Russian Supercomputing Days

NATIONAL RESEARCH UNIVERSITY

RUSSIAN FOUNDATION FOR BASIC RESEARCH

* This work was partially supported by
Russian Foundation for Basic Research (grant #16-37-00416).

Retrospective Satellite Data in the Cloud: An Array DBMS Approach*

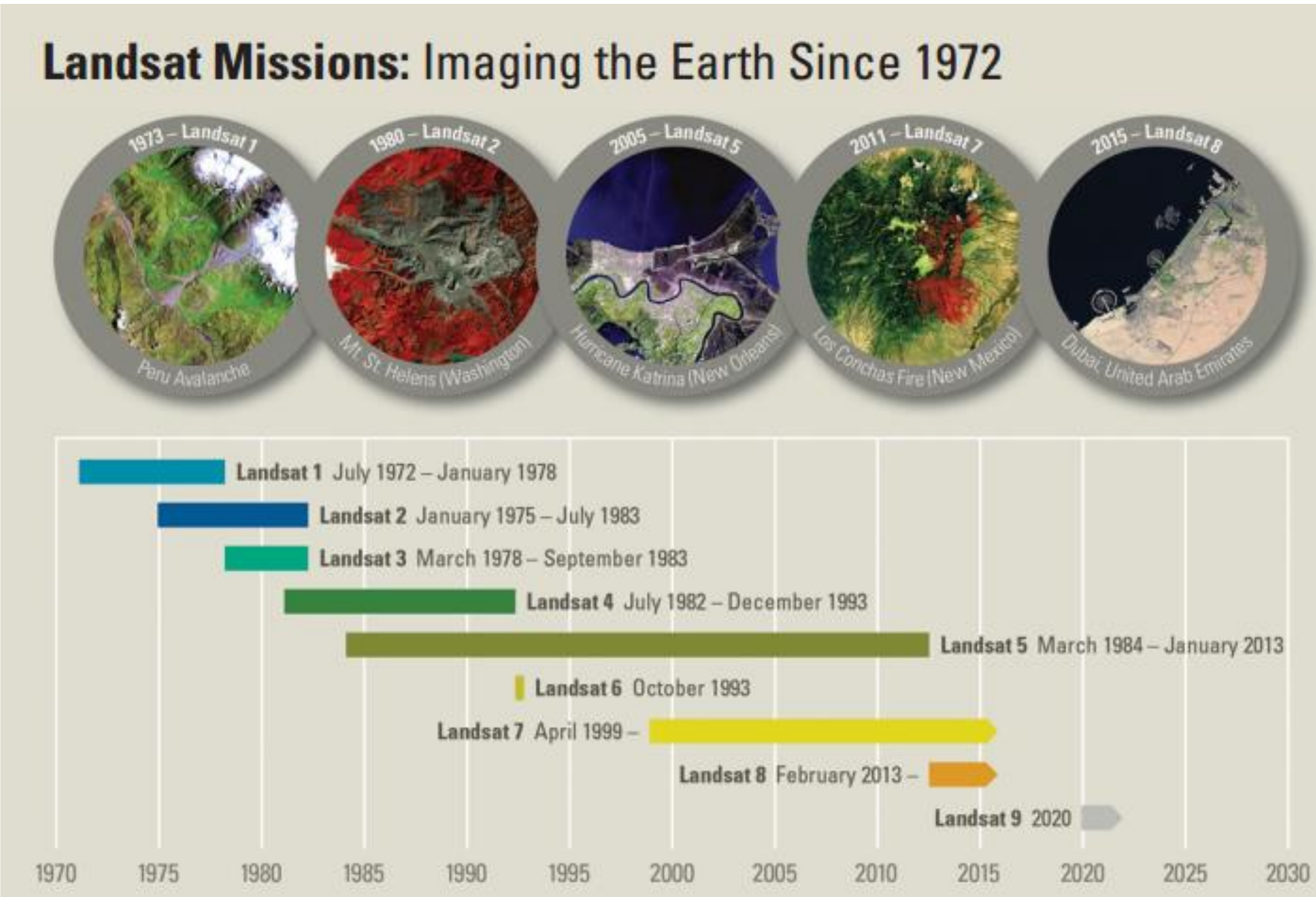
Ramon Antonio Rodriges Zalipynis, Anton Bryukhov and Evgeniy Pozdeev

rodriges@wikience.org

Big Data: Satellite Imagery

Landsat Program is the longest continuous space-based record of Earth's land in existence running from 1972 onwards.

https://landsat.usgs.gov/



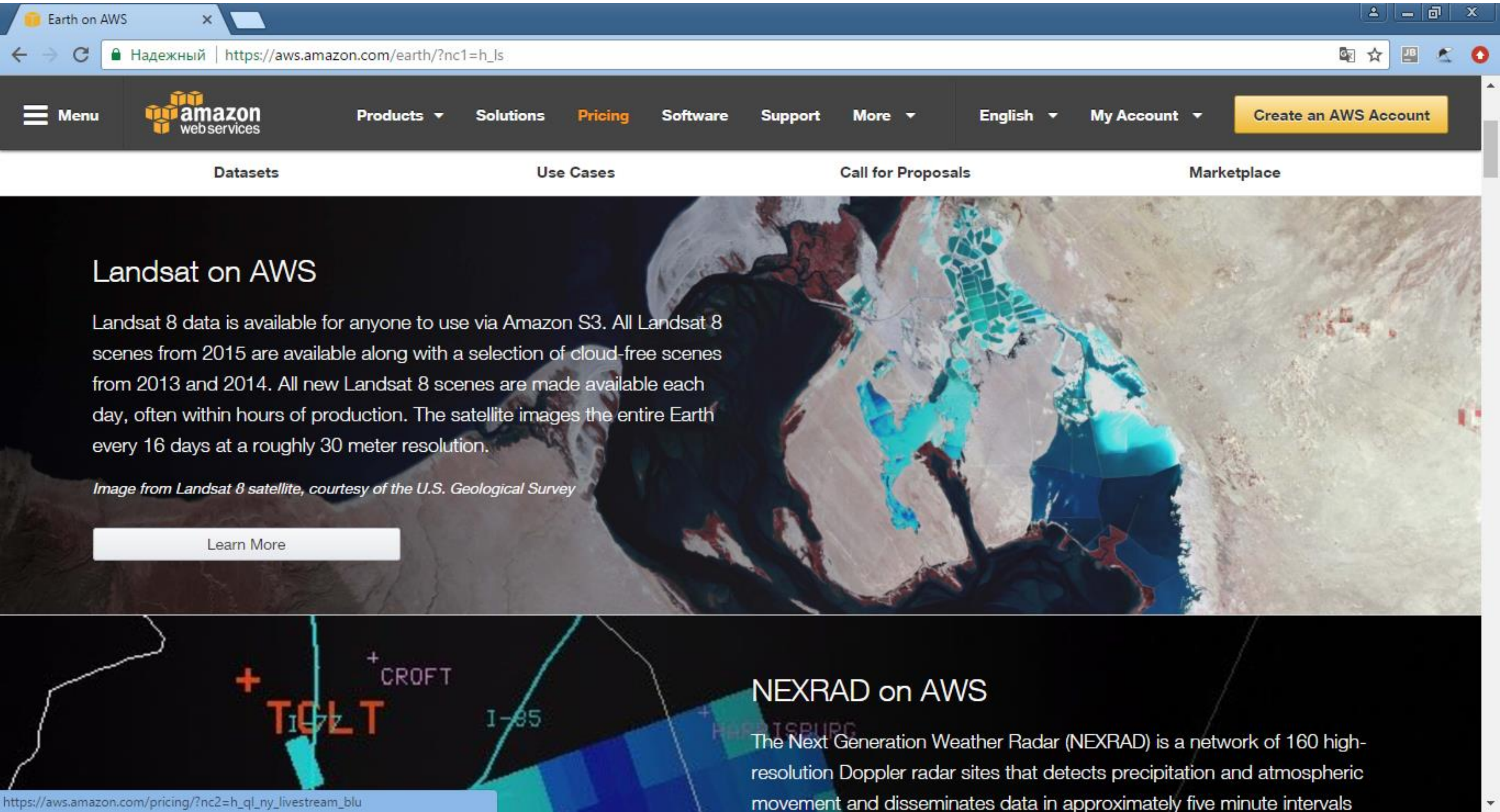
To date:
$> 6.8 \times 10$ M scenes mostly in GeoTIFF format

One scene is $\approx$1 GB

Landsat @ Amazon and Google

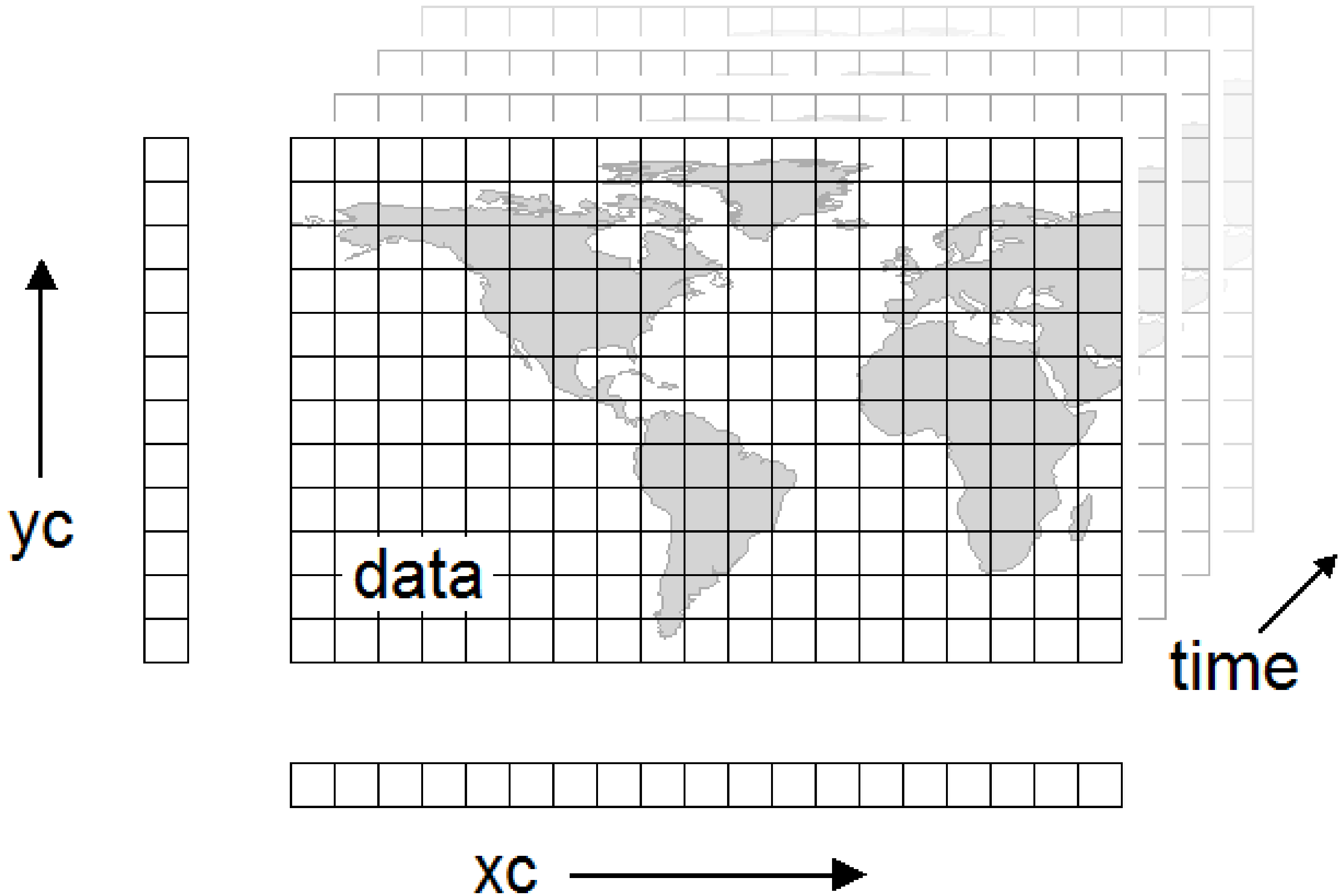
Satellite sector is data-rich, practically important and commercially attractive.

https://aws.amazon.com/earth



Array or Raster DBMS

Manages N-dimensional arrays



http://www.narccap.ucar.edu/users/user-meeting-08/handout/netcdf-diagram.png

State-of-the-art

	in situ	delegate	distributed	1st release
ChronosServer	YES	Direct	YES	?
SciDB	NO	Streaming	YES	~ 2008**
Oracle Spatial*	NO	NO	YES	< 2005
ArcGIS IS*	YES	NO	NO****	> 2000
RasDaMan	YES/NO***	NO	YES/NO***	~ 1999
MonetDB SciQL	NO	NO	NO	?
Intel TileDB	NO	NO	NO	04 04 2016

SciDB is the only free and distributed array DBMS available for comparison

* Commercial

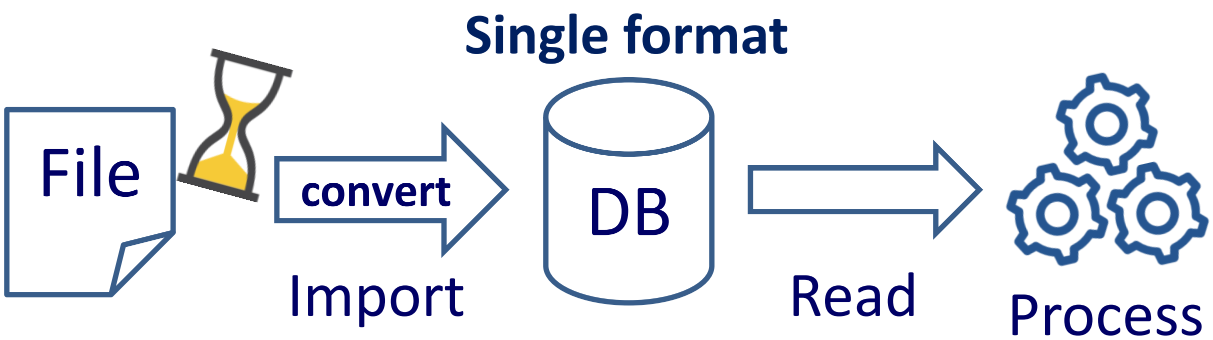
** Now (in 2017, since 9 years) it still has very limited set of operations

*** YES in *payed*, enterprise version

**** Data are the same on each server or retrieved from centralized storage

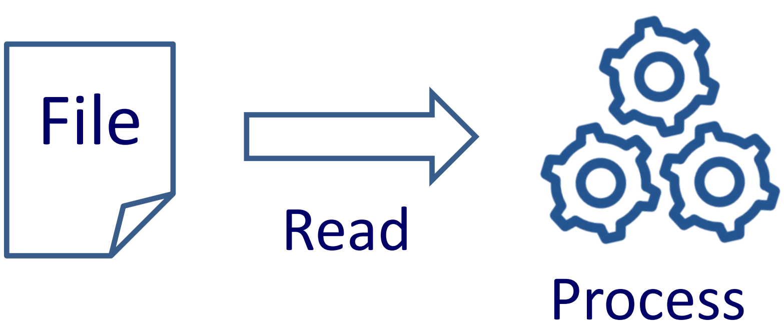
2 key approaches

Import, then process



Time-consuming, error-prone

In-situ processing



– Diverse file formats

No import

Reason for in-situ approach: powerful raster file formats

File-based raster data storage resulted in a broad set of highly optimized raster file formats

GeoTIFF represents an effort by over 160 different companies and organizations to establish interchange format for georeferenced raster imagery http://trac.osgeo.org/geotiff/

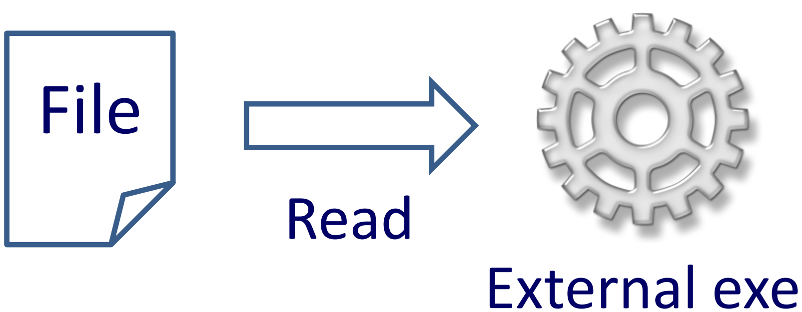
Delegation to command line tools (ChronosServer)

Decades of development resulted in many elaborate tools for processing these files **optimized mostly for a single machine.**

GDAL (Geospatial Data Abstraction Library)

≈10^6 lines of code, 100s of contributors
https://scan.coverity.com/projects/gdal

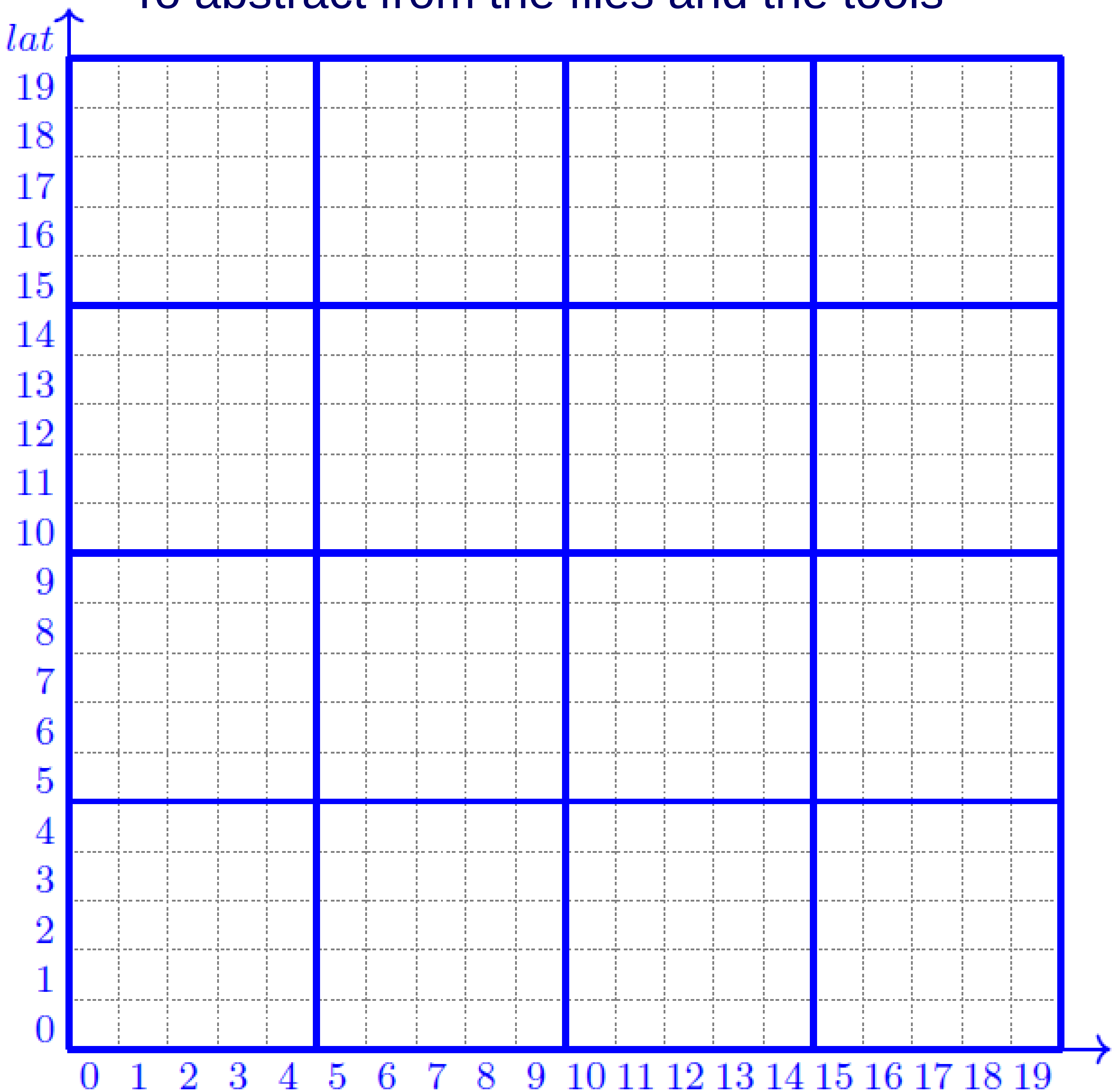
NetCDF Operators (NCO): started in 1995
http://nco.sourceforge.net/



ChronosServer

Data Model

To abstract from the files and the tools



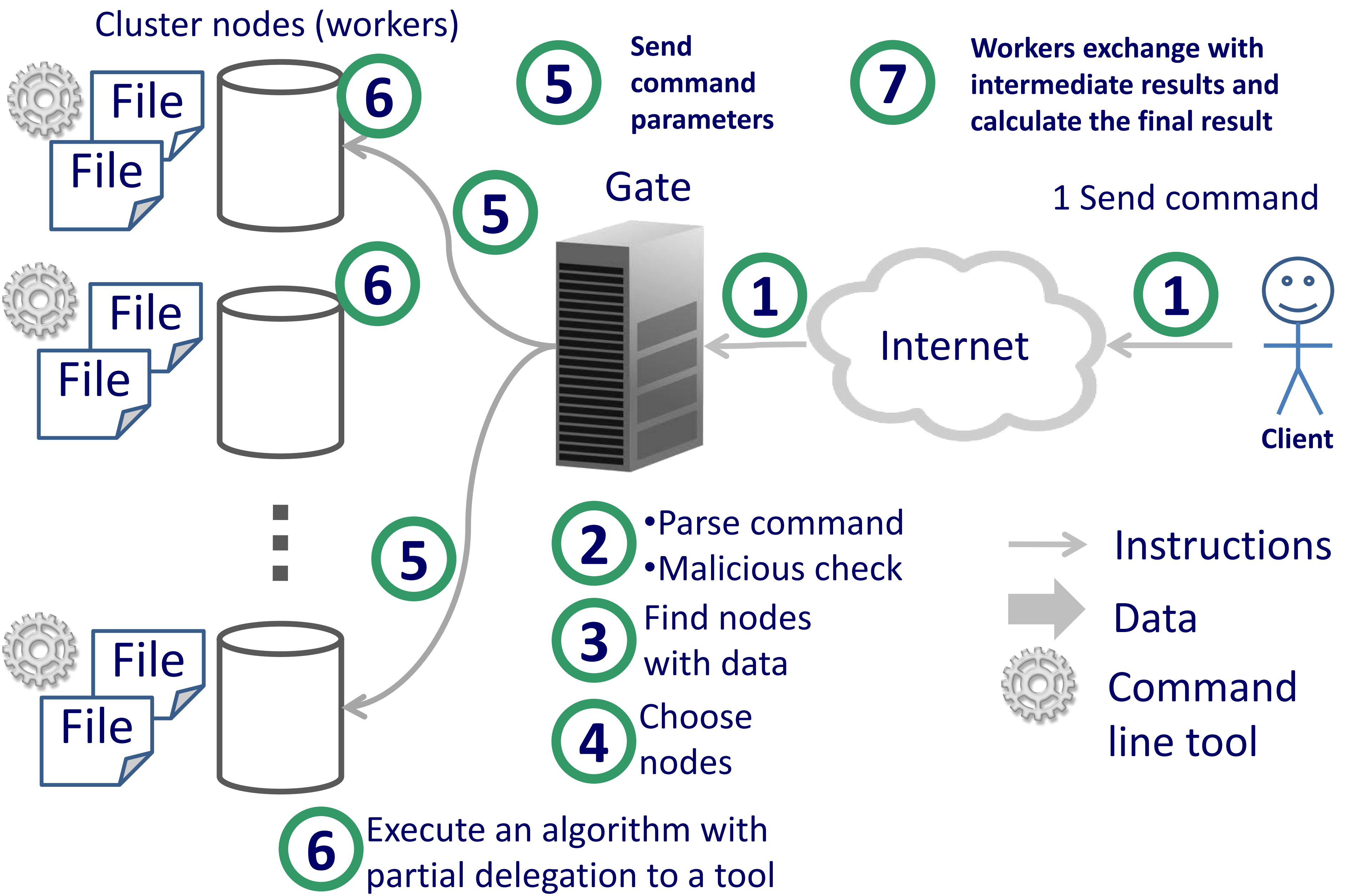
Two-level:

- User-level array
- System-level arrays (separated with thick blue lines)

An operation on a user-level array is mapped to a sequence of operations with respective subarrays

Each system-level array is stored as a regular raster file

Distributed execution of a single raster processing operation

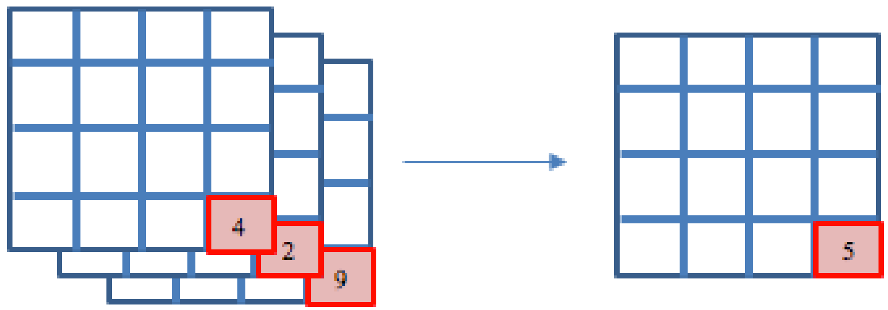


Array Operations for Performance Evaluation

(operations specifically targeted at retrospective data)

Aggregation

Create a single 2-d array from a time series of arrays: ***min, max, avg***



Algorithm 1 Distributed in situ array aggregation with delegation to an external command line tool (procedure AGGREGATE is executed on workers).

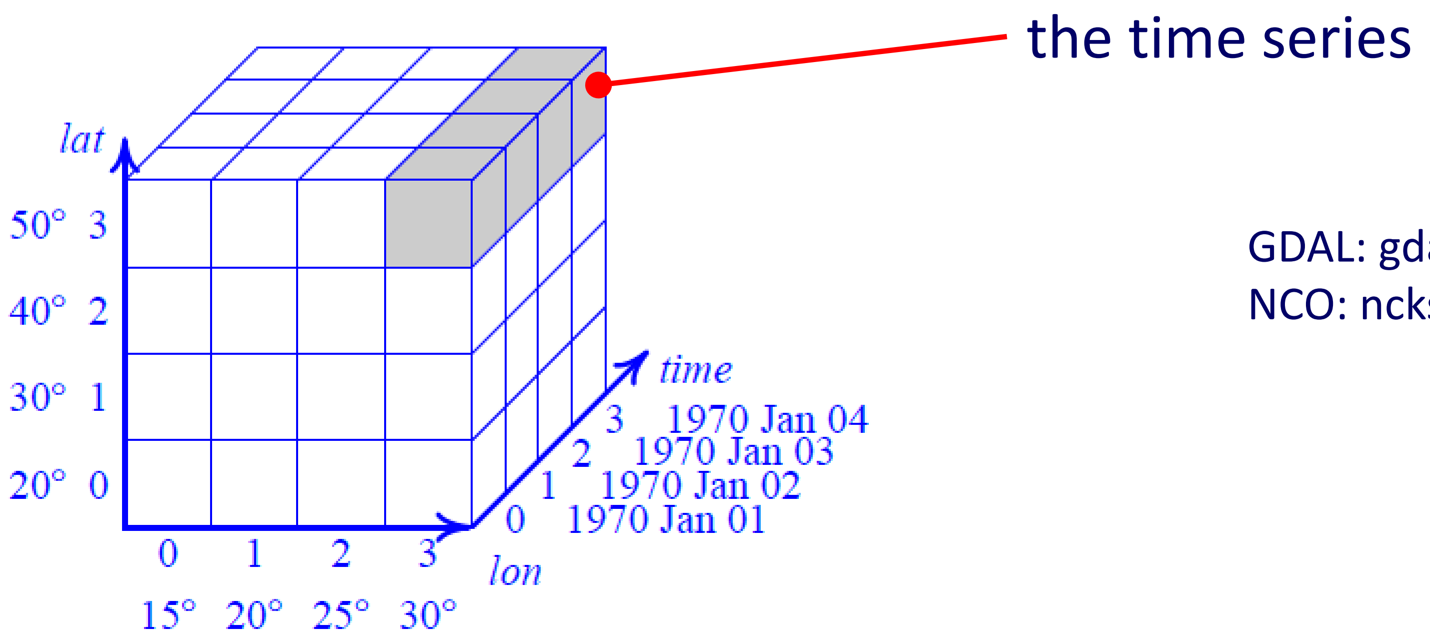
Input: wid is the identifier of the worker performing final aggregation

procedure AGGREGATE($\mathbb{D}$, f_{aggr}, wid) $\triangleright$ $\mathbb{D}$ is a dataset, see section 2.2
  aggregate all $p \in P$ residing on this worker into p' $\triangleright$ DELEGATION
  if the id of this worker equals to wid **then**
    accept subarrays from other workers: P_{aggr}
    aggregate p' and all $p \in P_{aggr}$ into p_{aggr}
    report success to Gate
  else send p' to worker with id $= wid$

GDAL: gdal calc.py
NCO: ncra

Hyperslabbing

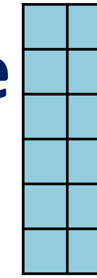
Extract a subarray from an N-d array



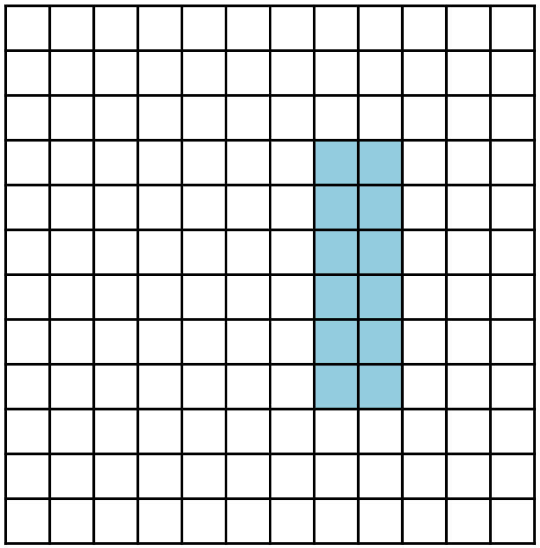
GDAL: gdal_translate
NCO: ncks

For the performance evaluation we extract subarrays from 3-d arrays (the time series of Landsat scenes)

Chunking: row-major disk layout, read 6×1 slice

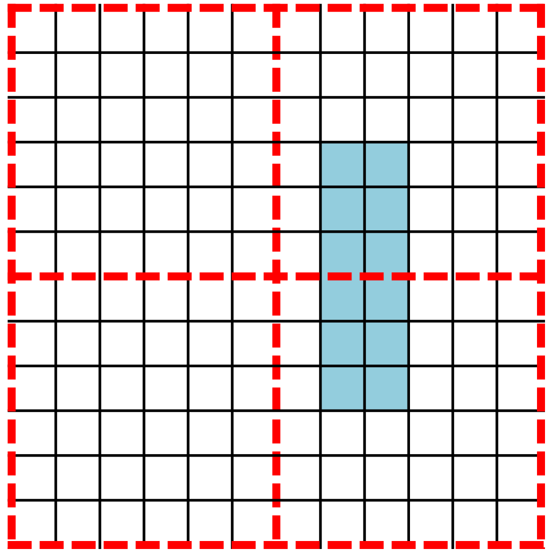


no chunking *
12 × 12 raster



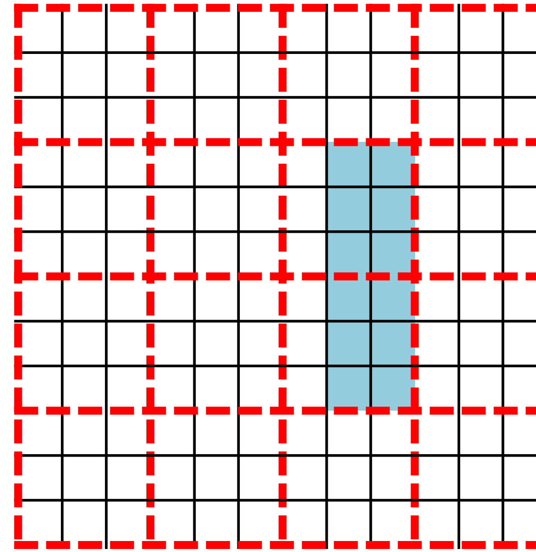
read 1 × 2 portions
6 storage requests
(a)

2 × 2 chunks
one chunk 6 × 6



read 2 chunks
50% of all data
(b)

4 × 4 chunks
one chunk 3 × 3



read 2 chunks
12.5% of data
(c)

* the whole array is a single chunk

** reads may also involve uncompressing data

Note that SSD is not the solution

Performance Evaluation

State-of-the-art

	in situ	delegate	distributed	1st release
ChronosServer	YES	Direct	YES	?
SciDB	NO	Streaming	YES	~ 2008**
Oracle Spatial*	NO	NO	YES	< 2005
ArcGIS IS*	YES	NO	NO****	> 2000
RasDaMan	YES/NO***	NO	YES/NO***	~ 1999
MonetDB SciQL	NO	NO	NO	?
Intel TileDB	NO	NO	NO	04 04 2016

SciDB is the only free and distributed array DBMS available for comparison

* Commercial

** Now (in 2017, since 9 years) it still has a very limited set of operations

*** YES in *payed*, enterprise version

**** Data are the same on each server or retrieved from centralized storage

SciDB

paradigm4

Known for	Ingres, Postgres, Vertica, Streambase, Illustra, VoltDB, SciDB
Notable awards	IEEE John von Neumann Medal (2005) ACM Turing Award (2014)

https://en.wikipedia.org/wiki/Michael_Stonebraker

SciDB – Scientific DB

NoSQL
AQL – Array Query Language
AFL – Array Functional Language

Distributed
General-purpose multidimensional array DBMS

Test data: time series of Landsat 8 scenes

Scenes cover Rome, Italy

- 9 scenes
- 30 m/pixel
- Band 1
- Path 190, Row 31
- Cloud cover < 20%
- 585 MB
- UTM 33N projection

Large SciDB import time prevents the evaluation on a larger data portion

Test data: time series of Landsat 8 scenes

Scenes cover Rome, Italy

- SciDB array
  18 × 7971 × 7941

To increase the data volume and to avoid waiting for loading more scenes, we attached SciDB array to itself to get the time dimension of size 18.

We could not attach the resulting array to itself again. We tried in many ways including array import with different chunk shapes but SciDB had been always failing with “not enough memory error”. As of 29-May-2017, we did not receive any feedback from SciDB developers on this issue.

Experimental setup (1)

Microsoft Azure

Computer Cluster in MS Azure Cloud

8 VMs (16 CPU cores)

- **SciDB 16.9, Nov. 2016 (the latest)**
- **Ubuntu 14.04 LTS**

  (SciDB 16.9 does not run on a newer Ubuntu)

Although Azure states the disk to be SSD, after the creation of such a disk Azure displays the disk to be a standard HDD disk backed by a magnetic drive.

D2_V2 Standard	
2	Cores
7	GB
	4 Data disks
	4x500 Max IOPS
	100 GB Local SSD
	Load balancing

Intel Xeon E5-2673 v3 (Haswell) 2.4 GHz

6 324,00

RUB/MONTH (ESTIMATED)

Experimental setup (2)

ChronosServer

- 100% Java
- Java 1.8
- OracleJDK 1.8.0_111 64 bit
- -Xmx 978 MB (max heap size)
- 1 worker per machine

The tools

available from the standard Ubuntu 14.04 repository were used: **GDAL** v1.10.1 (released 2013/08/26), **NCO v4.4.2**, last modified 2014/02/17

SciDB

- v16.09, latest (Nov 2016), C++

Parameters:

- 2 instances per machine
- 0 redundancy
- 5 execution and prefetch threads
- 1 prefetch queue size
- 1 operator threads
- 1024 MB array cache
- etc.

SciDB data import

1. No out-of-the-box import tool
   - Supports import from CSV files
2. Requires software development for data import
   - Took 3 weeks to develop and debug Java self-crafted SciDB import tool
3. Import procedure looks like this (*very simplified*):

```
Open GeoTiff file
Read metadata (band shape, etc.)
Create respective SciDB arrays to add data into
Read 2D band
Convert band to CSV
Save CSV string to CSV file
Feed CSV file to SciDB
```

4. Significant manual intervention, error-prone, slow (next slides)
5. Does not guarantee to be imported by SciDB (sometimes fail due to large array size)

Initialization phase

SciDB import

≈ 40 minutes of 1 scene

CODING & DEBUGGING TIME NOT TAKEN INTO ACCOUNT

ChronosServer tiling

≈ 410 seconds at most

Estimate: linear scalability on number of files

Data import lesson

Cannot import large data volumes

into SciDB 16.09 in a reasonable time frame.

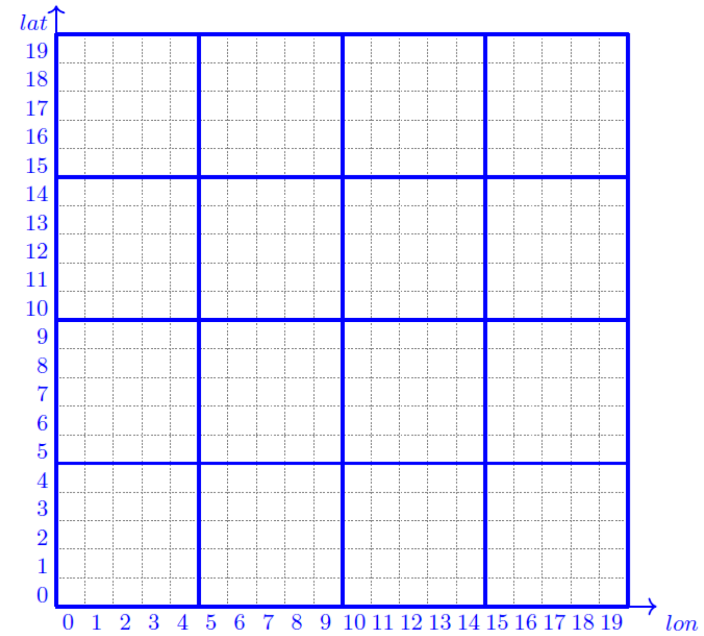
Data Cooking

Preprocessing Landsat data: 18 scenes, 1 cluster node

Target Shape	Time, sec.	Target Shape	Time, sec.
$4 \times 512 \times 512$	410.25	$9 \times 1024 \times 1024$	216.62
$9 \times 512 \times 512$	376.32	$4 \times 4096 \times 4096$	56.87
$4 \times 1024 \times 1024$	187.41	$9 \times 4096 \times 4096$	55.45

GeoTIFF → NetCDF

Tiling (cutting) the single array 18 × 7971 × 7941 → several T × N × M smaller subarrays



Experimental Results

Operation	Time, sec.			Ratio, SciDB/Chronos	
	ChronosServer (raw data)	ChronosServer ("cooked" data)	SciDB		
Average	38.36	8.12	230.74	6.02	28.42
Maximum	38.83	4.56	127.71	3.29	28.00
Minimum	38.98	4.63	125.70	3.22	27.15
Cut 512 × 512	1.79	1.01	1.98	1.11	1.96
Cut 1024 × 1024	3.34	2.14	3.41	1.02	1.59
Time series	0.53	0.31	0.84	1.58	2.71
Chunk 1 × 64 × 64	22.37	—	—	—	
Chunk 1 × 128 × 128	22.49	—	—	—	

Raw: GeoTIFF, time series of 18 scenes
Cooked: NetCDF, smaller subarrays

SciDB fails to chunk the array 18 × 7971 × 7941. It had been always failing with "not enough memory error". As of 29-May-2017, we did not receive any feedback from SciDB developers on this issue.

References

[1] Rodriges Zalipynis, R.A.: Distributed in situ processing of big raster data in the cloud. In: Perspectives of System Informatics – 11th International Andrei Ershov Informatics Conference, PSI 2017, Moscow, Russia, June 27–29, 2017, Revised Selected Papers. Lecture Notes in Computer Science, Springer (2017), in press

[2] Rodriges Zalipynis, R.A.: ChronosServer: Fast in situ processing of large multidimensional arrays with command line tools. In: Voevodin, V., Sobolev, S. (eds.) Supercomputing: Second Russian Supercomputing Days, RuSCDays 2016, Moscow, Russia, September 26–27, 2016, Revised Selected Papers. Communications in Computer and Information Science, vol. 687, pp. 27–40. Springer International Publishing, Cham (2016), http://dx.doi.org/10.1007/978-3-319-55669-7_3

[2] Rodriges Zalipynis, R.A.: Chronosserver: real-time access to “native” multi-terabyte retrospective data warehouse by thousands of concurrent clients. Inform., Cybern. Comput. Eng. 14(188), 151–161 (2011)

Some slides were modified from [1, 2]; original slides are available at http://doi.org/10.13140/RG.2.2.26922.21444

NATIONAL RESEARCH UNIVERSITY

Thank you for your attention!

Contributions

Rodriges:

- all slides,
- all text and figures,
- design and implementation of algorithms and ChronosServer,
- ChronosServer data model,
- Azure management code,
- SciDB import code,
- experimental setup.

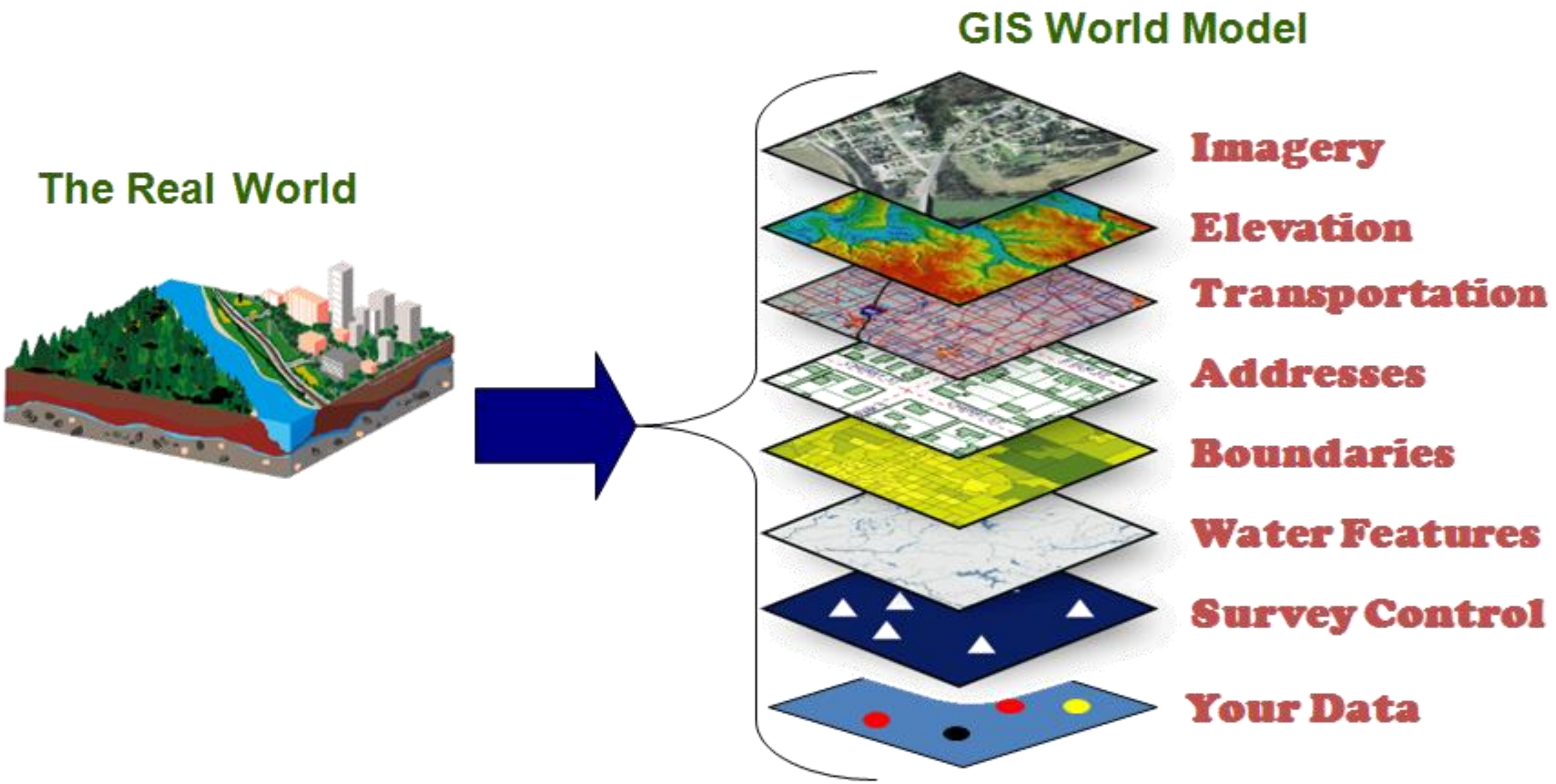
Pozdeev: SciDB cluster deployment.

Bryukhov: adapted SciDB import code to Landsat data.

All authors: experiments.

Satellite Data Applications

Numerous practically important projects

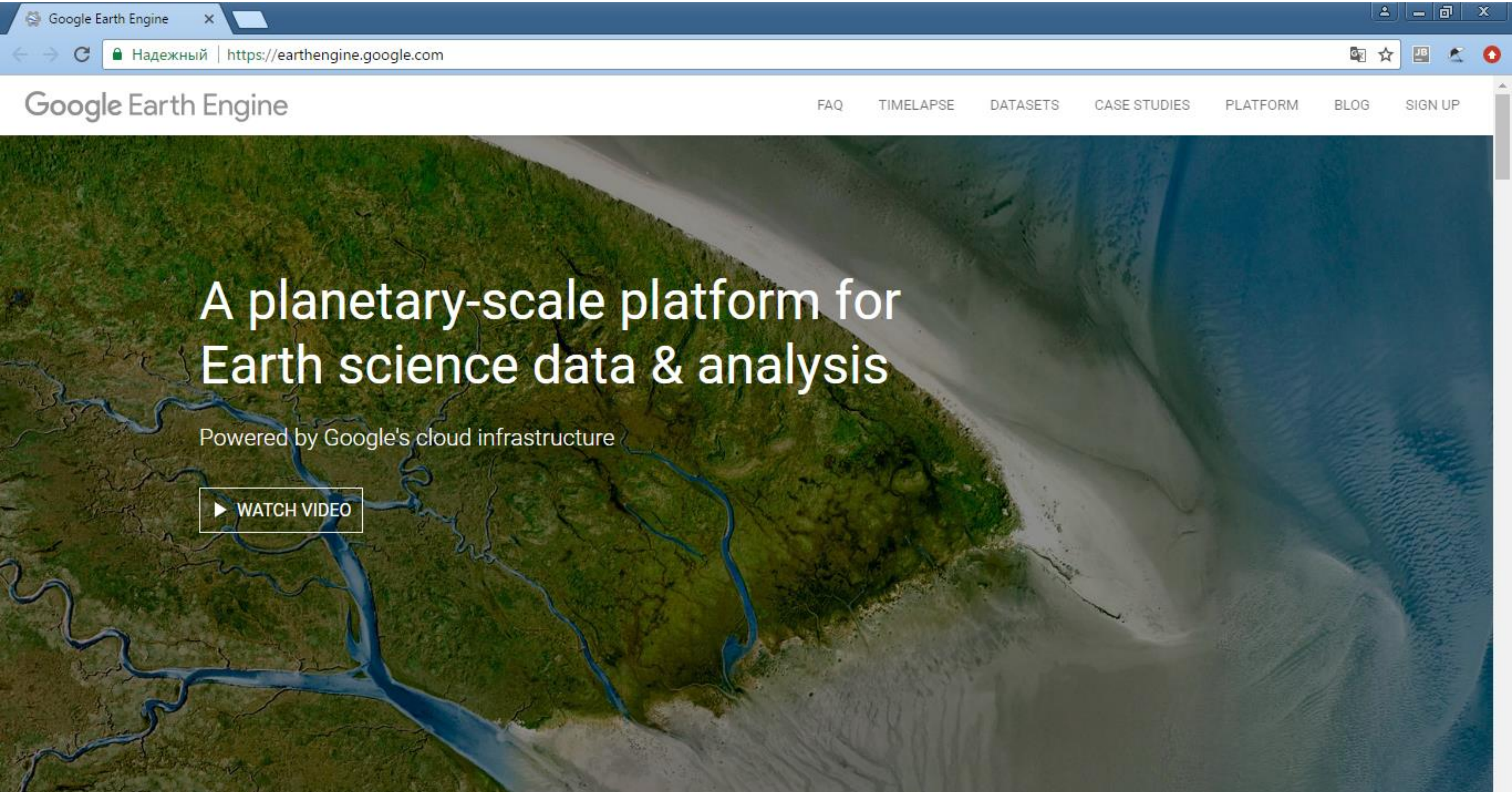


http://www.teachoceanscience.net/teaching_resources/education_modules/applications_of_landsat_data/getting_started/

Google Earth Engine

Released on the 40th anniversary of the Landsat Program

https://earthengine.google.com/



Results from other paper

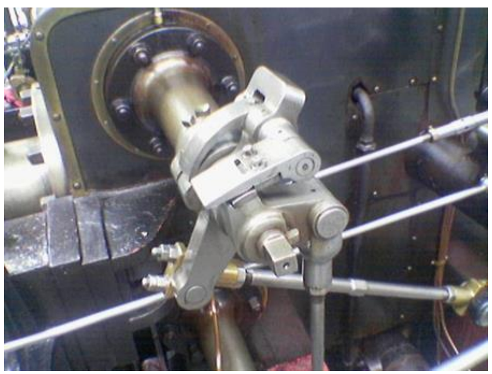
Rodriges Zalipynis R. A., Pozdeev E., Bryukhov A.
Satellite Imagery and Array DBMS: Towards Big Raster Data in the Cloud,
in: Analysis of Images, Social Networks and Texts.
6th International Conference, AIST 2017,
Lecture Notes in Computer Science, Revised Selected Papers.
Springer, 2017. (in press)

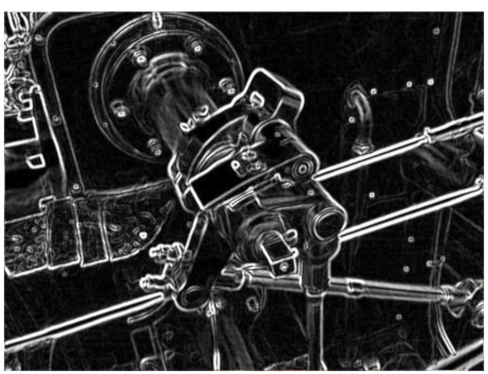
Other operations

Convolution

$$\mathbf{G} = \sqrt{{\mathbf{G}_x}^2 + {\mathbf{G}_y}^2}$$

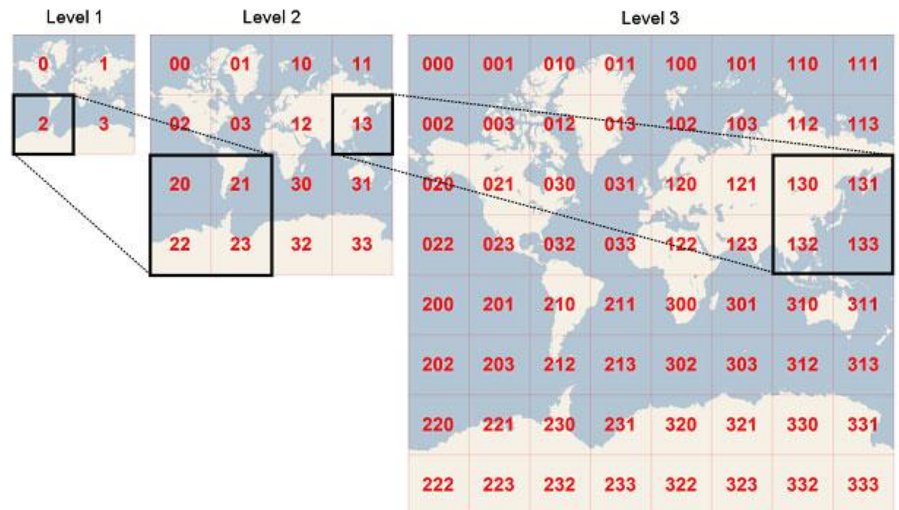
$$\mathbf{G}_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * \mathbf{A}$$



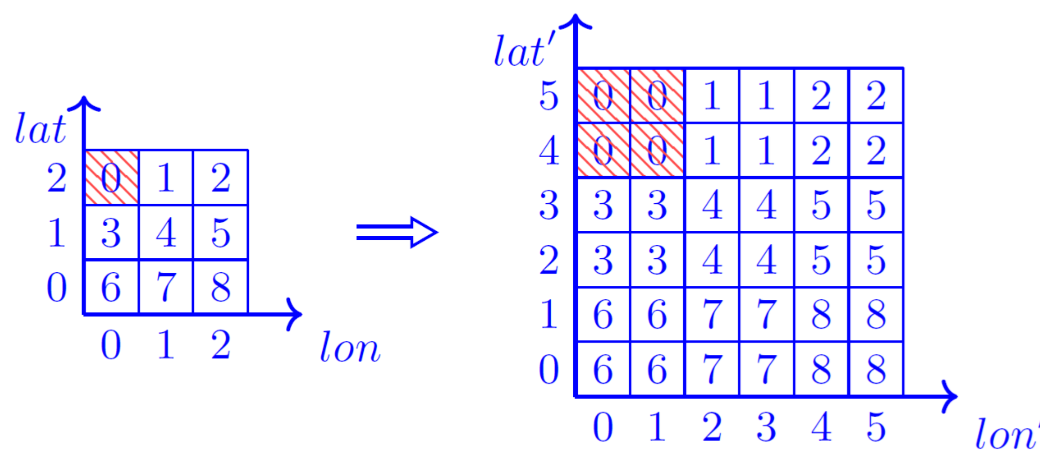


https://en.wikipedia.org/wiki/Sobel_operator

Multiresolution pyramid



Interpolation



Test data: Landsat 8 mosaic

- 4×4 scenes
- 30 m/pixel
- 02–11/07/2015
- 2279.94 MB
- cloud cover ≈23.46%
- UTM 31N projection

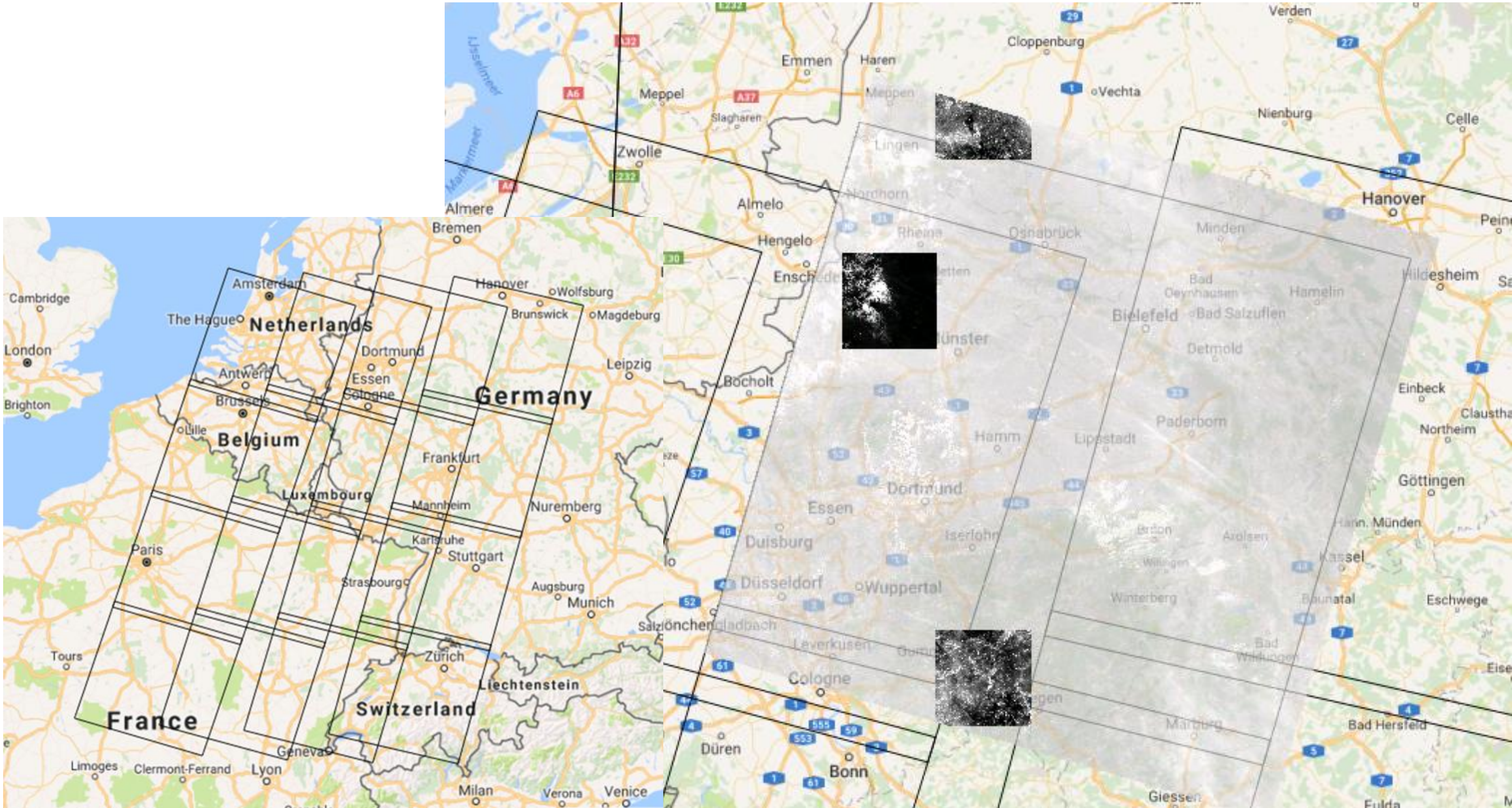
- SciDB array 24937×22855
- chunk 512×512

Test data: Landsat 8 mosaic

Scenes are cut into tiles (subarrays)

Overlapping tiles from distinct scenes are merged into a single tile

Tiles are evenly distributed among cluster nodes



Experimental Results

Pyramid (3 levels)	8 nodes	16 nodes
ChronosServer	13.41	8.28
SciDB	148.57	76.28
Ratio, SciDB/Chronos	**11.08**	**9.21**
Interpolation 2×	**8 nodes**	**16 nodes**
ChronosServer	24.00	11.65
SciDB	190.83	108.36
Ratio, SciDB/Chronos	**7.95**	**9.30**
Hyperslabbing[1]	**8 nodes**	**16 nodes**
ChronosServer	3.24	1.52
SciDB	5.67	3.47
Ratio, SciDB/Chronos	**1.75**	**2.28**
Sobel Filter	**8 nodes**	**16 nodes**
ChronosServer	179.22	92.71
SciDB	82087.2[2]	7527.06[3]
Ratio, SciDB/Chronos	**458.02**	**81.19**

1. Extract 1/4th of the image from its center
2. 1/256 size of the original image
3. 1/64 size of the original image; SciDB fails on full 4 × 4 scenes mosaic