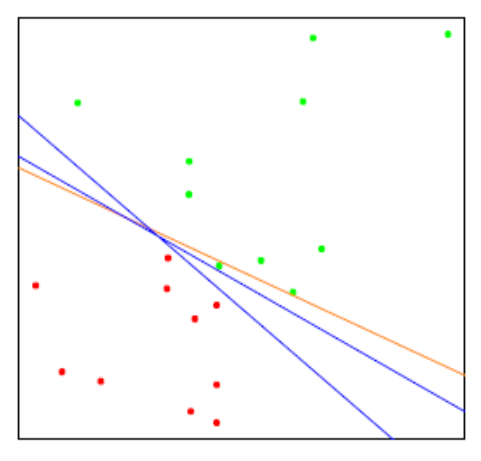


Parallel & Scalable Machine Learning

Introduction to Machine Learning Algorithms

Dr. Gabriele Cavallaro

Postdoctoral Researcher

High Productivity Data Processing Group

Juelich Supercomputing Centre

LECTURE 12

Unsupervised Clustering and Applications

February 27th, 2019

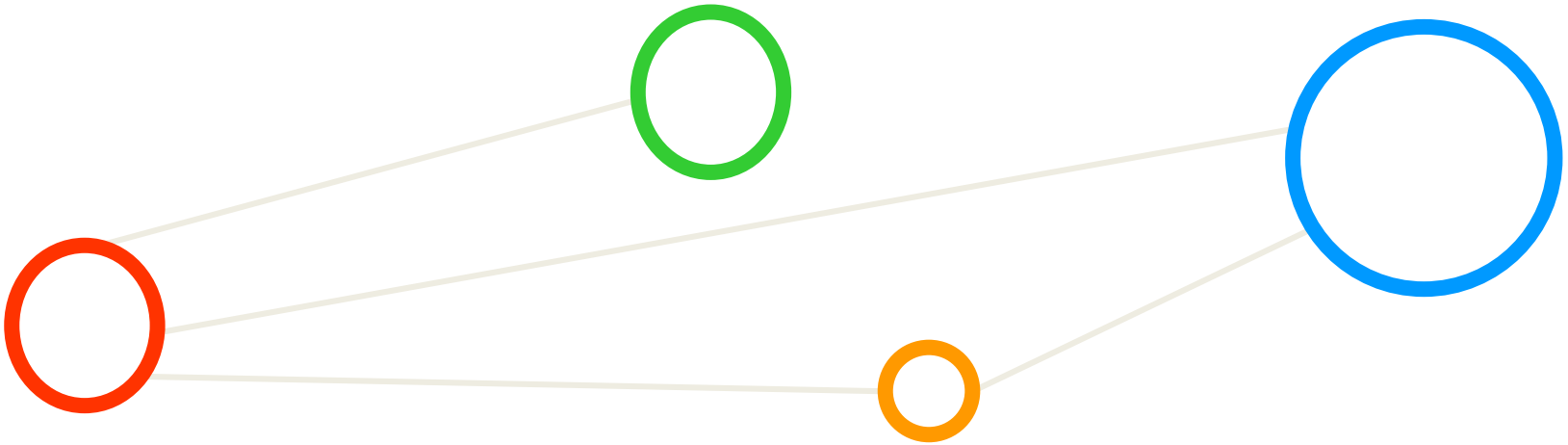
JSC, Germany

UNIVERSITY OF ICELAND

SCHOOL OF ENGINEERING AND NATURAL SCIENCES

FACULTY OF INDUSTRIAL ENGINEERING, MECHANICAL ENGINEERING AND COMPUTER SCIENCE

Outline



Outline of the Course

1. Parallel & Scalable Machine Learning driven by HPC
2. Introduction to Machine Learning Fundamentals
3. Introduction to Machine Learning Fundamentals
4. Feed Forward Neural Networks
5. Feed Forward Neural Networks
6. Validation and Regularization
7. Validation and Regularization
8. Data Preparation and Performance Evaluation
9. Data Preparation and Performance Evaluation
10. Theory of Generalization
11. Unsupervised Clustering and Applications
12. Unsupervised Clustering and Applications
13. Deep Learning Introduction

Theoretical Lectures

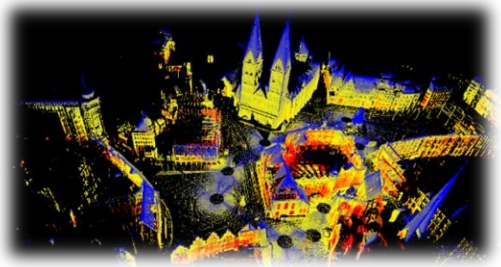
Practical Lectures

Outline

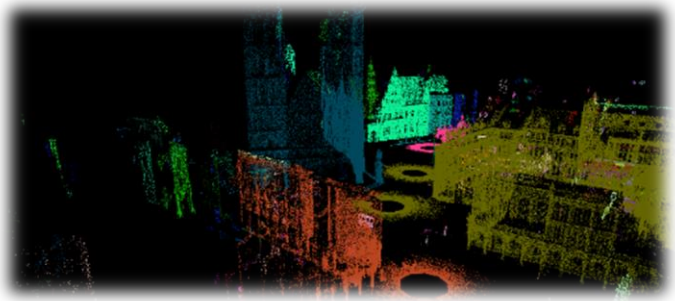
- Point Cloud Applications
  - Actueel Hoogtebestand Nederland
  - Bremen Dataset
- HPDBSCAN
  - HDF5 Parallel I/O
  - Parallel strategy
  - HDFView
- MPI Collective Functions

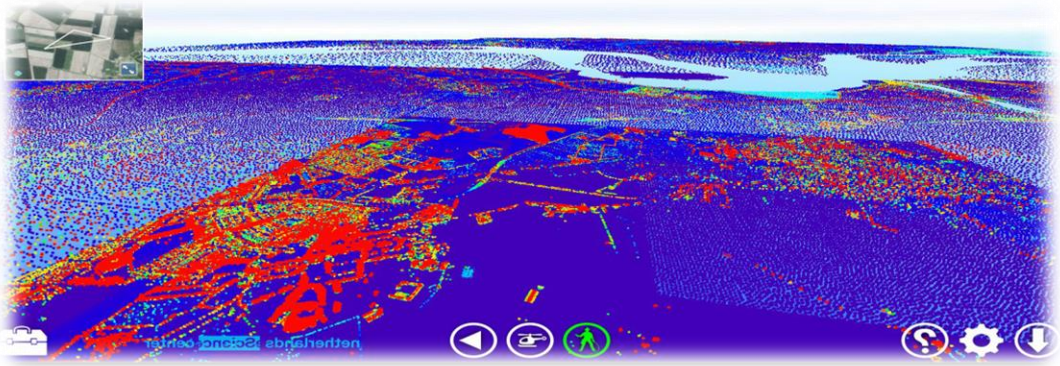
Point Cloud Applications

- ‘Big Data‘: 3D/4D laser scans
  - Captured by robots or drones
  - Millions to billion entries
  - Whole countries (e.g. Netherlands)
  - Inner cities (e.g. Bremen inner city)



- Selected Scientific Cases
  - Filter noise to better represent real data
  - Grouping of objects (e.g. buildings)
  - Study level of continous details

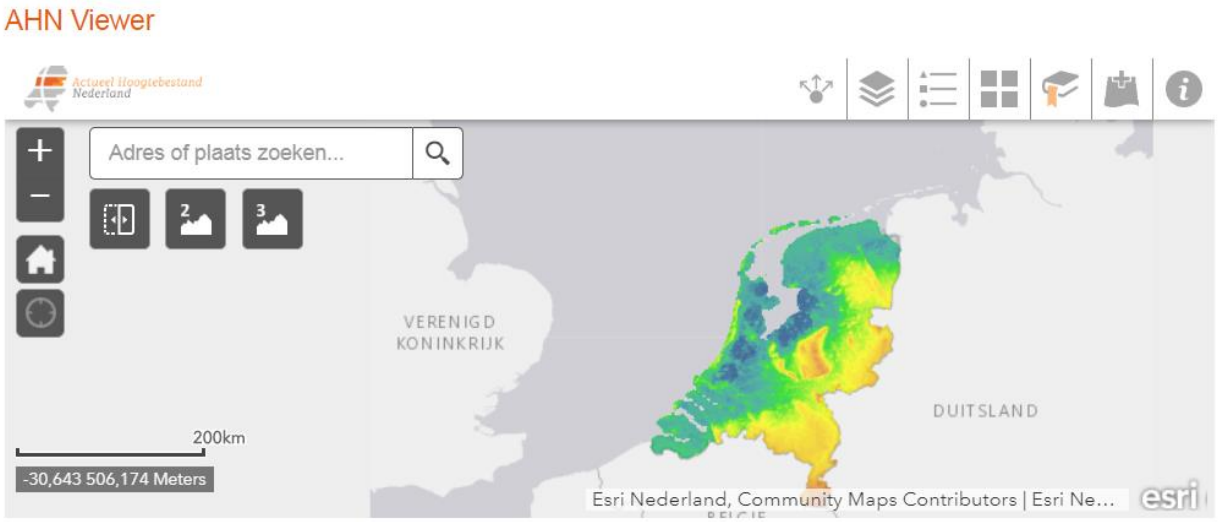




Actueel Hoogtebestand Nederland

- AHN is the digital height map for the whole of the Netherlands
- It contains detailed and precise height data
  - Average of 8 heights (points) measurements per square meter

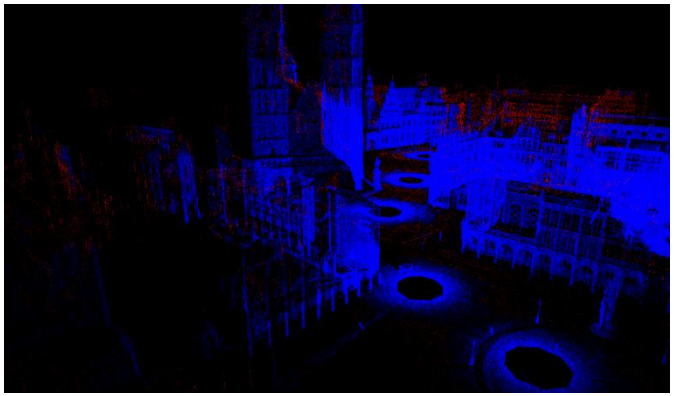
Name	Points	LAS files	Disk size [GB]	Area [km²]	Description
20M	20,165,862	1	0.4	1.25	TU Delft campus
210M	210,631,597	16	4.0	11.25	Major part of Delft city
2201M	2,201,135,689	153	42.0	125	City of Delft and surroundings
23090M	23,090,482,455	1,492	440.4	2,000	Major part of Zuid-Holland province
639478M	639,478,217,460	60,185	11,644.4	40,000	The Netherlands

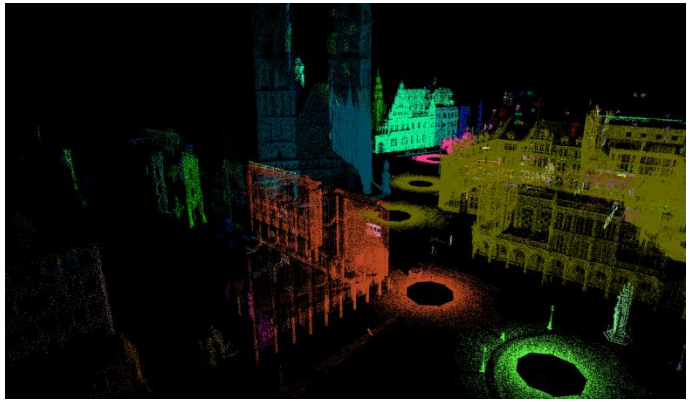


[1] AHN

Bremen Dataset

- Different clusterings of the inner city of Bremen
  - Using smart visualizations of the point cloud library (PCL)
  - Big Bremen (81 mio points) & sub sampled Small Bremen (3 mio points)





- **The Bremen Dataset is encoded in the HDF5 format (binary)**
- **You need your own copy of the file**

[2] Bremen Dataset

Bremen Dataset Available in B2SHARE

1. Go to https://b2share.eudat.eu/
2. Search for HPDBSCAN

[2] Bremen Dataset

HPDBSCAN Benchmark test files

by [Unknown]

Jan 13, 2017

Last updated at Jan 11, 2018

Abstract: Four data sets used for the benchmark of HPDBSCAN. Collection of geo-tagged Twitter data: This set has been collected and made available to us by Junjun Yin form the National Center for Supercomputing Application (NCSA). The dataset was obtained using the free twitter streaming API and contains exactly one percent of all geo-tagged of the United Kingdom in June 2014. It was initially created to investigate the possibility of mining peoples trajectories and to identify hotspots and points of interest (clusters of people) through monitoring tweet density. The full collections spans roughly 16.6 million tweets. A smaller subset of this was generated by filtering the entire set for the first week of June only. Oldtown of Bremen point cloud This data has been collected and made available by Dorit Borrmann and Andreas Nüchter from the Institute of Computer Science at the Jacobs University Bremen, Germany. It is a 3D point cloud of the oldtown of Bremen, Germany. A point cloud is a set of points and its represening coordinate system that often model the surface of objects. This particular point cloud of Bremen was recorded using a laser scanner system mounted onto an autonomous robotic vehicle. It has stopped at 11 different locations, performing each time a full 360° scan of the surrounding area. Given the GPS triangulized position and perspective of the camera the sub-point clouds where combined to one monolithic.

Keywords: dbscan; clustering; twitter; bremen; machine learning; unsupervised;

DOI: 10.23728/b2share.7f0c22ba9a5a44ca83cdf4fb304ce44e Copy
PID: 11304/7ee4b9c3-7ab5-423f-94d9-c3e920f656d6 Copy

Files

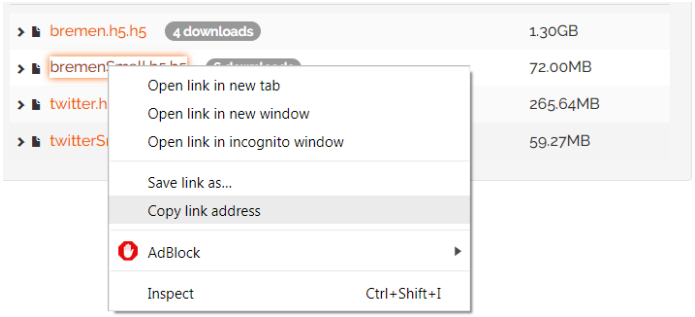
Name	Size
bremen.h5.h5 4 downloads	1.30GB
bremenSmall.h5.h5 6 downloads	72.00MB
twitter.h5.h5 4 downloads	265.64MB
twitterSmall.h5.h5 11 downloads	59.27MB

Basic metadata

Open Access	True ✔
License	
Contact Email	c.bodenstein@fz-juelich.de
Publication Date	2015-03-02
Contributors	
Resource Type	Category Other

Get your Own Copy

1. Copy the link address of the dataset



EUDAT

[2] Bremen Dataset

B2SHARE
Store and Share Research Data

2. Create your own folder in the project folder

$ mkdir /p/project/training1904/**your_username**/

3. Access your folder

$ cd /p/project/training1904/**your_username**/

4. Download the dataset

Paste the link address

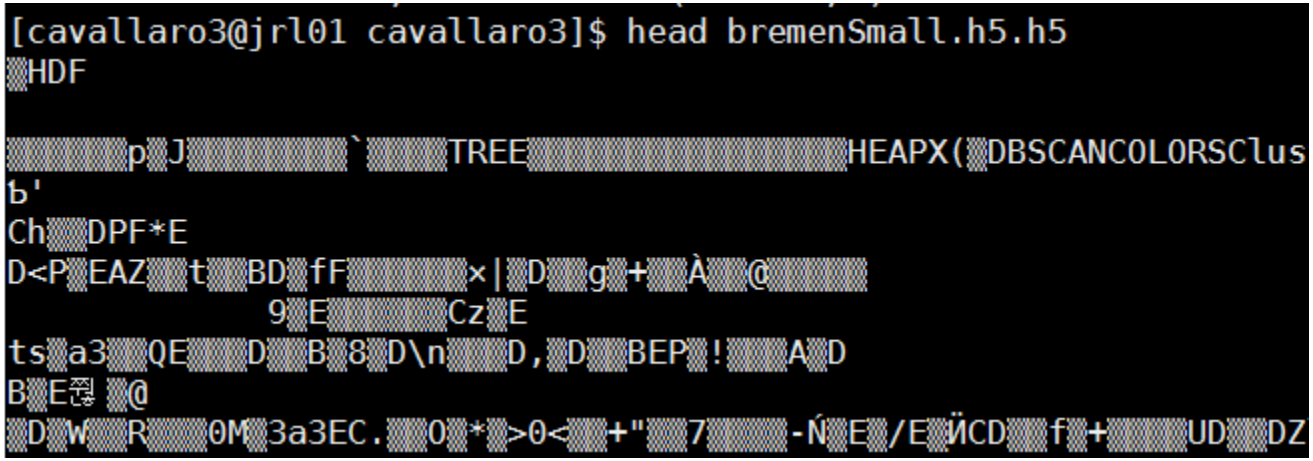
$ wget https://b2share.eudat.eu/api/files/189c8eaf-d596-462b-8a07-93b5922c4a9f/bremenSmall.h5.h5

```
[cavallaro3@jrl01 ~]$ mkdir /p/project/training1904/cavallaro3/
[cavallaro3@jrl01 ~]$ cd /p/project/training1904/cavallaro3/
[cavallaro3@jrl01 cavallaro3]$ wget https://b2share.eudat.eu/api/files/189c8eaf-d
--2019-02-21 09:43:57--  https://b2share.eudat.eu/api/files/189c8eaf-d596-462b-8a
Resolving b2share.eudat.eu (b2share.eudat.eu)... 86.50.166.74
Connecting to b2share.eudat.eu (b2share.eudat.eu)|86.50.166.74|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 72002416 (69M) [application/octet-stream]
Saving to: 'bremenSmall.h5.h5'

100%[=======================================================================

2019-02-21 09:43:58 (52.4 MB/s) - 'bremenSmall.h5.h5' saved [72002416/72002416]
```

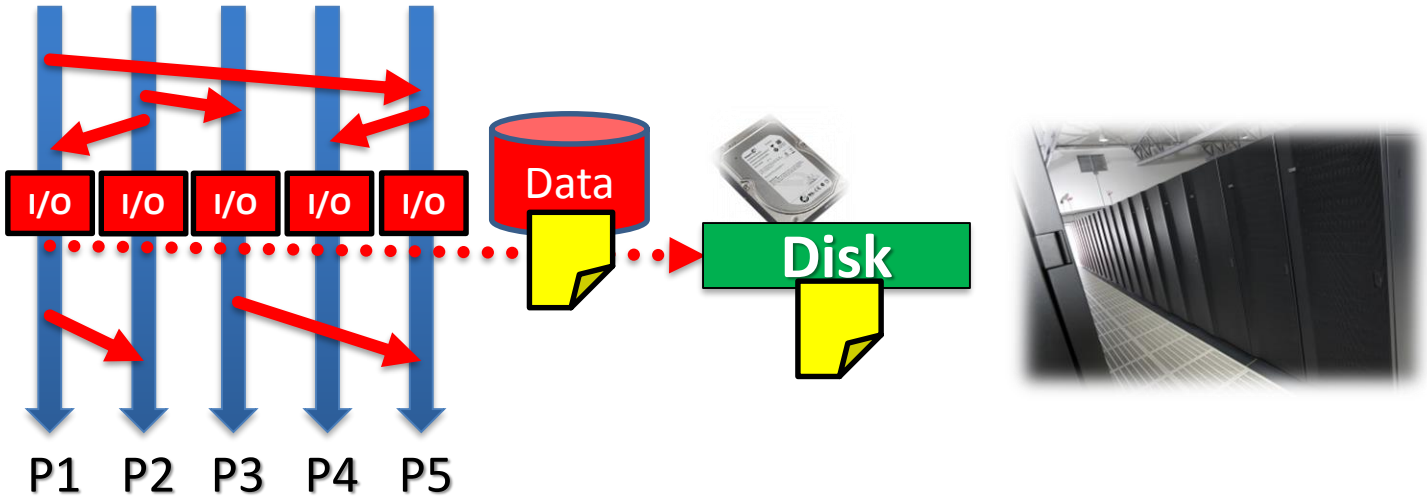
Exercises – Explore Bremen HDF5 Datasets (binary)

- **Notice binary content**

$ head /p/project/training1904/your_username/bremenSmall.h5



HDF5 – Parallel I/O: Shared file



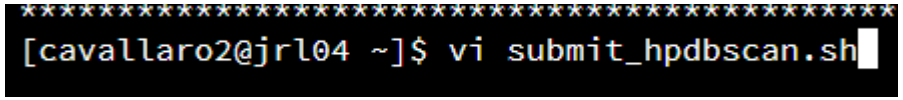
- Each process performs I/O to a single file
  - The file access is 'shared' across all processors involved
  - E.g. MPI/IO functions represent 'collective operations'
- Scalability and Performance
  - 'Data layout' within the shared file is crucial to the performance
  - High number of processors can still create 'contention' for file systems

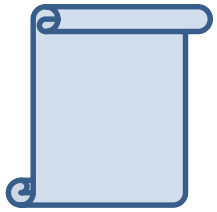
- **Parallel I/O: shared file means that processes can access their 'own portion' of a single file**
- **Parallel I/O with a shared file like MPI/IO is a scalable and even standardized solution**

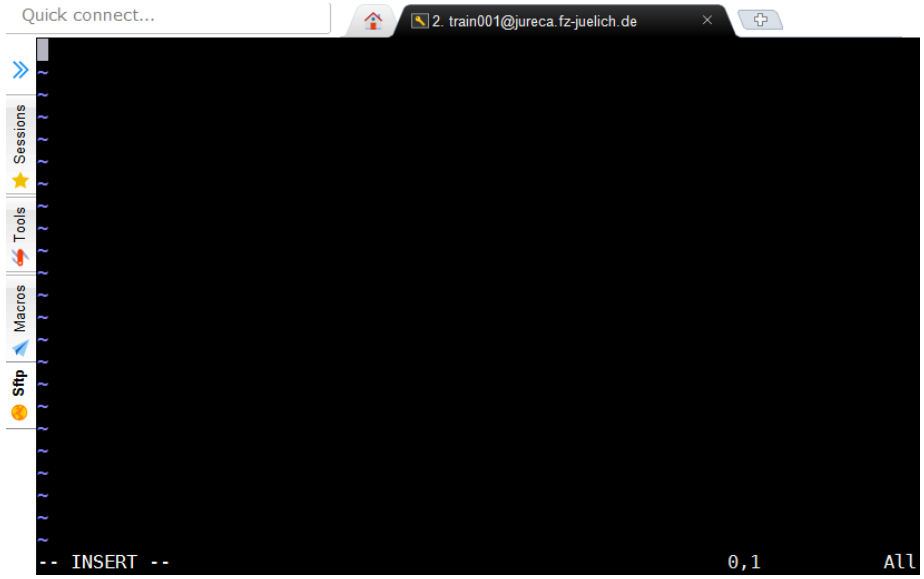
Job Script - File Creation with vi

- **vi** (visual editor): is the default editor that comes with the UNIX operating system
- It has two modes of operation:
  - **Command mode** commands: cause action to be taken on the file
  - **Insert mode:** entered text is inserted into the file

- Do the following steps:
  - **$ vi submit_hello_world.sh**
  - **Press i on your keyboard**







Create and Save your Job Script

```
#!/bin/bash
#SBATCH --job-name=HPDBSCAN
#SBATCH --output=HPDBSCAN-%j.out
#SBATCH --error=HPDBSCAN-%j.err
#SBATCH --mail-user=your_email

#SBATCH --time=01:00:00
#SBATCH--partition=batch
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=24
#SBATCH --account=training1904
#SBATCH --reservation=prace_ml_cpus_wed

export OMP_NUM_THREADS=${SLURM_CPUS_PER_TASK}
module load Stages/2018b
module load GCC/8.2.0 ParaStationMPI/5.2.1-1 HDF5/1.10.1

# location executable
HPDBSCAN=/p/project/training1904/.HPDBSCAN/source/dbscan

# location bremen dataset
DATASET=/p/project/training1904/your_username/bremenSmall.h5.h5

srun $HPDBSCAN -m 300 -e 500 -t 12 $DATASET
```

(parameters of DBSCAN and file to be clustered)

- **Job submit using command: sbatch <jobscript>**
- **Remember your <jobid> that is returned from the sbatch command**
- **Show status of the job then with: squeue –u <your-user-id>**
- **Reservations: prace_ml_cpus_wed**

JURECA HPC System – HPDBSCAN Job Submit

- Submit job via jobscript
  - $ sbatch submit_hpdbscan.sh
- Check job status (and cancel if needed)
  - $ squeue –u your_username
  - $ scancel jobid

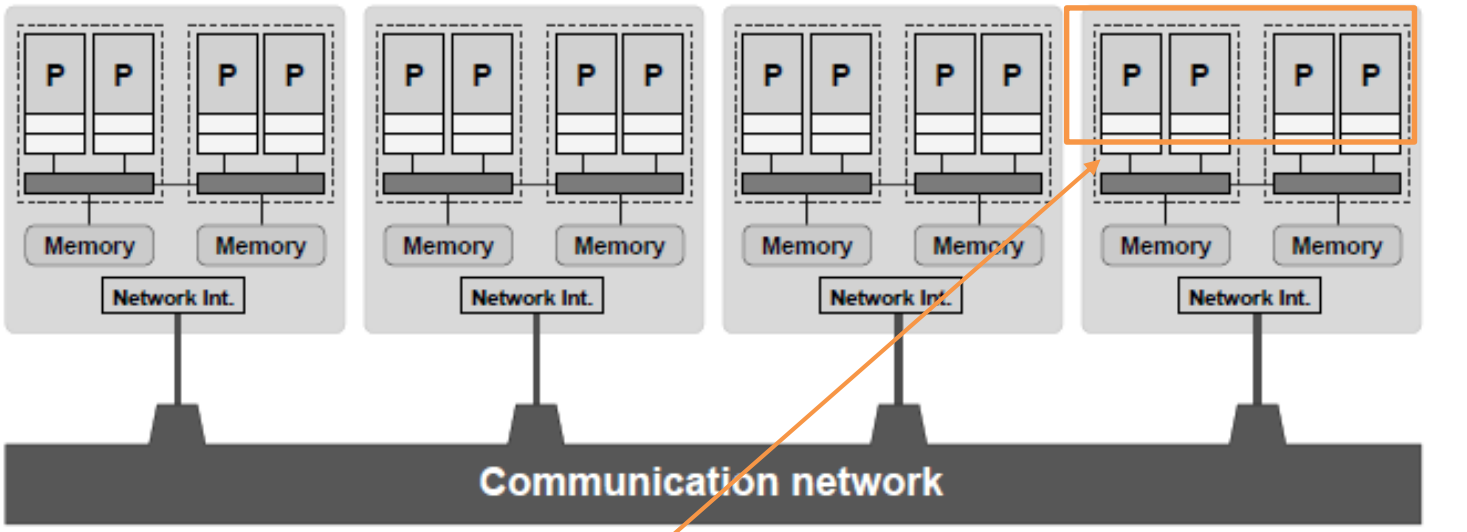
(scancel might take a second or two to take effect)

```
[cavallaro3@jrl12 ~]$ sbatch submit_hpdbscan.sh
Submitted batch job 6794501
[cavallaro3@jrl12 ~]$ squeue -u cavallaro3
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           6794501     devel HPDBSCAN cavallar CF       0:07      2 jrc[0039-0040]
[cavallaro3@jrl12 ~]$ squeue -u cavallaro3
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           6794501     devel HPDBSCAN cavallar  R       0:20      2 jrc[0039-0040]
[cavallaro3@jrl12 ~]$
```

HPDBSCAN - MPI + OpenMP

#SBATCH --nodes=N
#SBATCH --ntasks=N

number of MPI processes



#SBATCH --cpus-per-task=K

Number of CPUs that each MPI process can use

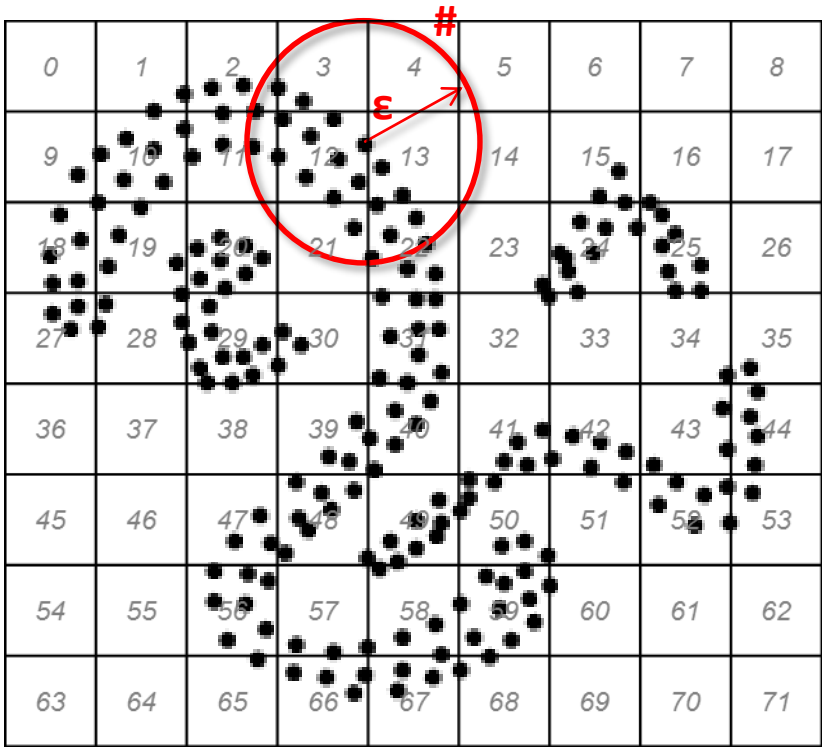
Review of Parallel DBSCAN Implementations

Technology	Platform Approach	Analysis
HPDBSCAN (authors implementation)	C; MPI; OpenMP	Parallel, hybrid, DBSCAN
Apache Mahout	Java; Hadoop	K-means variants, spectral, no DBSCAN
Apache Spark/MLlib	Java; Spark	Only k-means clustering, No DBSCAN
scikit-learn	Python	No parallelization strategy for DBSCAN
Northwestern University PDSDBSCAN-D	C++; MPI; OpenMP	Parallel DBSCAN

[3] M. Goetz, et al.

HDBSCAN Algorithm Details

- Parallelization Strategy
  - Smart ‘Big Data‘ Preprocessing into Spatial Cells (‘indexed‘)
  - OpenMP and HDF5 parallel I/O
  - MPI (+ optional OpenMP hybrid)
- Preprocessing Step
  - Spatial indexing and redistribution according to the point localities
  - Data density based chunking of computations

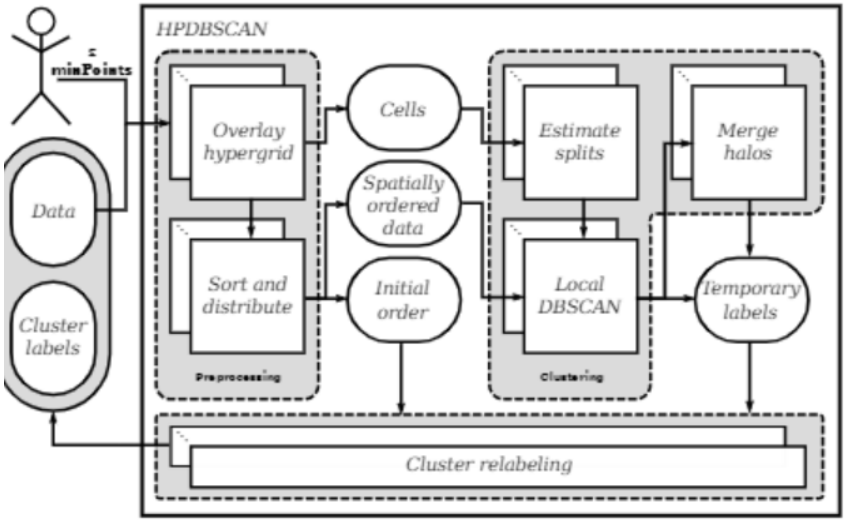


- Computational Optimizations
  - Caching of point neighborhood searches
  - Cluster merging based on comparisons instead of zone reclustering

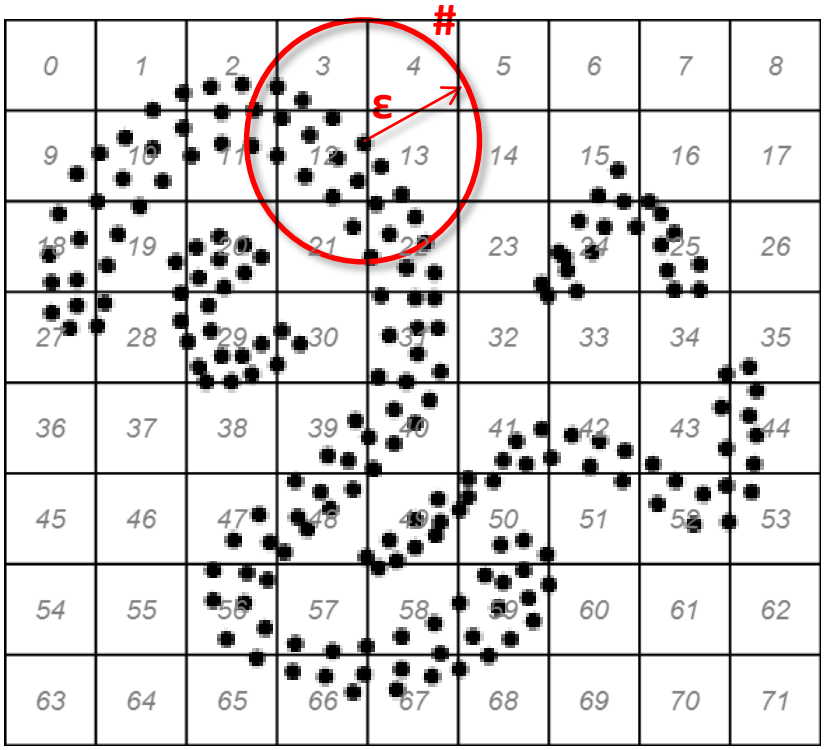
[4] M.Goetz, et al.

HPDBSCAN Algorithm

1. Load the entire dataset in equal-sized chunks by all processors in parallel
2. Assign the data to a unique spatial cell related to their location within the data space,
   - With respect to the given distance function
3. Perform the local clustering: assign a temporary cluster label to each of the data points
4. Generate cluster relabeling rules that decide if the temporary label assigned by a processing unit disagrees with the ones in the halo areas of the neighboring processors
   - The rules are **broadcasted (MPI)** and applied locally



[4] M.Goetz, et al.

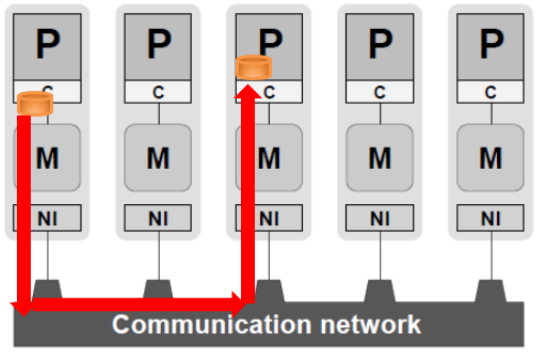


What is Message Passing Interface (MPI)?

- **'Communication library' abstracting from low-level network view**
  - Offers 500+ available functions to communicate between computing nodes
  - Practice reveals: parallel applications often require just ~12 (!) functions
  - Includes routines for efficient 'parallel I/O' (using underlying hardware)

- **Supports 'different ways of communication'**
  - 'Point-to-point communication' between two computing nodes (<- -> P)
  - Collective functions involve 'N computing nodes in useful communication'

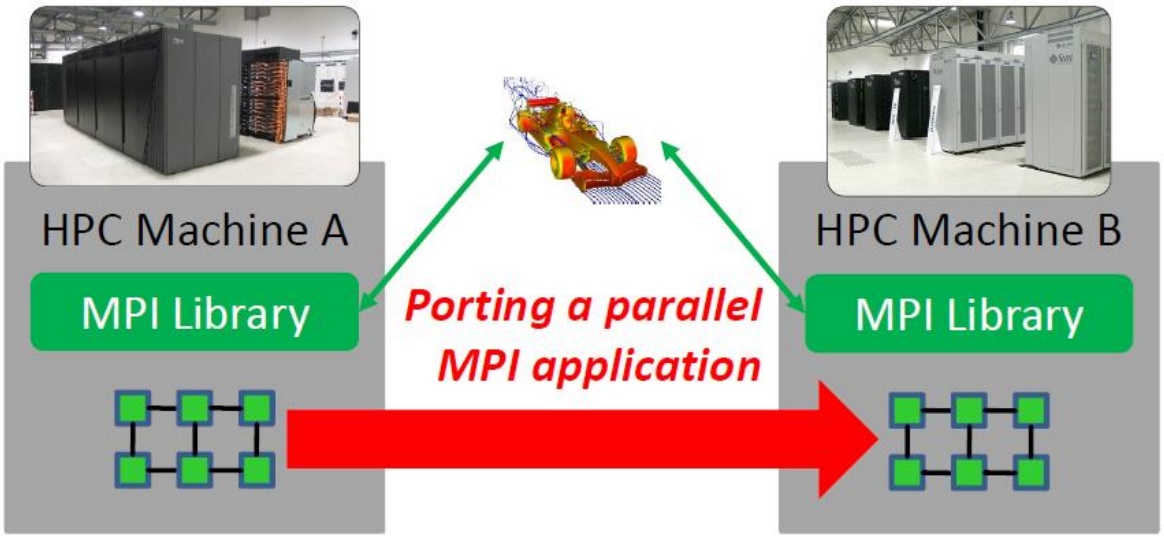
- **Deployment on Supercomputers**
  - Installed on (almost) all parallel computers
  - Different languages: C, Fortran, Python, R, etc.
  - Careful: different versions exist



Recall 'computing nodes' are independent computing processors (that may also have N cores each) and that are all part of one big parallel computer

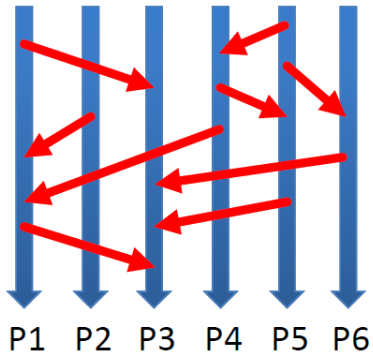
Key Keatures of MPI

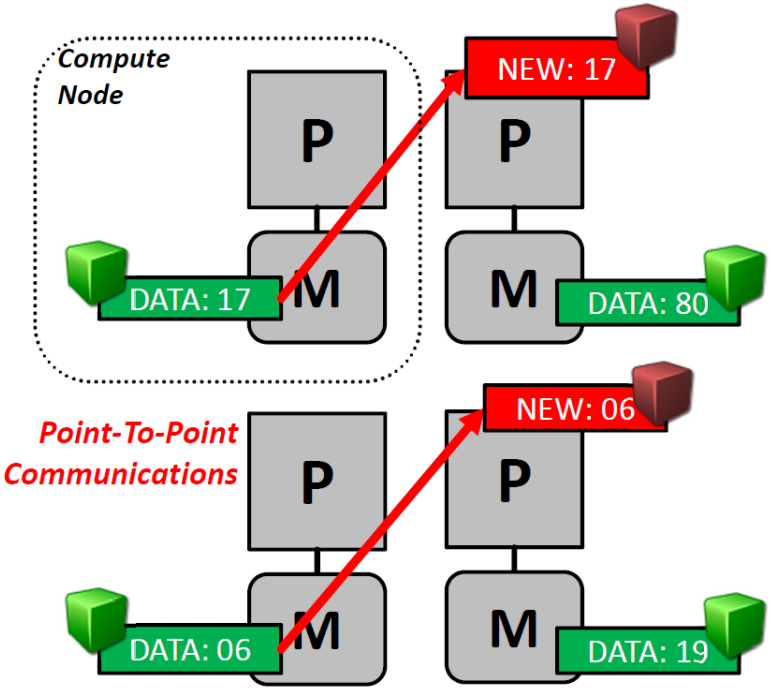
- Simplify programming in parallel programming, **focus on applications**
- It is not designed to handle any communication in computer networks
- Designed for performance within large parallel computers (e.g. no security)
- Several open-source well-tested implementations of MPI
- It enables portability of parallel applications



Message Passing: Exchanging Data

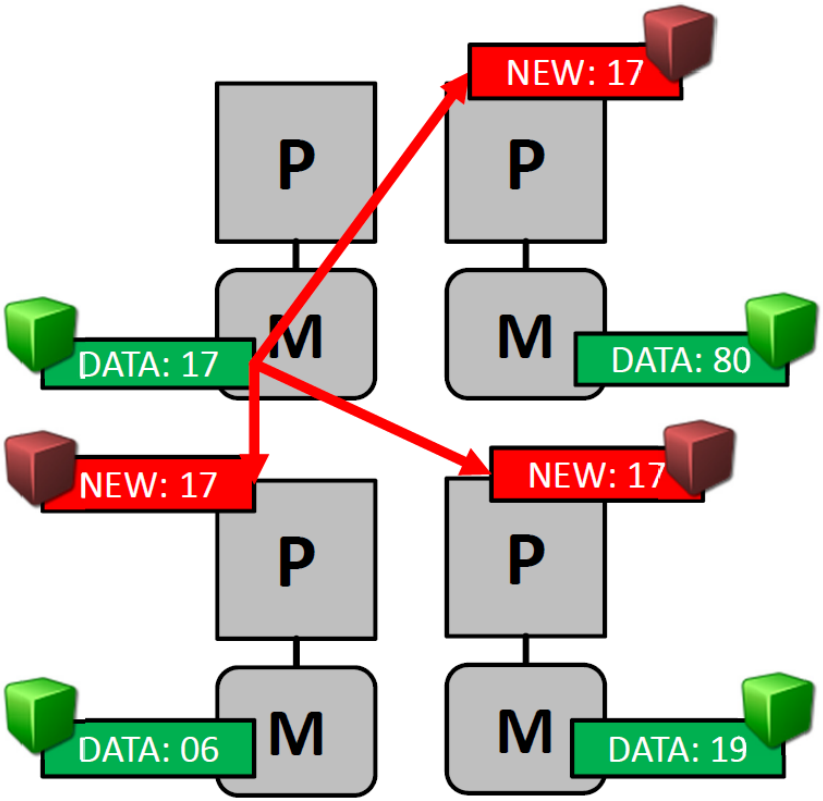
- Each processor has its own data and memory that cannot be accessed by other processors





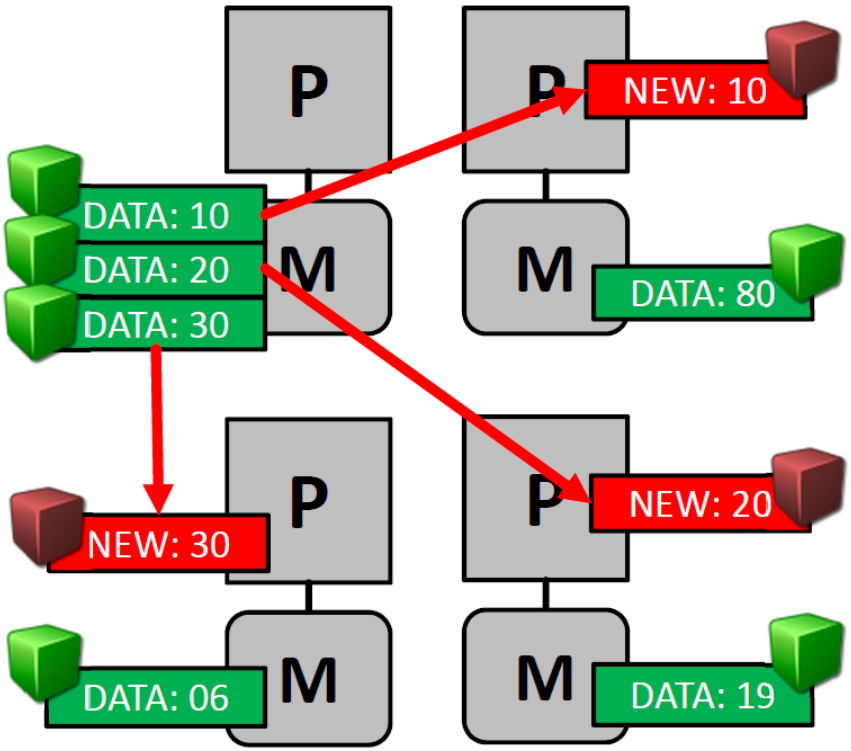
Collective Functions: Broadcast (one-to-many)

- Broadcast distributes the **same** data to many or even all other processors



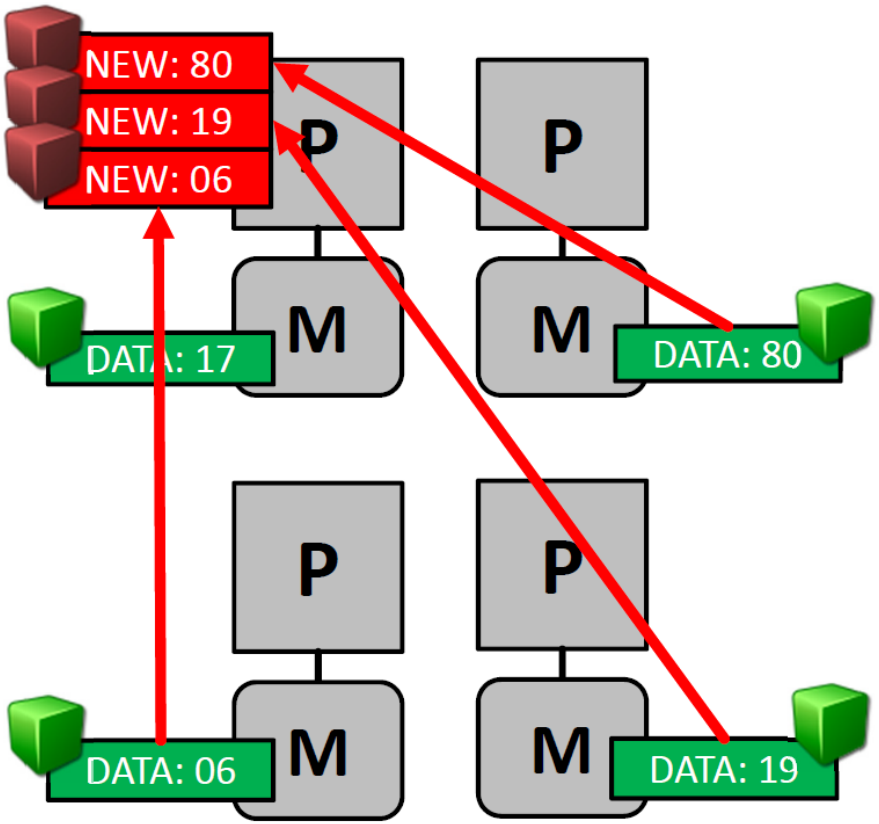
Collective Functions: Scatter (one-to-many)

- Scatter distributes **different** data to many or even all other processors



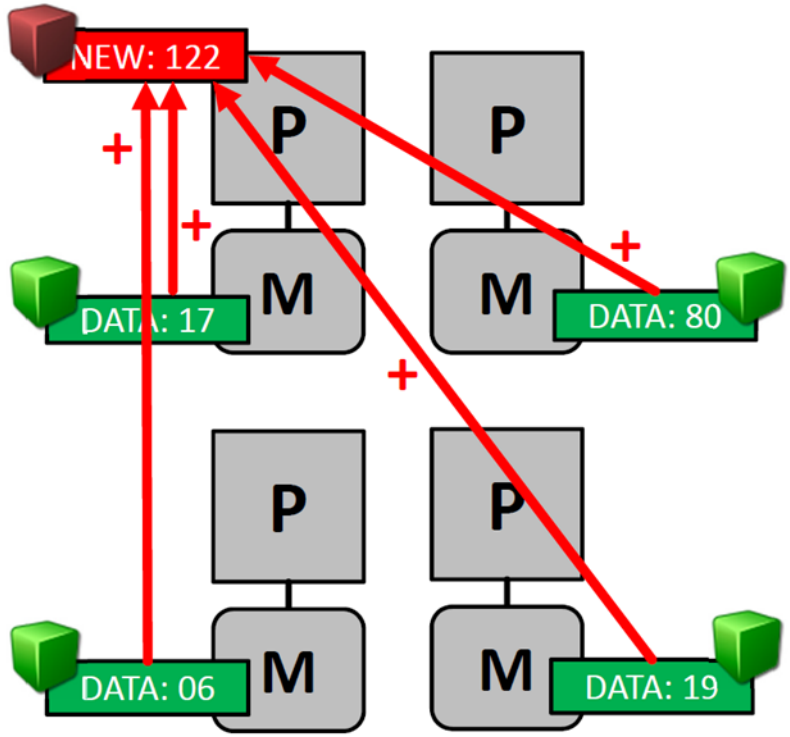
Collective Functions: Gather (many-to-one)

- Gather **collects** data from many or even all other processors to one specific processor



Collective Functions: Reduce (many-to-one)

- Each Reduce combines collection with computation based on data from many or even all other processors
- Usage of reduce includes finding a **global minimum or maximum, sum, or product** of the different data located at different processors



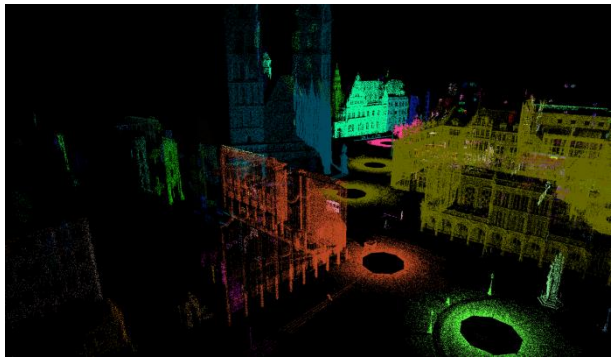
+ global sum as example

JURECA HPC System – HPDBSCAN Check Outcome

```
Calculating Cell Space...
        Computing Dimensions... [OK] in 0.001823
        Computing Cells...      [OK] in 0.018203
        Sorting Points...       [OK] in 0.214498
        Distributing Points...  [OK] in 0.126684
DBSCAN...
        Local Scan...           [OK] in 102.478086
        Merging Neighbors...    [OK] in 0.006435
        Adjust Labels ...       [OK] in 0.006754
        Rec. Init. Order ...    [OK] in 0.939040
        Writing File ...        [OK] in 0.019757
Result...
        21      Clusters
        2974394 Cluster Points
        25606   Noise Points
        2949094 Core Points
Took: 104.192540s
```

```
[cavallaro3@jrl12 ~]$ ls
HPDBSCAN-6794501.err  HPDBSCAN-6794501.out
[cavallaro3@jrl12 ~]$
```

- **The outcome of the clustering process is written directly into the HDF5 file using cluster IDs and noise IDs**

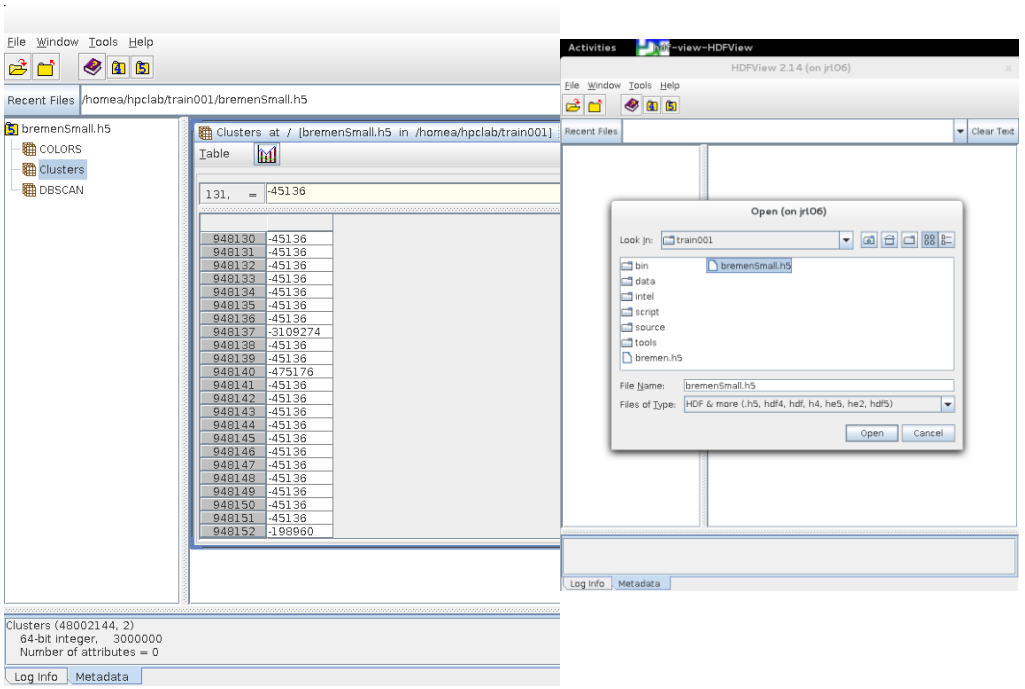


HDFView Example – Bremen Output

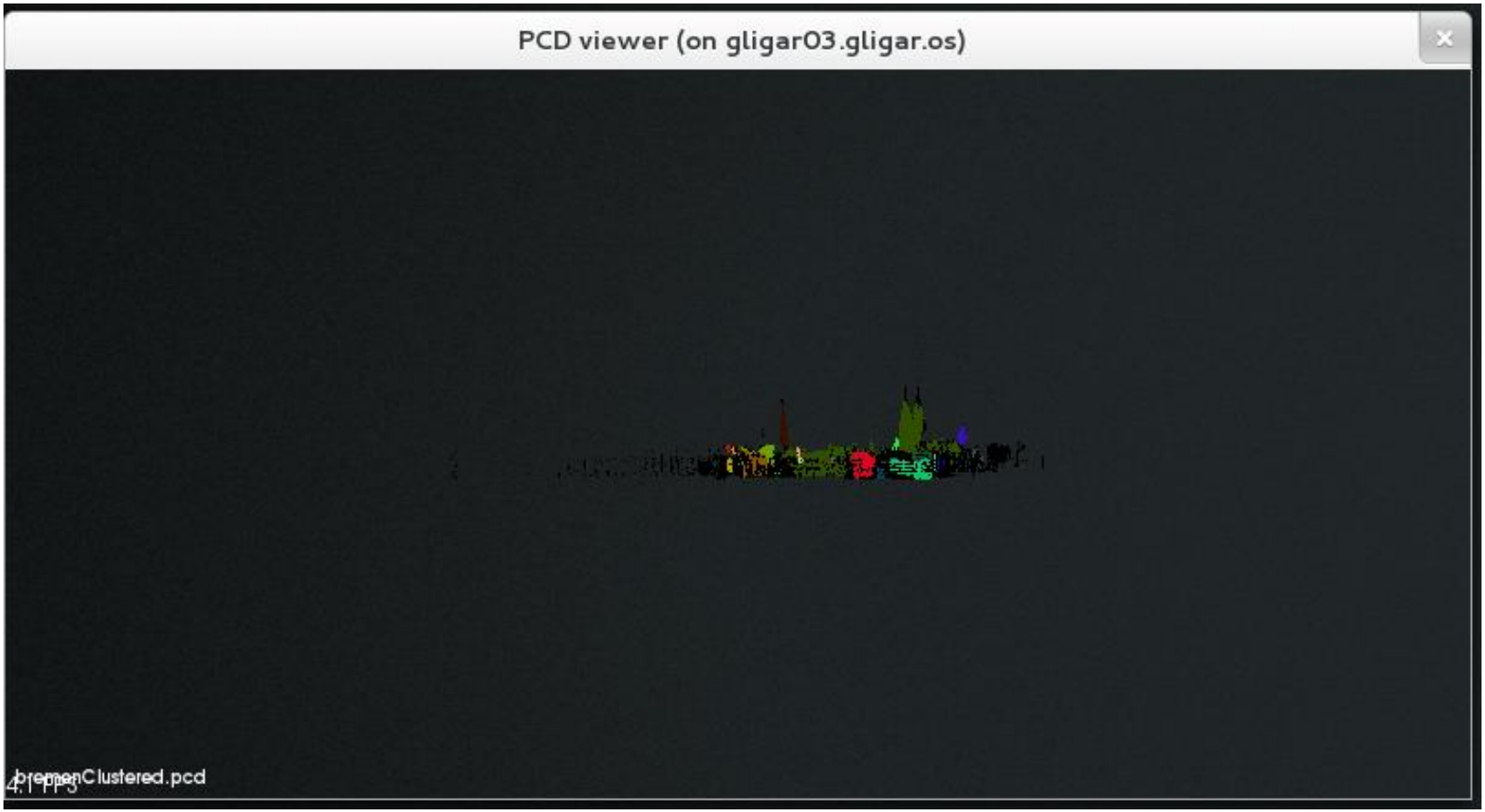
- HDFView is a visual tool for browsing and editing HDF files
  - Tools is using a GUI thus needs ssh –X when log into JURECA

```
[cavallaro1@jrl09 ~]$ module use /usr/local/software/jureca/OtherStages
[cavallaro1@jrl09 ~]$ module load Stages/2018b
       Preparing the environment for use of requested stage ( 2018b ).

[cavallaro1@jrl09 ~]$ module load HDFView/2.14
[cavallaro1@jrl09 ~]$
```

- DBSCAN folder: input data,
- Clusters folder: cluster labels
  - Which cluster each point belongs to (not consecutive)
- COLORS folder : please ignore it



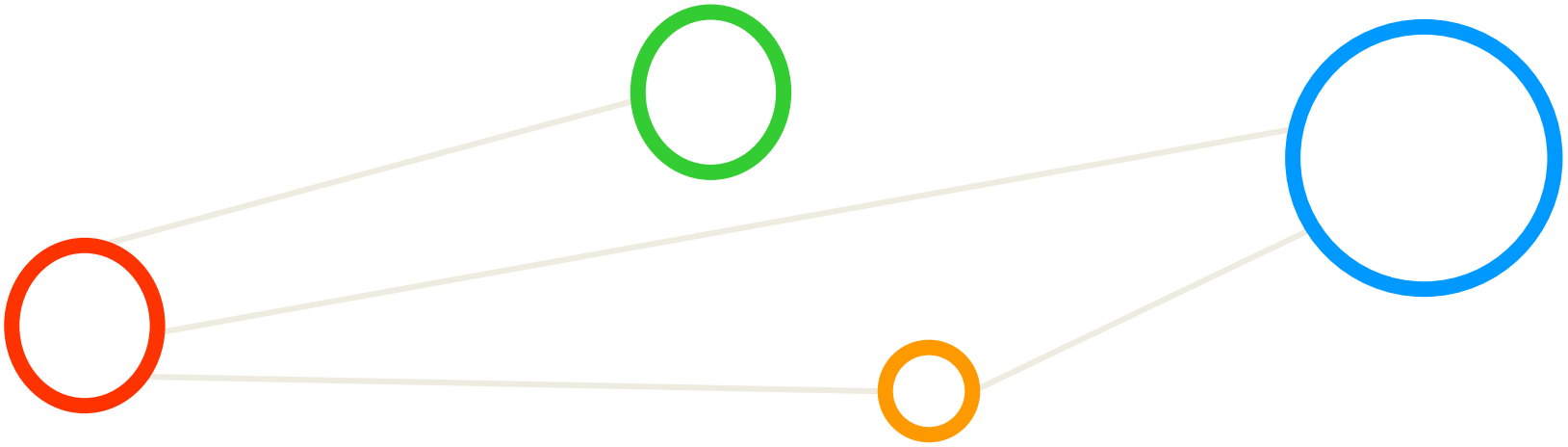
Point Cloud Viewer Example – Bremen Output



[5] Point Cloud Library (PCL)

- **Use Strg and Mouse Wheel to Zoom and use numbers of keyboard for different visualizations**

Lecture Bibliography



Lecture Bibliography (1)

- [1] Actueel Hoogtebestand Nederland (AHN)
  Online: http://www.ahn.nl/index.html
- [2] B2SHARE, 'HPDBSCAN Benchmark test files',
  Online: https://b2share.eudat.eu/records/7f0c22ba9a5a44ca83cdf4fb304ce44e
- [3] M. Goetz, M. Riedel et al., 'On Parallel and Scalable Classification and Clustering Techniques for Earth Science Datasets', 6th Workshop on Data Mining in Earth System Science, International Conference of Computational Science (ICCS), 2015
- [4] M.Goetz, M. Riedel et al., '*HPDBSCAN – Highly Parallel DBSCAN*', MLHPC Workshop at Supercomputing 2015
- [5] Point Cloud Library (PCL)
  Online: http://pointclouds.org/about/

Slides Available at http://www.morrisriedel.de/talks