Topics

Part I: BFAST R package optimizations

Part II: Scalable EO data management with SciDB

Part III: Hands-on with SciDB, Landsat, and BFAST

1. SciDB installation (with Docker)
2. Data ingestion
3. Analysis (practical part)

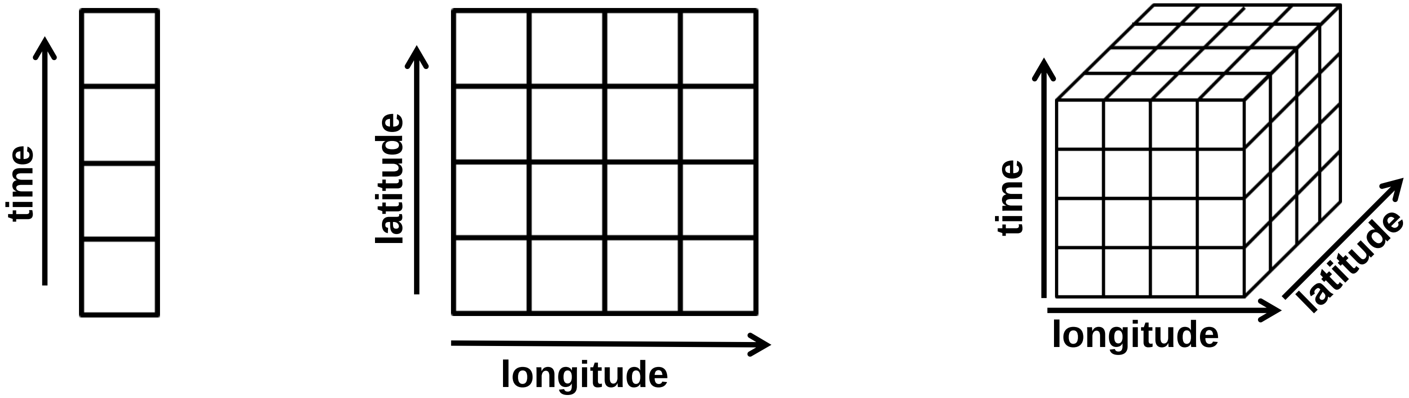
BFAST on large datasets: bfastSpatial and raster

- works well with out-of-memory data
- supports multicore parallel processing

- difficult to stack data from different tiles due to overlap and different recording dates
- does not scale beyond multiple machines on its own

SciDB for large EO datasets

- Array-based data management and analytical system [1]
- Runs on single computers as well as on large clusters
- Open-source version available
- Sparse storage
- Basic data representation as **multidimensional arrays**
- n dimensions, m attributes (bands) with different data types

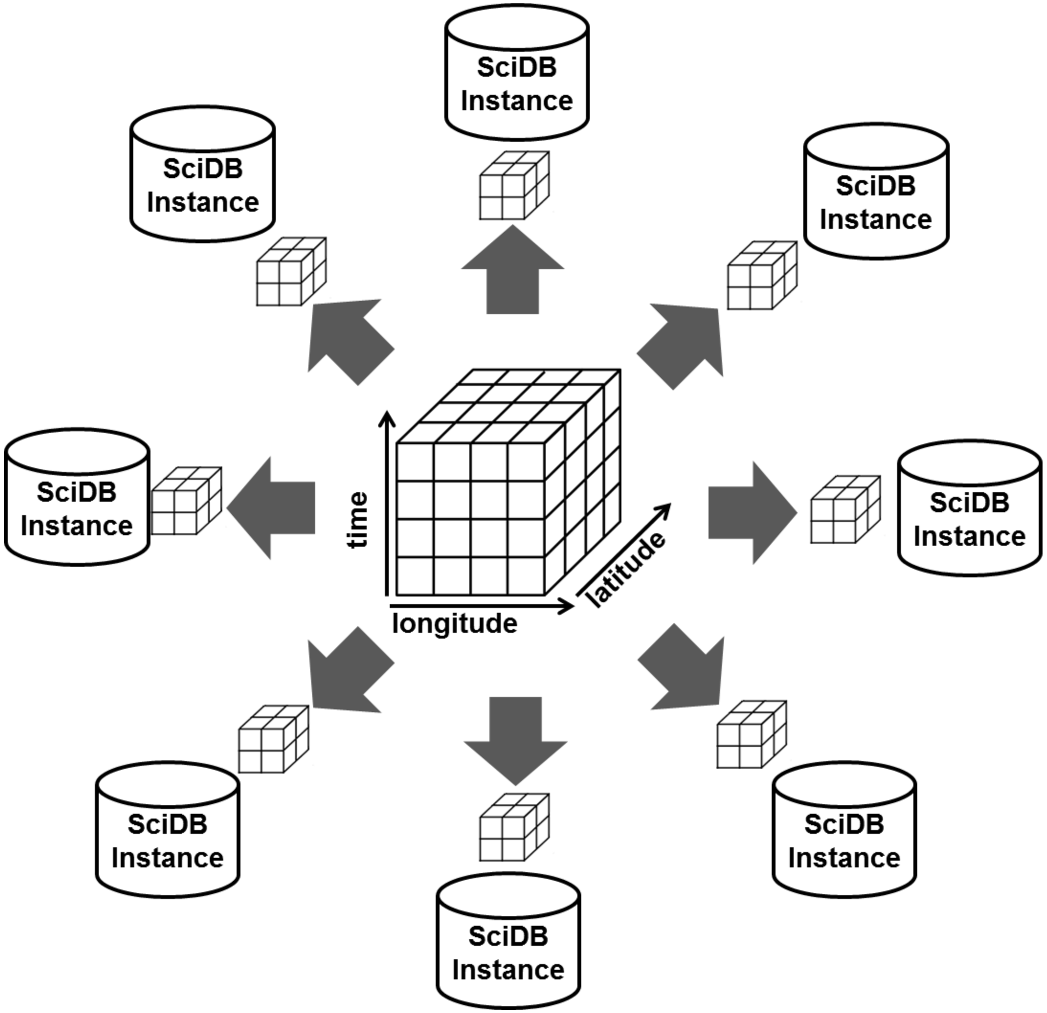


[1] Stonebraker, M., Brown, P., Zhang, D., & Becla, J. (2013). SciDB: A database management system for applications with complex analytics. *Computing in Science & Engineering*, *15*(3), 54-62.

Distributing arrays by chunking

- arrays are divided into equally sized chunks
- chunks are distributed over many SciDB instances
- instances may run on the same or different machines in a shared nothing cluster

→ distributing storage and computational load



Query language and functionality

- SciDB query language: Array Functional Language (AFL)

- Native functionality:
  - Load / write arrays from / to files
  - Arithmetic operations
  - Subsetting by dimensions and / or attributes
  - Aggregations (window, aggregate)
  - Array joins
  - Changing array schemas (repartitioning, redimensioning)
  - Linear algebra routines: (GEMM, GESVD, basic statistics)
  - …

SciDB: extensions for EO data

SciDB

- can load data from CSV and custom-binary files only
- does not understand spatial / temporal reference of arrays

→spacetime extensions [1]:

  – scidb4geo (https://github.com/appelmar/scidb4geo)
  – scidb4gdal (https://github.com/appelmar/scidb4gdal)

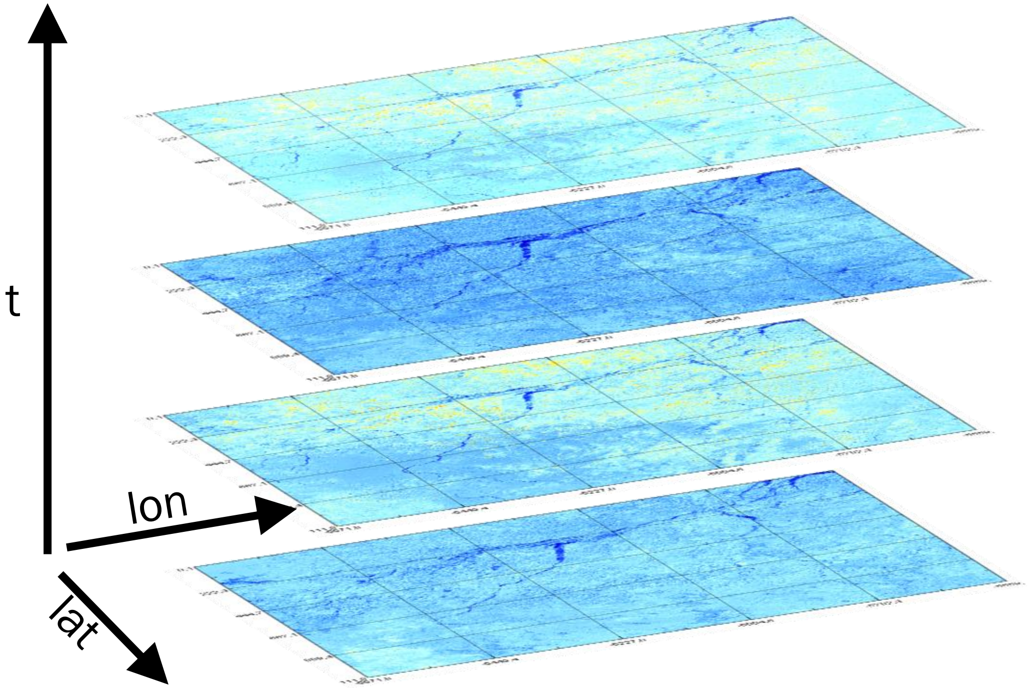
[1] Appel M., Lahn F., Pebesma E., Buytaert W., Moulds S. (2016). Scalable Earth-observation Analytics for Geoscientists: Spacetime Extensions to the Array Database SciDB. accepted for poster presentation at EGU General Assembly 2016, Vienna, Austria April 17-22, 2016.

scidb4geo

New AFL (Array Functional Language) operators	
Operator	Description
eo_arrays()	Lists geographically referenced arrays
eo_setsrs()	Sets the spatial reference of existing arrays
eo_getsrs()	Gets the spatial reference of existing arrays
eo_extent()	Computes the geographic extent of referenced arrays
eo_settrs()	Sets the temporal reference of arrays
eo_gettrs()	Gets the temporal reference of arrays
eo_setmd()	Sets key value metadata of arrays and array attributes
eo_getmd()	Gets key value metadata of arrays and array attributes
eo_over()	Overlays two arrays by space and / or time

scidb4gdal

- supports ingestion and download of images to and from SciDB
- GDAL supports > 100 raster formats
- ingestion automatically combines images by space and time (mosaicing)



Interfacing R

R as a client: packages `scidb`[1] and `scidbst`[2] works with proxy objects and lazy evaluation → starts computations when you want to read the data

- overwrites R methods, e.g. %*%
- limited to native SciDB functionality

Running R within SciDB: `stream`[3] and `r_exec`[4]

- apply arbitrary R functions in parallel on chunks

[1] https://github.com/Paradigm4/SciDBR
[2] https://github.com/flahn/scidbst
[3] https://github.com/Paradigm4/stream
[4] https://github.com/Paradigm4/r_exec

BFAST within SciDB

- **Idea**: organize chunk sizes such that one chunk contains the complete time-series of a small region, e.g. 50x50 pixels

- Use `stream` or `r_exec` to run bfast in parallel

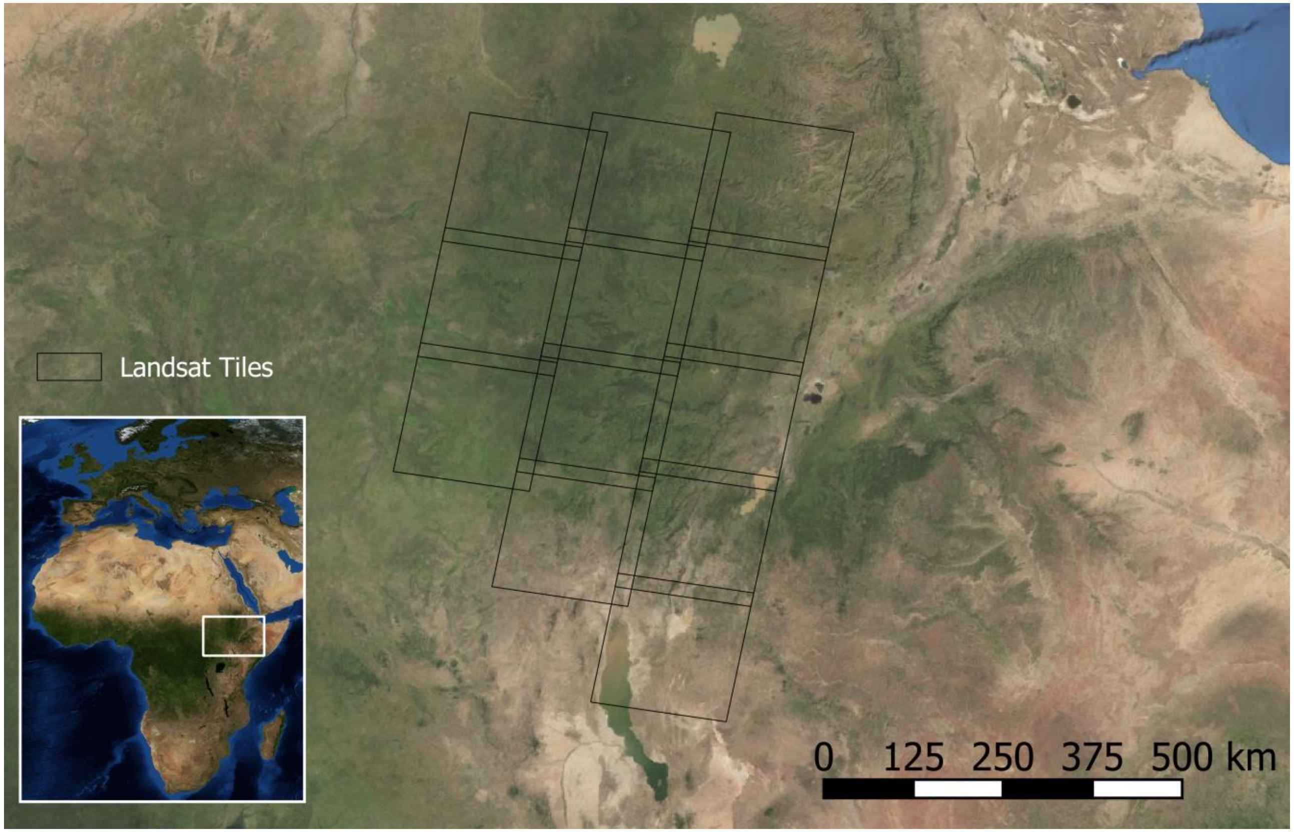
- R and the bfast package must be installed on all SciDB servers

→ scalability with relatively little amount of reimplementation needed

→ move computations to the data instead of move the data to the computations

Study case:
Monitoring changes in NDVI time series of Landsat 7 in south west Ethiopia

- Landsat 7 data from 12 tiles captured between 2003-07-21 and 2014-12-27 → 1975 scenes
- Derived NDVI product from ESPA
- approx. 325,000 km^2
- monitor changes starting with 2010-01-01, with ROC history model



Landsat 7 in SciDB

1. Ingestion:
   - For all *_ndvi.tif images:
     - extract date from filename
     - reproject / warp to the same spatial reference system
     - upload to SciDB

2. Repartition the array such that chunks contain complete time series of 64x64 pixels

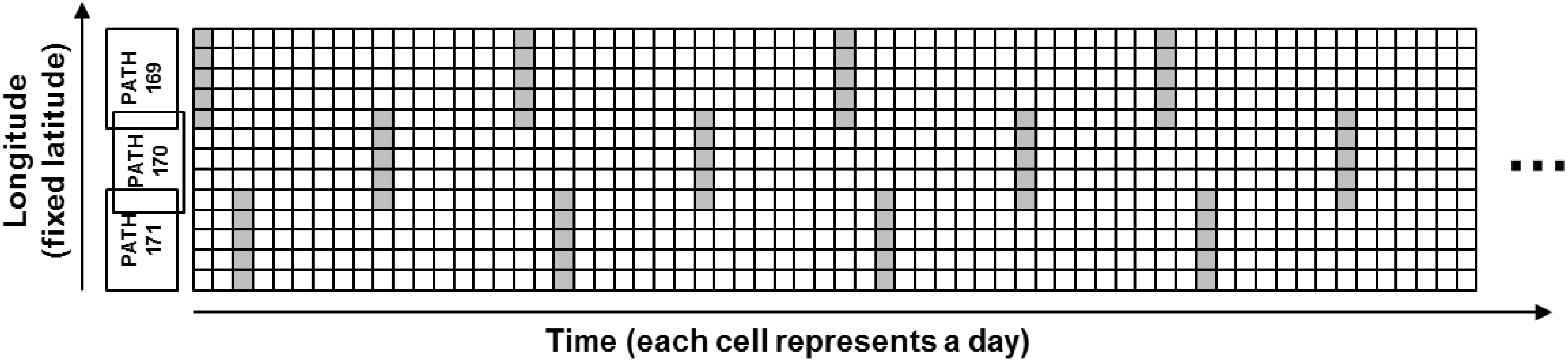
3. Preprocessing:
   - remove any values <= -9999 or >10000
   - unscale to -1, 1

- Ingestion of all scenes took around 4 days
- Repartitioning took around 2 days

Landsat 7 in SciDB

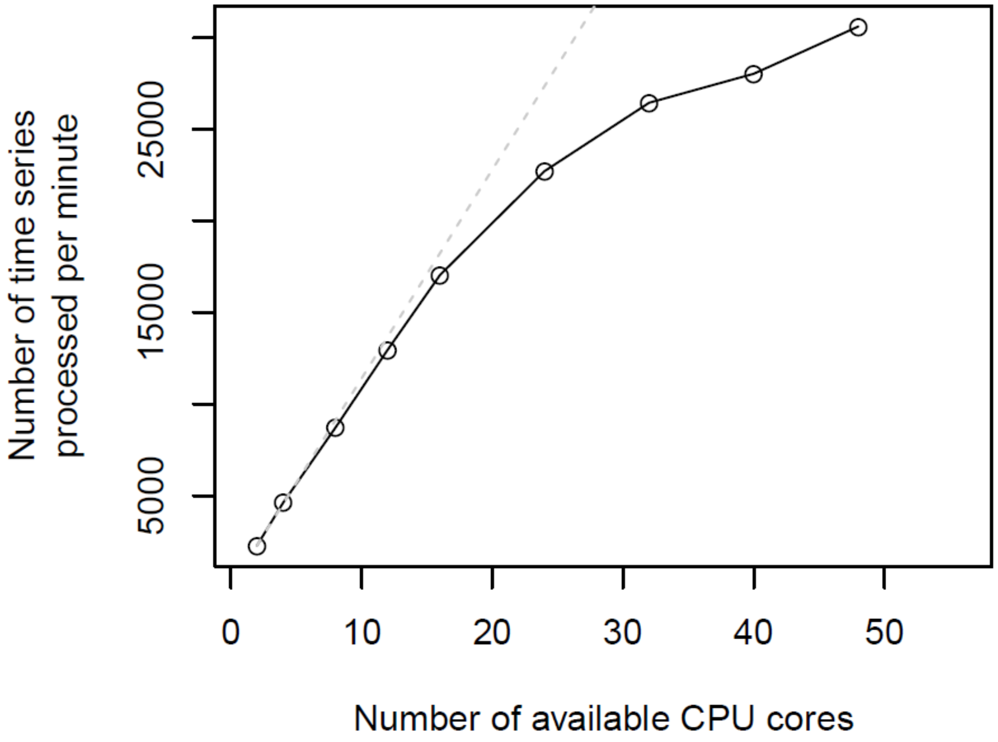
The data is represented in SciDB as a three-dimensional array with **daily temporal resolution** and

- 49548 x 47713 x 4177 cells in total
- 64 x 64 x 4177 cells per chunk
- Only 0.5% ($54 \cdot 10^9$) of the cells contain data
- SciDB has sparse storage



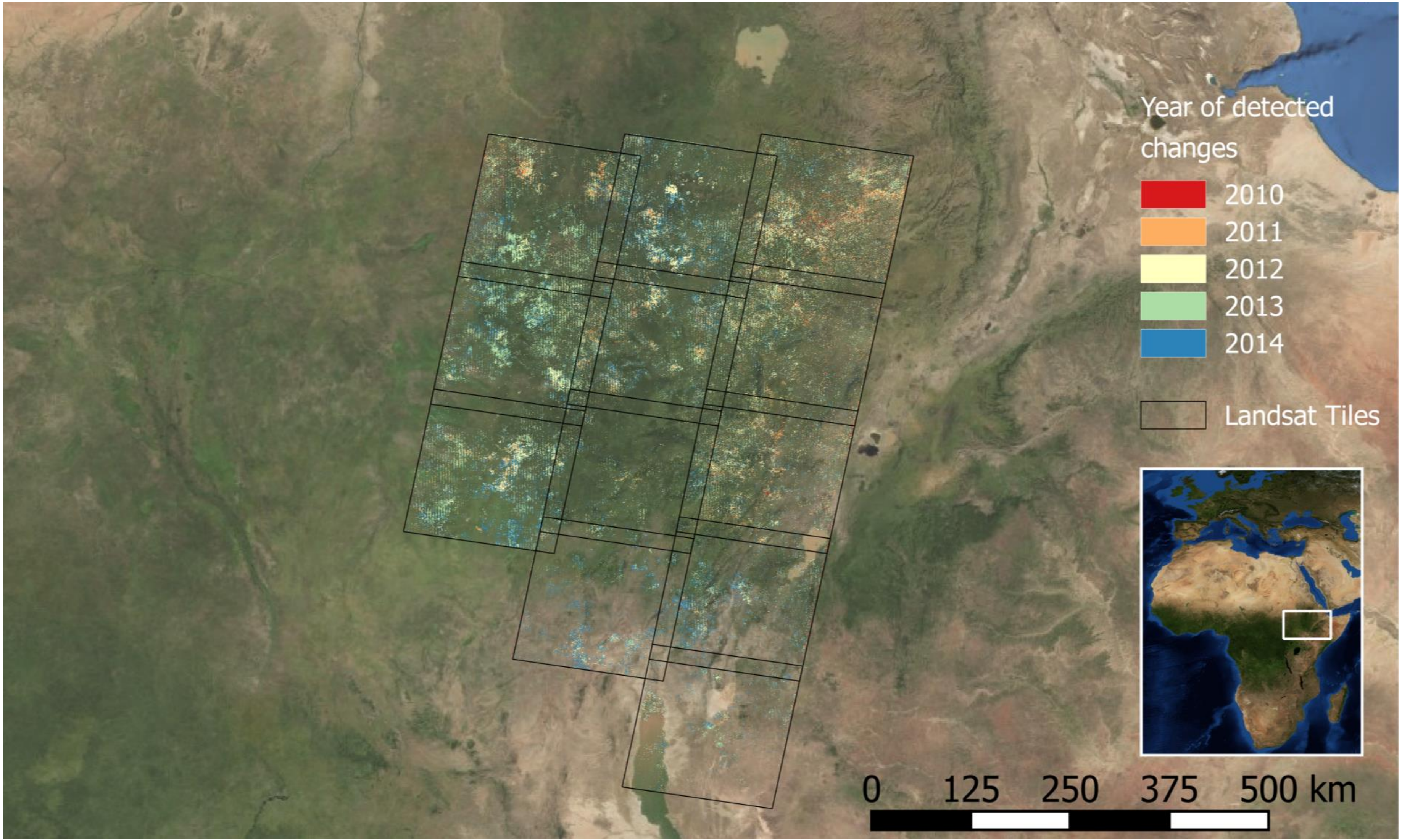
Scalability with SciDB instances

- 16 SciDB instances on one machine used (64 CPU cores, 256 GB main memory)
- running bfastmonitor repeatedly with different number of available CPU cores on a small subset



Study case: results

- Running bfastmonitor on the complete dataset took 8 days



Conclusions

- SciDB is able to make BFAST scalable even in large cluster environments
- The multidimensional array model, chunking, and sparse storage are well-suited to represent large EO datasets from many scenes
- Ingestion and data restructuring time consuming, alternatives to GDAL needed
- Installation and data ingestion not straightforward
- Analysis from R relatively easy to learn for experienced R users (see hands-on part)

Thank you

Questions?