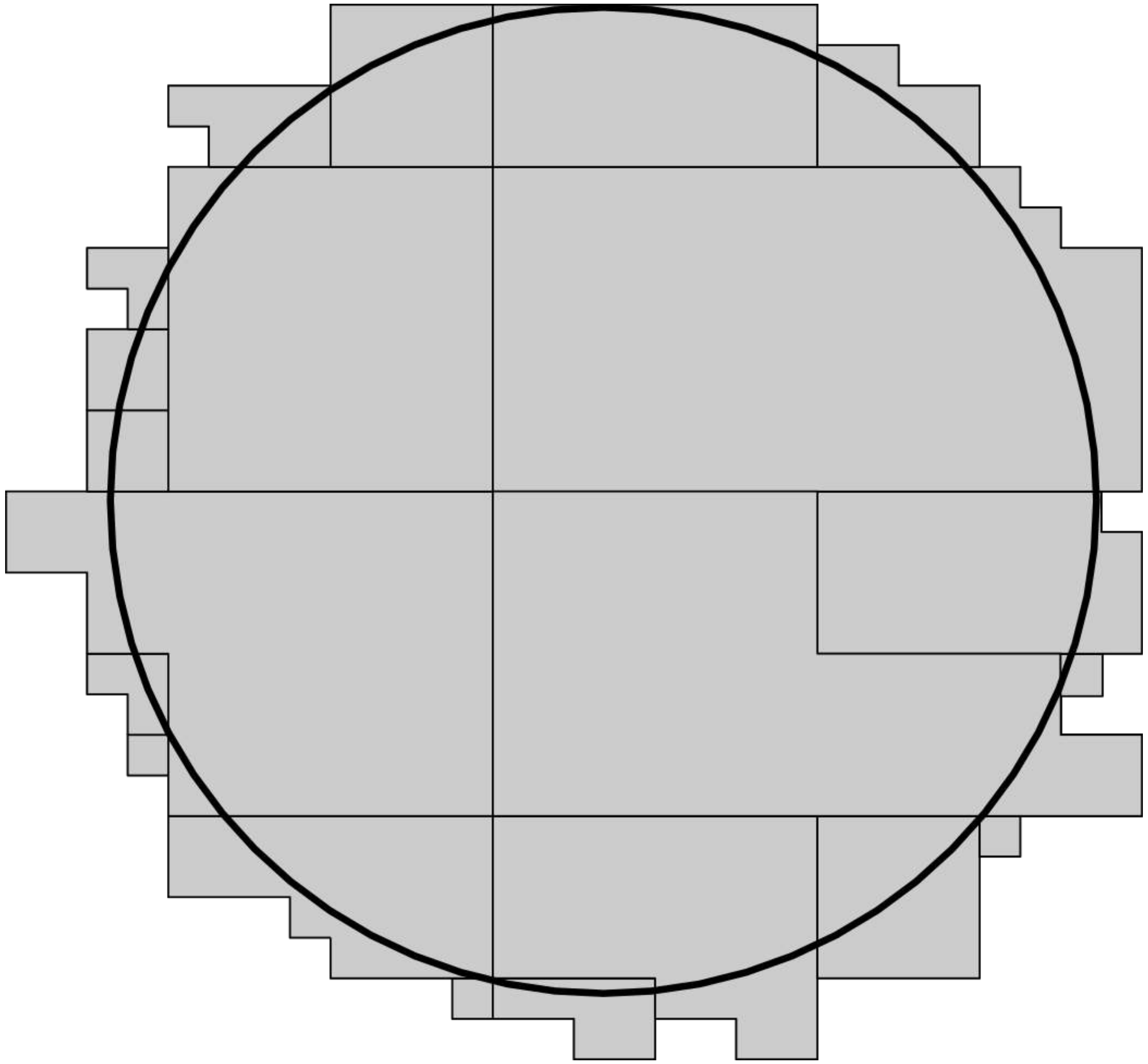
merged



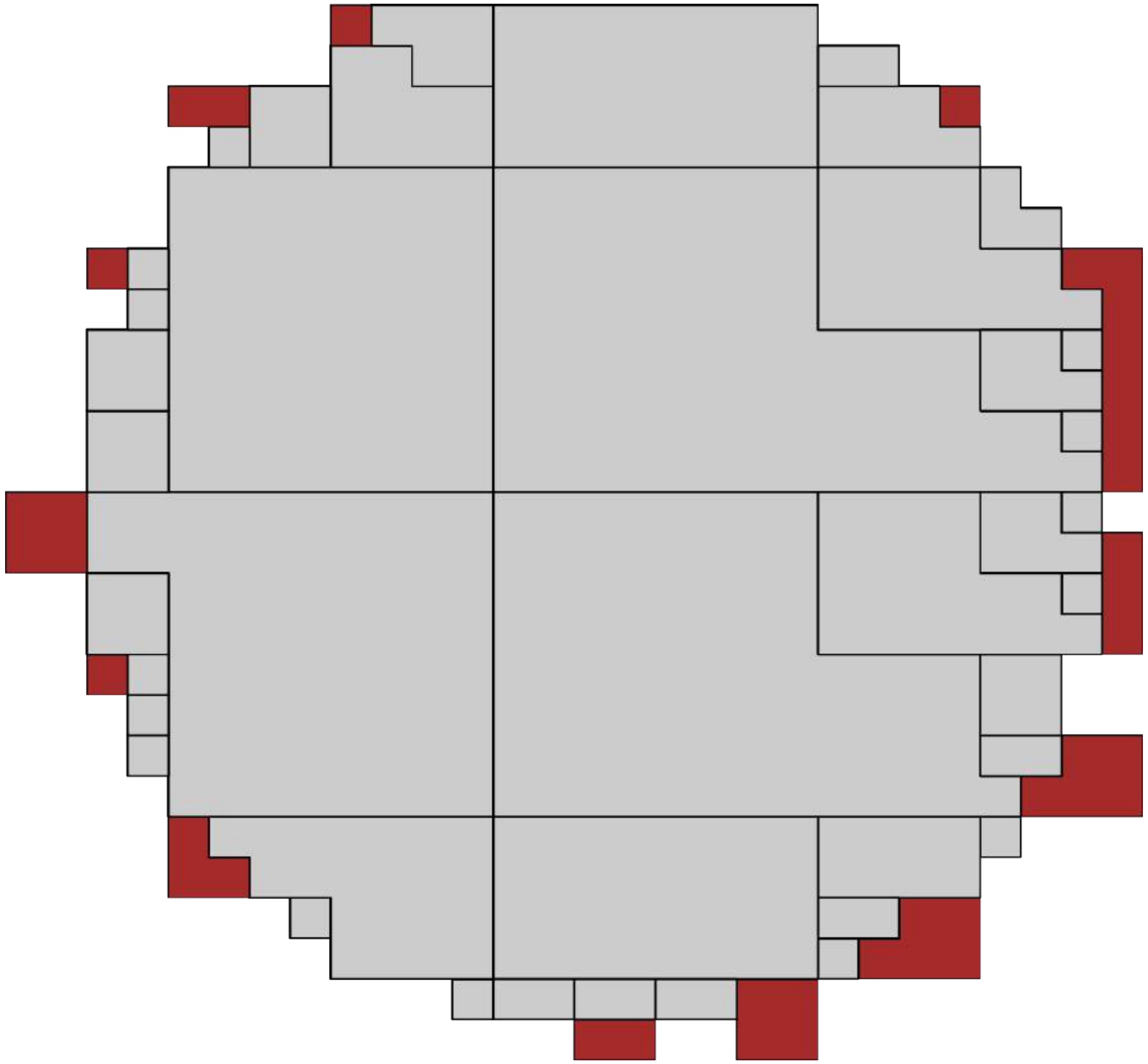
Expansion

Merge to max. number (30)

Merging to maximum number



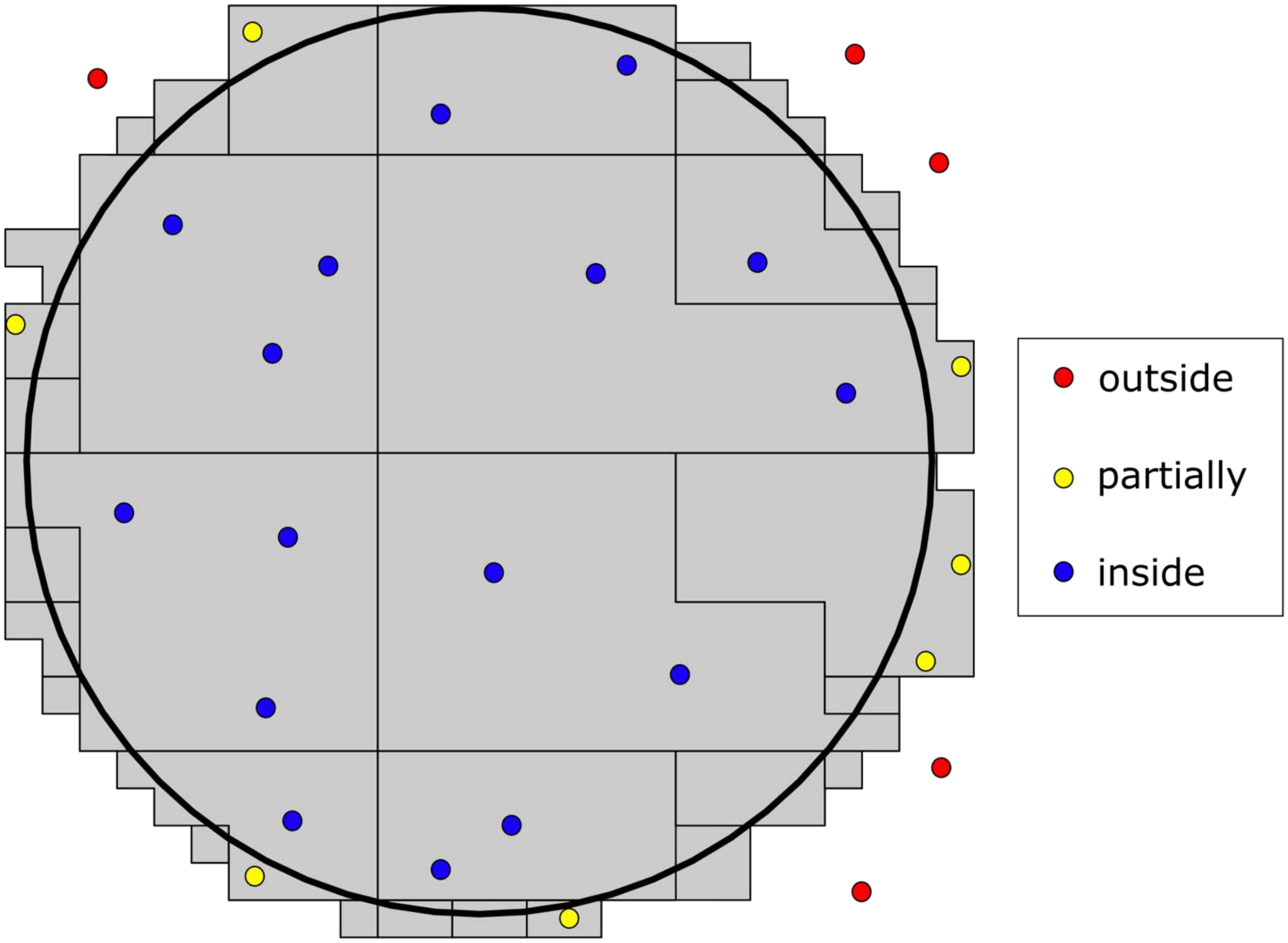
merged



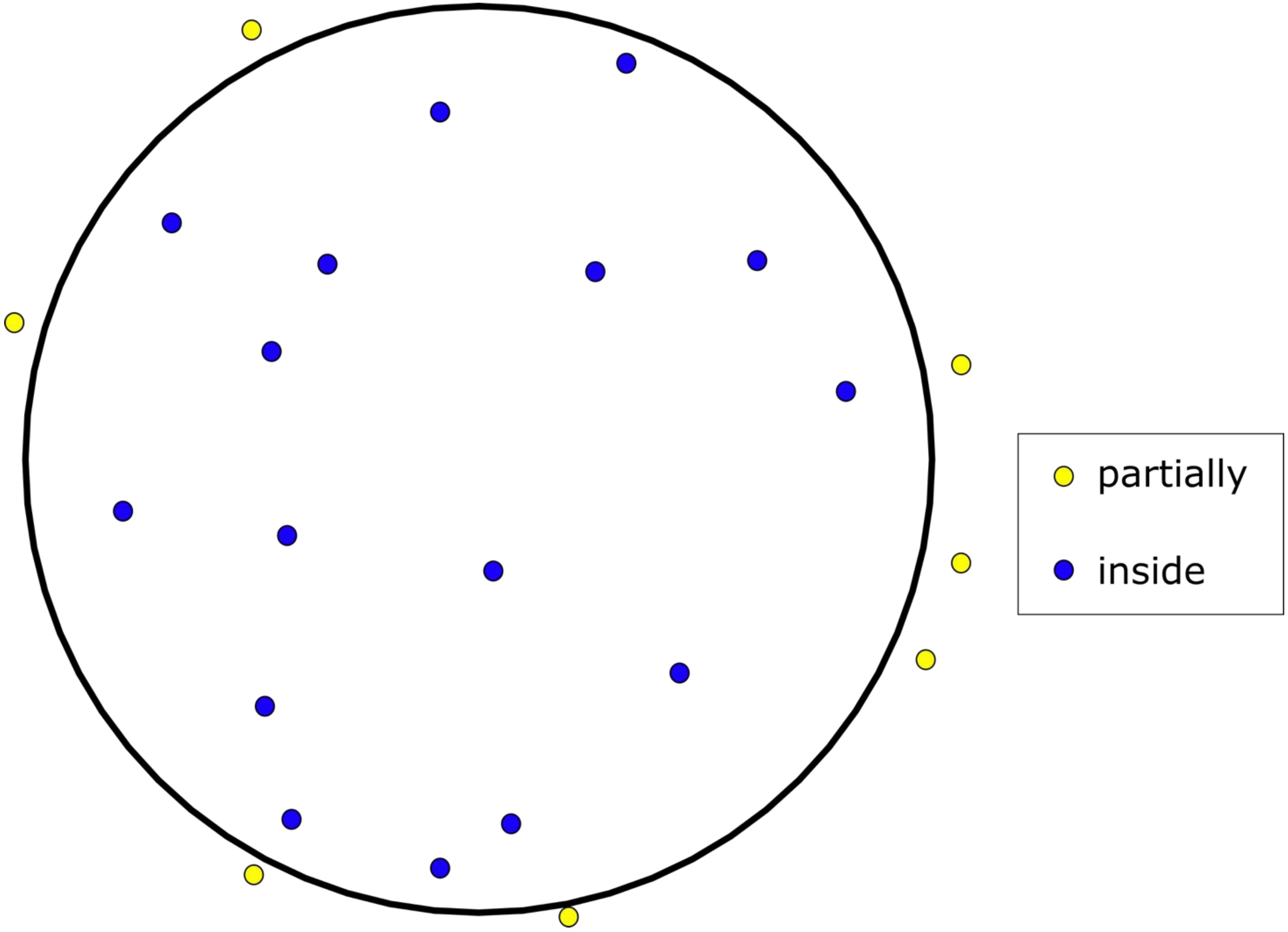
Expansion

Merge to max. number (20)

A SFC approach - Query



A SFC approach - Query



A SFC approach - Query

Multi-step query procedure

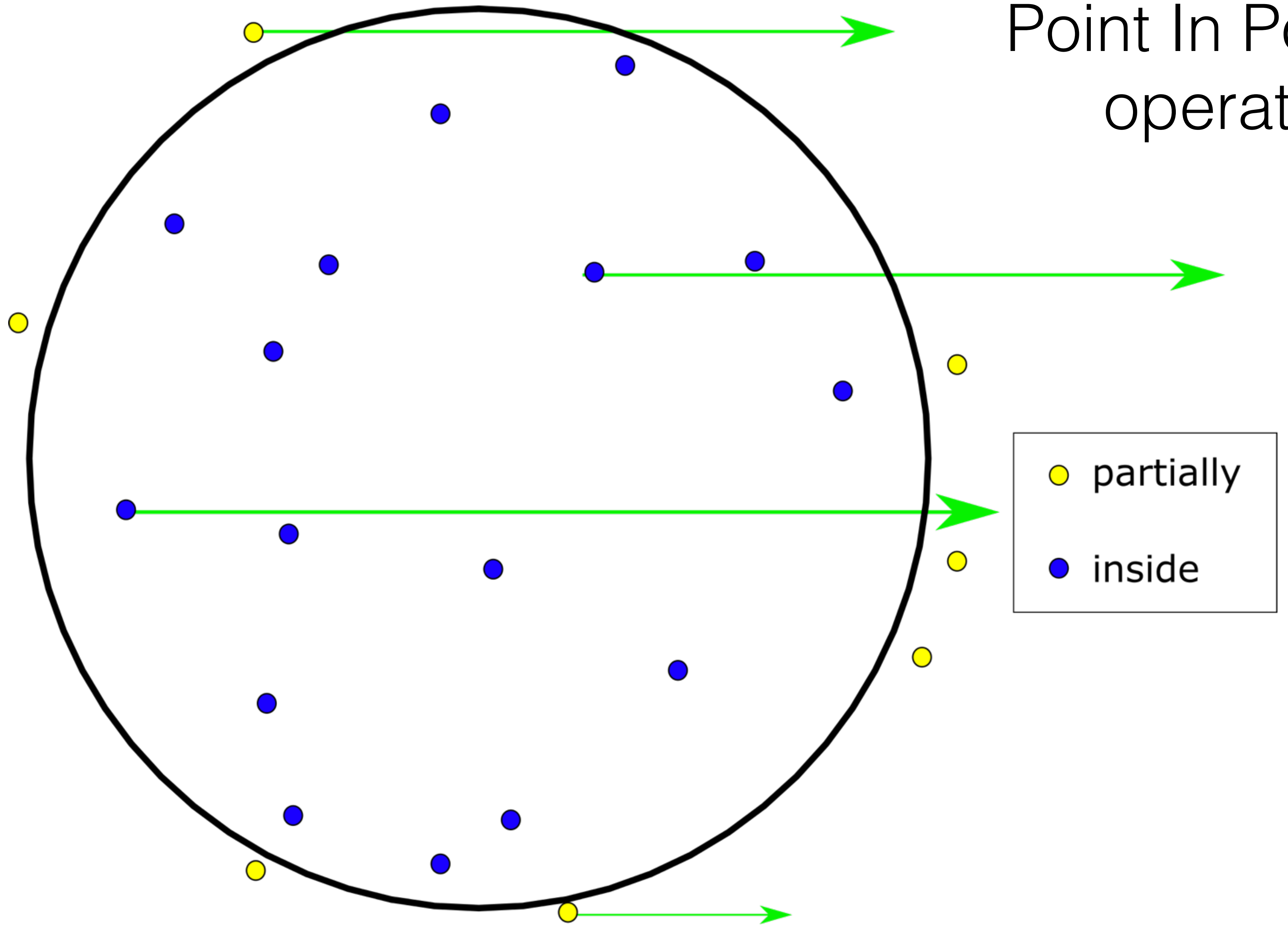
- Filter step: approximate query geometry using the 2^n-tree
- Fetch the approximated data and decode back to the original dimensions
- Refinement step: Detect the false hits using a Point in Polygon operation, or time and z refinement.

A SFC approach - Query

Multi-step query procedure

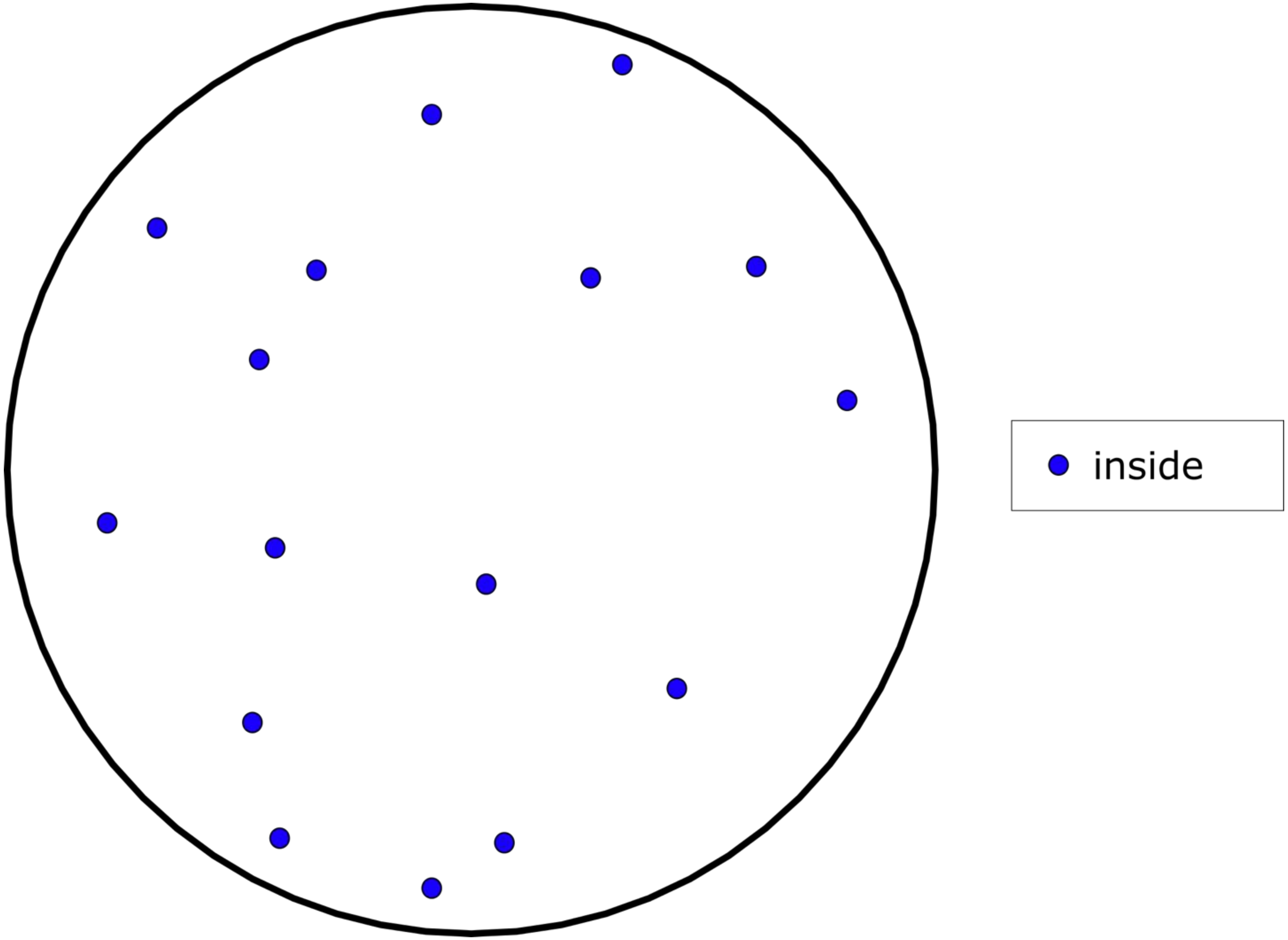
- Filter step: approximate query geometry using the 2^n-tree
- Fetch the approximated data and decode back to the original dimensions
- Refinement step: Detect the false hits using a Point in Polygon operation, or time and z refinement.

A SFC approach - Query



Point In Polygon operation

A SFC approach - Query



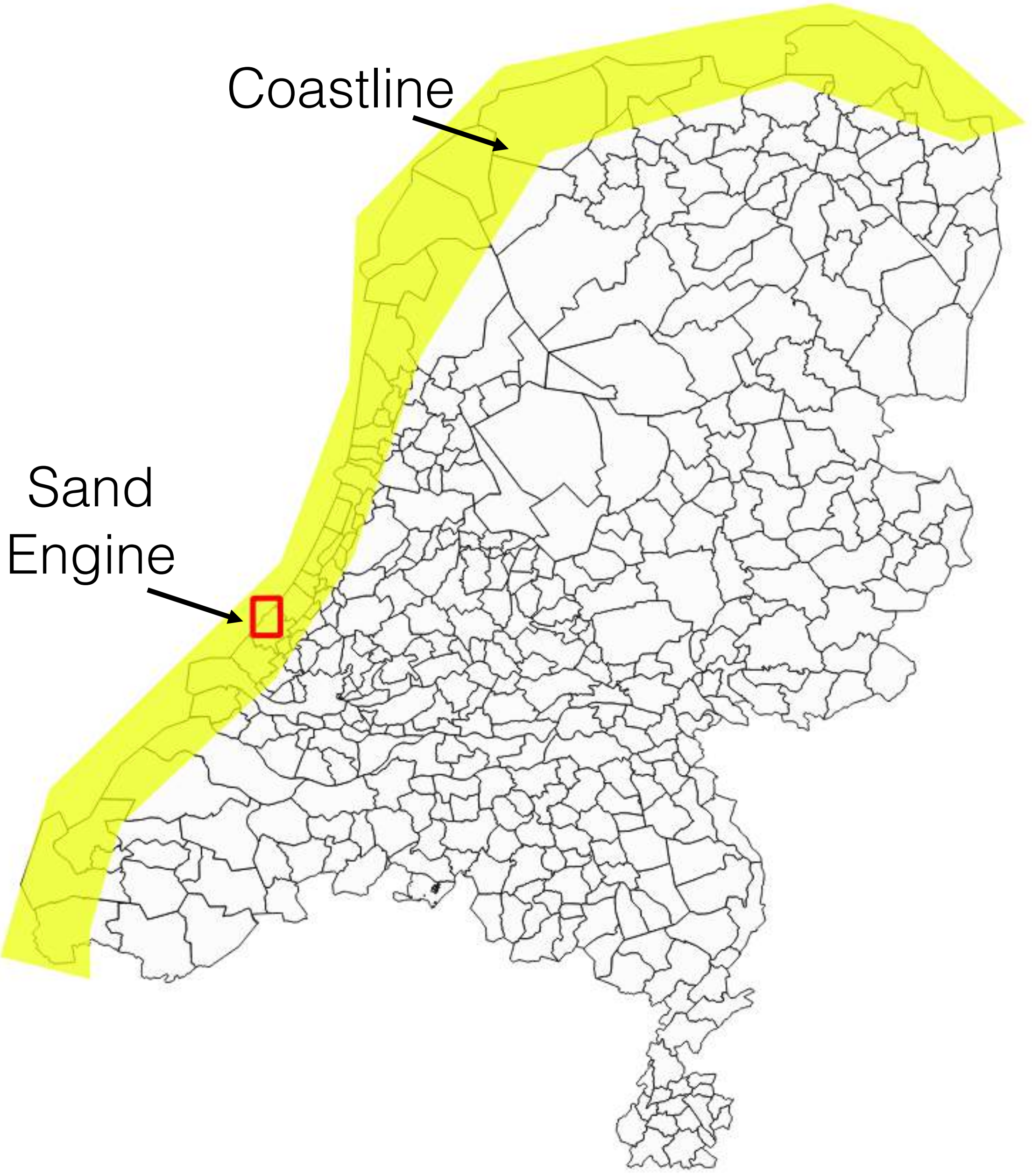
Results

Benchmark design

- Measure performance of storage space, loading time and query response time

- Datasets
  - Sand Engine
  - Coastline of the NL

Dataset	Time resolution	Spatial resolution	Points
Sand Engine	day	mm	100,000 pts/day
Coastline	year	cm	500 million pts/year



Benchmark design

- Benchmark stages

Table 1. The benchmark stages of the Sand Engine dataset

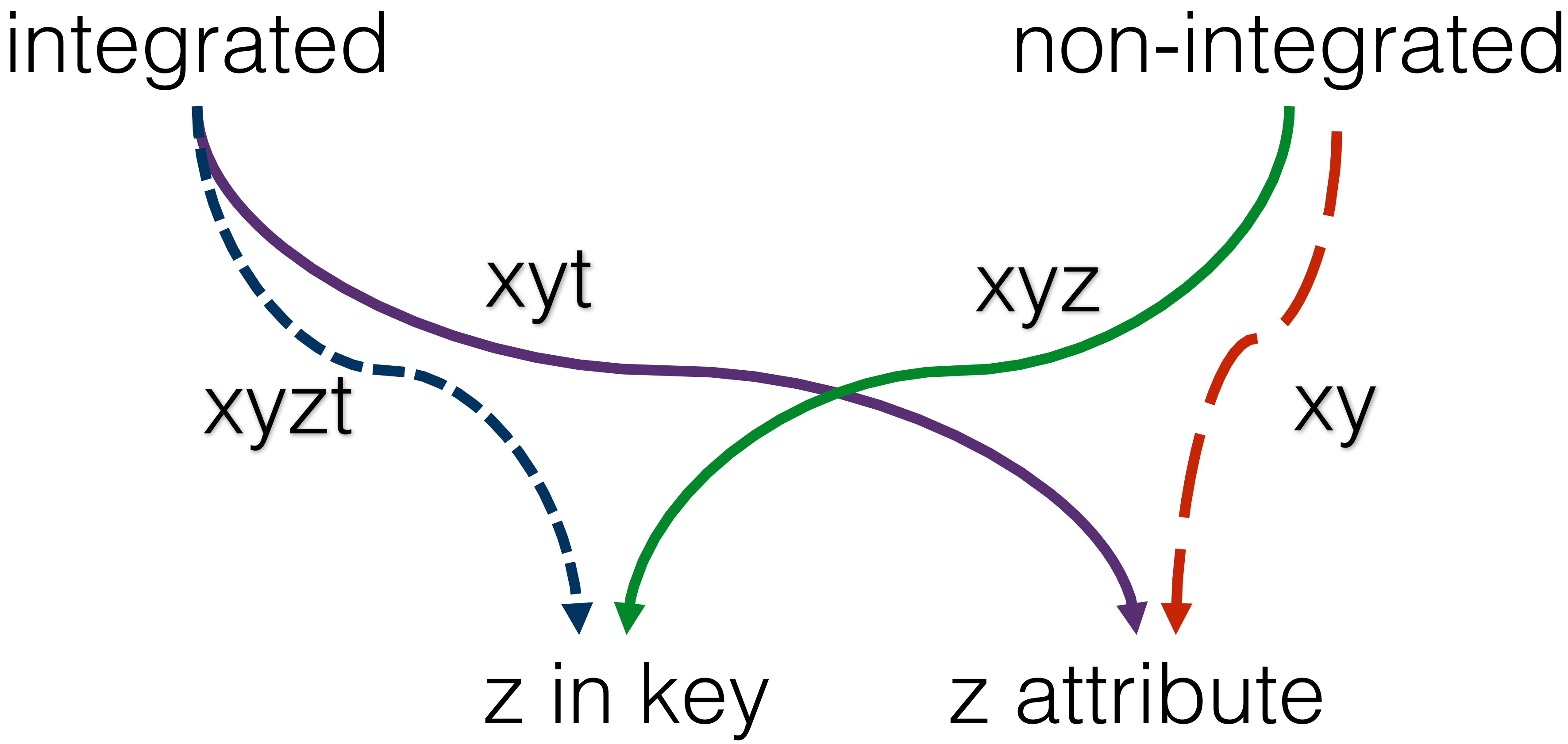
Benchmark	Points	Days	Size (MB)	Description
Small	18 M	230	347	2000 - 2002
Medium	44 M	554	836	2000 - 2006
Large	74 M	931	1414	2000 - 2015

Table 2. The benchmark stages of the Coastline dataset

Benchmark	Points	Years	Size (GB)	Description
Small	500 M	1	9.4	2012
Medium	995 M	2	18.7	2012 - 2013
Large	2020 M	4	37.9	2013 - 2015

Benchmark design

- 4 combinations



integrated

non-integrated

xyt

xyz

xyzt

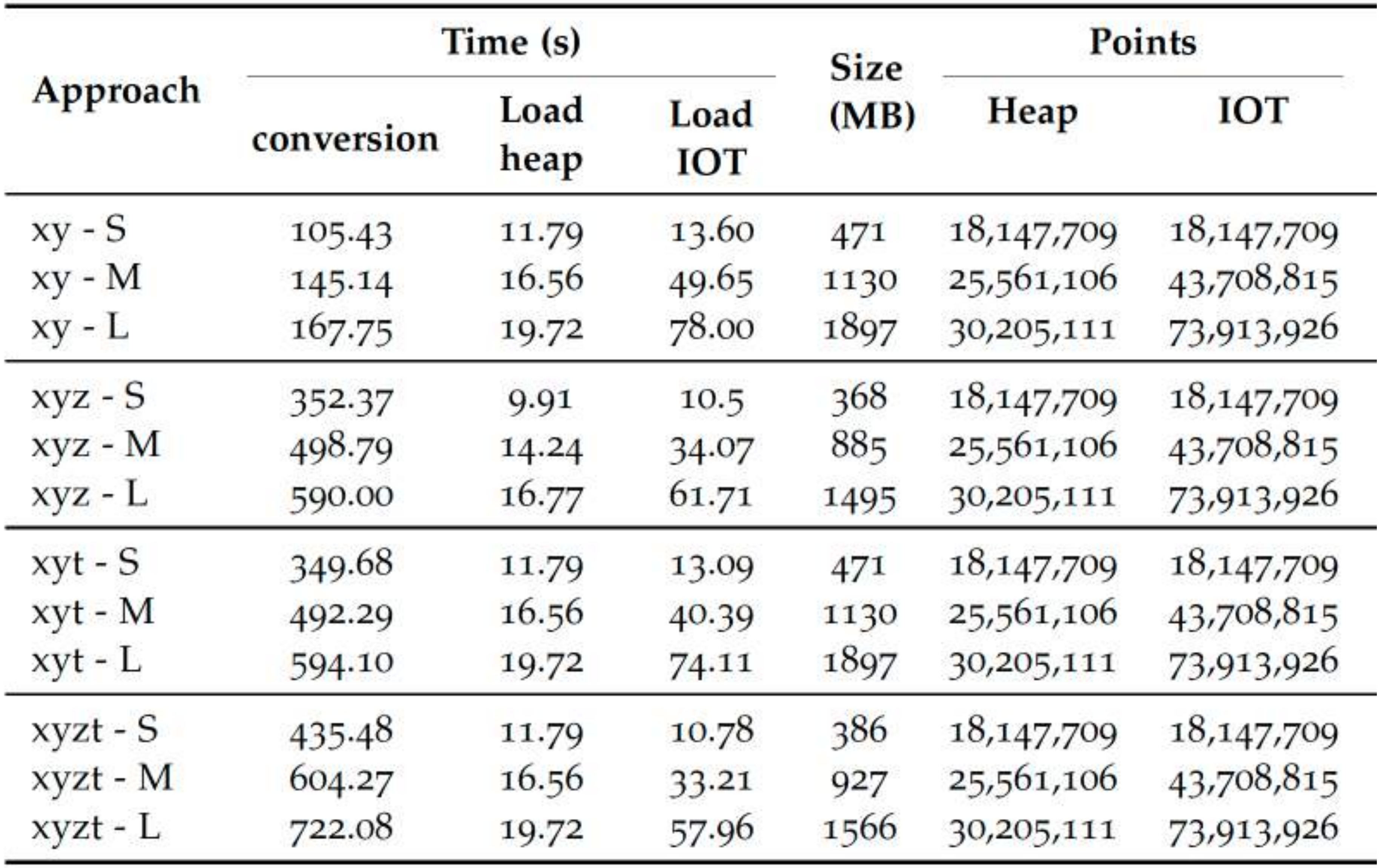
xy

z in key

z attribute

Results Loading

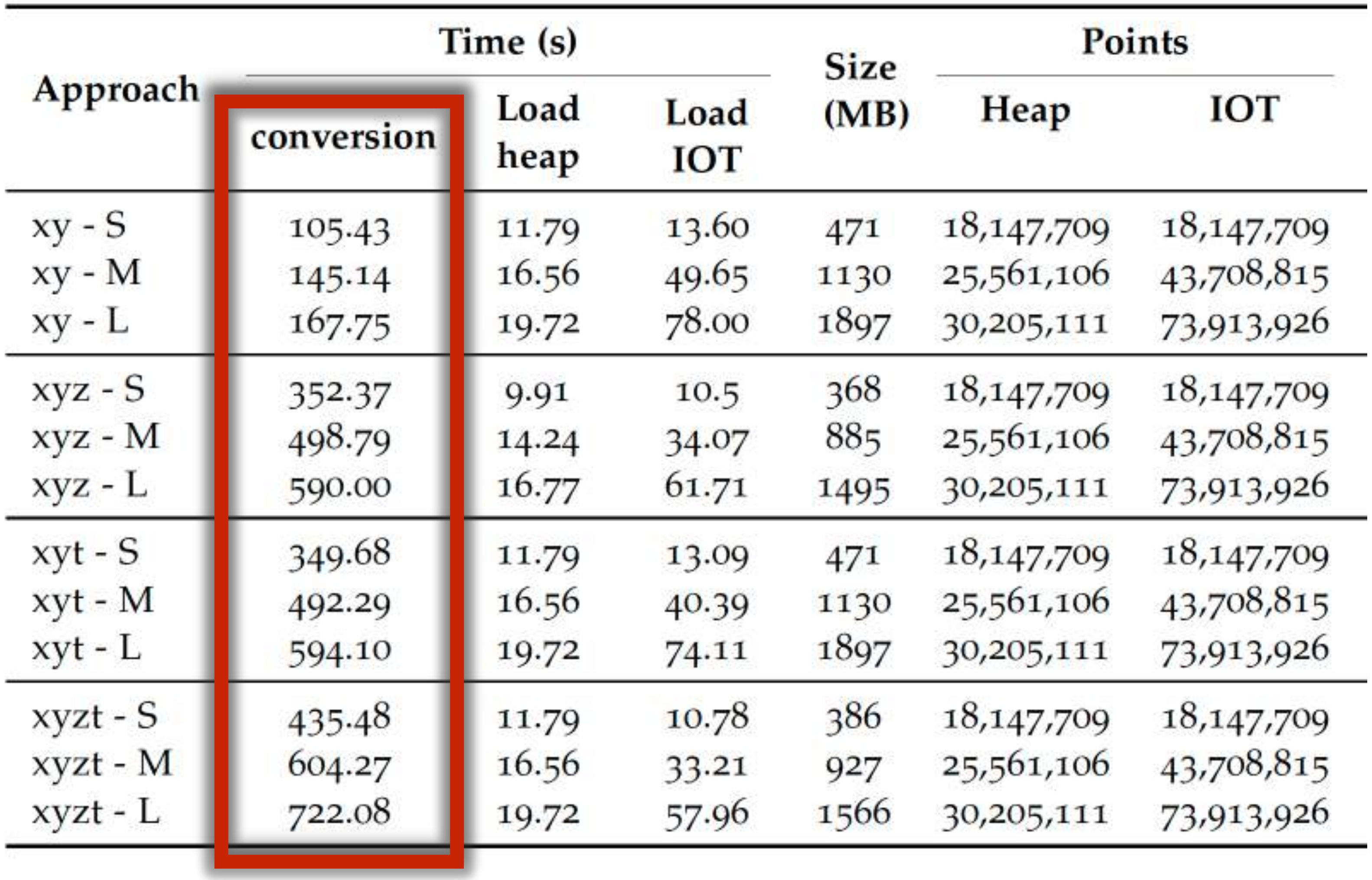
Sand Engine



Approach	Time (s)			Size (MB)	Points	
	conversion	Load heap	Load IOT		Heap	IOT
xy - S	105.43	11.79	13.60	471	18,147,709	18,147,709
xy - M	145.14	16.56	49.65	1130	25,561,106	43,708,815
xy - L	167.75	19.72	78.00	1897	30,205,111	73,913,926
xyz - S	352.37	9.91	10.5	368	18,147,709	18,147,709
xyz - M	498.79	14.24	34.07	885	25,561,106	43,708,815
xyz - L	590.00	16.77	61.71	1495	30,205,111	73,913,926
xyt - S	349.68	11.79	13.09	471	18,147,709	18,147,709
xyt - M	492.29	16.56	40.39	1130	25,561,106	43,708,815
xyt - L	594.10	19.72	74.11	1897	30,205,111	73,913,926
xyzt - S	435.48	11.79	10.78	386	18,147,709	18,147,709
xyzt - M	604.27	16.56	33.21	927	25,561,106	43,708,815
xyzt - L	722.08	19.72	57.96	1566	30,205,111	73,913,926

Results Loading

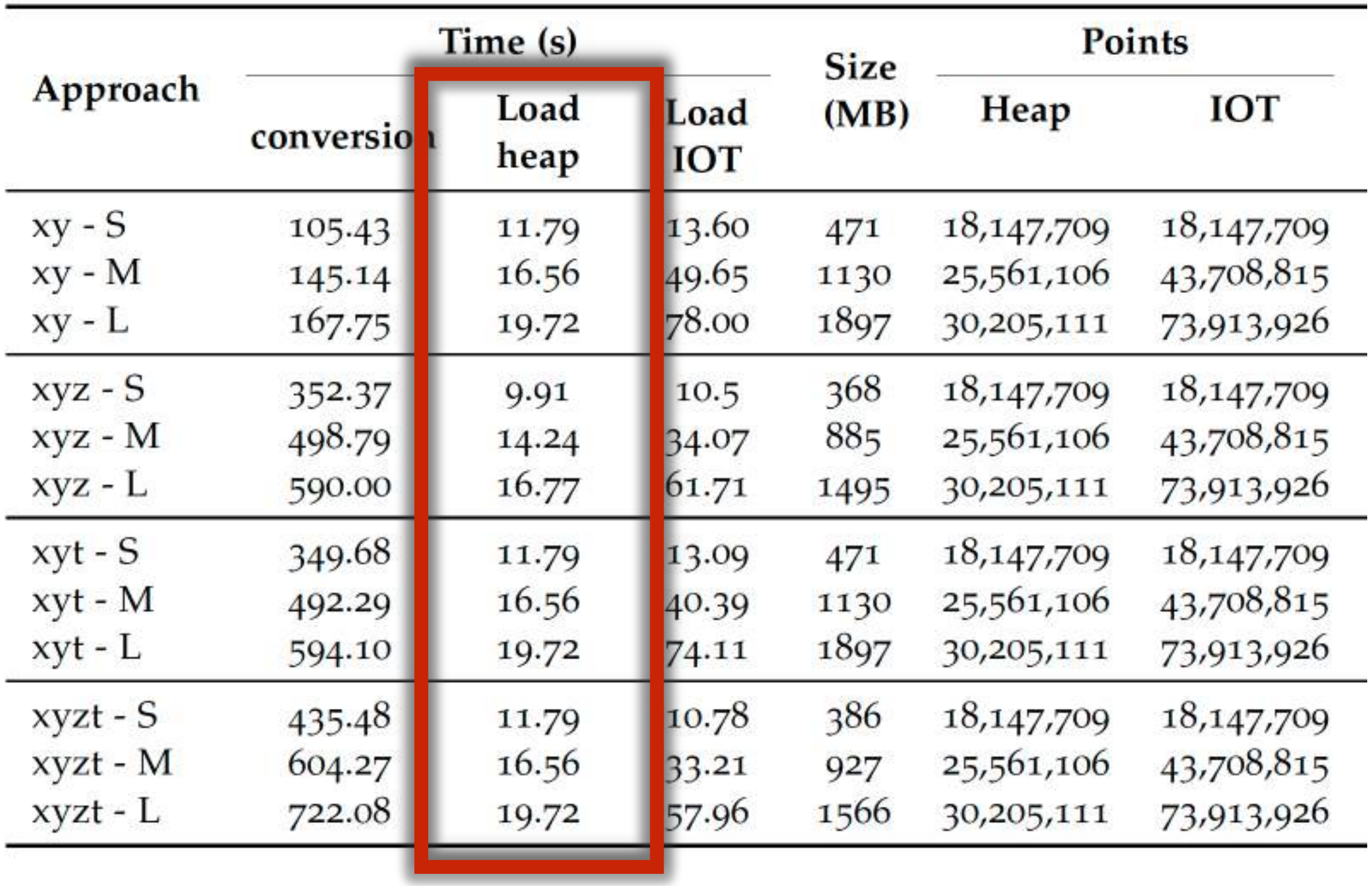
- The **SFC conversion** is the most expensive phase.
- Adding one more dimension in the key decreases the performance of the conversion.



Approach	Time (s)			Size (MB)	Points	
	conversion	Load heap	Load IOT		Heap	IOT
xy - S	105.43	11.79	13.60	471	18,147,709	18,147,709
xy - M	145.14	16.56	49.65	1130	25,561,106	43,708,815
xy - L	167.75	19.72	78.00	1897	30,205,111	73,913,926
xyz - S	352.37	9.91	10.5	368	18,147,709	18,147,709
xyz - M	498.79	14.24	34.07	885	25,561,106	43,708,815
xyz - L	590.00	16.77	61.71	1495	30,205,111	73,913,926
xyt - S	349.68	11.79	13.09	471	18,147,709	18,147,709
xyt - M	492.29	16.56	40.39	1130	25,561,106	43,708,815
xyt - L	594.10	19.72	74.11	1897	30,205,111	73,913,926
xyzt - S	435.48	11.79	10.78	386	18,147,709	18,147,709
xyzt - M	604.27	16.56	33.21	927	25,561,106	43,708,815
xyzt - L	722.08	19.72	57.96	1566	30,205,111	73,913,926

Results Loading

- **Loading into the heap** table is not affected by the benchmark case used.

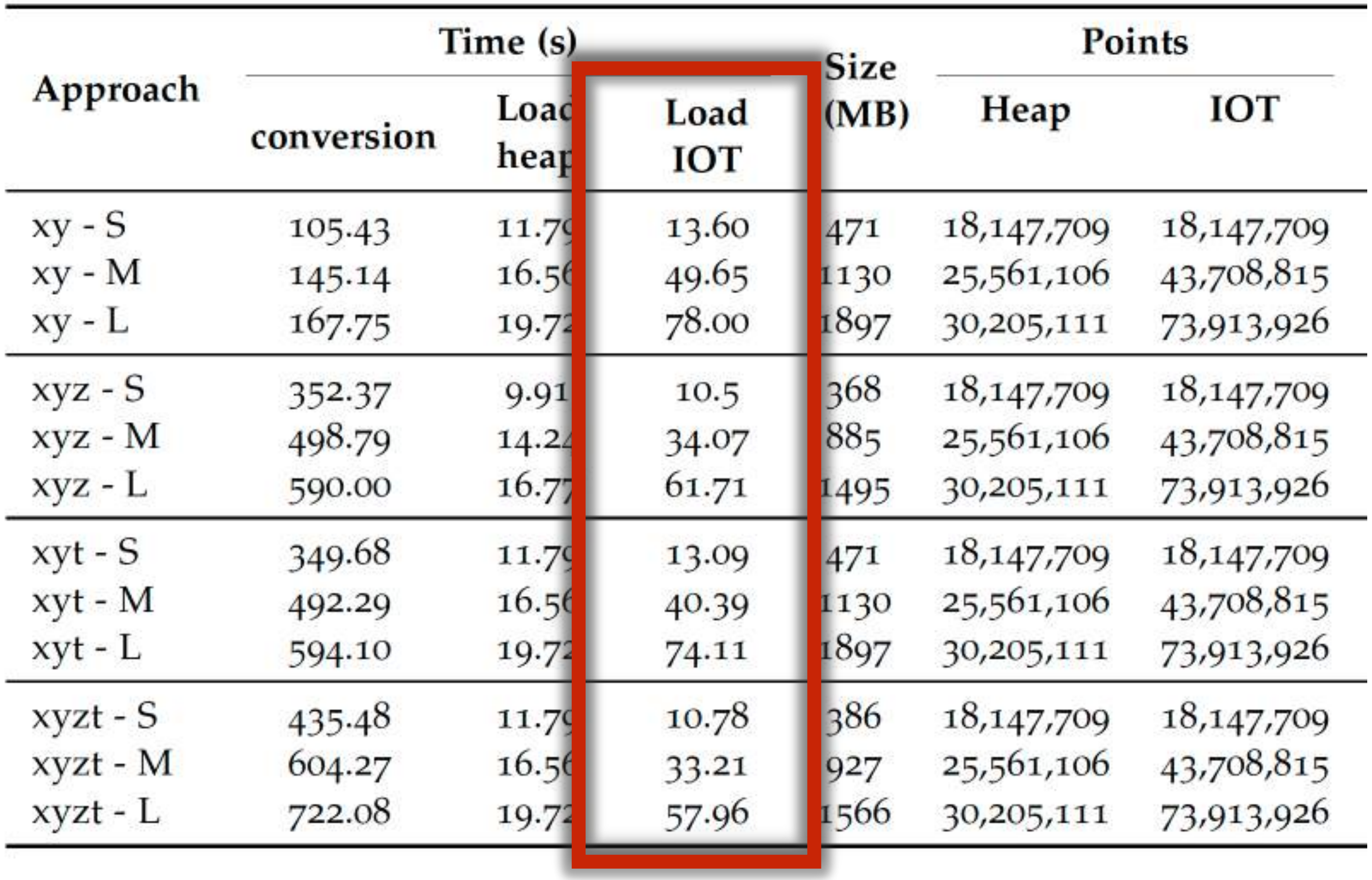


Approach	Time (s)			Size (MB)	Points	
	conversion	Load heap	Load IOT		Heap	IOT
xy - S	105.43	11.79	13.60	471	18,147,709	18,147,709
xy - M	145.14	16.56	49.65	1130	25,561,106	43,708,815
xy - L	167.75	19.72	78.00	1897	30,205,111	73,913,926
xyz - S	352.37	9.91	10.5	368	18,147,709	18,147,709
xyz - M	498.79	14.24	34.07	885	25,561,106	43,708,815
xyz - L	590.00	16.77	61.71	1495	30,205,111	73,913,926
xyt - S	349.68	11.79	13.09	471	18,147,709	18,147,709
xyt - M	492.29	16.56	40.39	1130	25,561,106	43,708,815
xyt - L	594.10	19.72	74.11	1897	30,205,111	73,913,926
xyzt - S	435.48	11.79	10.78	386	18,147,709	18,147,709
xyzt - M	604.27	16.56	33.21	927	25,561,106	43,708,815
xyzt - L	722.08	19.72	57.96	1566	30,205,111	73,913,926

Results Loading

- The **creation of the IOT** is dependent only on the treatment of z used.

- The IOT is created faster when treating z as part of the the key.

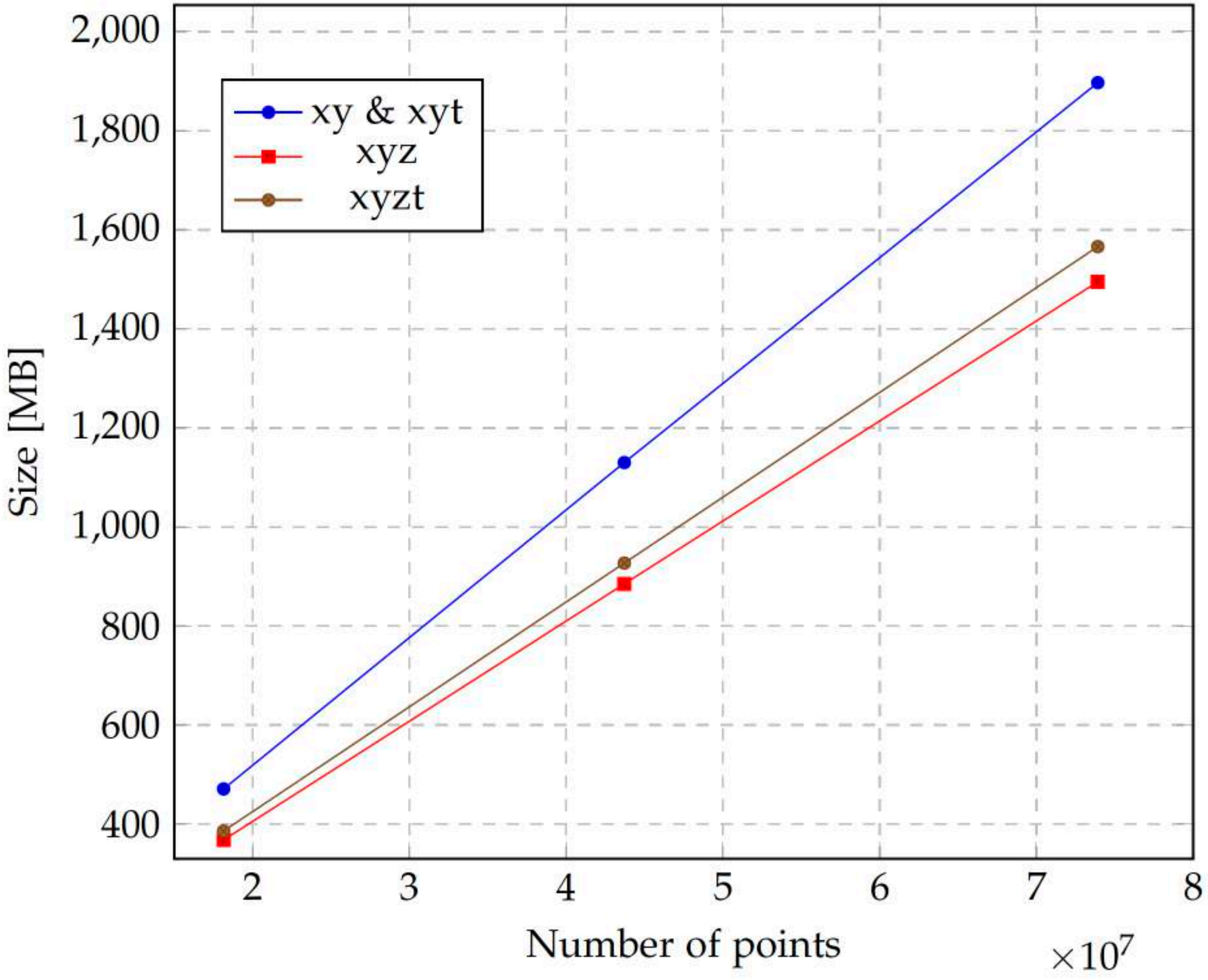


Approach	Time (s)			Size (MB)	Points	
	conversion	Load heap	Load IOT		Heap	IOT
xy - S	105.43	11.79	13.60	471	18,147,709	18,147,709
xy - M	145.14	16.56	49.65	1130	25,561,106	43,708,815
xy - L	167.75	19.72	78.00	1897	30,205,111	73,913,926
xyz - S	352.37	9.91	10.5	368	18,147,709	18,147,709
xyz - M	498.79	14.24	34.07	885	25,561,106	43,708,815
xyz - L	590.00	16.77	61.71	1495	30,205,111	73,913,926
xyt - S	349.68	11.79	13.09	471	18,147,709	18,147,709
xyt - M	492.29	16.56	40.39	1130	25,561,106	43,708,815
xyt - L	594.10	19.72	74.11	1897	30,205,111	73,913,926
xyzt - S	435.48	11.79	10.78	386	18,147,709	18,147,709
xyzt - M	604.27	16.56	33.21	927	25,561,106	43,708,815
xyzt - L	722.08	19.72	57.96	1566	30,205,111	73,913,926

Results Loading

- The **storage requirements** are affected only by the treatment of z.

- Treating z as an attribute increases the storage.

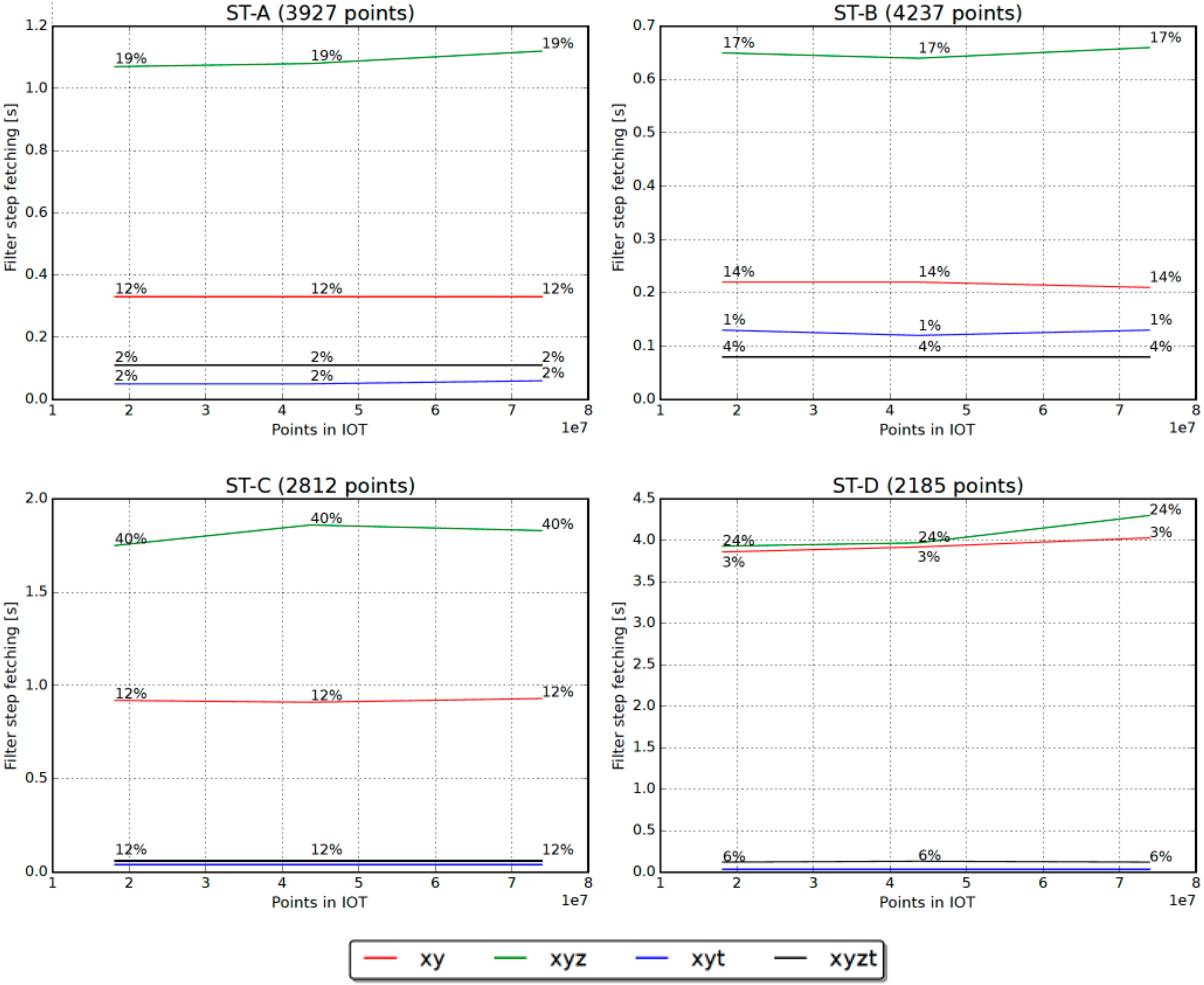


Results Querying

- Test the scalability of the queries
- Focus on the fetching time of the filter step that directly uses the structure. The rest of the steps can be improved in performance and are not analysed.

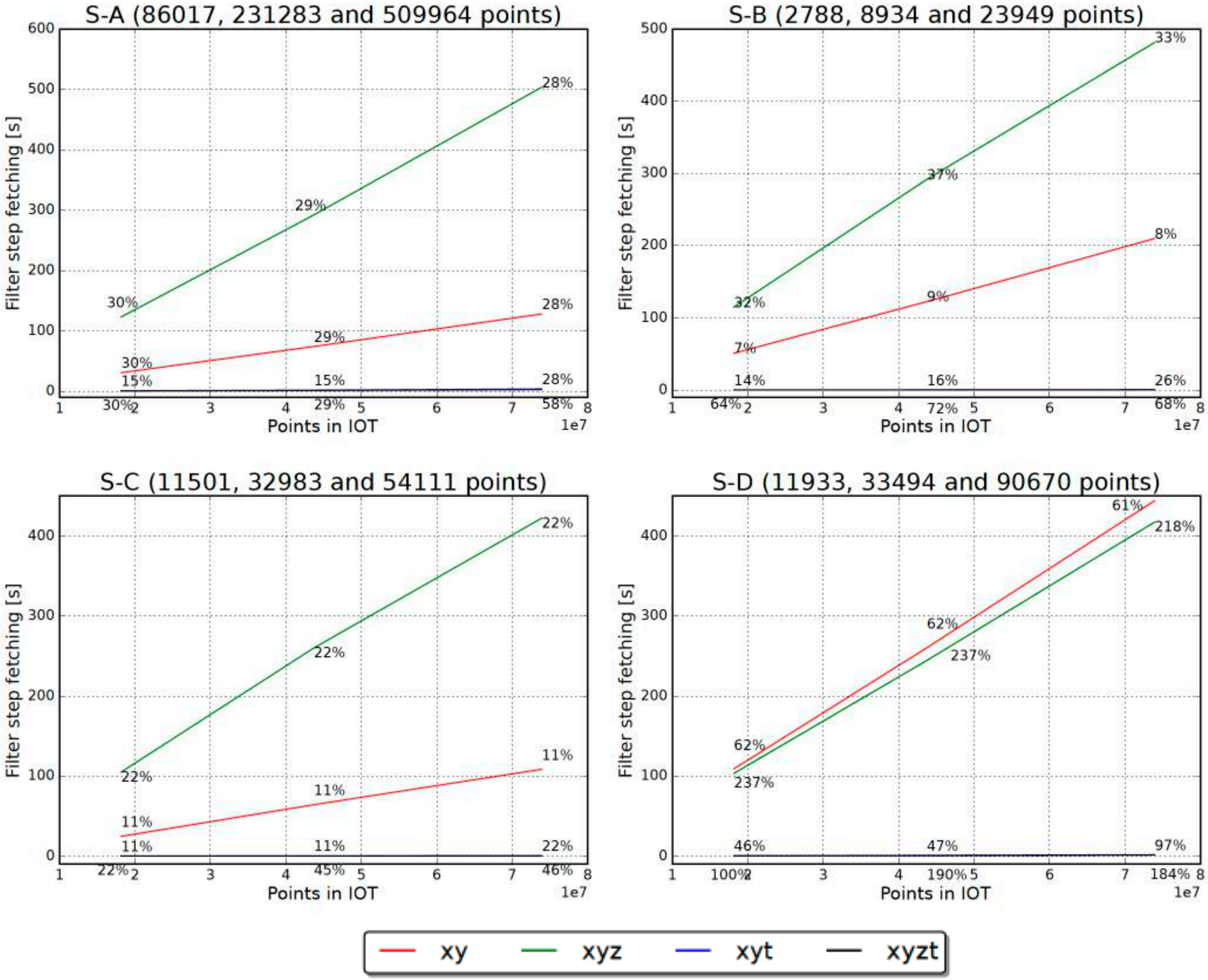
Results Querying

Space – time queries



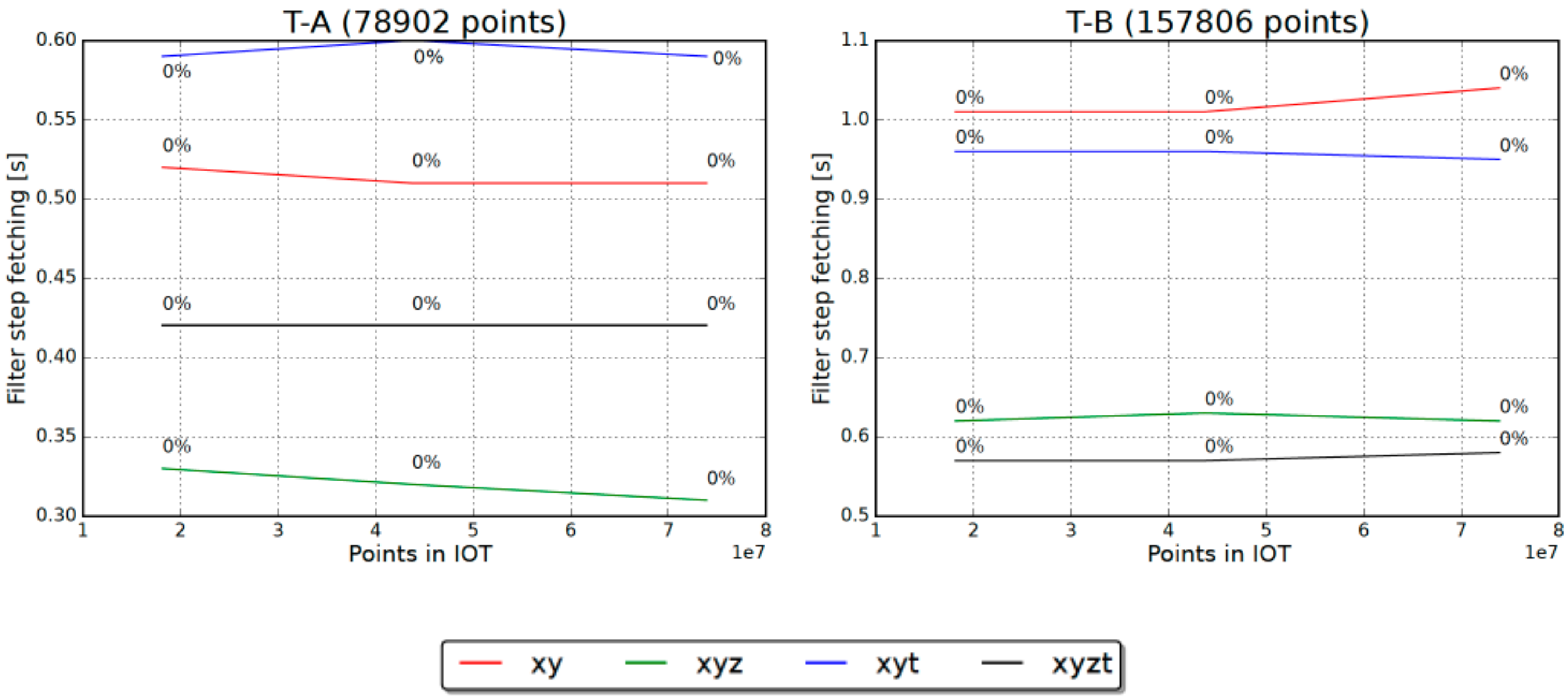
Results Querying

Space only queries



Results Querying

Time only
queries



Conclusion & Future work

Conclusions

- Designed and executed a benchmark for dynamic point clouds

- Two integrations of space and time and, two treatments of z

- Integrated approach presented better query response times, compared to non-integrated for the specific use case (both treatments of z possible)

- Key aspect of the implementation: Index Organised Table

Future work

- Native database functionality (encoding, decoding, range generation)
- Investigate a different SFC
- Investigate parallel processing
- Up-scaled benchmark of trillion points
- Investigate the generation of blocks: compression

Thank you for your attention!

References

- Gaede, V. and Gunther,O. (1998). Multidimensional access methods. ACM Computing Surveys (CSUR), 30(2):170–231.

- van Oosterom, P., Martinez-Rubi, O., Ivanova, M., Horhammer, M., Geringer, D., Ravada, S., Tijssen, T., Kodde, M., and Gonc¸alves, R. (2015). Massive point cloud data management: Design, implementation and execution of a point cloud benchmark. Computers & Graphics, 49 :92 –125 .