ILLINOIS INSTITUTE OF TECHNOLOGY

Department of Computer Science

The globus compute dataset: An open function-as-a-service dataset from the edge to the cloud

Serverless Computing

- Allows to run applications and functions without being concerned about the underlying hardware
  - Pick a runtime
  - Write function
  - Run (and scale)
- Advantages
  - On-demand
  - Elastic scaling
  - ….

What about scientific computing?

Globus Compute

- Former FuncX

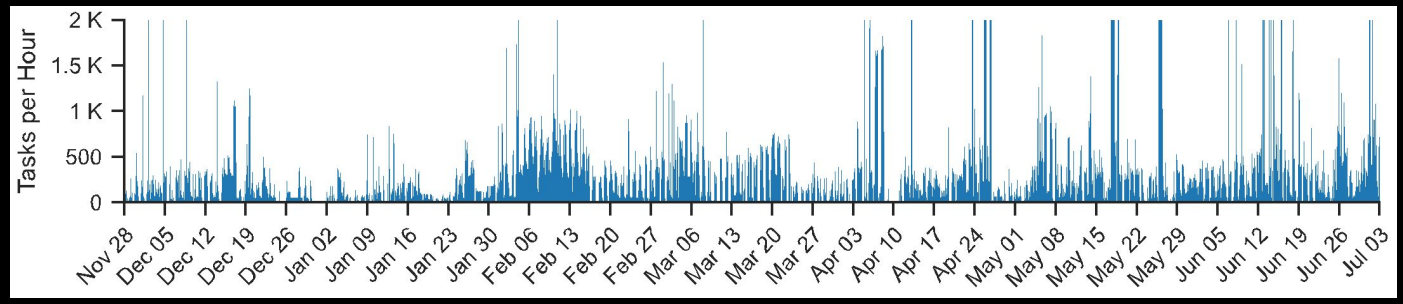
  Managed, federated Function-as-a-Service for reliably, scaleably and securely executing functions on remote endpoints from laptops to supercomputers

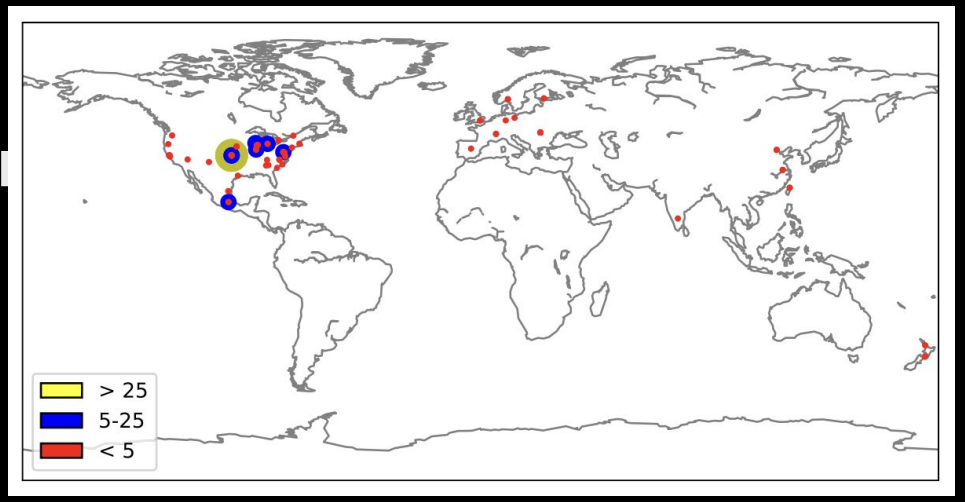
- Scientific computing
  - Researchers primarily work in high level languages
  - FaaS allow researchers to work in a familiar language (e.g., Python) using familiar interfaces (e.g., Jupyter notebooks)

- Goal: Move closer to researchers' environments

Globus Compute Dataset

- Dataset
  - 31 weeks of data
  - 2.1 million function calls
  - 270 thousand functions
  - 252 active users
  - 580 geographically distributed endpoi

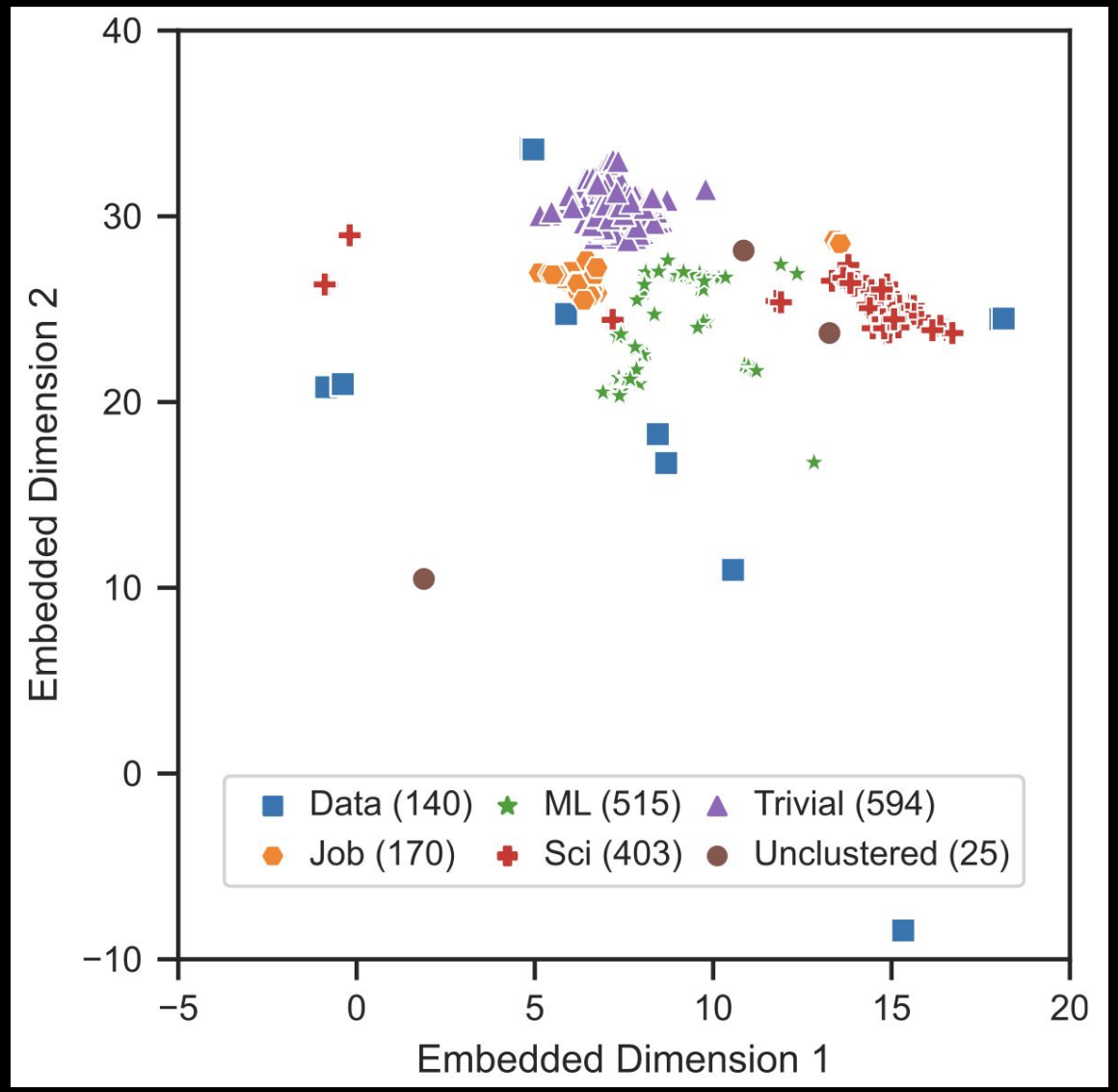




- Uniqueness
  - Fine grained timestamps
  - Function source code analysis
  - Endpoint analysis

Analysis Overview

- Statistical analysis
  - Systems performance
  - Interarrival times
  - Task invocations
  - Function bodies
  - User behavior
- Further analysis
  - Task submission patterns
  - Cold-start
  - Clustering of functions

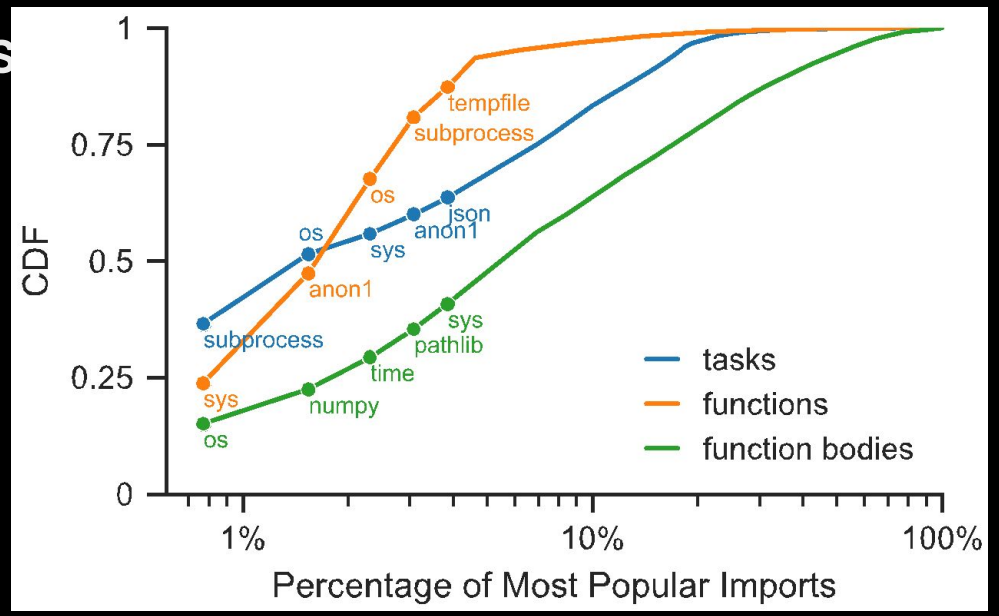


Statistical Analysis

Characteristic	Central tendency		Measure of variability	
	Mean	Median	SD	Range
System performance				
Arrival rate [req/h]	404.31	179.00	1.46e+03	[0e+00; 4.53e+04]
Avg. arrival rate per endpoint [req/h]	110.75	2.03	634.45	[0.33; 8.11e+03]
End-to-end time [s]	1.36e+03	0.34	1.66e+04	[1.57e−03; 1.17e+06]
Interarrival times				
Received (t_{re}) → Wait for node (t_{wn}) [s]	414.83	0.10	1.48e+04	[1.02e−06; 1.17e+06]
Wait for node (t_{wn}) → Wait for launch (t_{wl}) [s]	260.58	7.23e−03	1.88e+03	[1.77e−04; 1.31e+05]
Wait for launch (t_{wl}) → Execution starts (t_{es}) [s]	298.89	9.02e−03	2.08e+03	[4.91e−04; 1.31e+05]
Execution starts (t_{es}) → Execution ends (t_{ee}) [s]	49.04	0.03	300.37	[7.6e−05; 1.04e+05]
Execution ends (t_{ee}) → Results received (t_{rr}) [s]	5.42	0.13	51.13	[3.74e−05; 4.9e+04]
Tasks				
Avg. function idle time [s]	2.13e+03	61.38	5.55e+04	[5.12e−06; 5.44e+06]
Argument size [Bytes]	1.73e+04	62.00	2.14e+05	[30.00; 1.03e+07]
Function bodies				
# Lines of code	35.68	48.00	29.47	[1.00; 467.00]
Cyclomatic complexity	5.91	6.00	3.96	[1.00; 20.00]
Imported libraries	1.50	1.00	1.52	[0.00; 18]
Users				
Avg. task submission interval [s]	1.89e+05	3.25e+03	8.1e+05	[2.67e−06; 7.48e+06]
# Tasks submitted	8.42e+03	22.50	5.12e+04	[1.00; 6.78e+05]
# Functions submitted	1.1e+03	7.50	1.07e+04	[1.00; 1.23e+05]
# Used endpoints	3.08	1.00	4.56	[1.00; 29.00]

Analysis of Interest

- Function performance
  - End-to-end time is on average 23 minutes
  - 85% of functions are completed within 10 seconds
  - Cold start for 93% of functions can be avoided by "warming" them for 5 minutes
- Most popular imports
- Provider types for endpoints
  - 76% Local
  - 11% Slurm
  - 5% Kubernetes



In a Nutshell

- Foster research in serverless computing and related fields
- Analyzed and released a scientific computing dataset
  - 31 weeks
  - 2 million function invocations
  - Fine-grained time stamps
- FAIR
- Rolling update