Scaling geospatial processes on Harvard's high-performance cluster

Devika Kakkar and Ben Lewis

Harvard Center for Geographic Analysis

Datafest 2020

Harvard University, January 21st

“Big” geospatial data: even a million features

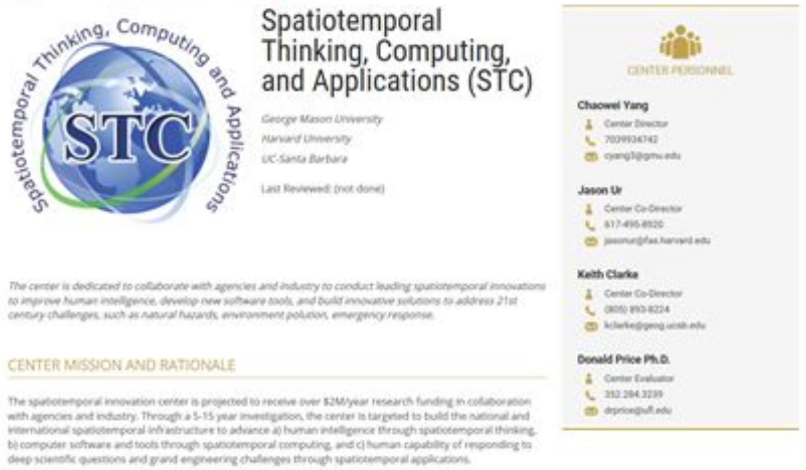
- Most sets of geographic features are modest: thousands to millions in size. But...
- Increasing spatial resolution is changing this: e.g. National Hydro Datasets Medium Res -> ~3M reaches, High Res -> ~30M reaches. Similar for gridded data, e.g. 10m DEM -> 1m Lidar-based 3DEP increases volume 100x.
- Time is changing this: multiple observations and predictions for multiple feature properties quickly combine into billions of records.
- Traditional GIS software struggles to access and visualize, let alone analyze such scales and structures of datasets.
- Datasets with 1-100 billion records are becoming common in academic, business, and government domains

FAS Research Computing (FASRC)

- FAS Research Computing offers:
    - Supercomputing Environment (Cannon HPC Cluster)
    - Lab Storage
    - Instrument Computing Support
    - Hosted Machines (Virtual and Physical)

I/UCRC Spatio-temporal Consortium

- George Mason University
- University of California Santa Barbara
- Harvard University



Spatiotemporal Thinking, Computing, and Applications (STC)

George Mason University

Harvard University

UC-Santa Barbara

Last Reviewed: (not done)

The center is dedicated to collaborate with agencies and industry to conduct leading spatiotemporal innovations to improve human intelligence, develop new software tools, and build innovative solutions to address 21st century challenges, such as natural hazards, environment polution, emergency response.

CENTER MISSION AND RATIONALE

The spatiotemporal innovation center is projected to receive over $2M/year research funding in collaboration with agencies and industry. Through a 5-15 year investigation, the center is targeted to build the national and international spatiotemporal infrastructure to advance a) human intelligence through spatiotemporal thinking, b) computer software and tools through spatiotemporal computing, and c) human capability of responding to deep scientific questions and grand engineering challenges through spatiotemporal applications.

CENTER PERSONNEL

Chaowei Yang
Center Director
7039934742
cyang3@gmu.edu

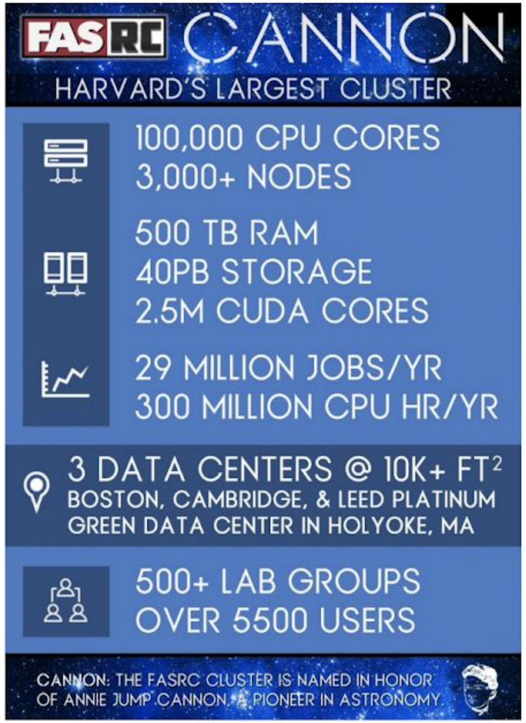
Jason Ur
Center Co-Director
617-495-8920
jasonur@fas.harvard.edu

Keith Clarke
Center Co-Director
(805) 893-8224
kclarke@geog.ucsb.edu

Donald Price Ph.D.
Center Evaluator
352.284.3239
drprice@ufl.edu

Supercomputing Environment: Cannon [1]

- Compute: 100,000 compute nodes, 8-64 cores/node, 12Gb to 512Gb memory/node, 2,500,000 NVIDIA GPU cores
- Software: CentOS 7 operating system, Slurm job manager, Singularity, 1000+ scientific tools and programs
- Storage: 100 GB (Home dir), 4TB+ (Lab storage), 70Gb/node (Local scratch), 2.4PB (Global scratch), 3PB (Persistent Research data)
- #144 in TOP500 Supercomputers in world



FASRC CANNON

HARVARD'S LARGEST CLUSTER

100,000 CPU CORES
3,000+ NODES

500 TB RAM
40PB STORAGE
2.5M CUDA CORES

29 MILLION JOBS/YR
300 MILLION CPU HR/YR

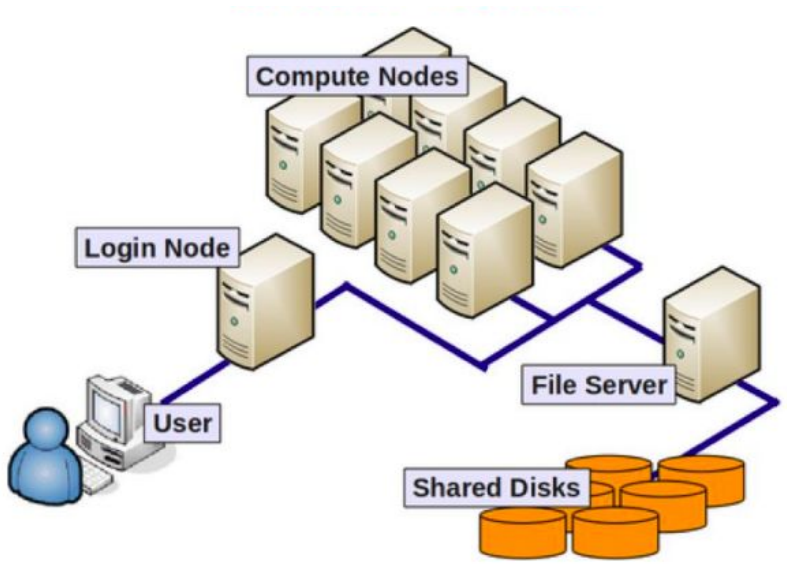
3 DATA CENTERS @ 10K+ FT²
BOSTON, CAMBRIDGE, & LEED PLATINUM GREEN DATA CENTER IN HOLYOKE, MA

500+ LAB GROUPS
OVER 5500 USERS

CANNON: THE FASRC CLUSTER IS NAMED IN HONOR OF ANNIE JUMP CANNON, A PIONEER IN ASTRONOMY.

Cannon Cluster [1]

Cluster Basics



Basic Architecture of FASRC Cluster [1]

Slurm Workload Manager

- Slurm is an open-source cluster management and job scheduling system for Linux clusters
- Slurm is the workload manager on about 60% of the TOP500 supercomputers.
- It performs the following key functions:
  - Allocates resources
  - Provides a framework for managing jobs
  - Resolves conflicts for resources
- FASRC uses Slurm to manage workload on the Cannon cluster

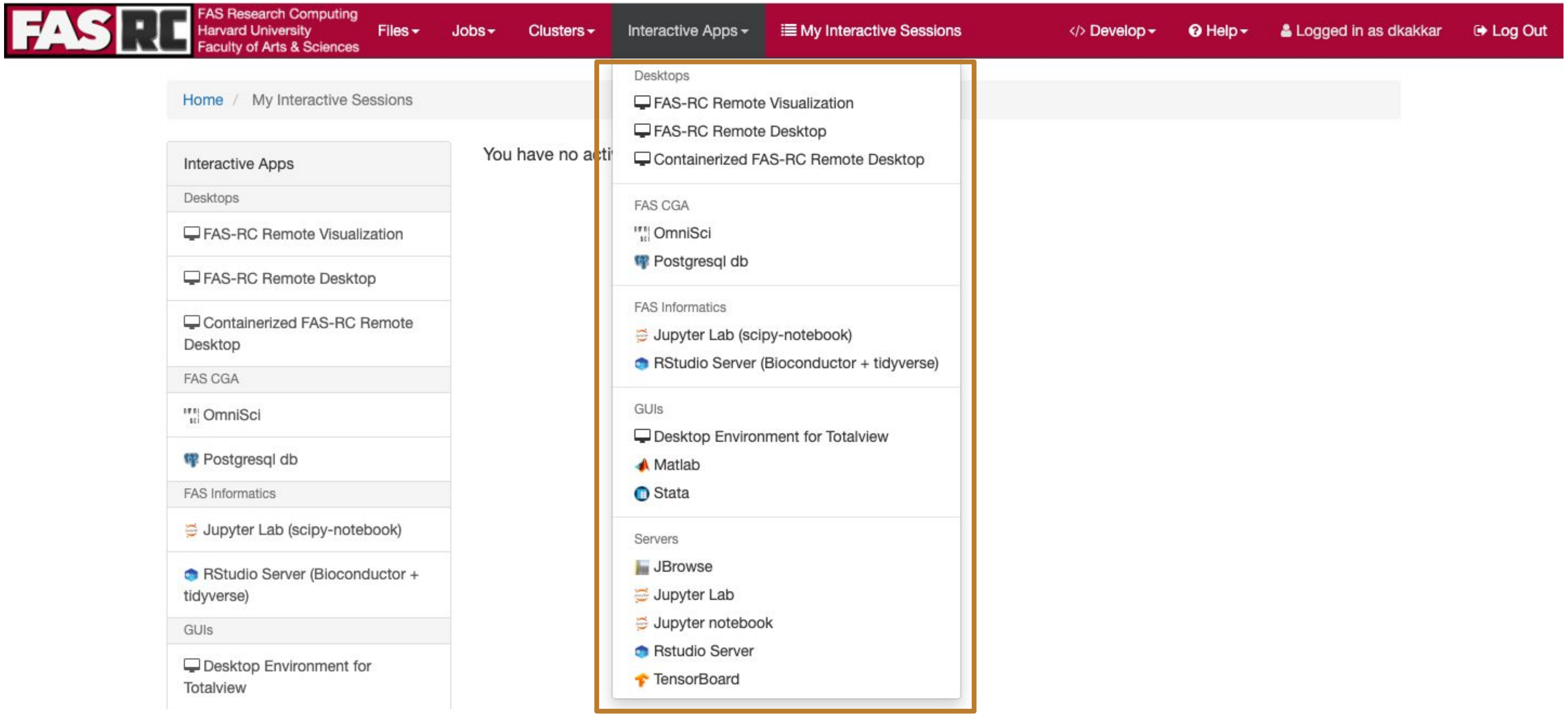
Slurm scheduler

- FASRC uses Slurm built-in job accounting and fairshare system to ensure that resources are used fairly
- Every lab has a base Share of the community-wide system
- Fairshare score of a lab is then calculated based off of their Share versus the amount of the cluster they have actually used
- Fairshare score is then utilized to assign priority to their jobs relative to other users on the cluster
- TRES: allows the scheduler to charge back users for how much they have used different features on the cluster
- sshare: A tool that can be used to see your current fairshare

Singularity on the cluster

- Singularity is a free, cross-platform and open-source technology used for containerization of workloads
- As of 2018, the Singularity user base is estimated to be greater than 25,000 installations and includes users at academic institutions such as Ohio State University, and Michigan State University, as well as top HPC centers like Texas Advanced Computing Center, San Diego Supercomputer Center, and Oak Ridge National Laboratory.
- Singularity has been deployed on FASRC cluster
- Singularity enables users to have full control of their operating system environment
- FASRC uses Singularity instead of Docker for security reasons
- Singularity can import Docker containers

Data Science apps on FASRC



FAS Research Computing
Harvard University
Faculty of Arts & Sciences

Files Jobs Clusters Interactive Apps My Interactive Sessions Develop Help Logged in as dkakkar Log Out

Home / My Interactive Sessions

You have no acti

Interactive Apps

Desktops
- FAS-RC Remote Visualization
- FAS-RC Remote Desktop
- Containerized FAS-RC Remote Desktop

FAS CGA
- OmniSci
- Postgresql db

FAS Informatics
- Jupyter Lab (scipy-notebook)
- RStudio Server (Bioconductor + tidyverse)

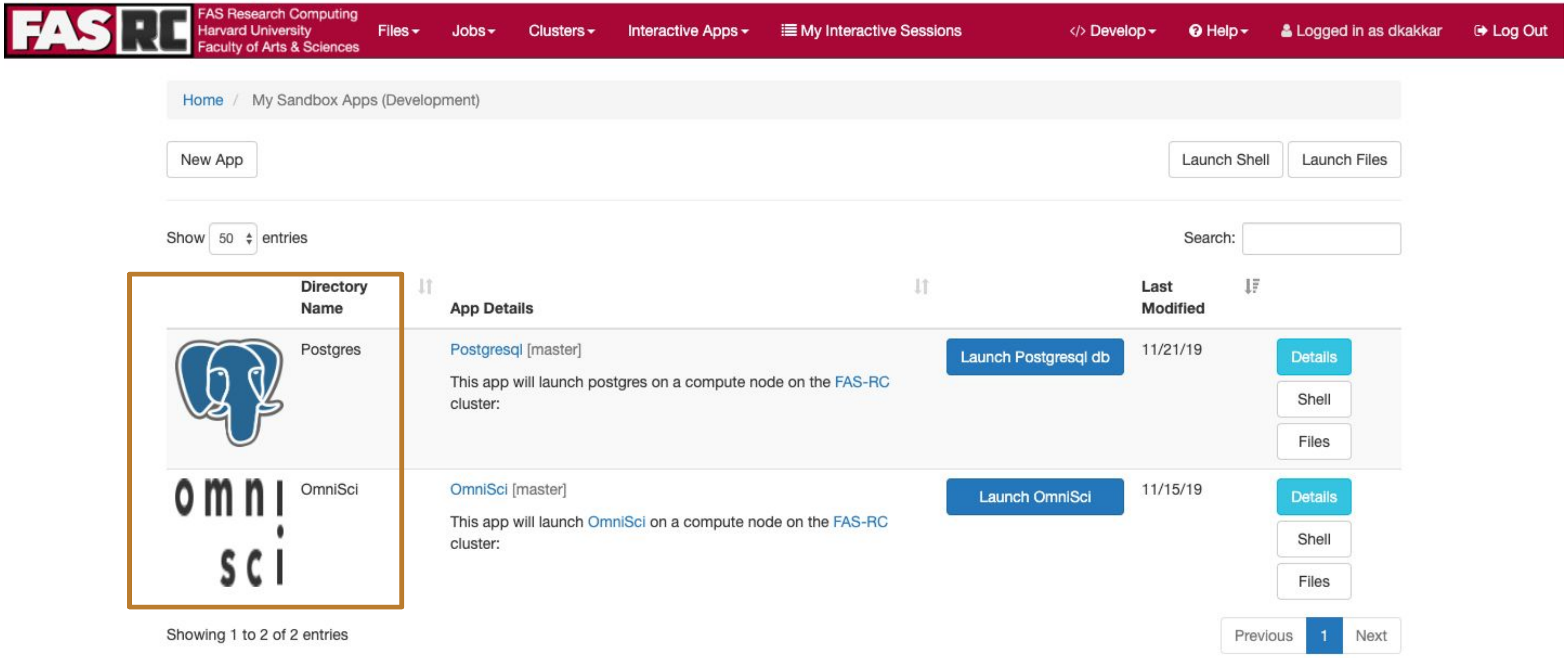
GUIs
- Desktop Environment for Totalview
- Matlab
- Stata

Servers
- JBrowse
- Jupyter Lab
- Jupyter notebook
- Rstudio Server
- TensorBoard

GIS Databases for Big Data

- **PostGreSQL**: Powerful, open source object-relational database system
- **PostGIS**: Provides spatial objects for the PostgreSQL database, allowing storage and query of information about location and mapping
- **OmniSci**:
  - Designed to overcome the scalability and performance limitations of legacy analytics tools
  - Super fast queries/analytics (including machine learning) of unindexed data (open source)
  - Super fast interactive rendering (free for educational use) of millions or billions of features, on-the-fly on a map
  - Leverages the massively parallel processing of GPUs alongside traditional CPU compute

GIS apps that will be available in the future



Home / My Sandbox Apps (Development)

Show 50 entries

Search:

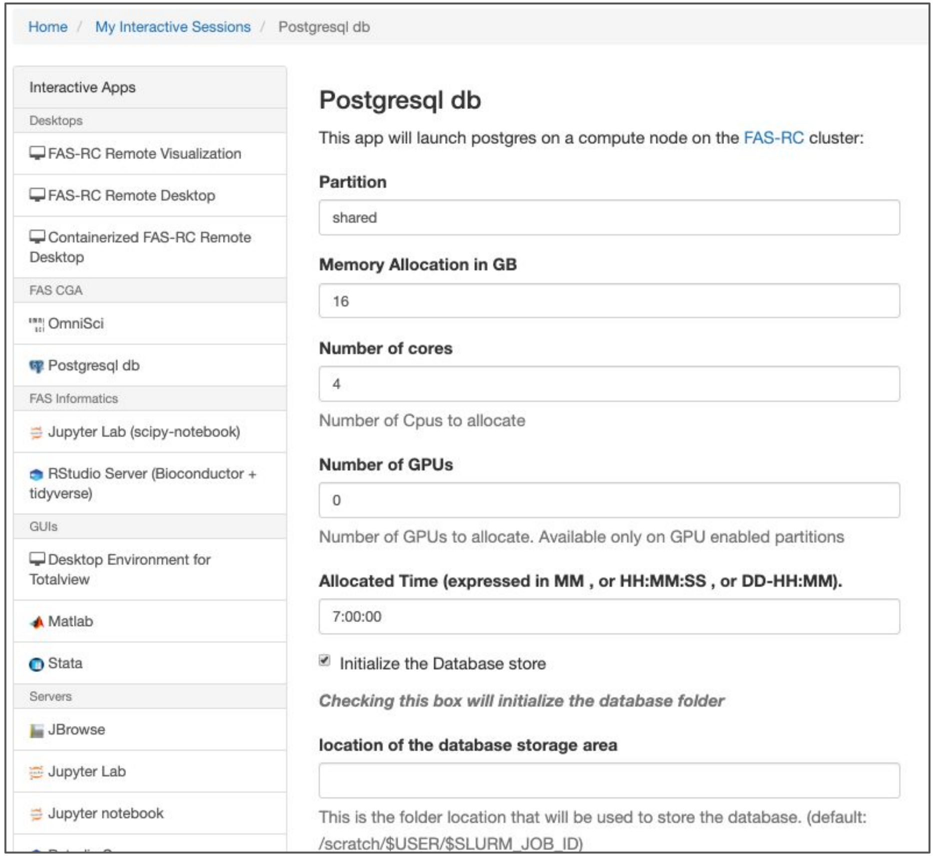
Directory Name	App Details		Last Modified	
Postgres	Postgresql [master] This app will launch postgres on a compute node on the FAS-RC cluster:	Launch Postgresql db	11/21/19	Details Shell Files
OmniSci	OmniSci [master] This app will launch OmniSci on a compute node on the FAS-RC cluster:	Launch OmniSci	11/15/19	Details Shell Files

Showing 1 to 2 of 2 entries

GIS Apps on FASRC

- An app on FASRC remote desktop vdi.rc.fas.harvard.edu to run the server
- Code for the app will be in user home folder
- The apps in the personal ~/fasrc/dev/ folder will show up in the "Develop" menu and under "My Interactive Sessions" once you login to vdi.rc.fas.harvard.edu
- One can launch the app, and upon successful start of the job, on the dashboard you will get info on how to connect
- The location where the job is staged and the log is stored is reachable clicking next to "session ID"
- To make changes in that app,create a branch, make changes and we can then merge it
- Github repo: https://github.com/cga-harvard/OmniSci_on_HPC

POSTGRES on FASRC



Home / My Interactive Sessions / Postgresql db

Interactive Apps
Desktops
FAS-RC Remote Visualization
FAS-RC Remote Desktop
Containerized FAS-RC Remote Desktop
FAS CGA
OmniSci
Postgresql db
FAS Informatics
Jupyter Lab (scipy-notebook)
RStudio Server (Bioconductor + tidyverse)
GUIs
Desktop Environment for Totalview
Matlab
Stata
Servers
JBrowse
Jupyter Lab
Jupyter notebook

Postgresql db

This app will launch postgres on a compute node on the FAS-RC cluster:

Partition

shared

Memory Allocation in GB

16

Number of cores

4

Number of Cpus to allocate

Number of GPUs

0

Number of GPUs to allocate. Available only on GPU enabled partitions

Allocated Time (expressed in MM , or HH:MM:SS , or DD-HH:MM).

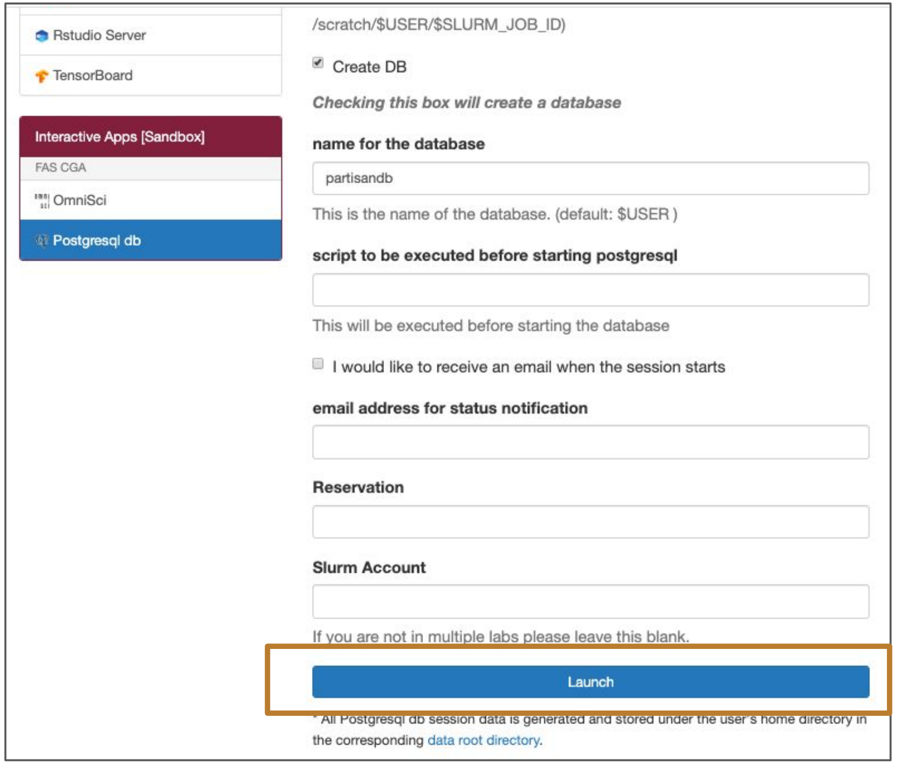
7:00:00

☑ Initialize the Database store

Checking this box will initialize the database folder

location of the database storage area

This is the folder location that will be used to store the database. (default: /scratch/$USER/$SLURM_JOB_ID)



/scratch/$USER/$SLURM_JOB_ID)

☑ Create DB

Checking this box will create a database

name for the database

partisandb

This is the name of the database. (default: $USER)

script to be executed before starting postgresql

This will be executed before starting the database

☐ I would like to receive an email when the session starts

email address for status notification

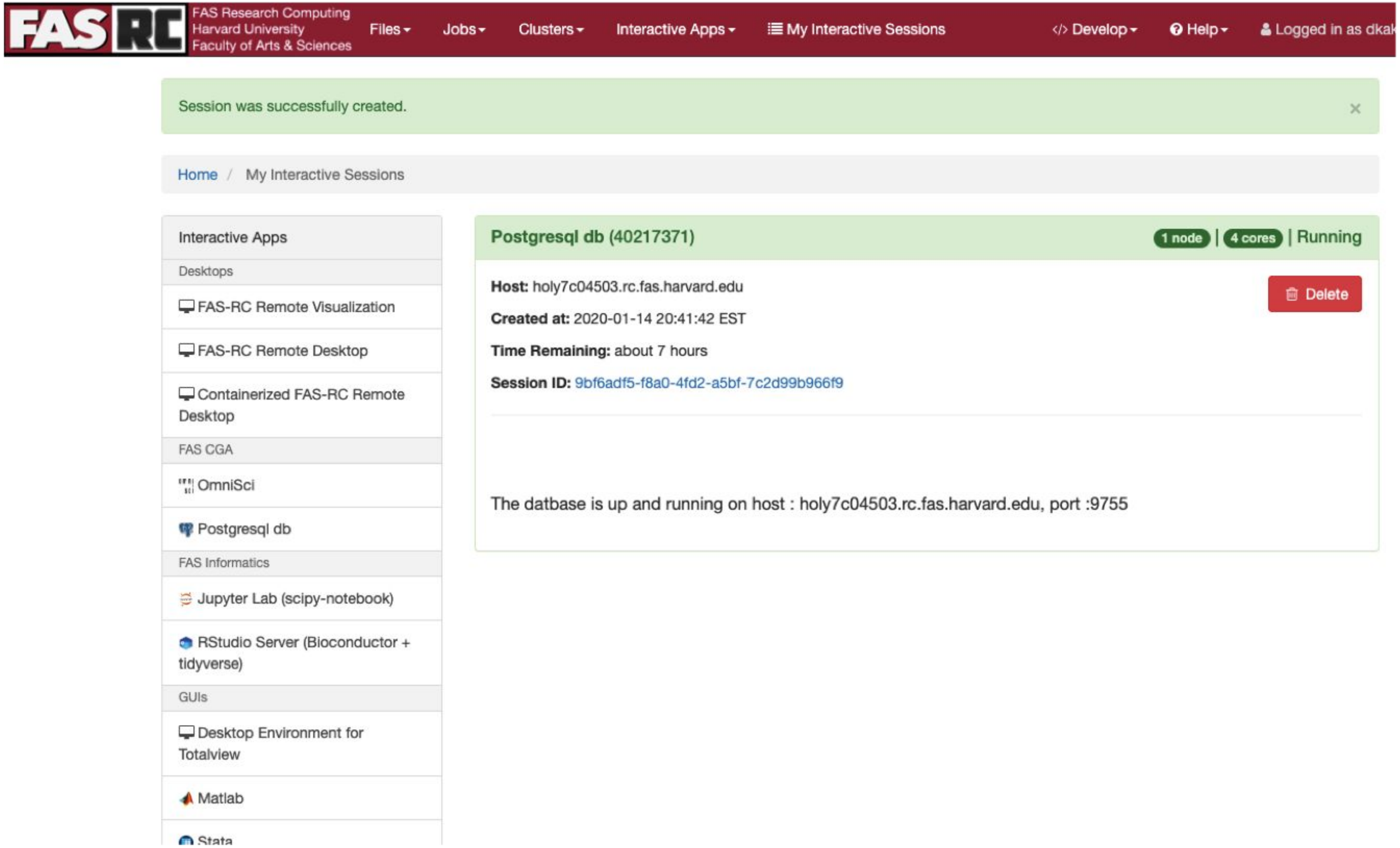
Reservation

Slurm Account

If you are not in multiple labs please leave this blank.

Launch

* All Postgresql db session data is generated and stored under the user's home directory in the corresponding data root directory.



Session was successfully created.

Home / My Interactive Sessions

Interactive Apps
Desktops
FAS-RC Remote Visualization
FAS-RC Remote Desktop
Containerized FAS-RC Remote Desktop
FAS CGA
OmniSci
Postgresql db
FAS Informatics
Jupyter Lab (scipy-notebook)
RStudio Server (Bioconductor + tidyverse)
GUIs
Desktop Environment for Totalview
Matlab
Stata

Postgresql db (40217371) | 1 node | 4 cores | Running

Host: holy7c04503.rc.fas.harvard.edu

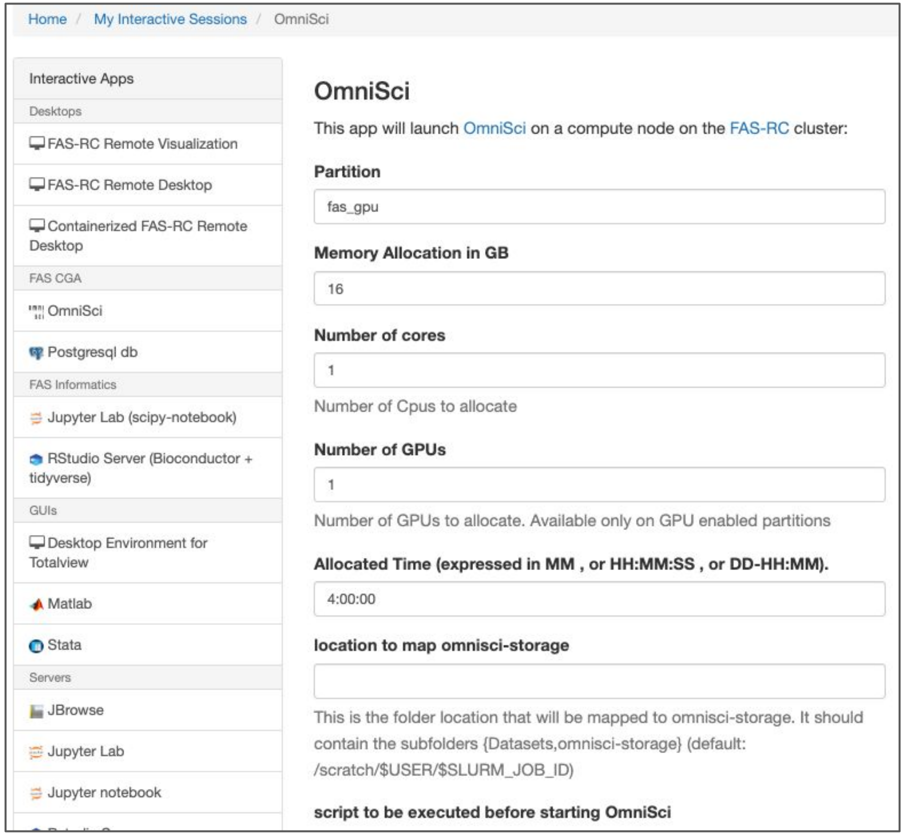
Created at: 2020-01-14 20:41:42 EST

Time Remaining: about 7 hours

Session ID: 9bf6adf5-f8a0-4fd2-a5bf-7c2d99b966f9

The datbase is up and running on host : holy7c04503.rc.fas.harvard.edu, port :9755

OmniSci on FASRC



Home / My Interactive Sessions / OmniSci

Interactive Apps

Desktops

- FAS-RC Remote Visualization
- FAS-RC Remote Desktop
- Containerized FAS-RC Remote Desktop

FAS CGA

- OmniSci
- Postgresql db

FAS Informatics

- Jupyter Lab (scipy-notebook)
- RStudio Server (Bioconductor + tidyverse)

GUIs

- Desktop Environment for Totalview
- Matlab
- Stata

Servers

- JBrowse
- Jupyter Lab
- Jupyter notebook

OmniSci

This app will launch OmniSci on a compute node on the FAS-RC cluster:

Partition

fas_gpu

Memory Allocation in GB

16

Number of cores

1

Number of Cpus to allocate

Number of GPUs

1

Number of GPUs to allocate. Available only on GPU enabled partitions

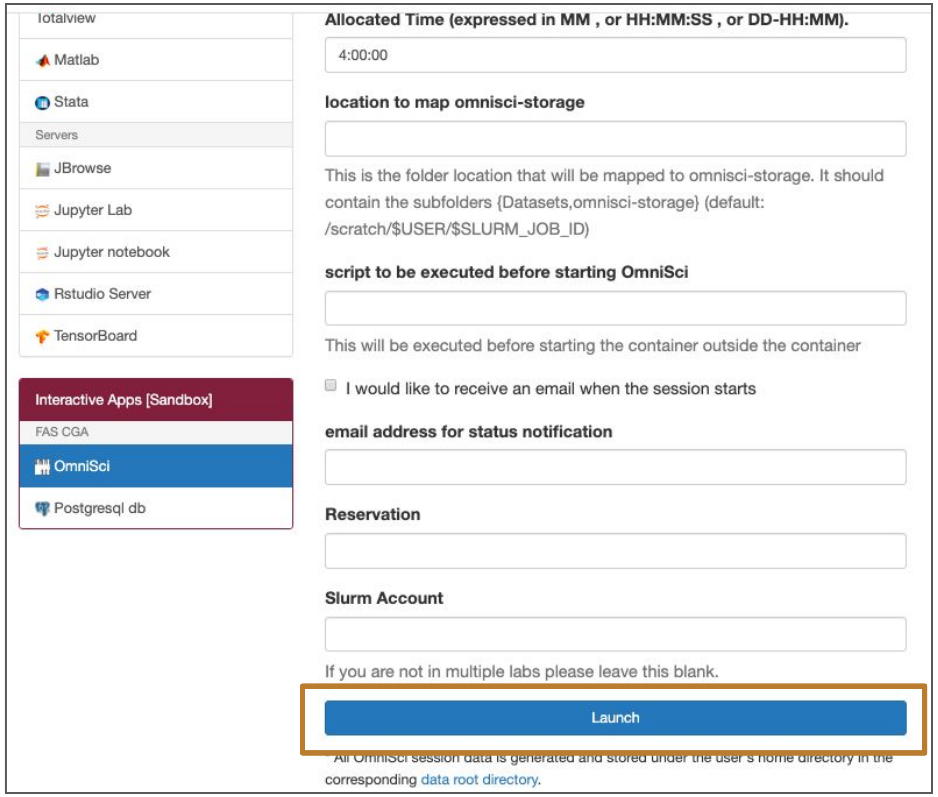
Allocated Time (expressed in MM , or HH:MM:SS , or DD-HH:MM).

4:00:00

location to map omnisci-storage

This is the folder location that will be mapped to omnisci-storage. It should contain the subfolders {Datasets,omnisci-storage} (default: /scratch/$USER/$SLURM_JOB_ID)

script to be executed before starting OmniSci



- Totalview
- Matlab
- Stata

Servers

- JBrowse
- Jupyter Lab
- Jupyter notebook
- Rstudio Server
- TensorBoard

Interactive Apps [Sandbox]

FAS CGA

- OmniSci
- Postgresql db

Allocated Time (expressed in MM , or HH:MM:SS , or DD-HH:MM).

4:00:00

location to map omnisci-storage

This is the folder location that will be mapped to omnisci-storage. It should contain the subfolders {Datasets,omnisci-storage} (default: /scratch/$USER/$SLURM_JOB_ID)

script to be executed before starting OmniSci

This will be executed before starting the container outside the container

☐ I would like to receive an email when the session starts

email address for status notification

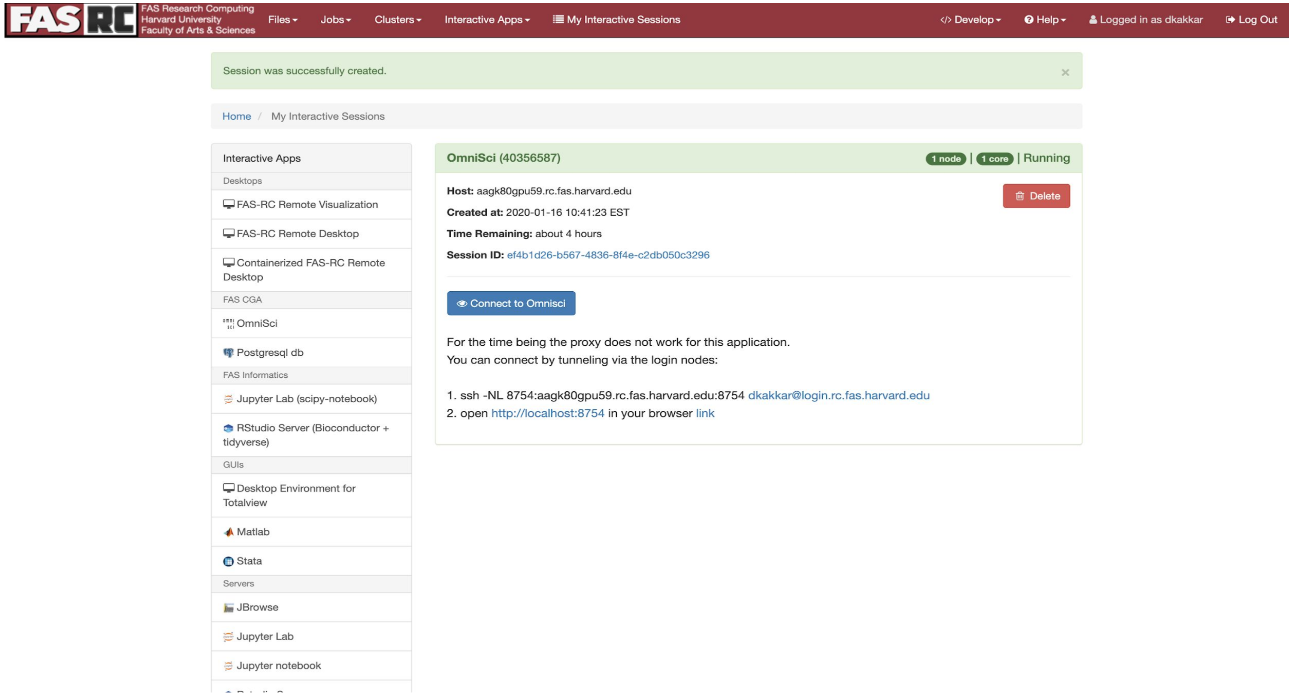
Reservation

Slurm Account

If you are not in multiple labs please leave this blank.

* All OmniSci session data is generated and stored under the user's home directory in the corresponding data root directory.

OmniSci on FASRC



Session was successfully created.

Home / My Interactive Sessions

Interactive Apps
Desktops
FAS-RC Remote Visualization
FAS-RC Remote Desktop
Containerized FAS-RC Remote Desktop
FAS CGA
OmniSci
Postgresql db
FAS Informatics
Jupyter Lab (scipy-notebook)
RStudio Server (Bioconductor + tidyverse)
GUIs
Desktop Environment for Totalview
Matlab
Stata
Servers
JBrowse
Jupyter Lab
Jupyter notebook

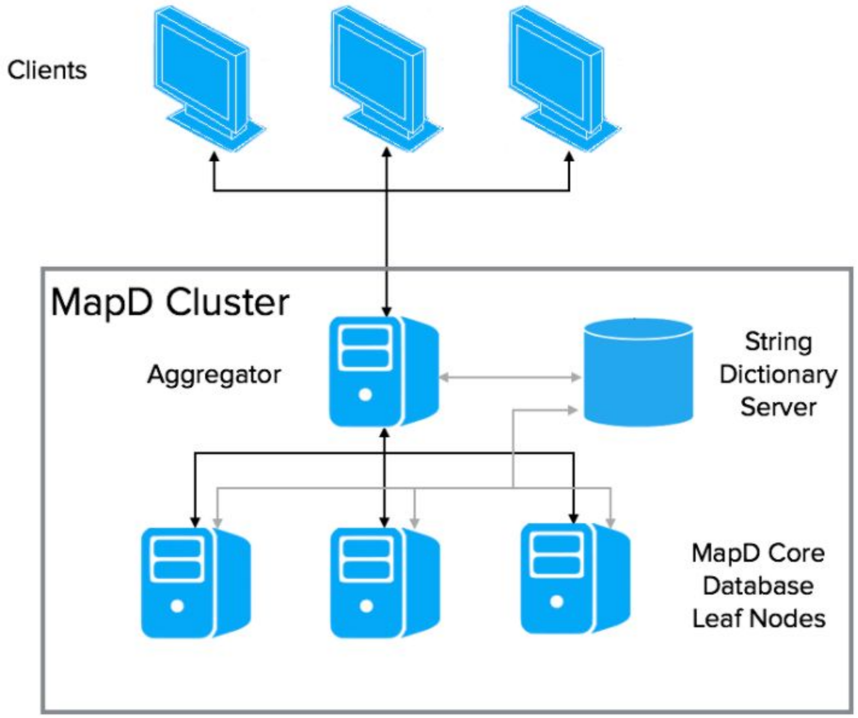
OmniSci (40356587) 1 node | 1 core | Running

Host: aagk80gpu59.rc.fas.harvard.edu
Created at: 2020-01-16 10:41:23 EST
Time Remaining: about 4 hours
Session ID: ef4b1d26-b567-4836-8f4e-c2db050c3296

For the time being the proxy does not work for this application.
You can connect by tunneling via the login nodes:

1. ssh -NL 8754:aagk80gpu59.rc.fas.harvard.edu:8754 dkakkar@login.rc.fas.harvard.edu
2. open http://localhost:8754 in your browser link

Distributed OmniSci on FASRC



Distributed Configuration OmniSci [5]

Illustrative Use Case

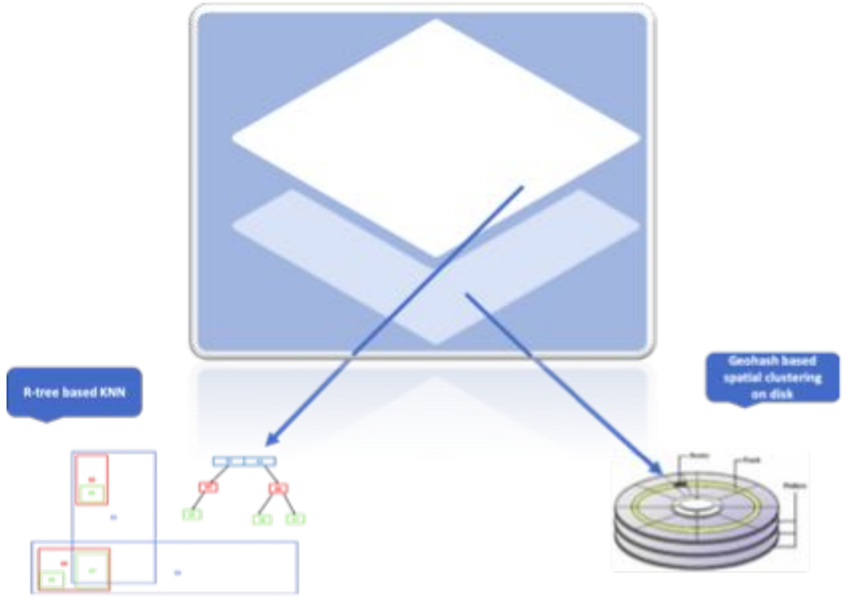
Problem Introduction

- K Nearest Neighbor (KNN) calculation is a well known GIS operation which is hard to scale
- The three common methods of finding KNN:
  - Traditional algorithms
  - Geohash based KNN
  - Index-based Search

96th St subway station, NY

Geohash Clustering with KNN search

- Combining two popular spatial index for best results: Geohash and R-tree
- A two layer approach:
  - Bottom layer : Geohash index based clustering
  - Top layer: R-tree index based searches
- **Extremely fast:** 200,000 distance calculations per second
- **On moderate resources:** m4.xlarge EC2 with 300 GB EBS
- **Cost efficient:** $175/month



Use Case: Partisan Analysis

- Construct individual levels of partisan exposure at several approximated geographies for each voter in US
- Performing k-means clustering on a voter dataset of 180 million, with k=1000
- Solution wrapped as Amazon AMI for easy and fast implementation
  - AMI replicates the entire KNN computation environment on launch
  - User friendly solution with no GIS expertise needed
  - Results stored as PostGIS dumps on S3 in instead of EBS
  - 87% compression in the result dataset obtained
  - 18TG of DB reduced to 1500 GB
  - Storage cost reduced form $1800/month to $35/month (98% reduction)

Objective

The main objective of this work are as follows:

- To make k-means calculations against large datasets fast and easy
- A variation on our approach could likely be used to speed up other spatial processes which require neighborhood search operations

Related work elsewhere

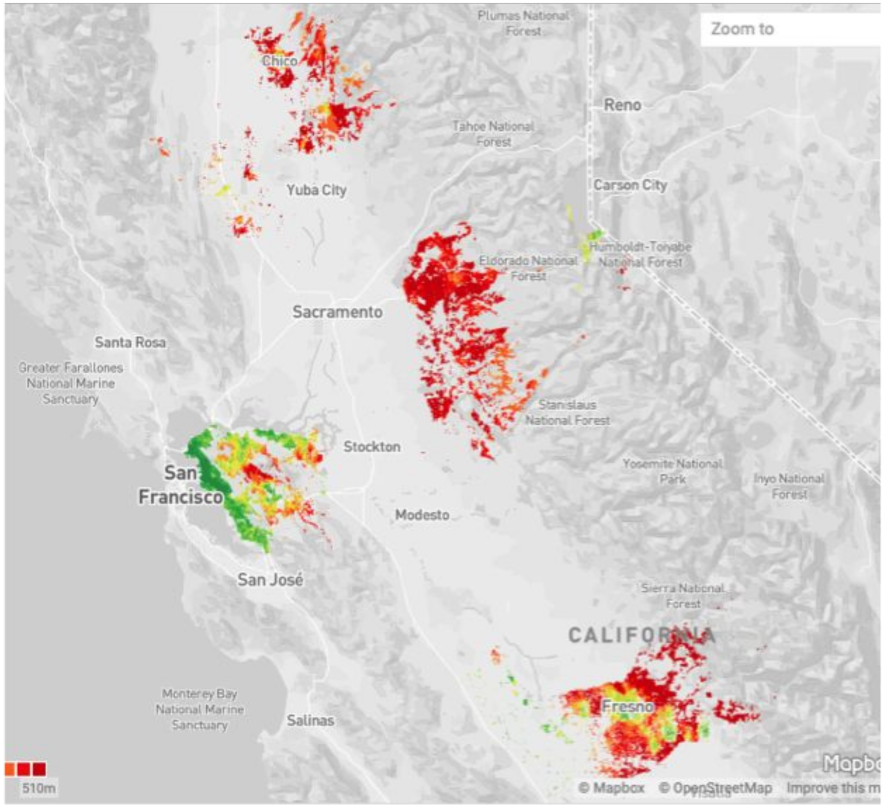
- Another way of calculating KNN on a big dataset is distributed processing on Spark using scikit-learn, however:
  - Spark's MLlib doesn't have built-in support for KNN calculations, but scikit-learn does
  - scikit-learn's k-NN kneighbors() method is a computational bottleneck for large datasets and needs parallelization.
  - scikit-learn's k-NN kneighbors() is inserted into a Spark map function and run in a distributed environment

How ours is different

- **Extremely Fast:** 200,000 distances/sec
- **Resource Optimized:** Basic EC2 m4.xlarge server
- **Two layered approach:**
    - Bottom layer of Geohash based clustering assisting fast data retrieval by the
    - Top layer of R-tree based KNN search
- **Dimensional reduction:** 2D coordinates reduced to 1D Geohash providing compressed storage
- **Cost Effective:** Infrastructure costs only $175/month
- **Hardware and Software Optimization:** Optimization of search algorithm and hard disk storage
- **Customized AMI:** Ready to use
- **Compressed Storage:** 18 TB reduced to 1.5 TB
- **GPU based:** Extremely fast GPU based processing of results

Related work using OmniSci

- **Challenge:** Processing 180 billion distances to calculate the partisan weights of each voter
- **Goal:** Minimize overall time for geospatial analysis of big data
- **Method:** Use GPUs instead of CPUs
- **Platform:** OmniSci
- Real time interactive visualization
- Time and cost saving over traditional approach

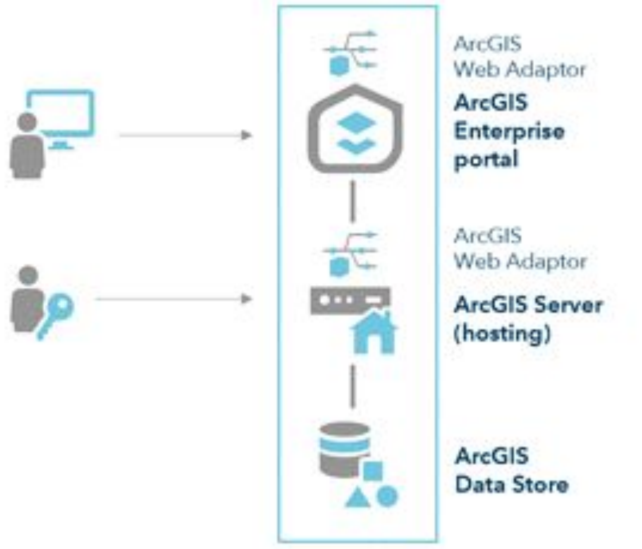


Solution using FASRC

- AWS setup is efficient but expensive to use long term
- FASRC provides a more sustainable long term solution
- Our plan is to install PostGIS and OmniSci as public apps on the FASRC cluster
    - Thereafter, transfer the PostGIS based KNN solution to FASRC
    - Transfer Omnisci based modelling solution to FASRC
- I/O is a big overhead in problems of these scales so loading the data once and utilizing it over and over again makes it more efficient
- Divide and Conquer approach: Currently, multiple AWS instances are launched using the AMI to replicate the database over multiple instances and hence save I/O time
- Replicate AWS: Run several asynchronous jobs on FASRC which will run independent calculations and then combine the results in the end

Future Plans

- Installing ArcGIS Enterprise which consists of:
  - ArcGIS Enterprise Portal
  - ArcGIS Server
  - ArcGIS Data Store
- Exploring other possible use cases such as:
  - Disease Surveillance
  - Global Internet access mapping
  - National Water Model
  - EPA Air Quality Modeling
- Buying paid infrastructure on FASRC for hosted apps



ArcGIS Enterprise [4]

References

[1] Introduction to Cluster Computing:
https://www.rc.fas.harvard.edu/wp-content/uploads/2019/12/Intro-to-Cannon.pdf

[2] Postgis- https://postgis.net/

[3] Slurm Workload Manager: https://slurm.schedmd.com/overview.html

[4] Singularity on Cluster:
https://www.rc.fas.harvard.edu/resources/documentation/software/singularity-on-the-cluster/#odyssey

[5] OmniSci Overview: https://docs.omnisci.com/latest/4_distributed.html

[6] ArcGIS Enterprise: https://enterprise.arcgis.com/en/documentation/install/

Evaluation: http://bit.ly/datafest_eval