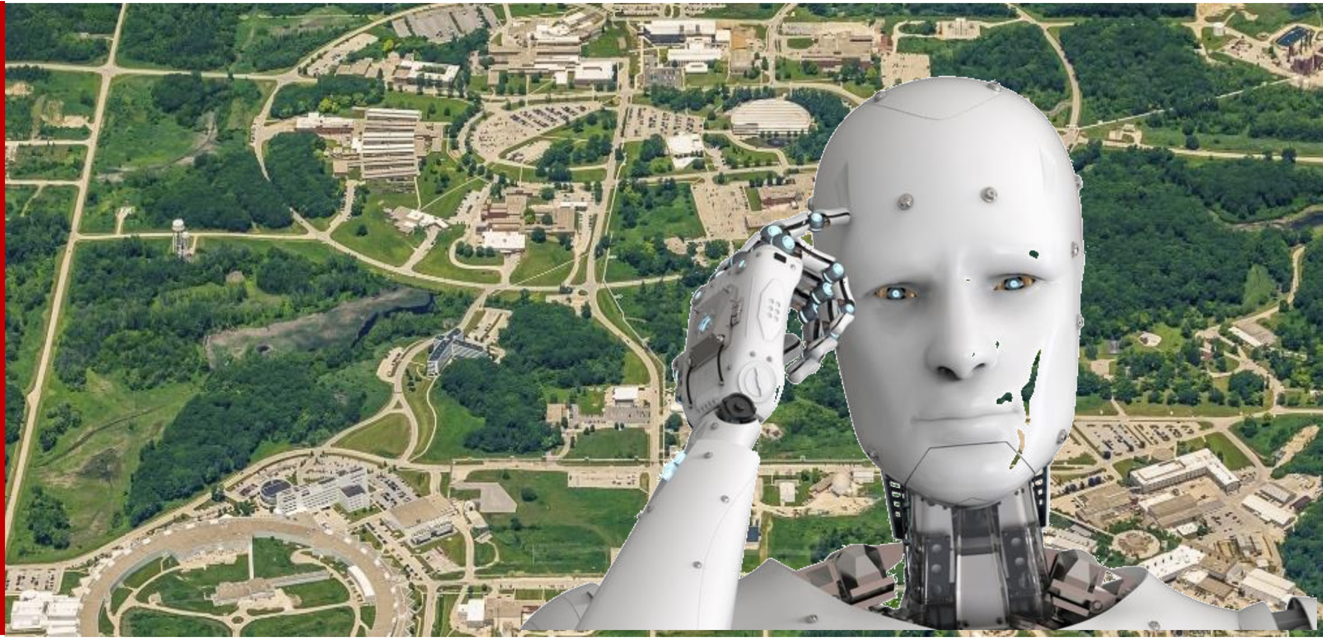
Argonne NATIONAL LABORATORY

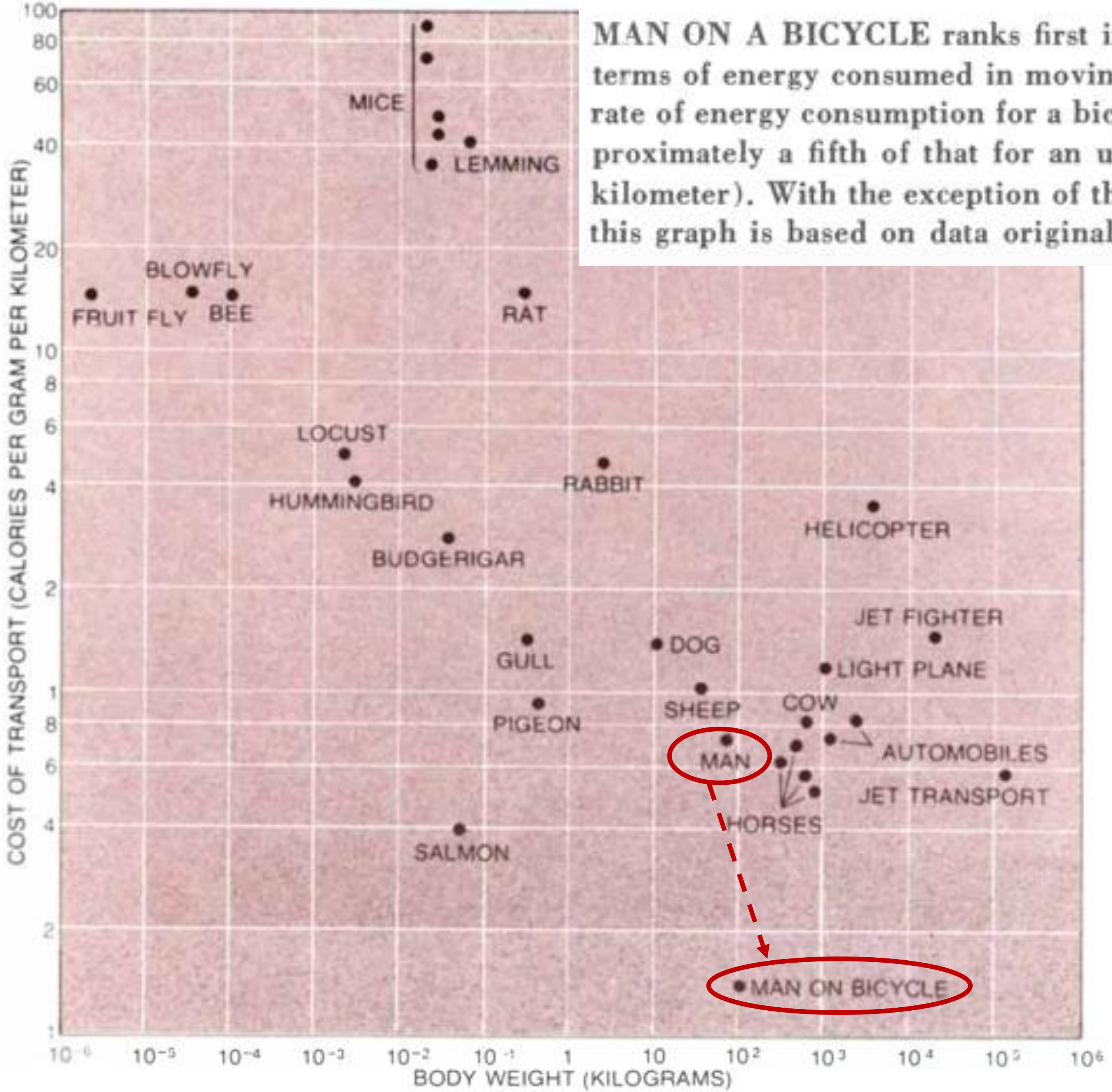
Mechanisms Empowering Discovery



IAN FOSTER

Distinguished Fellow and Director, Data Science and Learning Division, Argonne National Laboratory
Office of Science Distinguished Scientists Fellow
Professor of Computer Science, University of Chicago
foster@anl.gov

Advanced Scientific Computing Advisory Committee
January 13, 2020



MAN ON A BICYCLE ranks first in efficiency among traveling animals and machines in terms of energy consumed in moving a certain distance as a function of body weight. The rate of energy consumption for a bicyclist (about .15 calorie per gram per kilometer) is approximately a fifth of that for an unaided walking man (about .75 calorie per gram per kilometer). With the exception of the black point representing the bicyclist (*lower right*), this graph is based on data originally compiled by Vance A. Tucker of Duke University.

“A bicycle for the mind”

March 1973, Scientific American

PROGRAMMER'S REFERENCE MANUAL

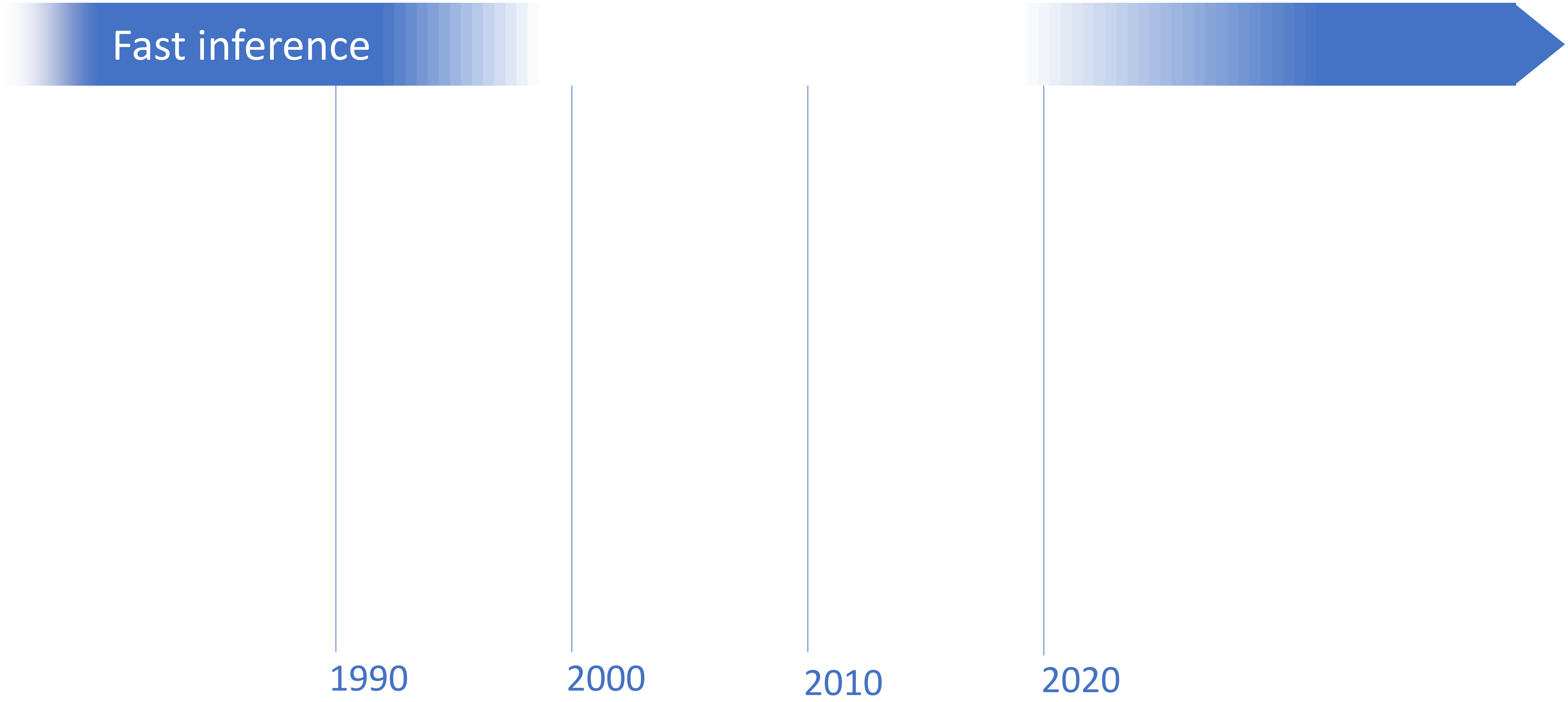
Fortran

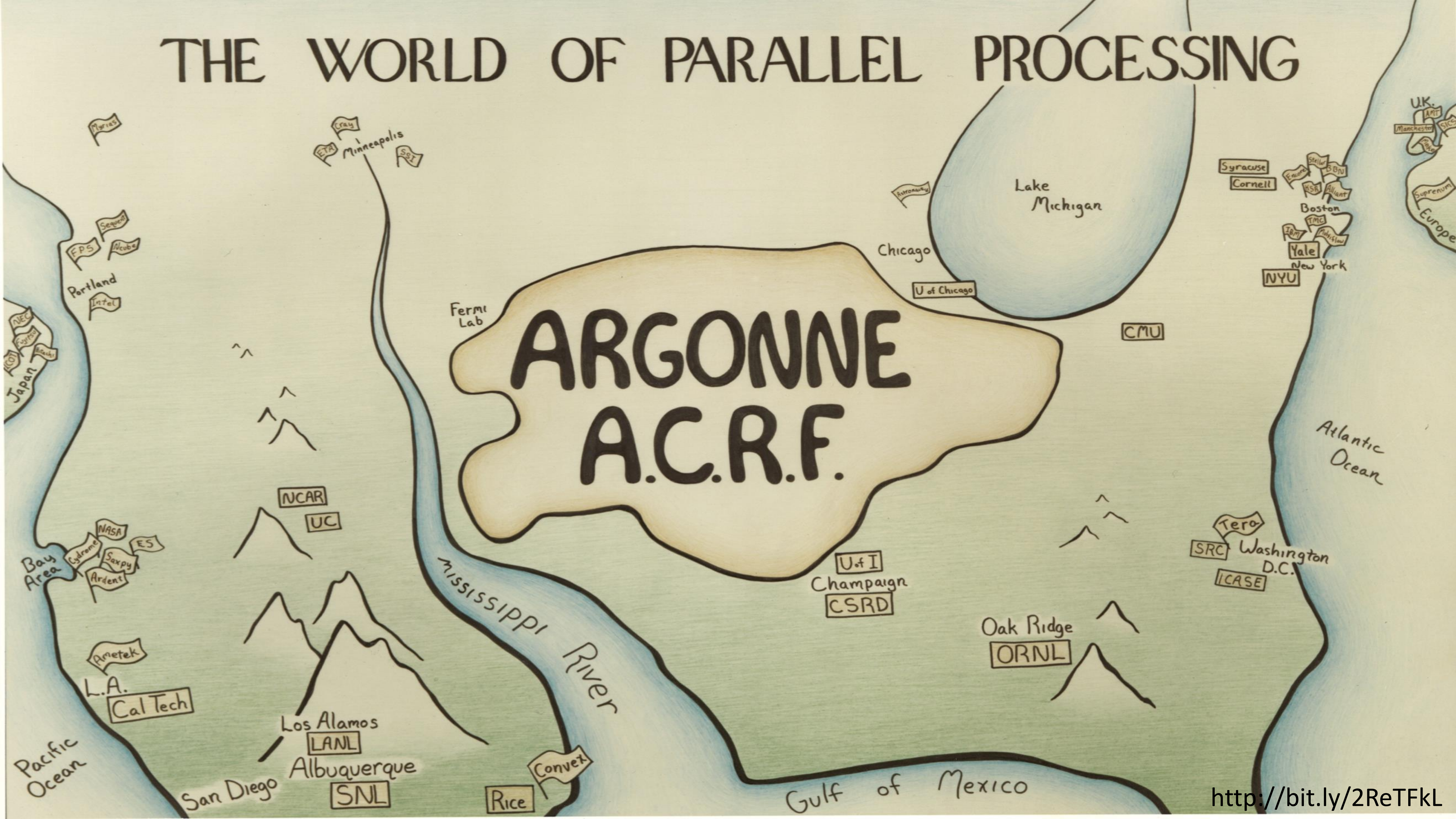
AUTOMATIC CODING SYSTEM

FOR THE IBM 704

https://www.fortran.com/FortranForTheIBM704.pdf

Areas of opportunity for scientific mechanisms





http://bit.ly/2ReTFkL

Computer Science in Economics and Management **3**: 137–145, 1990.

Is a GigaLIP Fast Enough?

TOM W. KELLER
Microelectronics and Computer Technology Corporation, 3500 West Balcones Center Dr., Austin, TX 78759, U.S.A.

(Recieved: December 1988)

Abstract. The capability of large, data-intensive expert systems is determined not only by the cleverness and expertise of their knowledge manipulation algorithms and methods but also by the fundamental speeds of the computer systems upon which they are implemented. To date, logical inferences per second (LIPS) is used as the "power metric" of the knowledge processing capacity of an expert system implementation. We show why this simplistic metric is misleading. We relate the power metrics for conventional computer systems to LIPS and demonstrate wide discrepancies. We review the 'power' of today's largest conventional mainframes, such as the IBM 3090/400 and the Cray Research Cray-2 and forecast the expected power of mainframes and specialized processors in the coming decade.

Credit: Lloyd DeGrane / The New York Times

Dr. William McCune at Argonne Labs, Illinois in his office with computer. The "Proof of Robbins Conjecture" problem is on the screen.

1996

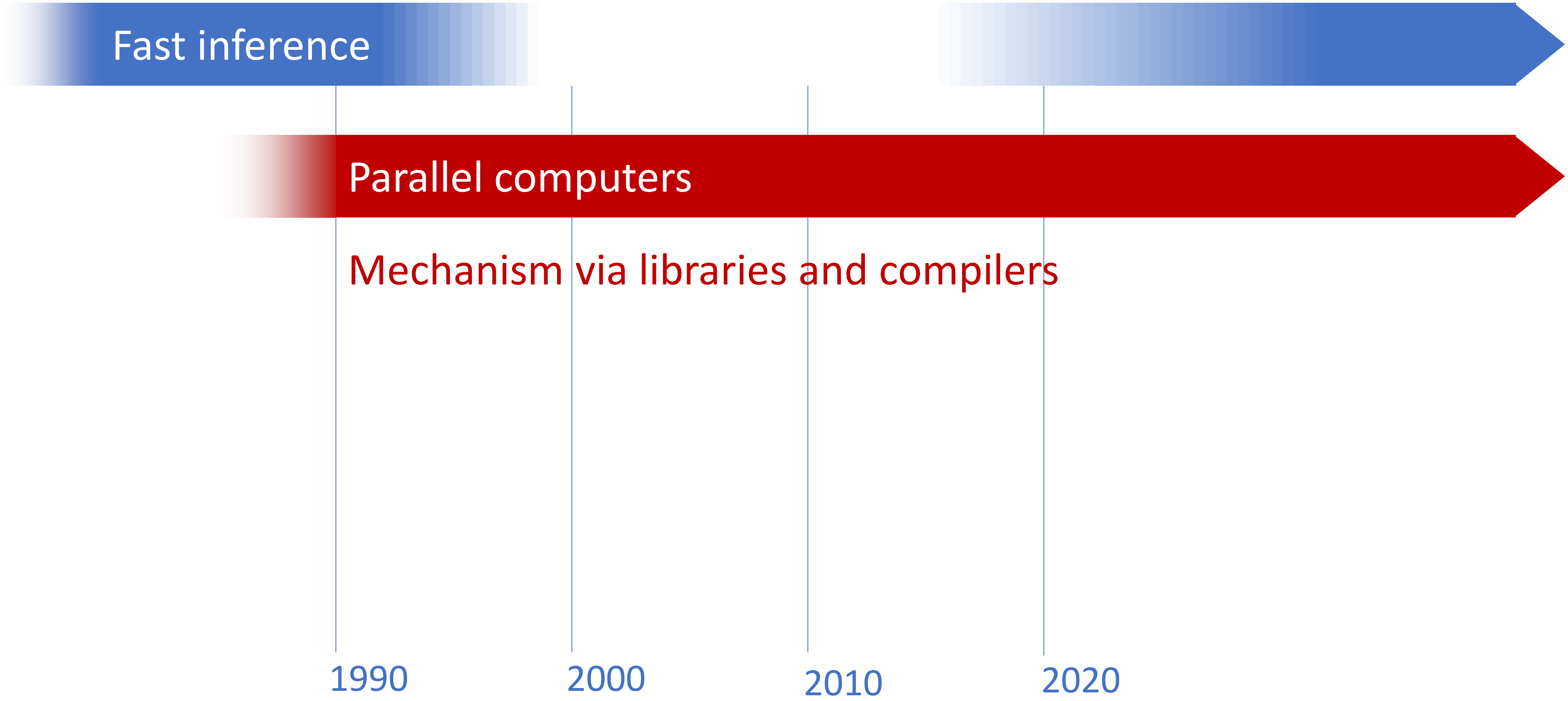
8 Days on an IBM RS/6000
30 megabytes of memory

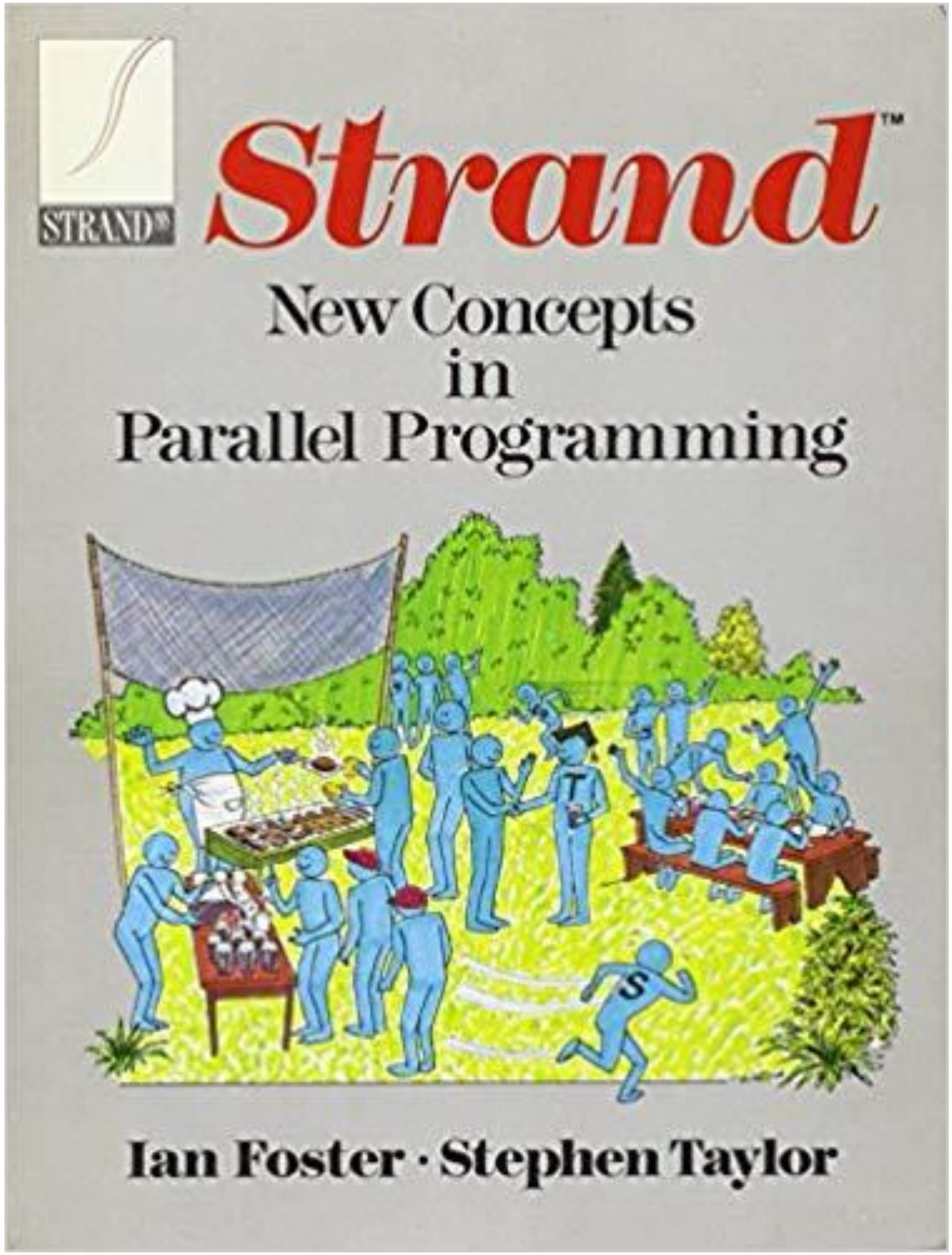
Powerful satisfiability modulo theory (SMT) solvers like Microsoft's Z3 adapt strategies from Otter

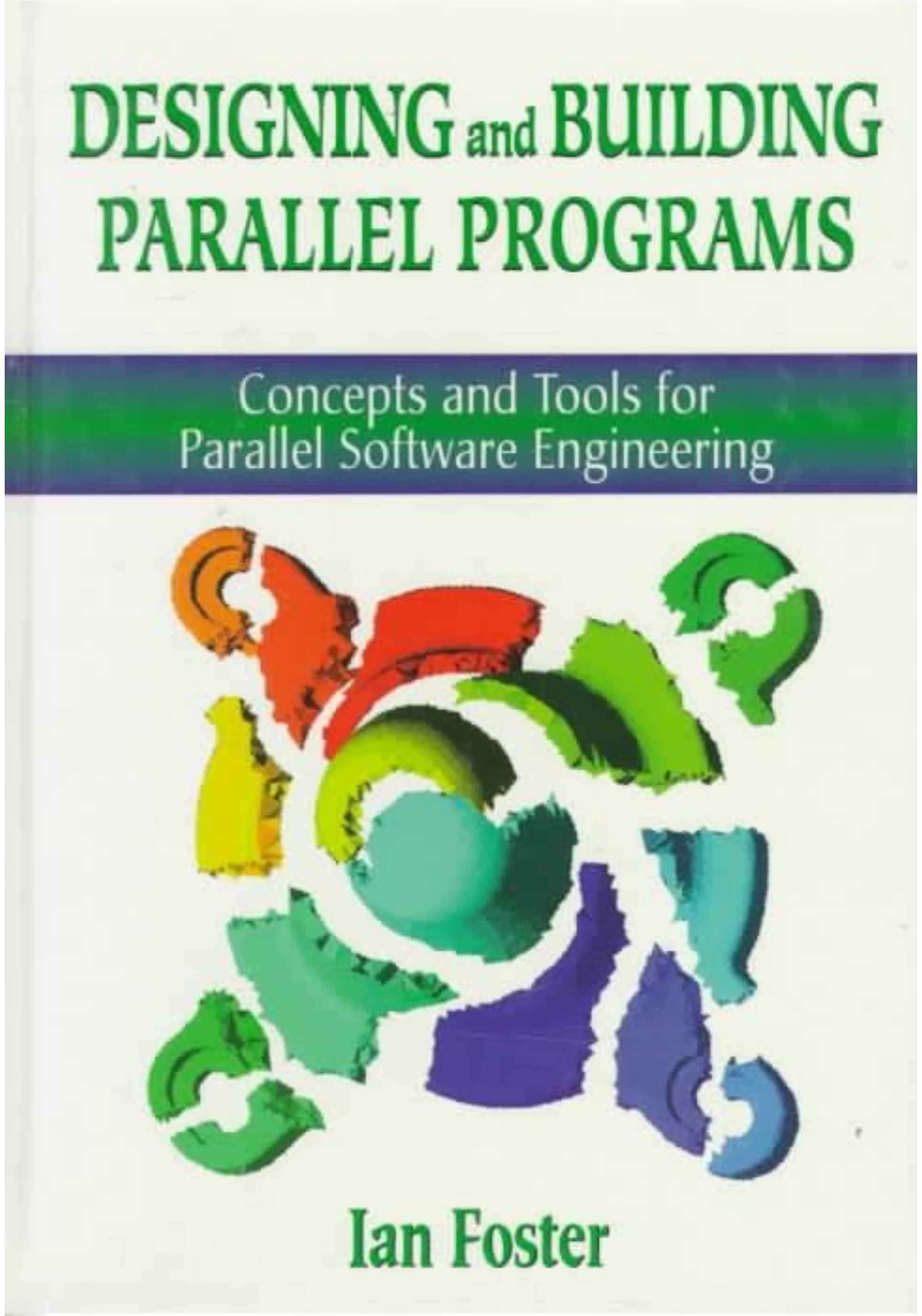
- Bounded model-checking of model programs 
- Termination 
- Security protocols  
- Business application modeling 
- Cryptography 
- Model Based Testing (SQL-Server)
- *Your killer-application here*

Sources: https://nyti.ms/37X8fUM, http://bit.ly/35J3lcE

Areas of opportunity for scientific mechanisms




STRAND
Strand™
New Concepts in Parallel Programming
T
S
Ian Foster · Stephen Taylor


DESIGNING and BUILDING PARALLEL PROGRAMS
Concepts and Tools for Parallel Software Engineering
Ian Foster

6 Gflop/s in 1995, for climate model proxy app on Intel Paragon

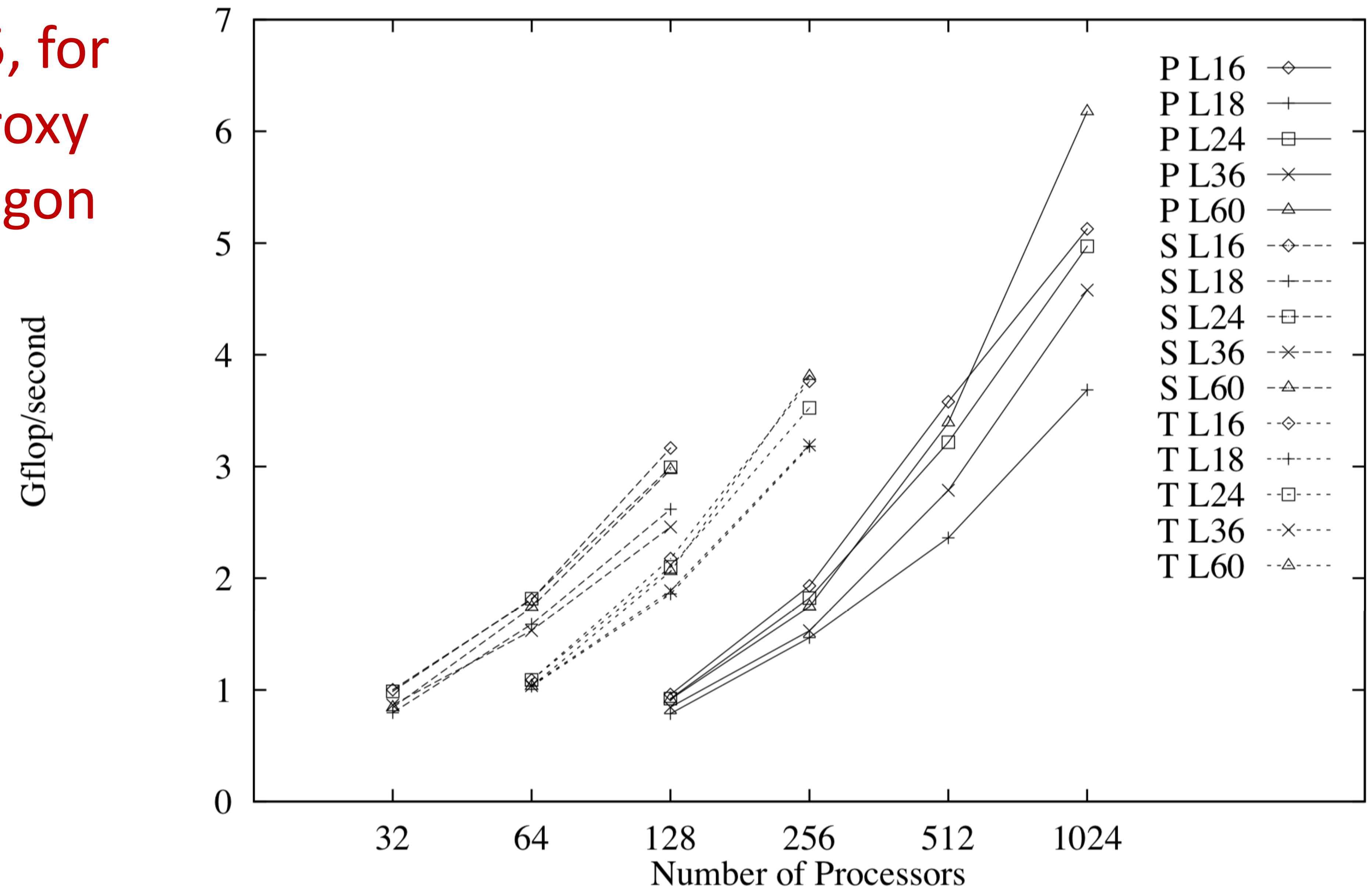
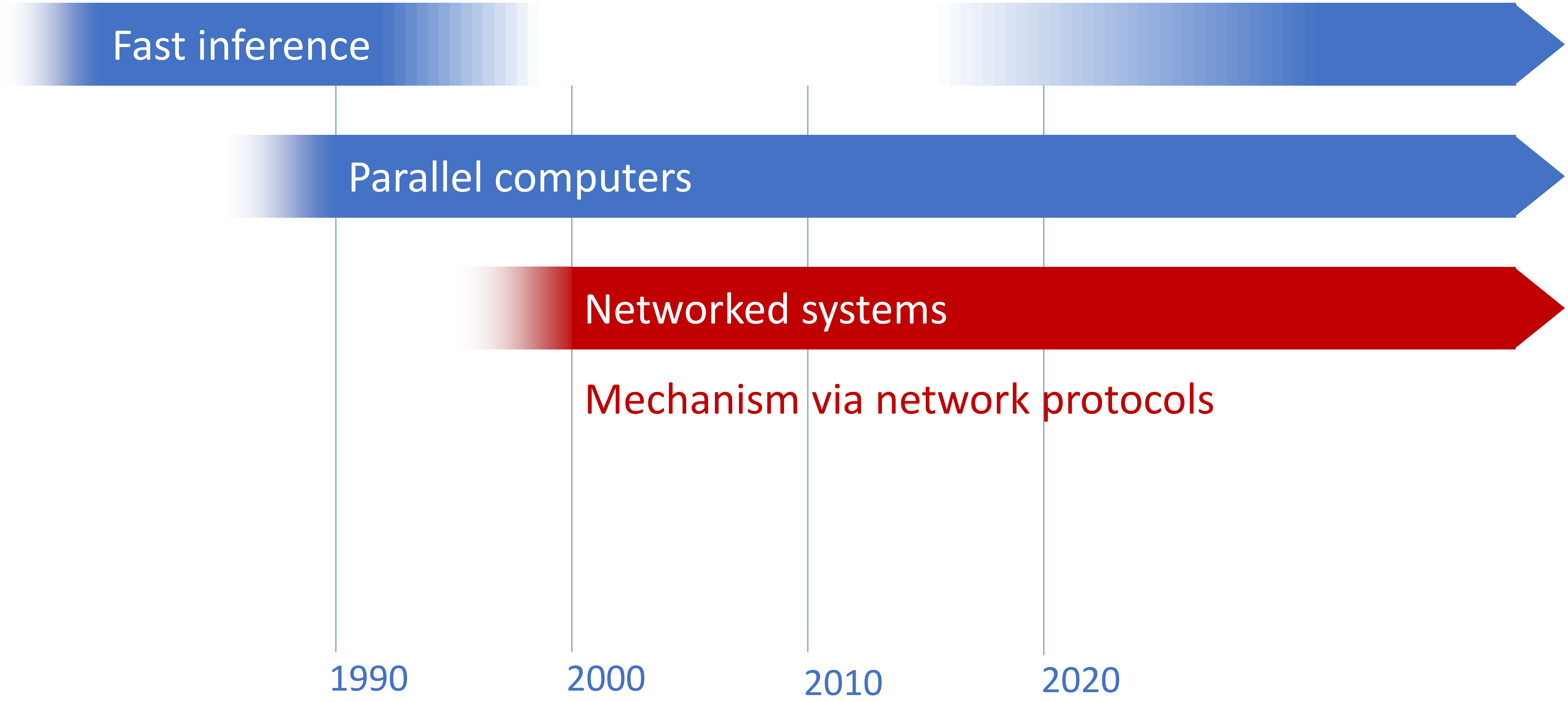


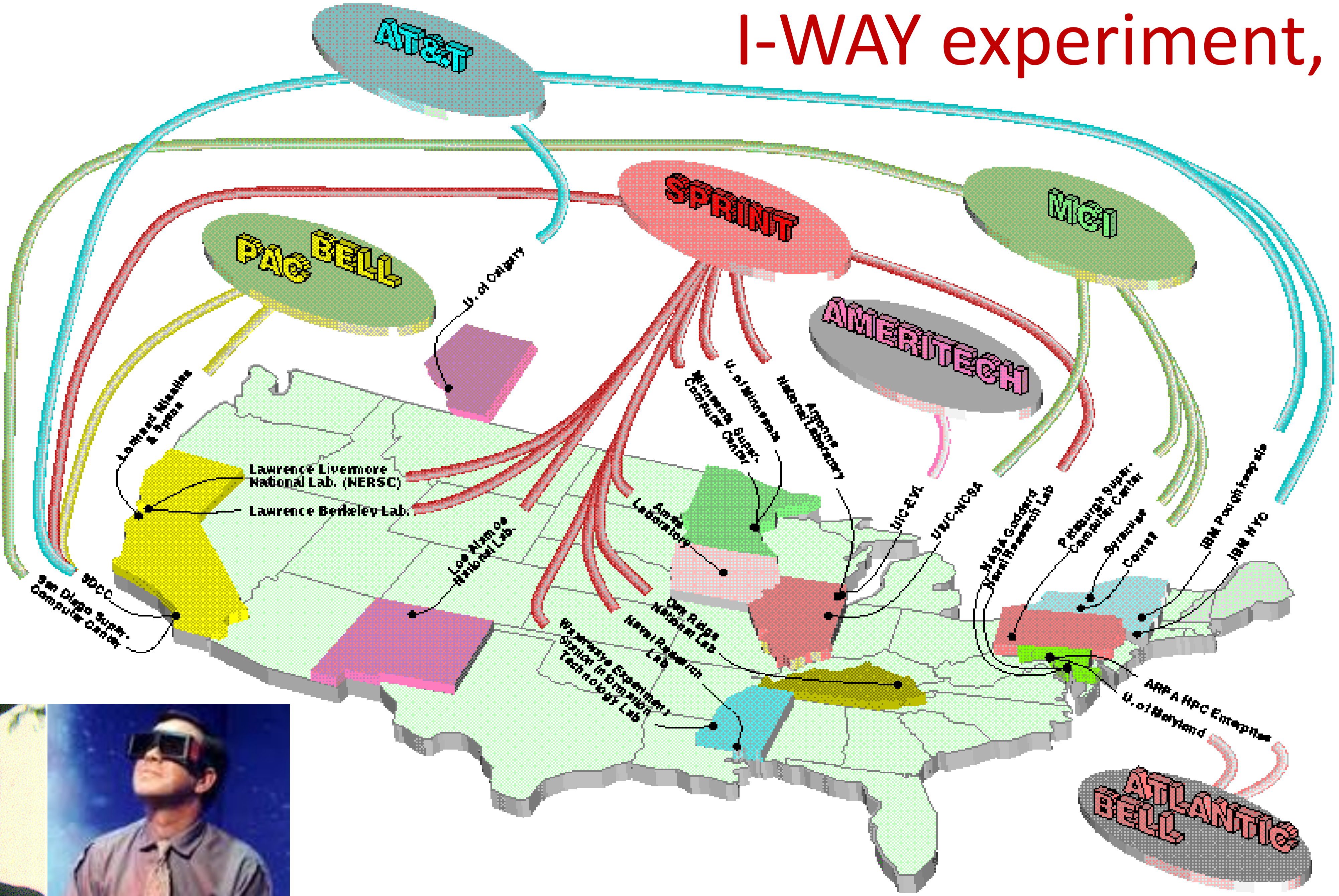
Figure 5: Execution rate for 5-day forecast at T85 resolution (double precision).

http://bit.ly/35GtA3r, 1995

Areas of opportunity for scientific mechanisms



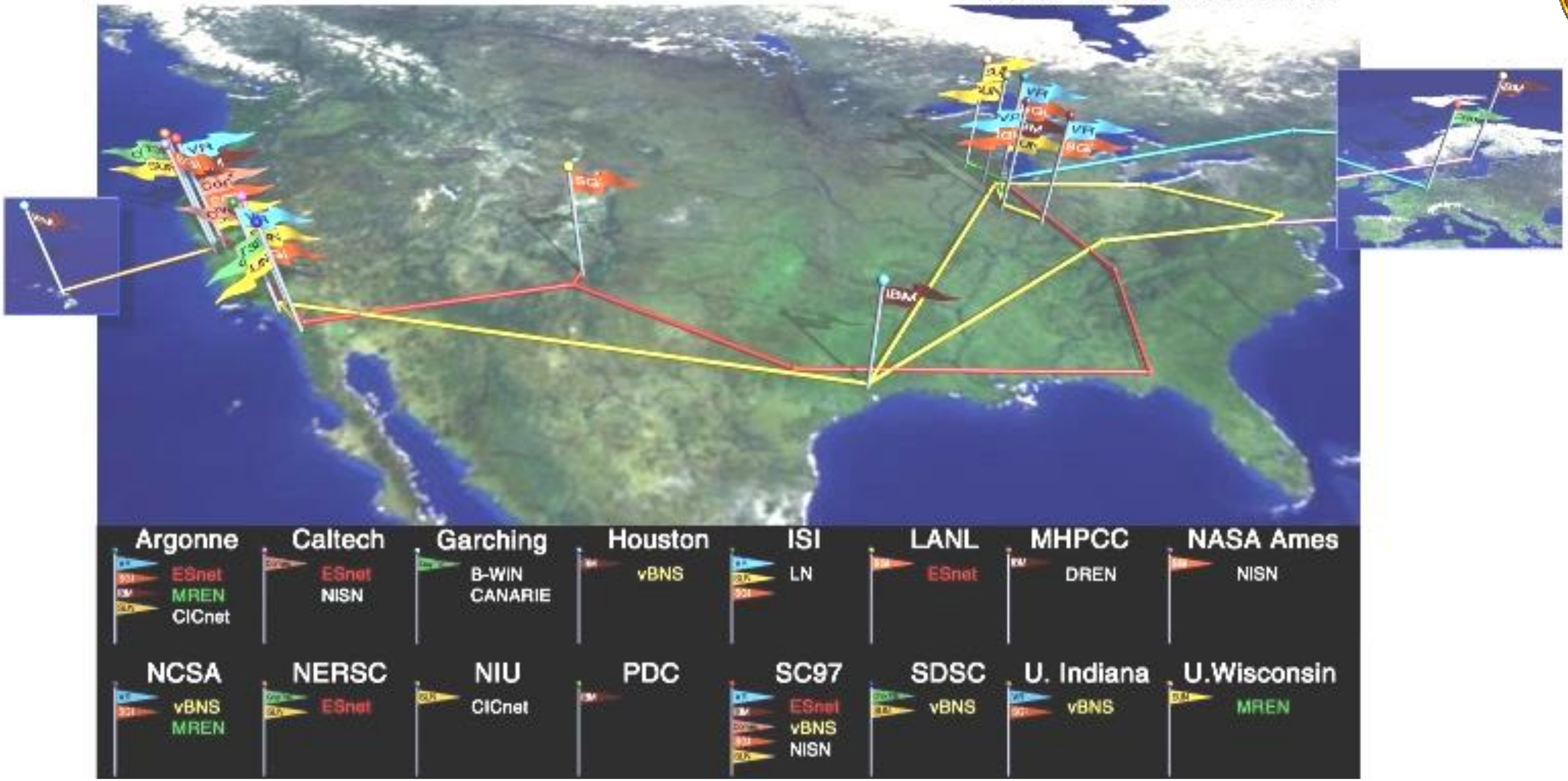
I-WAY experiment, 1995



Source: Linda Winkler and Richard Foster

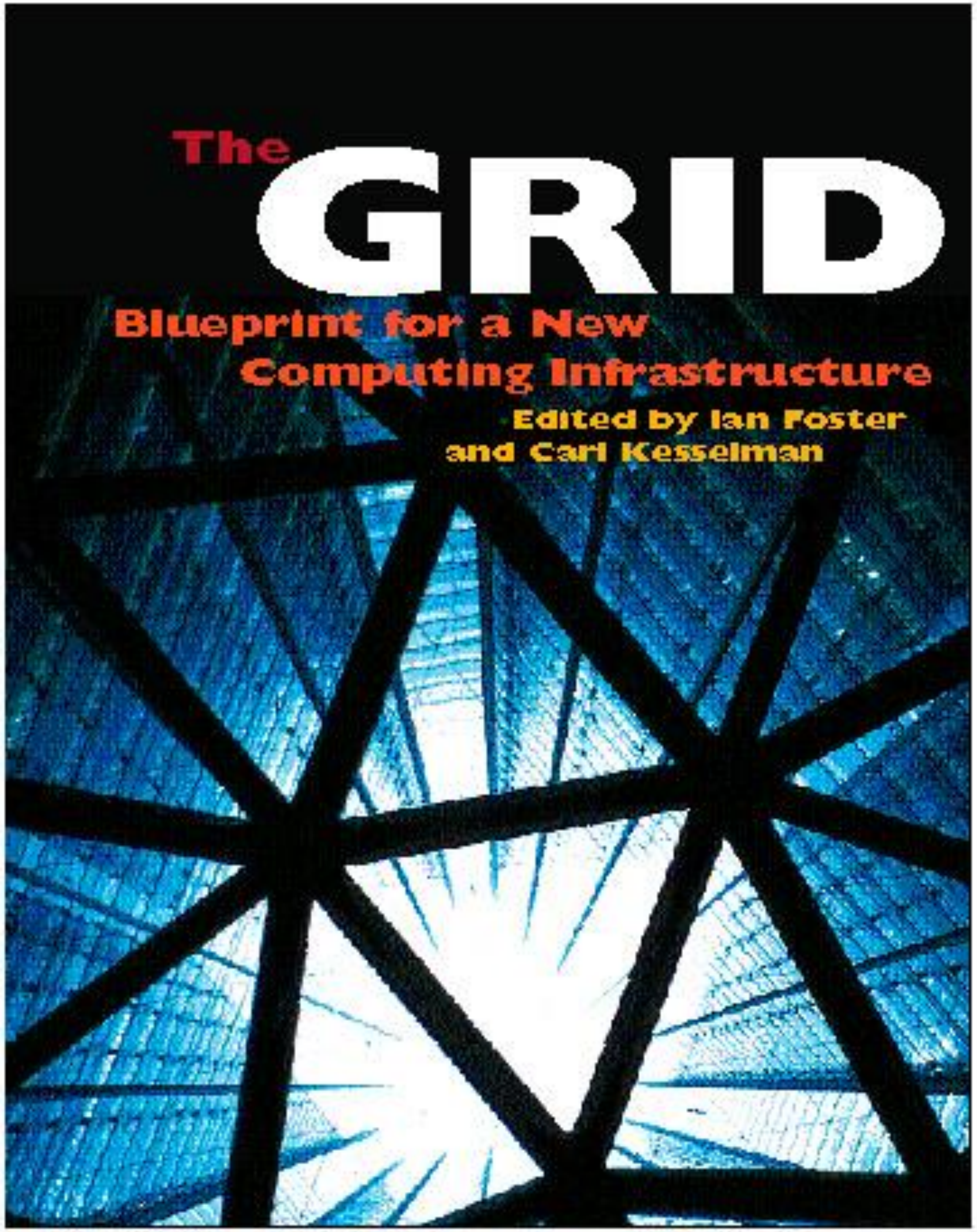
GUSTO Computational Grid Testbed

as of November 1997



16 sites, 330 computers, 3600 nodes, 2 Teraflop/s, 10 application partners

The grid vision



Accelerate discovery & innovation by providing **on-demand access to computing**

"if mechanisms are in place to allow reliable, transparent, and instantaneous access to high-end resources, then it is as if those resources are devoted to them"
(The Grid, Chapter 2)

Grid instrumental in 3 Nobel prizes

IPCC climate assessment: Peace, 2007

Earth System Grid enables sharing of simulation outputs

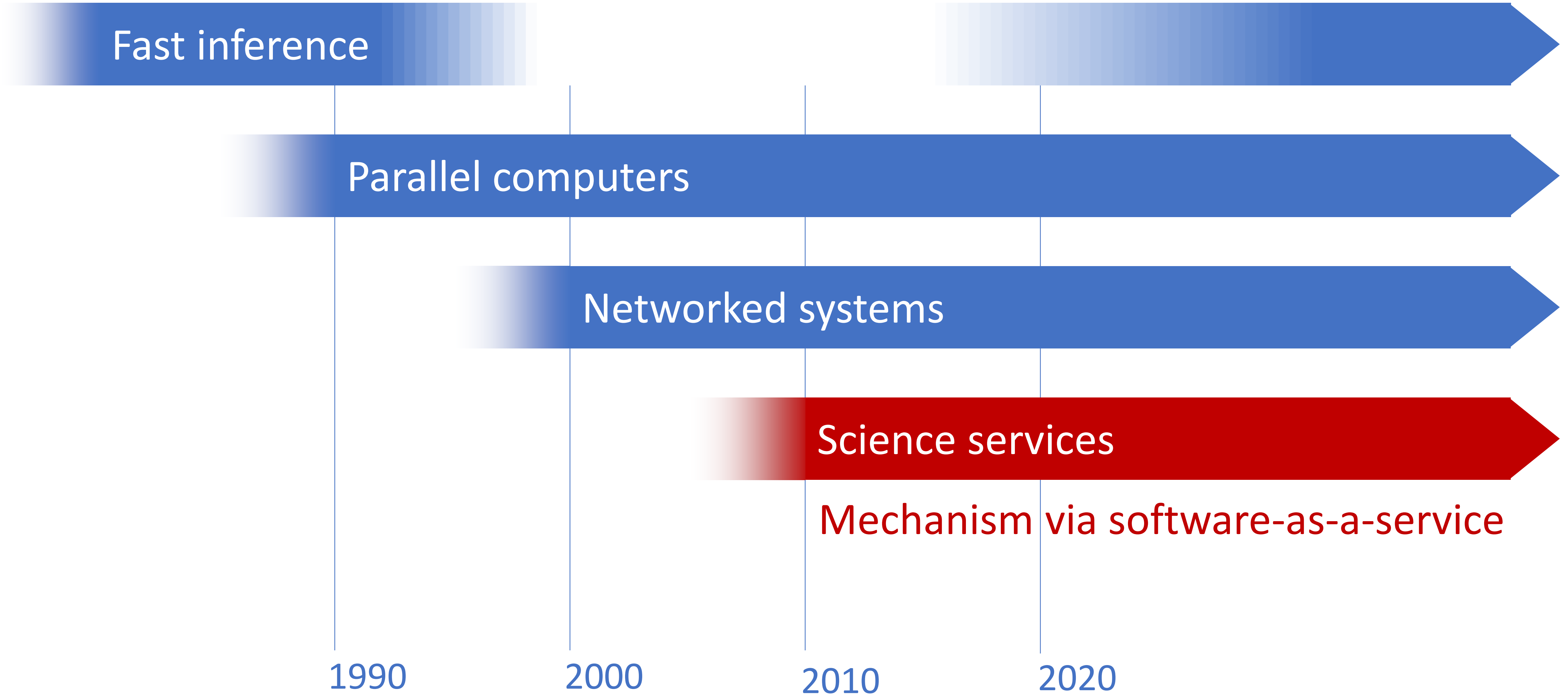
Discovery of Higgs Boson: Physics, 2013

"only possible because of the extraordinary achievements of … grid computing"—Rolf Heuer, CERN DG

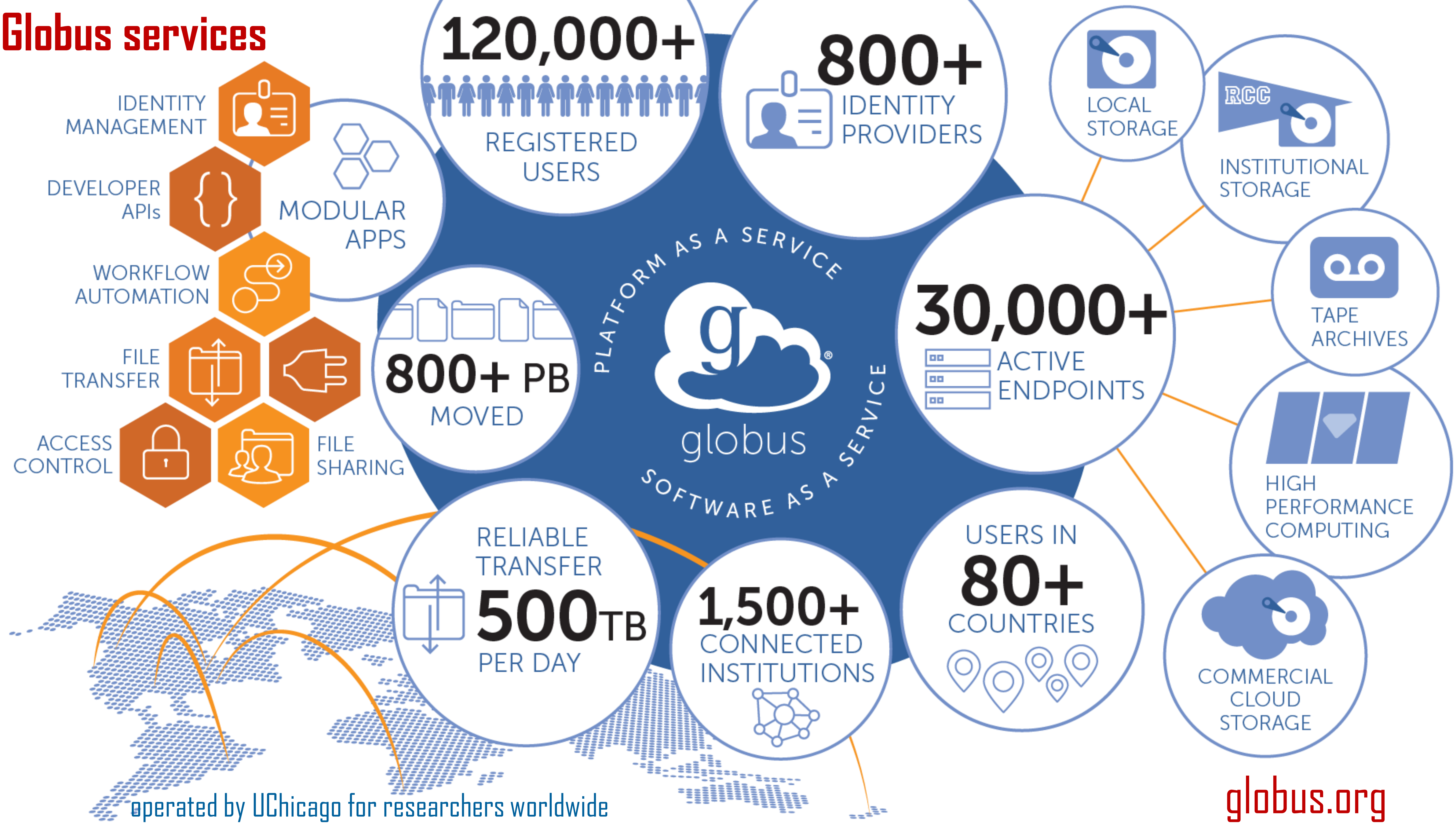
Detection of gravitational waves: Physics, 2017

LIGO scientific collaboration uses grid technologies to pool data and computing

Areas of opportunity for scientific mechanisms

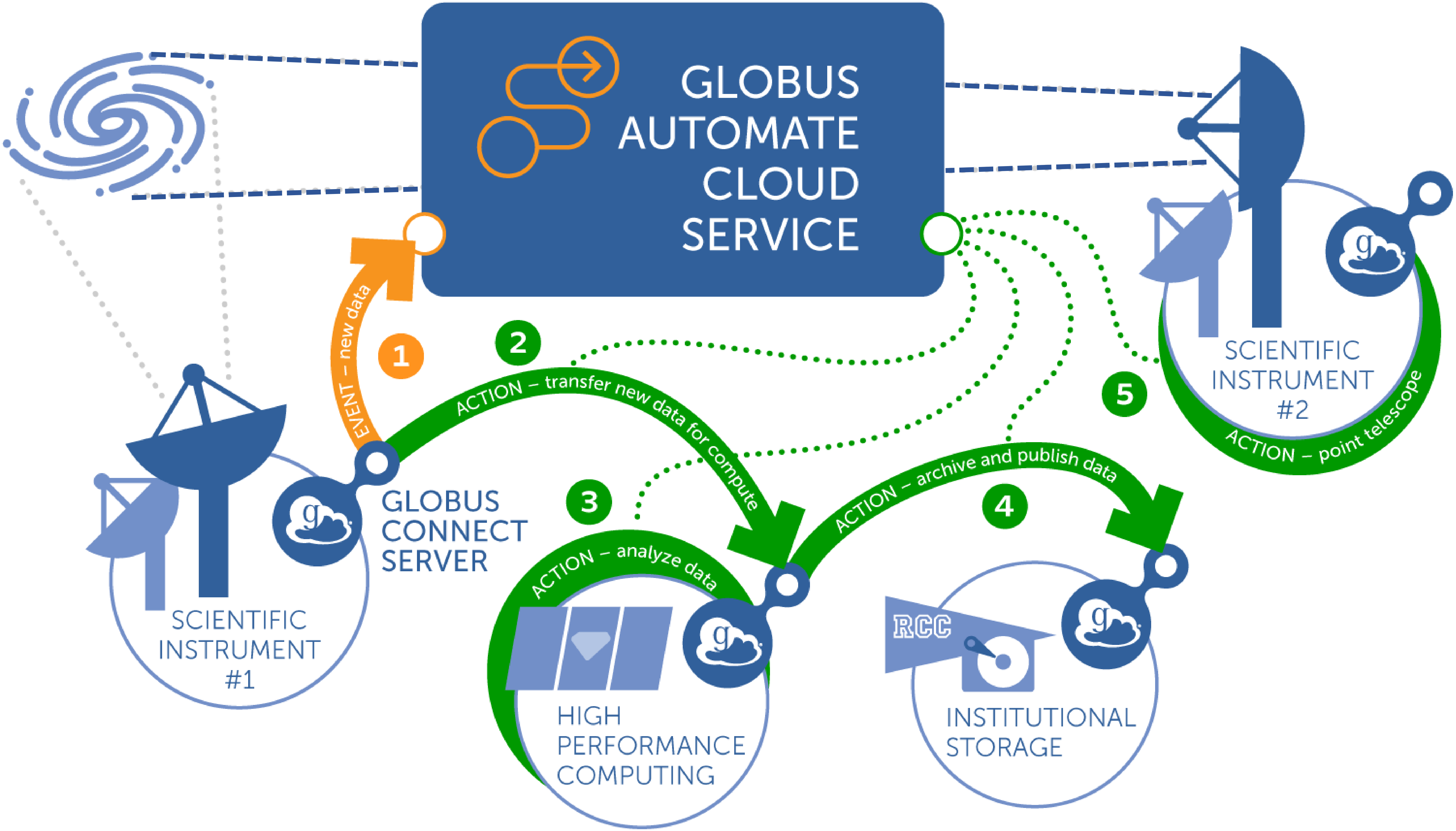


Globus services



operated by UChicago for researchers worldwide

globus.org



GLOBUS
AUTOMATE
CLOUD
SERVICE
1
EVENT – new data
2
ACTION – transfer new data for compute
3
ACTION – analyze data
4
ACTION – archive and publish data
5
ACTION – point telescope
GLOBUS
CONNECT
SERVER
SCIENTIFIC
INSTRUMENT
#1
SCIENTIFIC
INSTRUMENT
#2
HIGH
PERFORMANCE
COMPUTING
RCC
INSTITUTIONAL
STORAGE

Globus by the numbers...

1960
most shared endpoints at a single institution

793 PB
moved

103 billion
files processed

1710
active server endpoints

159
subscribers

124,887
total users

81
countries where Globus is used

25,881
active personal endpoints

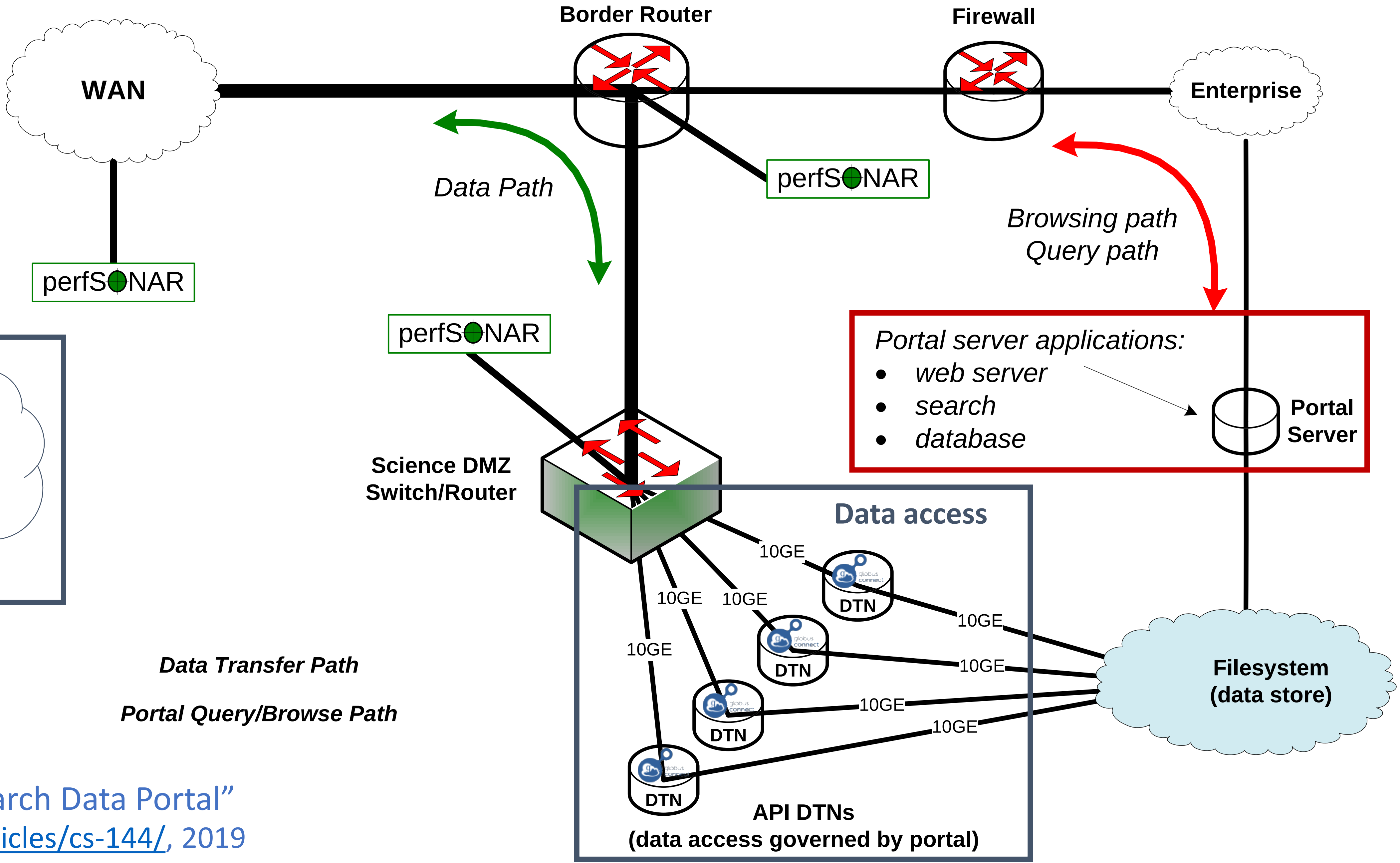
889
identity providers

2.9 PB
largest single transfer to date

6575
active shared endpoints

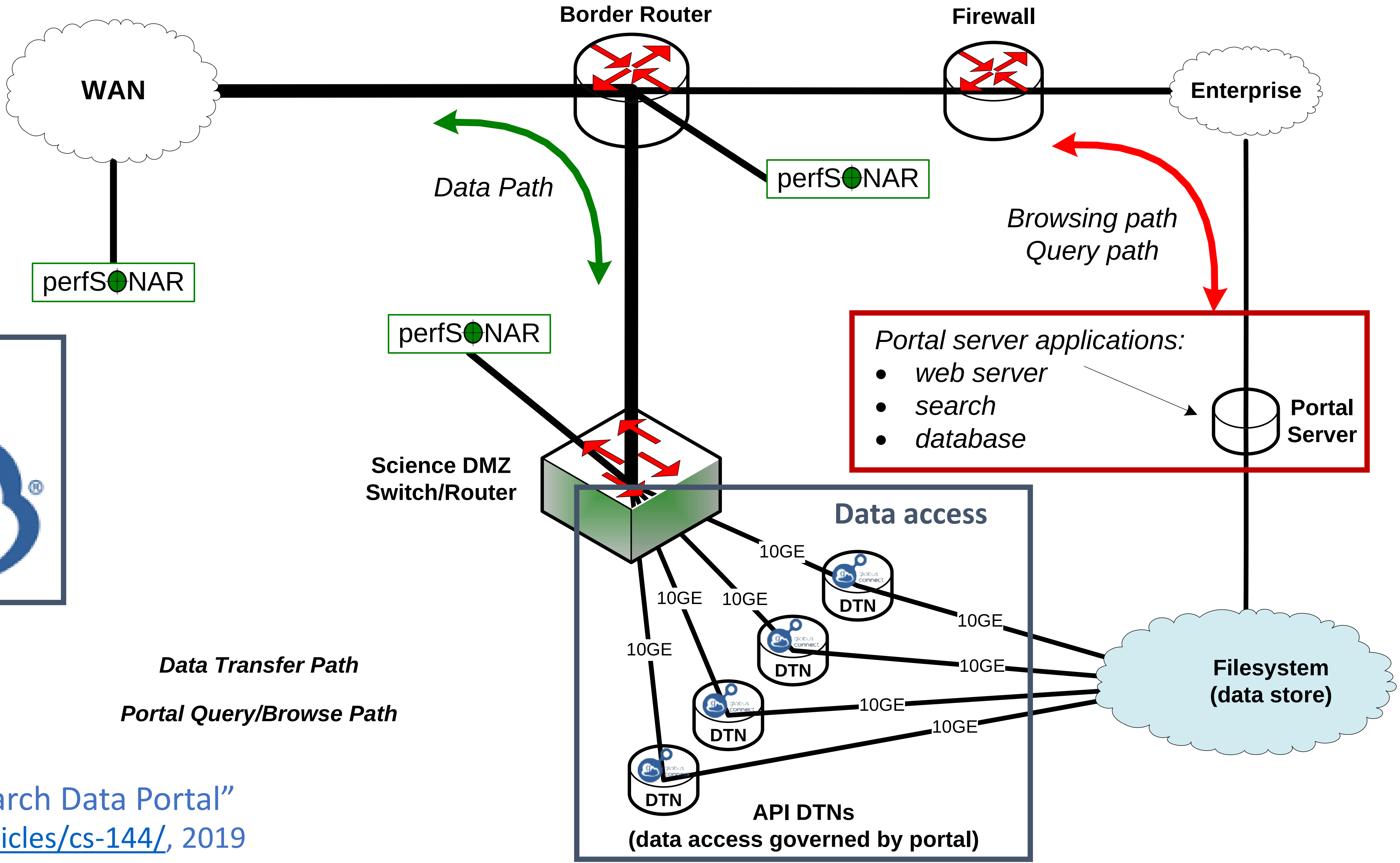
99.9%
availability

New mechanisms enable new design patterns



"The Modern Research Data Portal"
https://peerj.com/articles/cs-144/, 2019

New mechanisms enable new design patterns



Data Transfer Path

Portal Query/Browse Path

"The Modern Research Data Portal"
https://peerj.com/articles/cs-144/, 2019

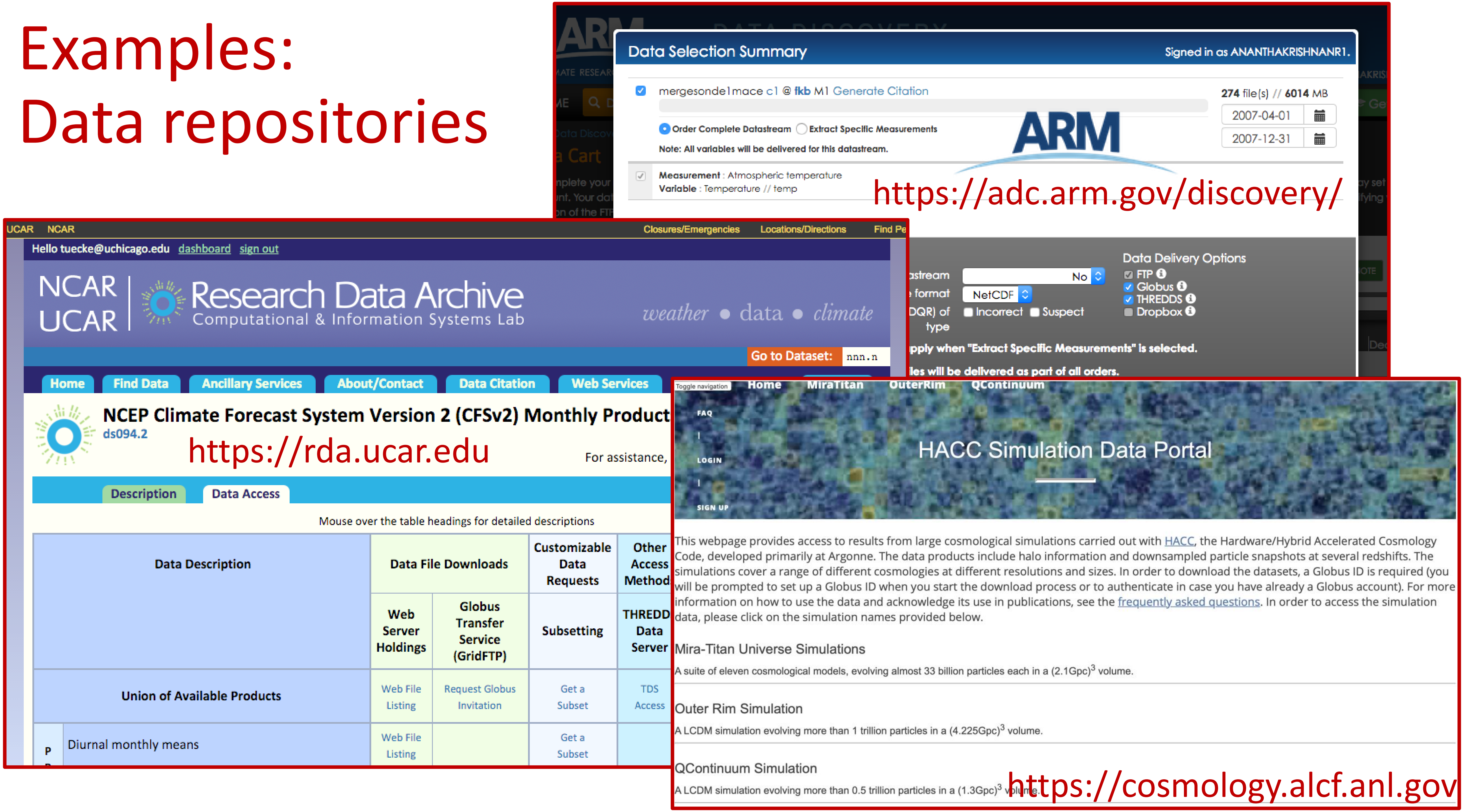
Implementation is trivial via Globus mechanisms

1. Create "shared endpoint"
2. Copy data to shared endpoint
3. Set permissions on shared endpoint for user; notify user
4. (Eventually) delete shared endpoint

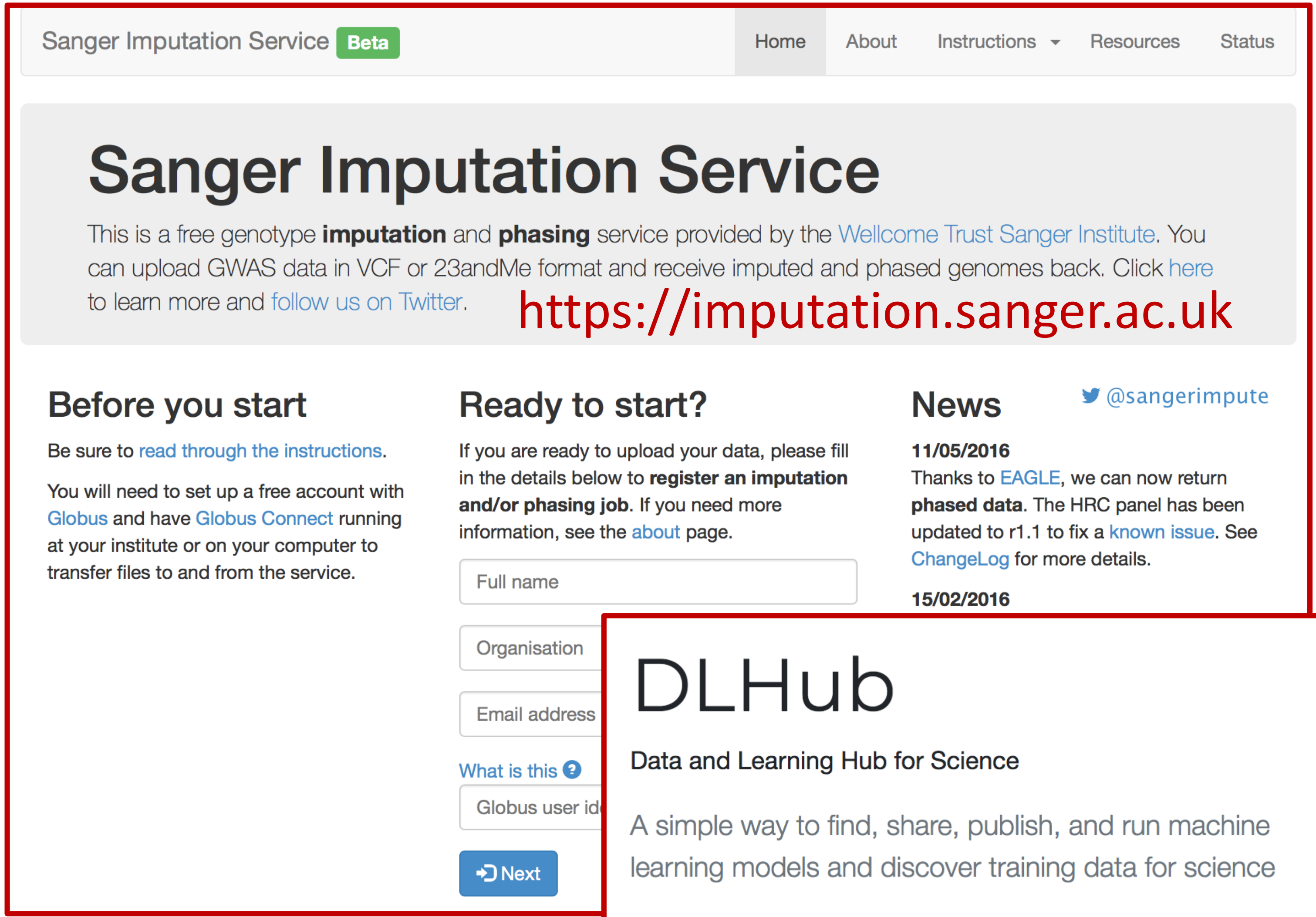
```
from globus_sdk import TransferClient, TransferData
from globus_sdk import AuthClient
import sys, random, uuid

def rdp(host_id,      # Endpoint for shared endpoint
        source_path,  # Directory to copy data from
        email):       # Email address to share with
    tc = TransferClient()
    ac = AuthClient()
    tc.endpoint_autoactivate(host_id)

    # (1) Create shared endpoint:
    # (a) Create directory to be shared
    share_path = '/~/' + str(uuid.uuid4()) + '/'
    tc.operation_mkdir(host_id, path=share_path)
    # (b) Create shared endpoint on directory
    shared_ep_data = {
      'DATA_TYPE': 'shared_endpoint',
      'host_endpoint': host_id,
      'host_path': share_path,
      'display_name': 'RDP shared endpoint',
      'description': 'RDP shared endpoint'
    }
    r = tc.create_shared_endpoint(shared_ep_data)
    share_id = r['id']
```

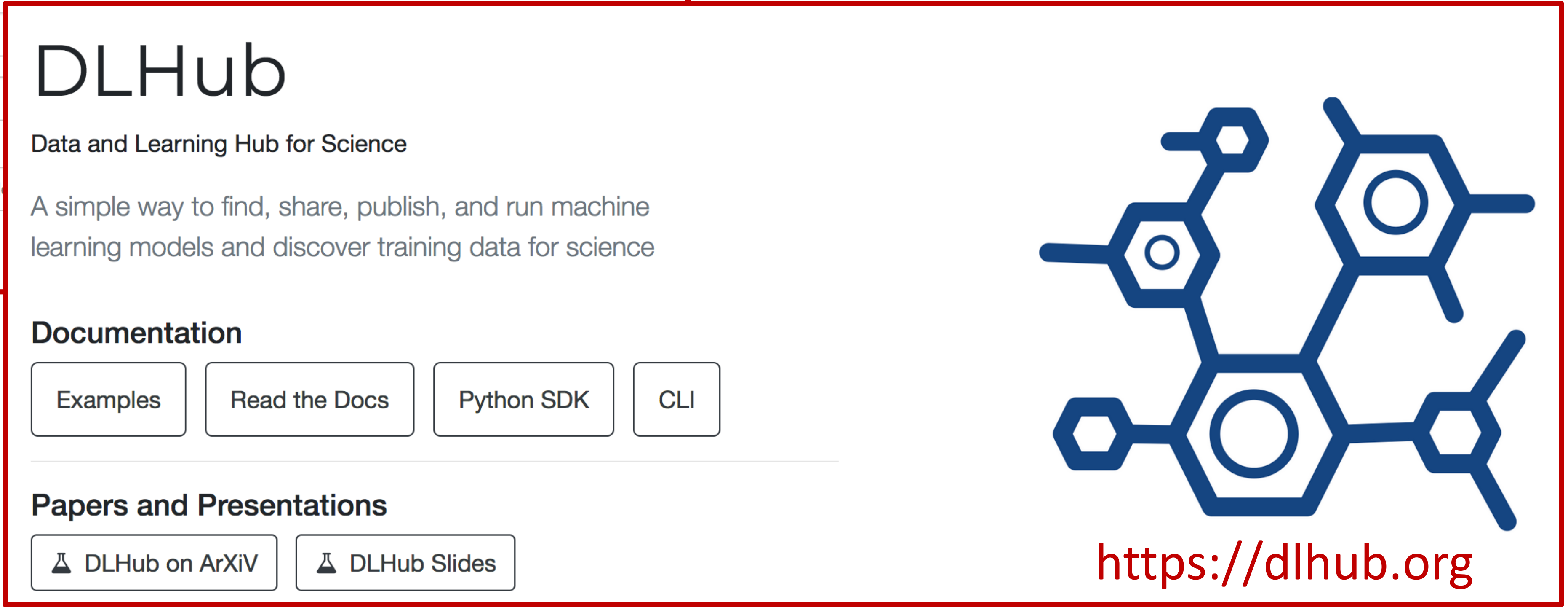
Connect to storage system

Examples:
Data repositories



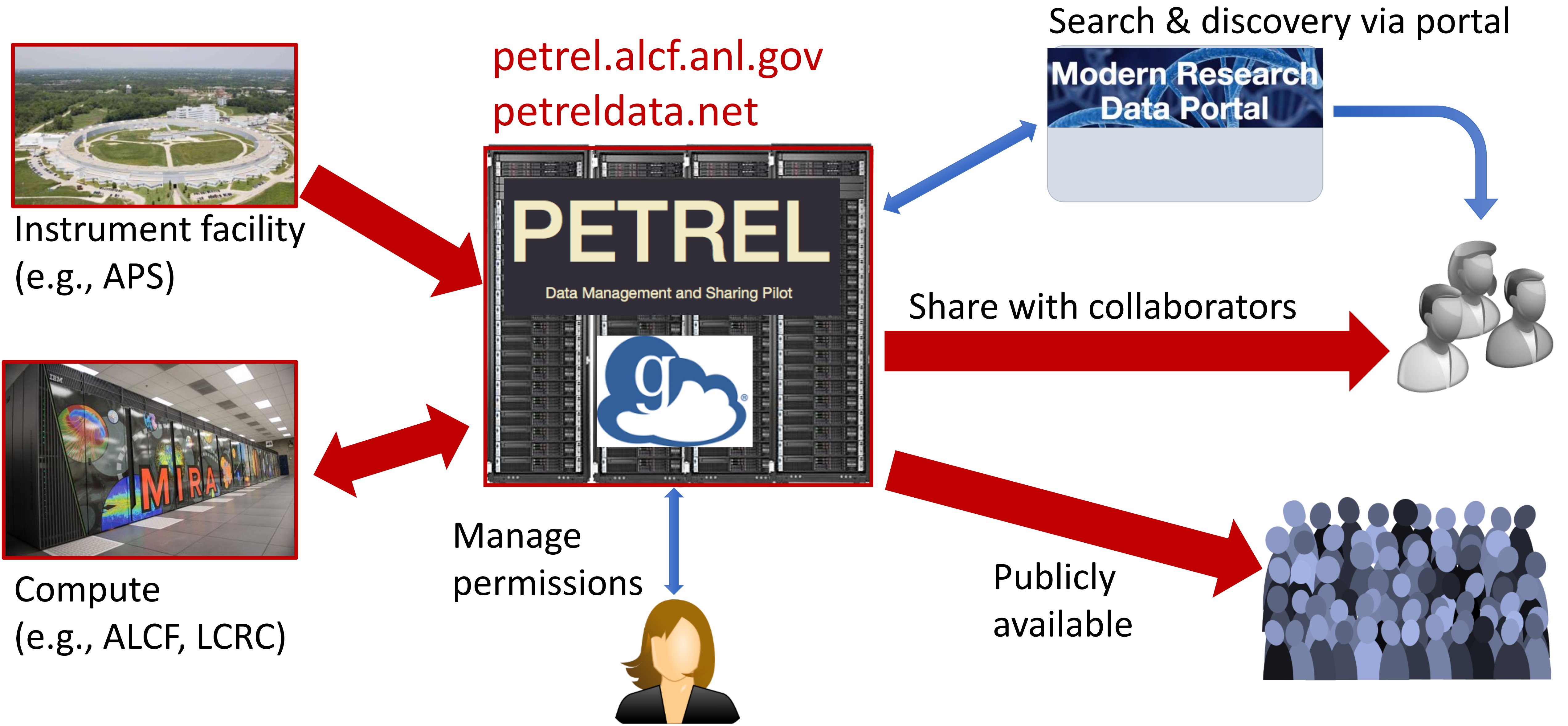
Examples: Analysis services





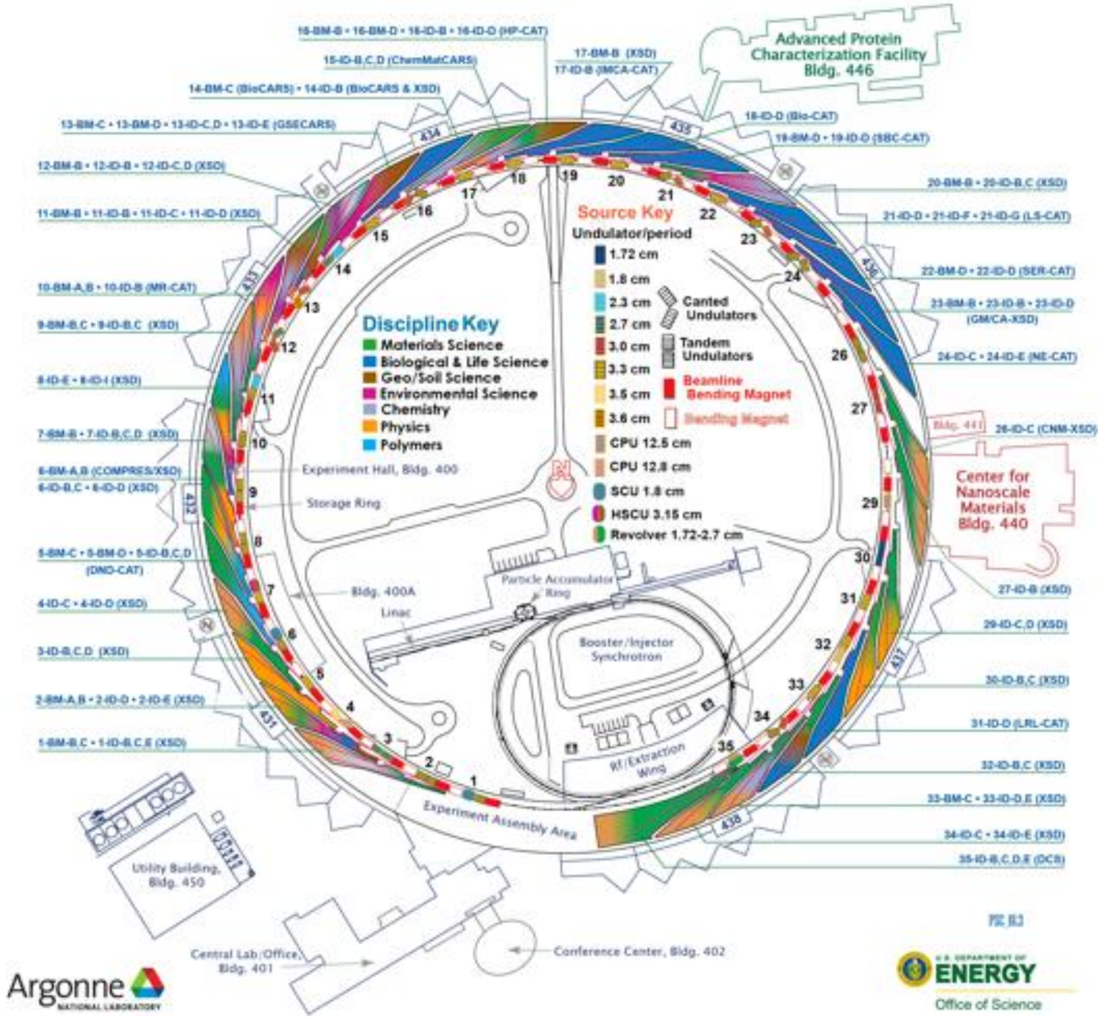
New mechanisms encourage higher-level capabilities

E.g., Petrel data sharing system



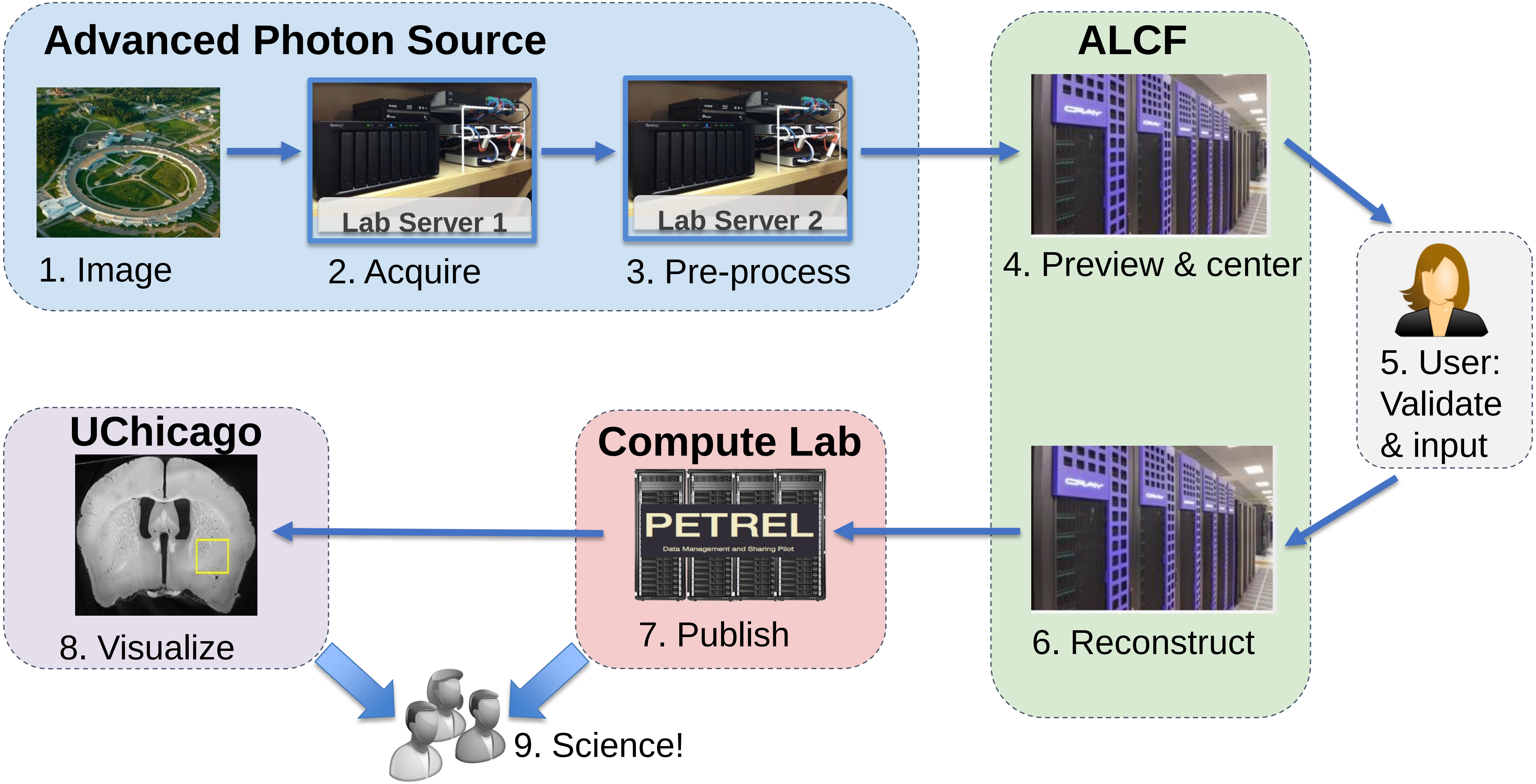
Examples: Instruments

DMagic: Automated data capture and distribution from tomography experiments

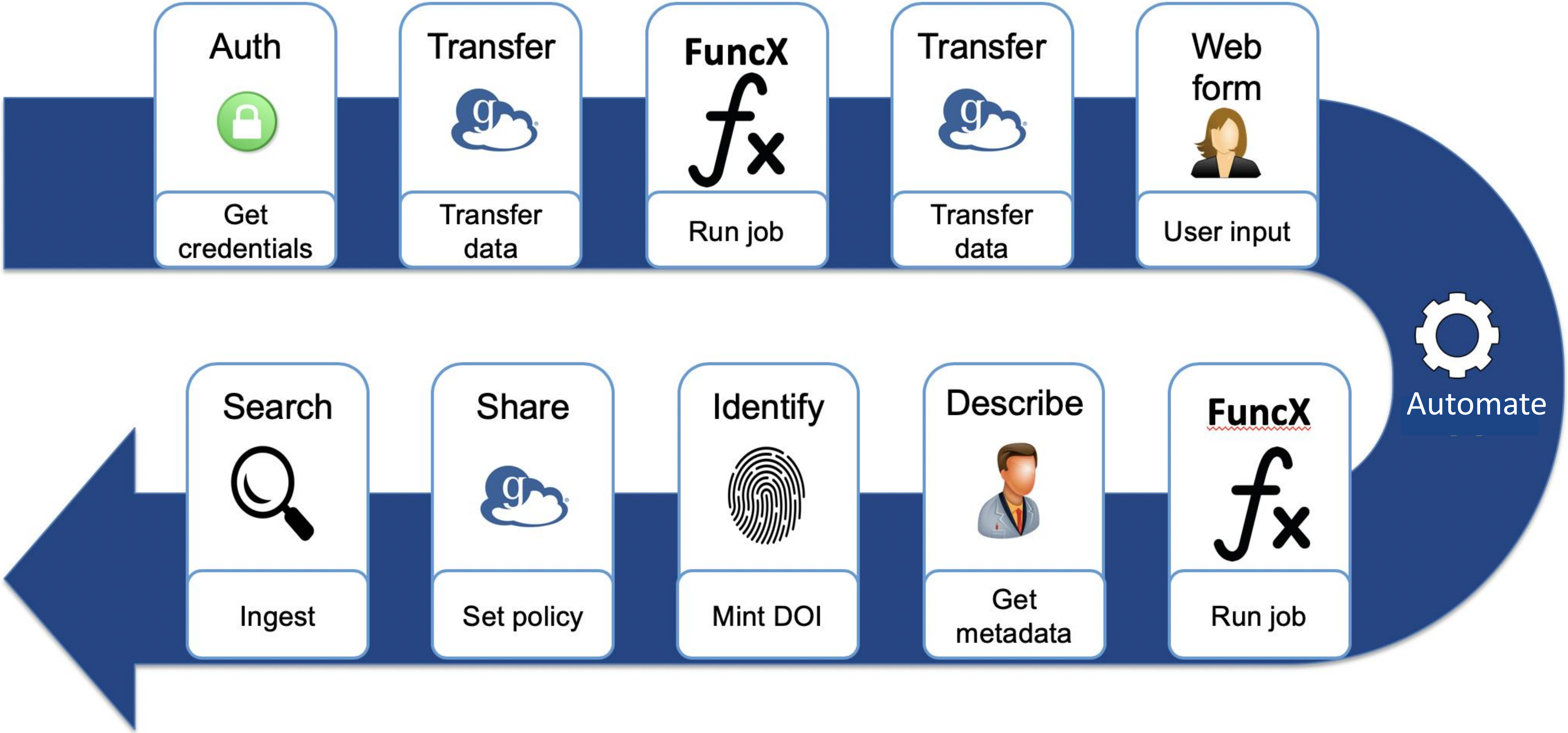


Francesco De Carlo http://dmagic.readthedocs.org

Flow automation in a neuroanatomy automation



Globus Automate: Event/rule-driven automation



Petrel by the numbers: January 2020

4.3 TB
largest file

857 Million
files moved

1.73 PB
data stored

683
total users

8.32 PB
transferred

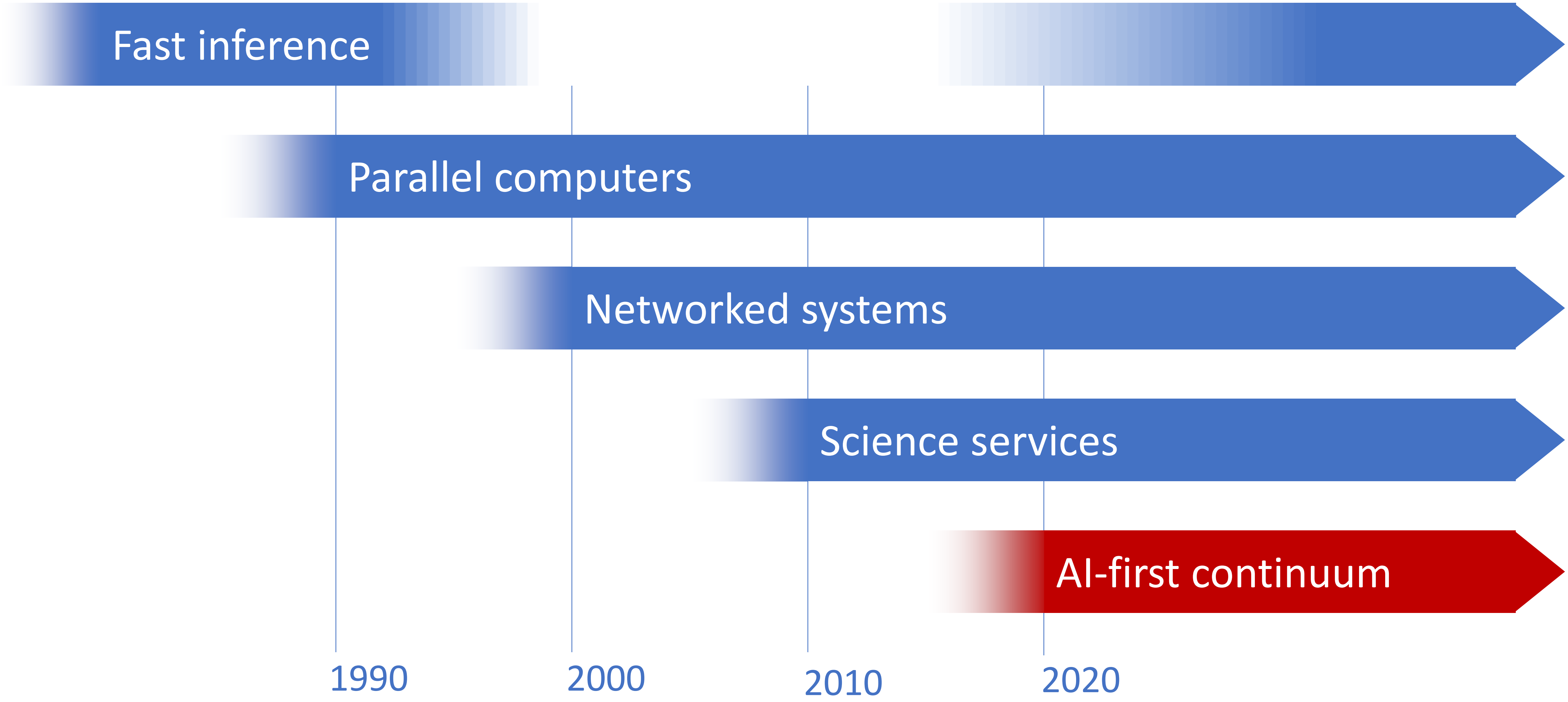
252 TB
largest project

312 users of single project

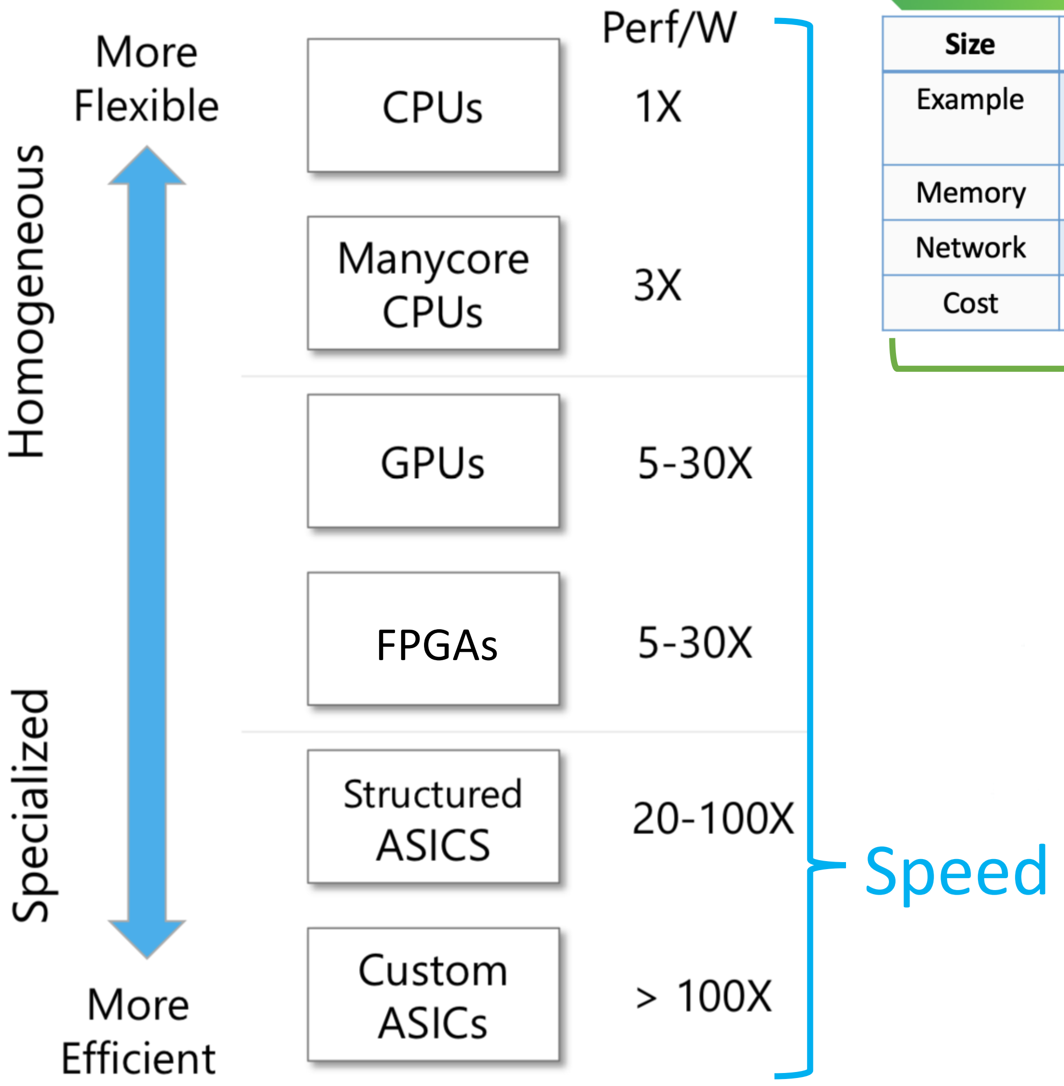
108,490
transfer tasks

47
projects

Areas of opportunity for scientific mechanisms



A rapidly evolving computing/data continuum



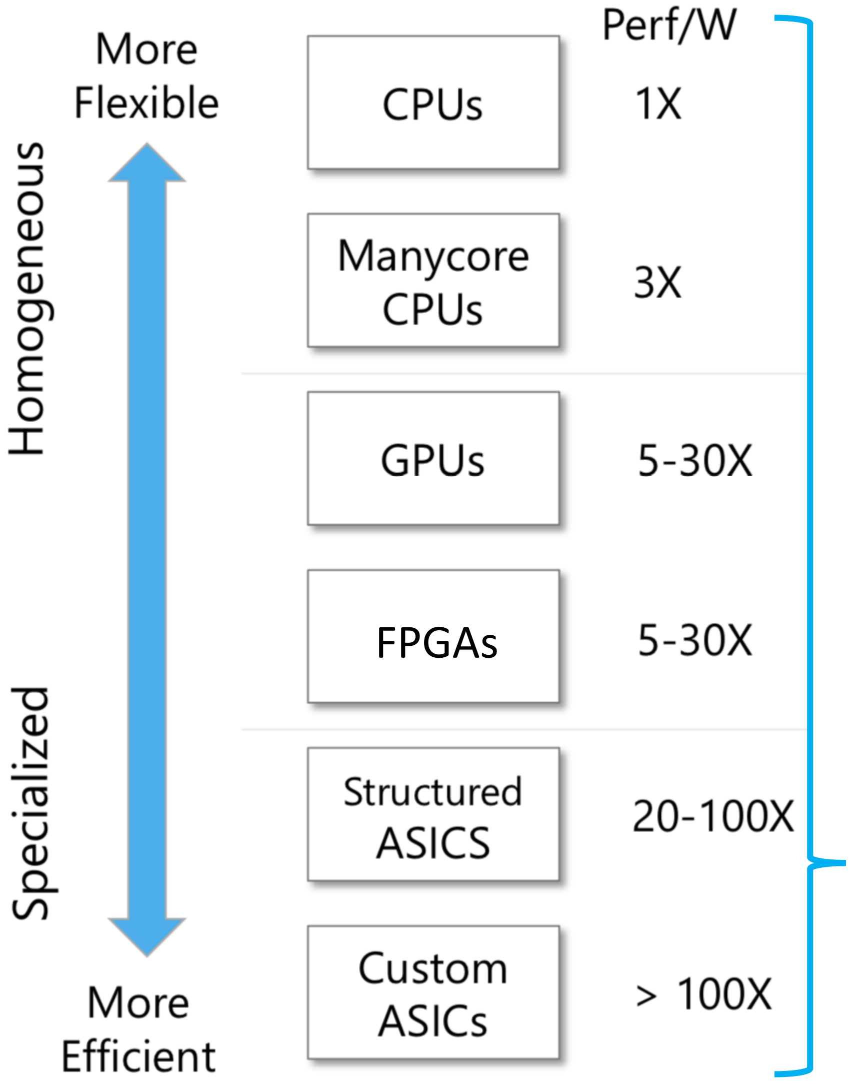
IoT/Edge — Fog — HPC/Cloud

Size	Nano	Micro	Milli	Server	Fog	Campus	Facility
Example	Adafruit Trinket	Particle.io Boron	Array of Things	Linux Box	Co-located Blades	1000-node cluster	Datacenter
Memory	0.5K	256K	8GB	32GB	256G	32TB	16PB
Network	BLE	WiFi/LTE	WiFi/LTE	1 GigE	10GigE	40GigE	N*100GigE
Cost	$5	$30	$600	$3K	$50K	$2M	$1000M

Size

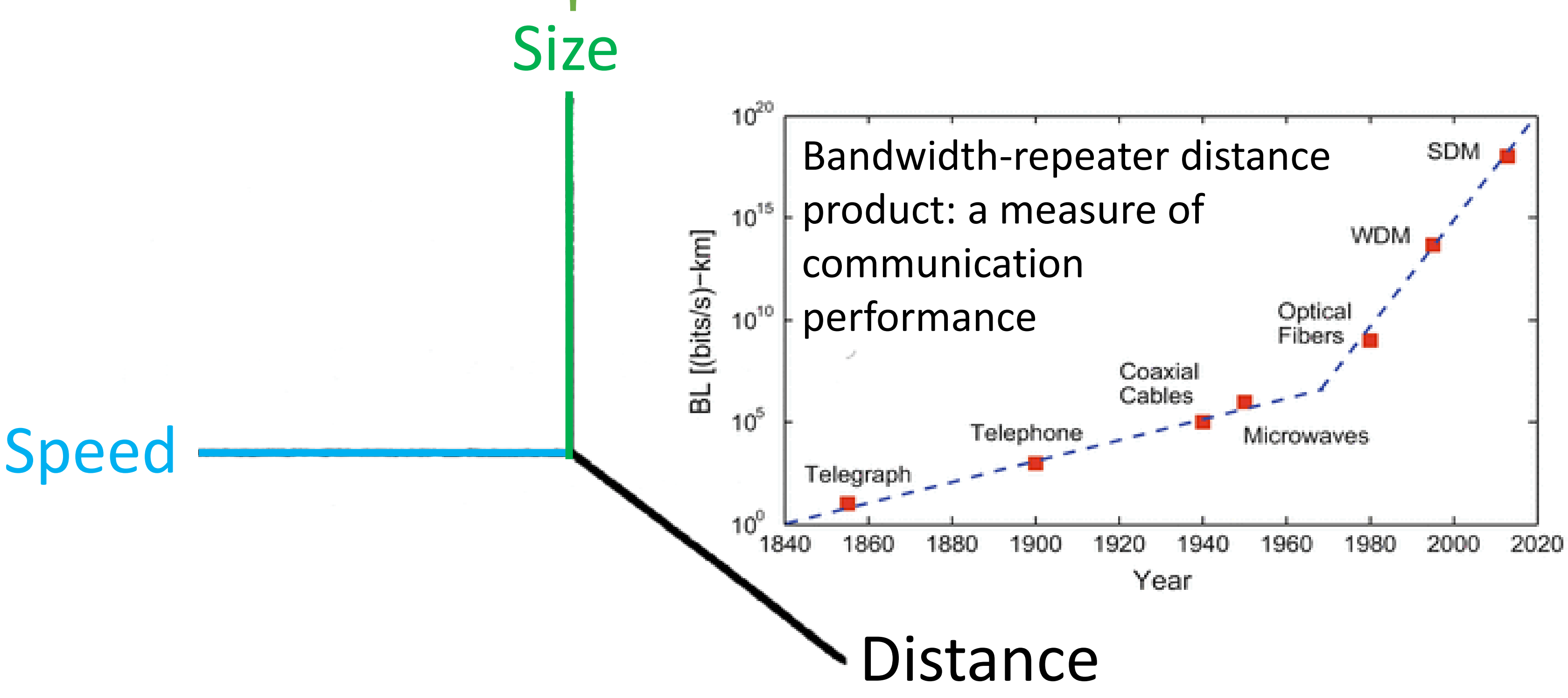
Sources: http://bit.ly/2SDGHzT, https://doi.org/10.1007/978-3-319-31903-2_8, Pete Beckman

A rapidly evolving computing/data continuum



IoT/Edge — Fog — HPC/Cloud

Size	Nano	Micro	Milli	Server	Fog	Campus	Facility
Example	Adafruit Trinket	Particle.io Boron	Array of Things	Linux Box	Co-located Blades	1000-node cluster	Datacenter
Memory	0.5K	256K	8GB	32GB	256G	32TB	16PB
Network	BLE	WiFi/LTE	WiFi/LTE	1 GigE	10GigE	40GigE	N*100GigE
Cost	$5	$30	$600	$3K	$50K	$2M	$1000M

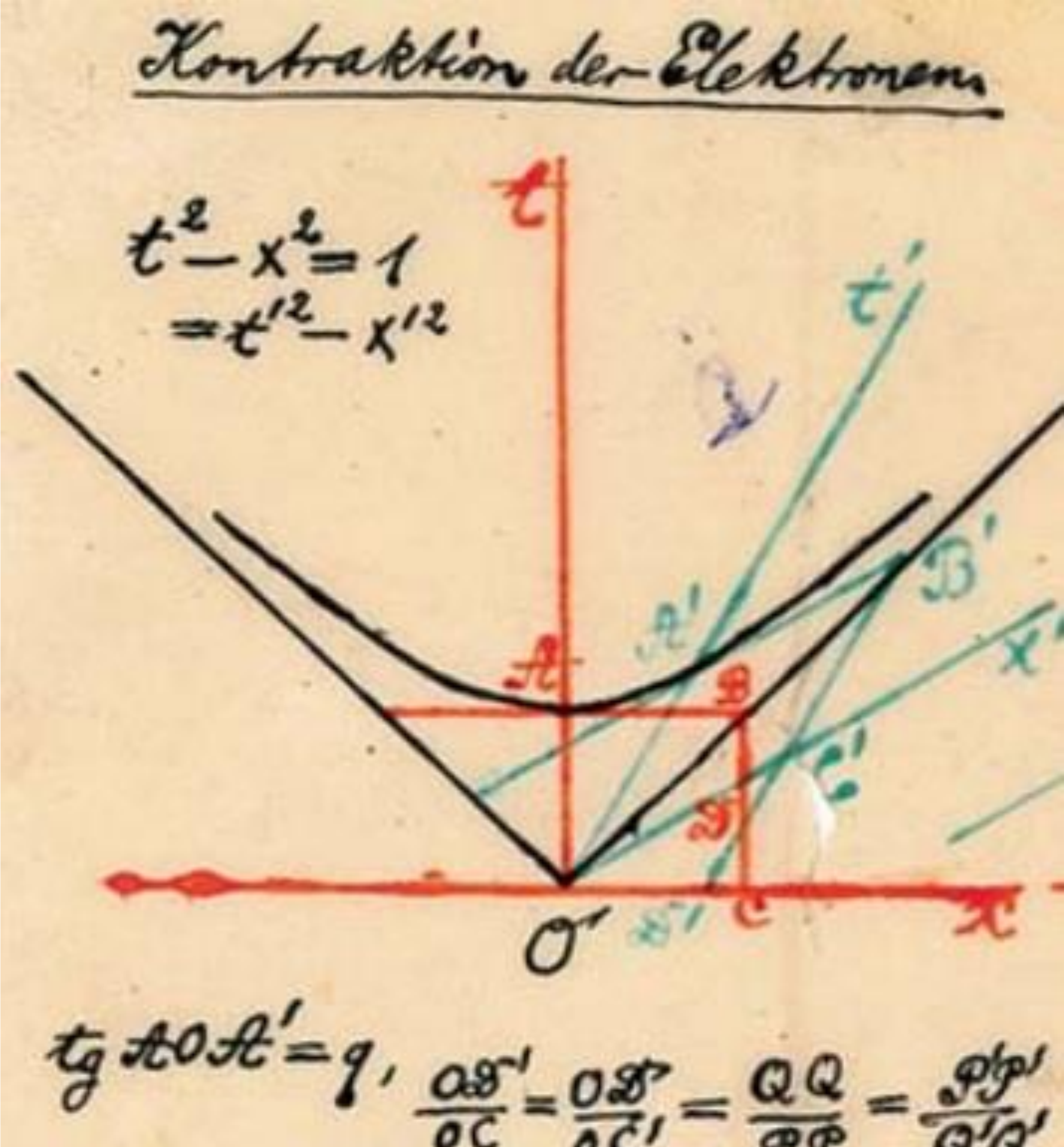


Sources: http://bit.ly/2SDGHzT, https://doi.org/10.1007/978-3-319-31903-2_8, Pete Beckman

The
space-time continuum

“space by itself, and time by itself, are doomed to fade away into mere shadows, and only a kind of union of the two will preserve an independent reality …”

H. Minkowski, 1908



Space-time diagram
https://en.wikipedia.org/wiki/Spacetime

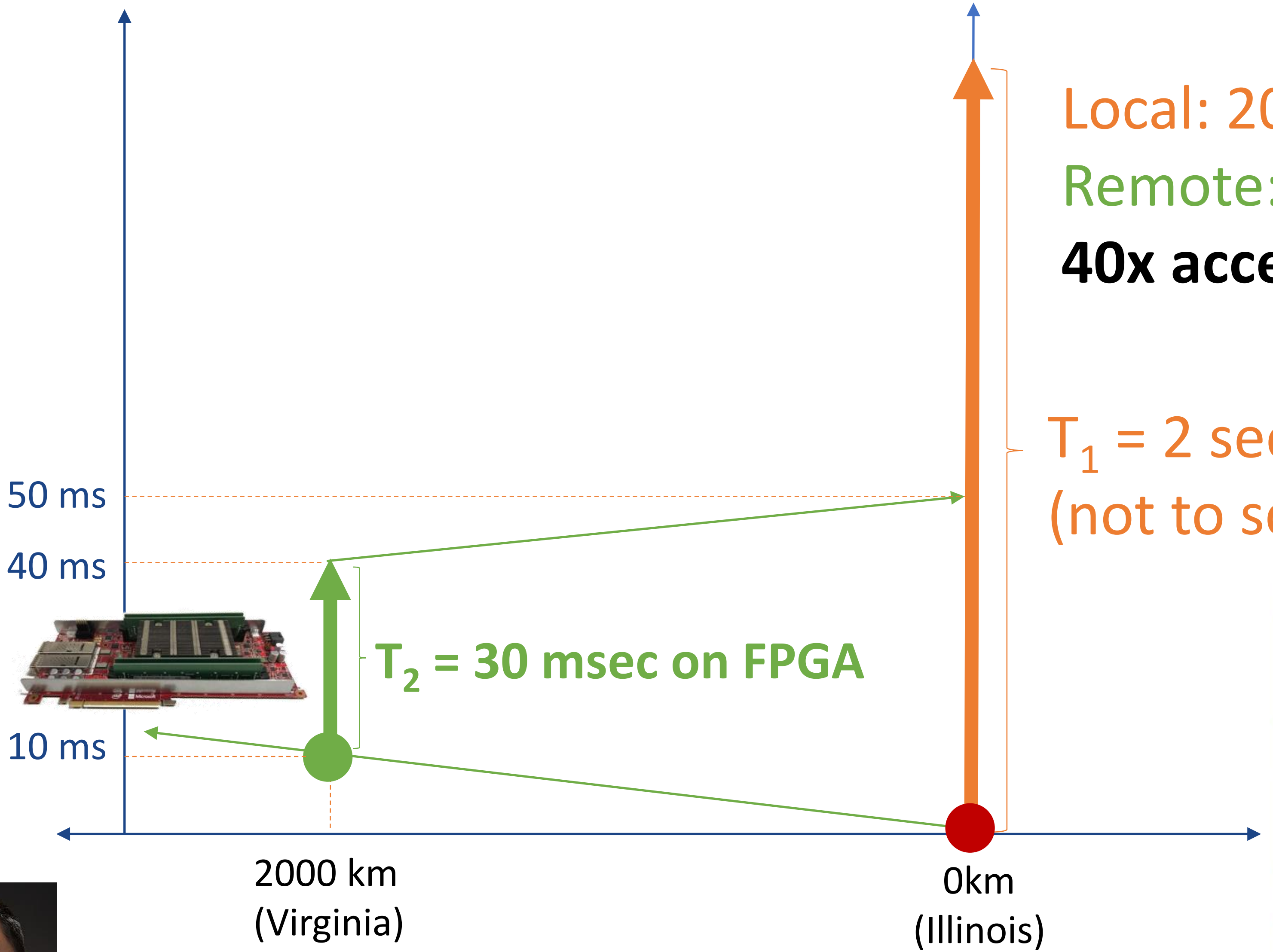
For example: High energy physics trigger analysis

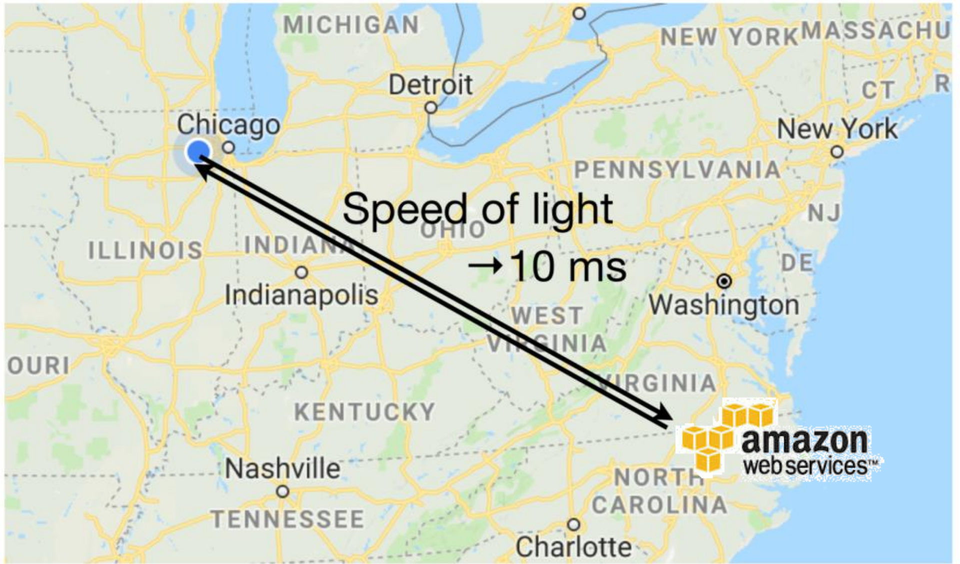
Local: 2000 msec

Remote: 30 + 10 + 10 = 50 msec

40x acceleration

T_1 = 2 seconds on CPU
(not to scale)





Nhan Tran, FermiLab, et al. arXiv:1904.08986

For example: High energy physics trigger analysis

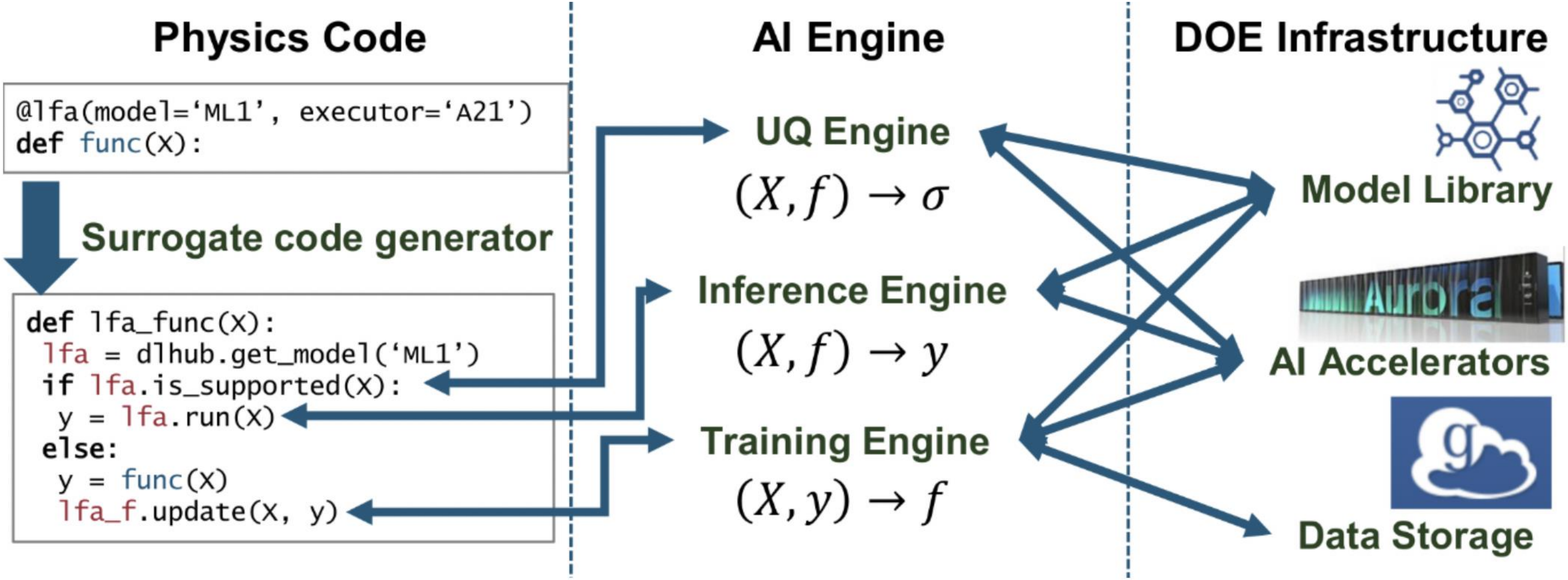
Misquoting Minkowski:

"**location** by itself, and **speed** by itself, are doomed to fade away into mere shadows, and only a kind of union of the two will preserve an independent reality …"

Nhan Tran, FermiLab, et al. arXiv:1904.08986

Robust Learned Function Accelerators (RLFAs)

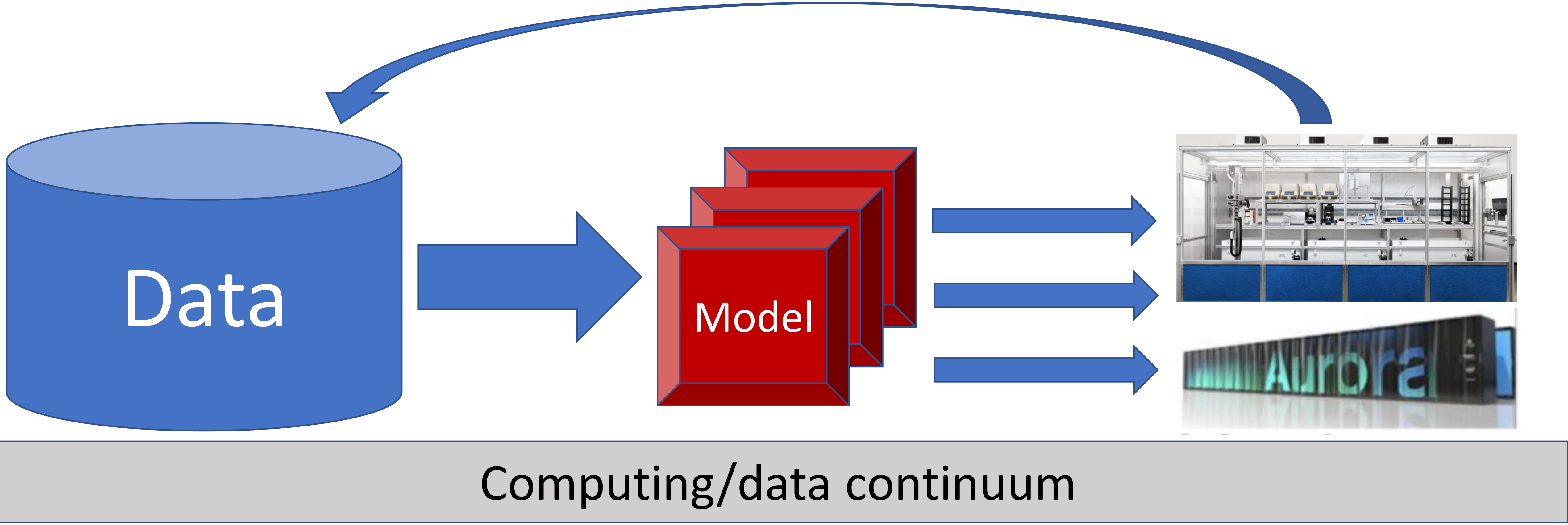
Fluidity between simulations and learned models



Logan Ward, Ben Blaiszik, et al.

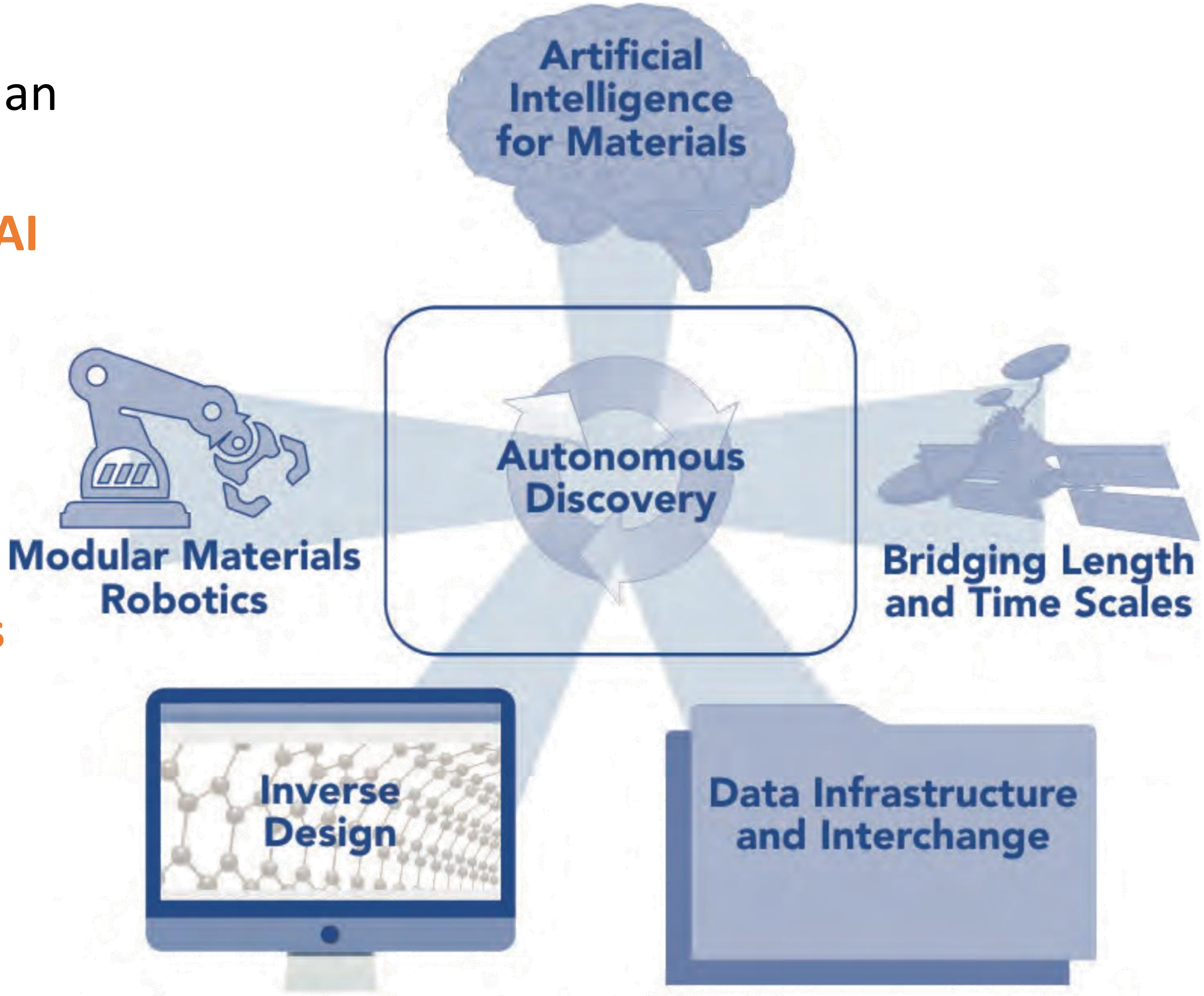
Intelligent storage systems

Fluidity between data and learned models



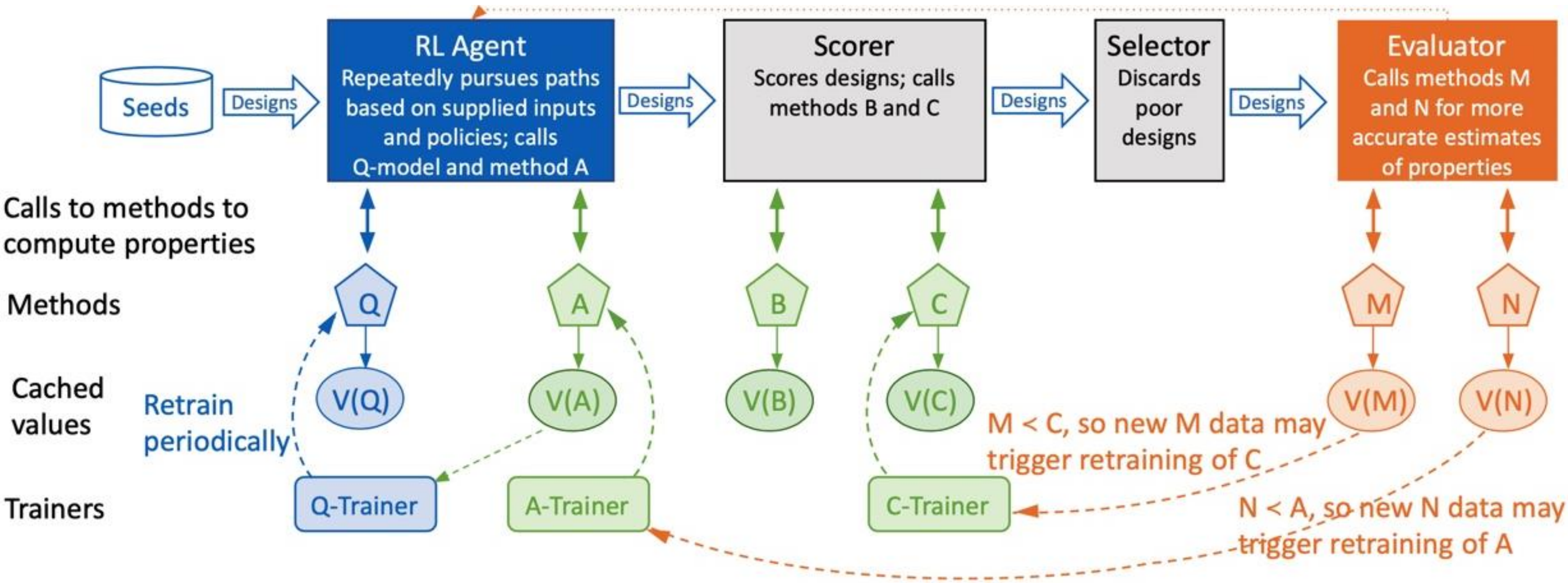
New methods: E.g., Materials Acceleration Platform

1) **Self-driving laboratories** that design, perform, and interpret experiments in an automated way
2) The development of **specific forms of AI for materials discovery**
3) **Modular materials robotics platforms** that can be assemblies of modular building blocks for synthesis and characterization
4) Research into **computational methods for inverse design**
5) New methodologies for **bridging the length and timescales** associated with materials simulation
6) Sophisticated **data infrastructure and interchange platforms**



Report: Materials Acceleration Platform, 2017 – http://bit.ly/2Z7kp9Q

Reinforcement learning pipeline for electrolyte design



A dynamic mix of simulation, model training, and inference, scaling to exascale

Project Celerity*
(My “Office of Science Distinguished Scientists Fellow” proposal)

Identity new mechanisms needed to bridge the gap between new (especially “AI-first”) scientific applications and the emerging data/computing continuum

Via a process of experimentation, discussion, and debate

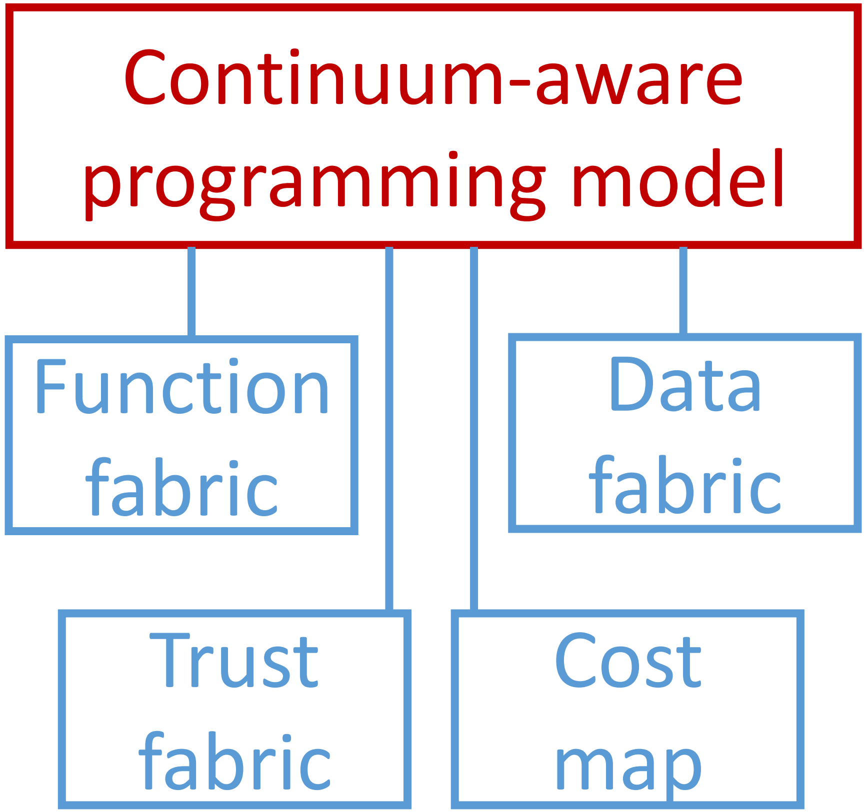
* **Celerity** (n) rapidity; swiftness; speed – from Latin celeritas, from which also **c** for speed of light in vacuum

Bridging the gap

What mechanisms will facilitate the programming of this distributed, heterogeneous, dynamically evolving AI-first continuum?

Some things that we surely need:

- **Function**: Compute wherever is fastest, cheapest, closest, most accurate, …
- **Data**: Access where fastest, cheapest, closest, most accurate, …
- **Trust**: Balance certainty vs. cost
- **Cost**: Useful estimates of the state of this dynamic system



Coding the continuum: Elements of an open solution

Thanks to colleagues, especially: globus labs


Rachana Ananthakrishnan


Yadu Babuji


Ben Blaiszik


Kyle Chard


Ryan Chard


Zhuozhao Li


Tyler Skluzacek


Steve Tuecke

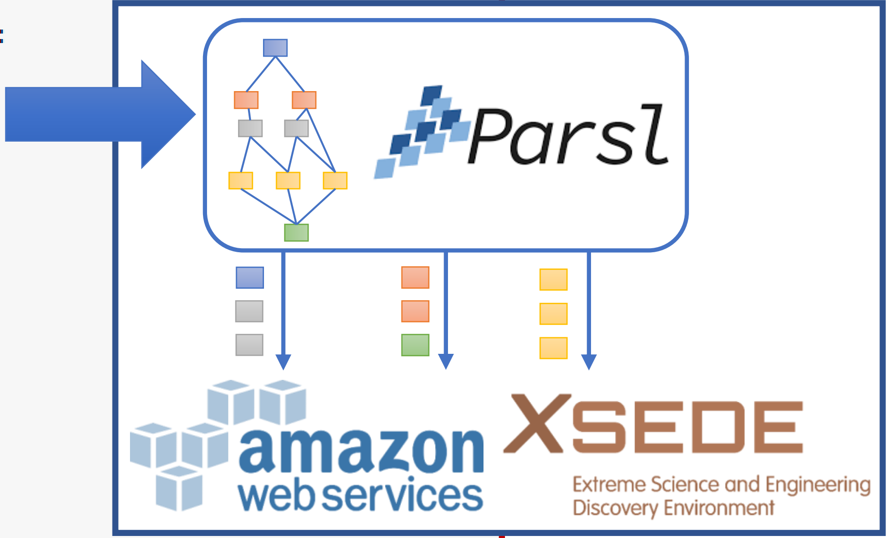

Anna Woodard


Logan Ward

Model registry	DLHub
Flows	Automate
Write programs	Parsl
Cost map	SCRIMP
Function fabric	funcX
Data fabric	Data services
Trust fabric	Auth

Coding the continuum: Elements of an open solution

```python
# App that estimates pi by placing points in a box
@python_app
def pi(total):
    import random

    # Set the size of the box (edge length) in which we drop random points
    edge_length = 10000
    center = edge_length / 2
    c2  = center ** 2
    count = 0

    for i in range(total):
        # Drop a random point in the box.
        x,y = random.randint(1, edge_length),random.randint(1, edge_length)
        # Count points within the circle
        if (x-center)**2 + (y-center)**2 < c2:
            count += 1

    return (count*4/total)

# App that computes the average of the values
@python_app
def avg_points(a, b, c):
    return (a + b + c)/3

# Estimate three values for pi
a, b, c = pi(10**6), pi(10**6), pi(10**6)

# Compute the average of the three estimates
avg_pi  = avg_points(a, b, c)
```



https://arxiv.org/pdf/1905.02158

http://parsl-project.org

Model registry	DLHub
Flows	Automate
Write programs	**Parsl**
Cost map	SCRIMP
Function fabric	funcX
Data fabric	Data services
Trust fabric	Auth

Coding the continuum: Elements of an open solution

```
In [1]: from funcx_sdk.client import FuncXClient

        fxc = FuncXClient()

In [2]: func = """
        def add(data):
            sum_val = sum(data['data'])
            return sum_val
        """

In [3]: fxc.register_function("add_func", func, description="Sum a list of numbers.")

In [4]: input_data = [1, 2, 3]

        res = fxc.run(input_data, "user#laptop", "add_func")

In [5]: print(res)

        6
```

Portable code

Python
Docker, Shifter, Singularity

→

Any access

SSH, Globus, cluster or HPC scheduler

→

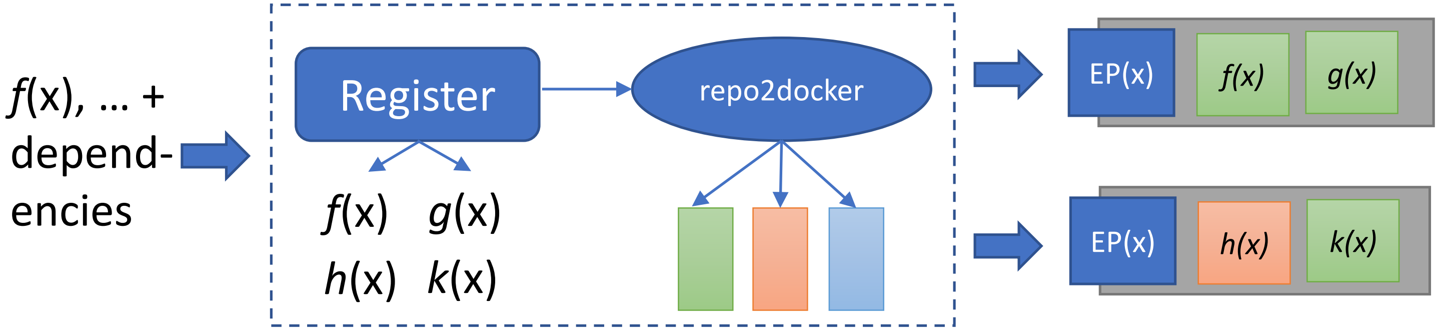
Any computer

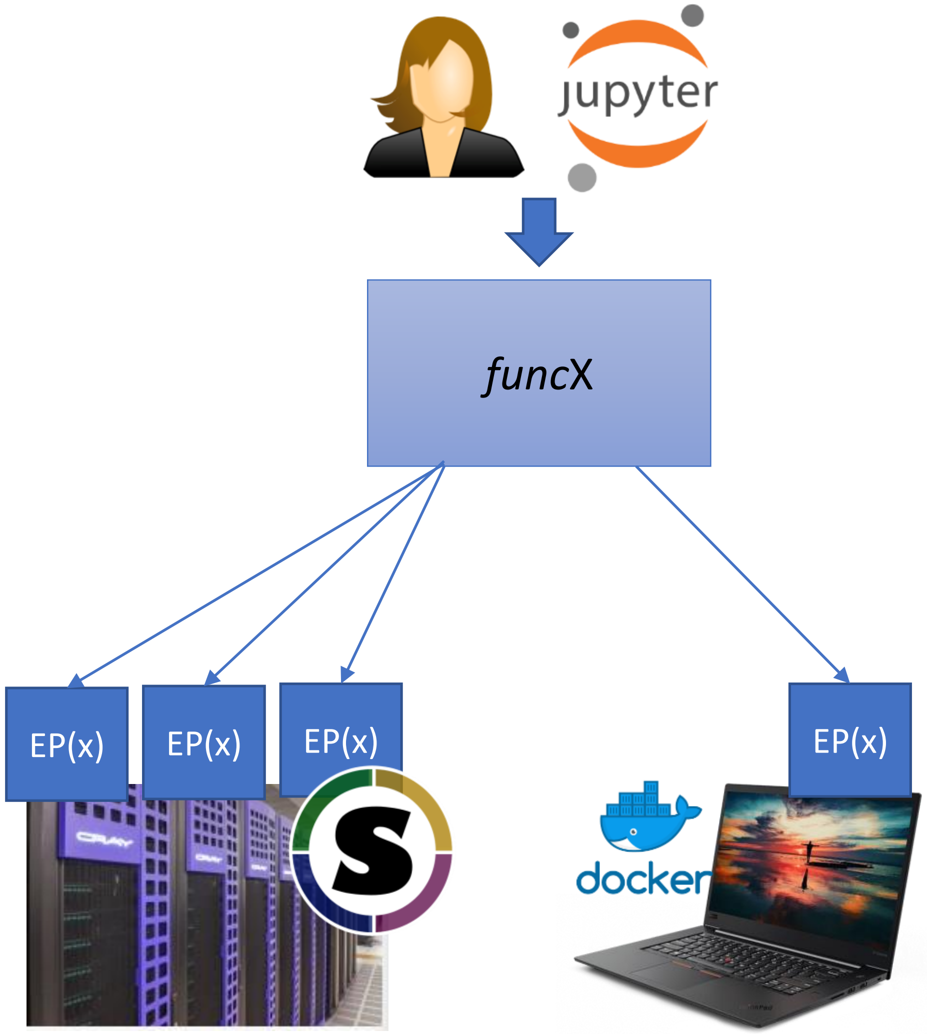
Clusters, clouds, HPC, accelerators

Model registry	DLHub
Flows	Automate
Write programs	Parsl
Cost map	SCRIMP
Function fabric	funcX
Data fabric	Data services
Trust fabric	Auth

funcX: Transform clouds, clusters, and supercomputers into high-performance function serving systems

```
In [1]: from funcx_sdk.client import FuncXClient

        fxc = FuncXClient()

In [2]: func = """
        def add(data):
            sum_val = sum(data['data'])
            return sum_val
        """

In [3]: fxc.register_function("add_func", func, description="Sum a list of numbers.")

In [4]: input_data = [1, 2, 3]

        res = fxc.run(input_data, "user#laptop", "add_func")

In [5]: print(res)

        6
```

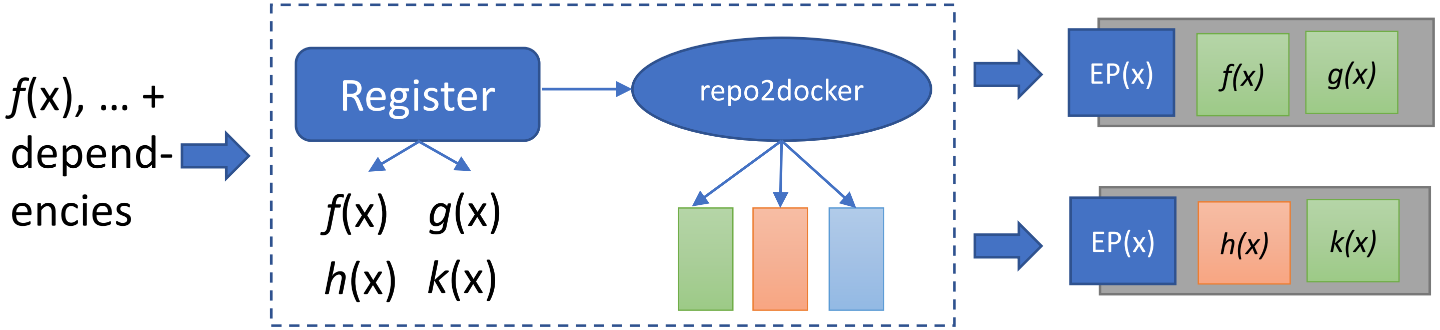


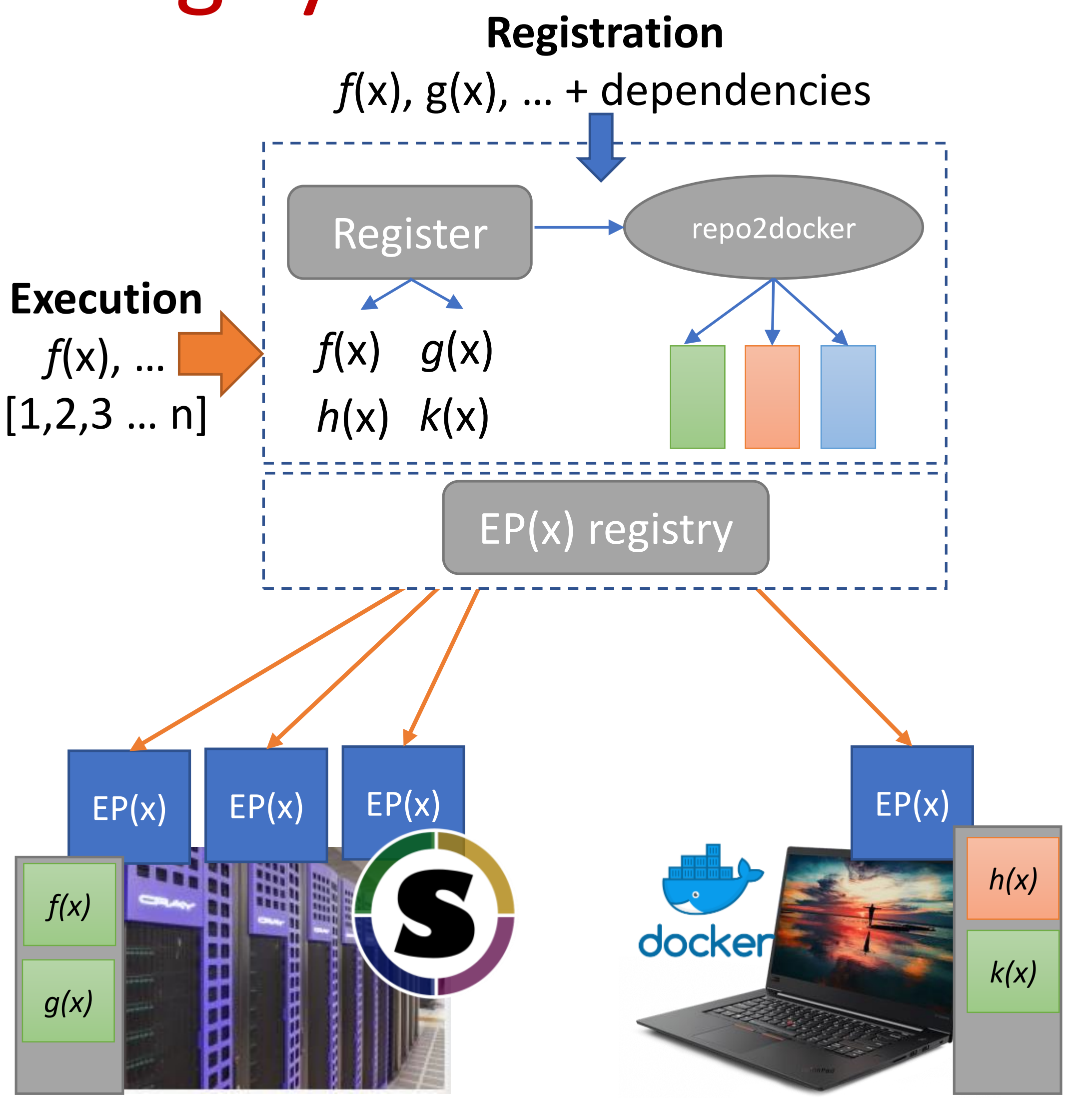


Simply deploy *funcX* endpoint to transform a computer into a function serving system

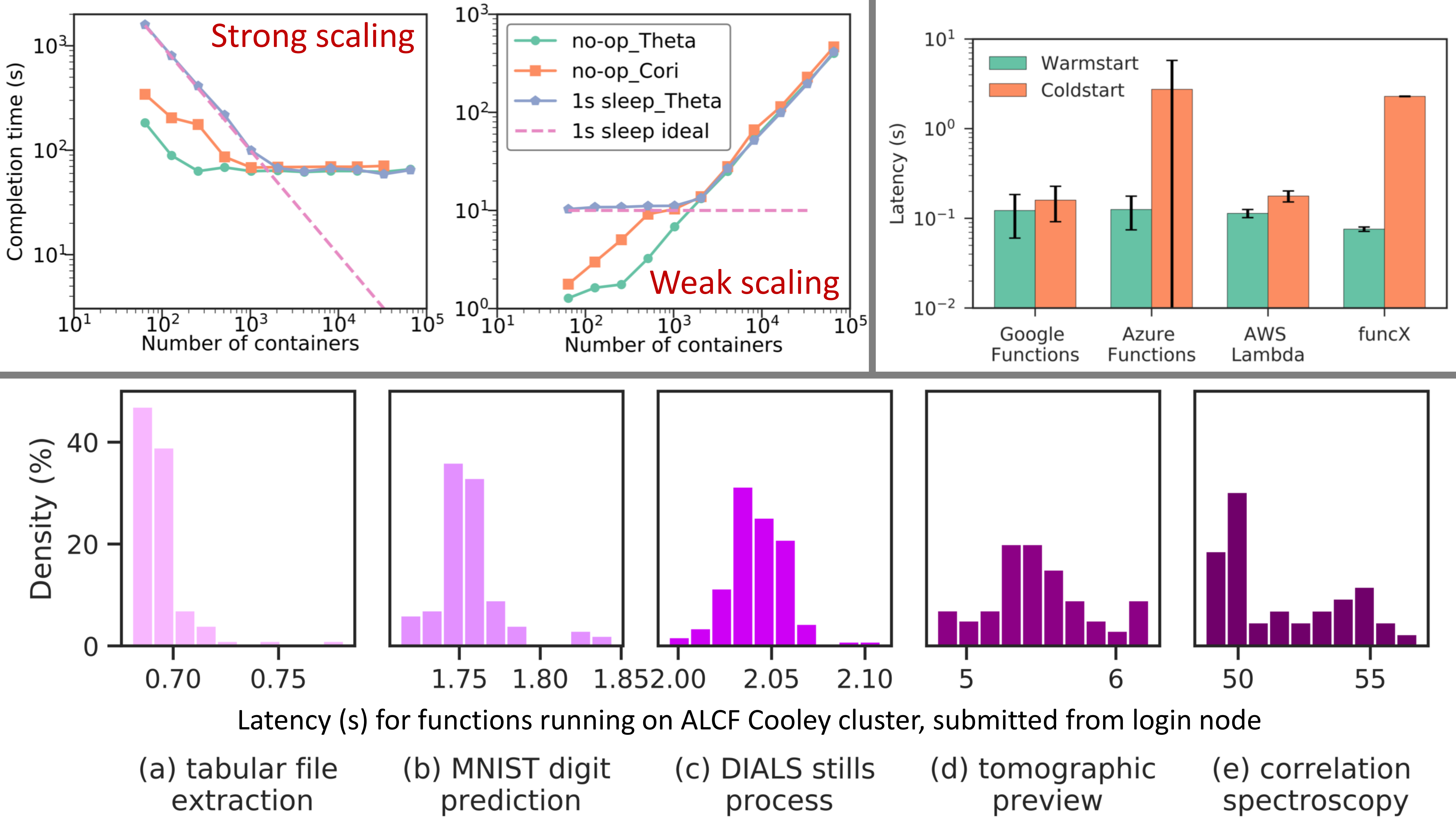
funcX: Transform clouds, clusters, and supercomputers into high-performance function serving systems

```
In [1]: from funcx_sdk.client import FuncXClient

        fxc = FuncXClient()

In [2]: func = """
        def add(data):
            sum_val = sum(data['data'])
            return sum_val
        """

In [3]: fxc.register_function("add_func", func, description="Sum a list of numbers.")

In [4]: input_data = [1, 2, 3]

        res = fxc.run(input_data, "user#laptop", "add_func")

In [5]: print(res)

        6
```





Simply deploy *funcX* endpoint to transform a computer into a function serving system



Strong scaling
Completion time (s)
Number of containers
no-op_Theta
no-op_Cori
1s sleep_Theta
1s sleep ideal
Weak scaling
Number of containers
Warmstart
Coldstart
Latency (s)
Google Functions
Azure Functions
AWS Lambda
funcX
Density (%)
Latency (s) for functions running on ALCF Cooley cluster, submitted from login node
(a) tabular file extraction
(b) MNIST digit prediction
(c) DIALS stills process
(d) tomographic preview
(e) correlation spectroscopy

DLHub: Organizing and Serving Models

DLHub
Data and Learning Hub for Science
https://www.dlhub.org

- Collect, publish, categorize models
- Serve models via API with access controls to simplify sharing, consumption, and access
- Leverage ALCF resources and prepare for Exascale ML
- Deploy and scale automatically
- Citable DOIs for reproducible science

Models and Processing Logic as a Service

Describe →

- Specify the model files
- Mark up the model with information to make it discoverable and usable

```
from dlhub_sdk.models.servables.keras import KerasModel
m = KerasModel.create_model("p1b1-example.h5")

m.set_title("CANDLE Pilot 1 - Benchmark 1")
m.set_name("candle_p1b1")
m.set_domains("genomics","biology","HPC")
```

Publish →

- Register with DLHub for containerization as a servable
- DLHub service creates unique endpoint for servable

```
from dlhub_sdk.client import DLHubClient
dl = DLHubClient()
dl.publish(m)
```

Discover

- Discover servables with advanced search capabilities through Python SDK or web UI

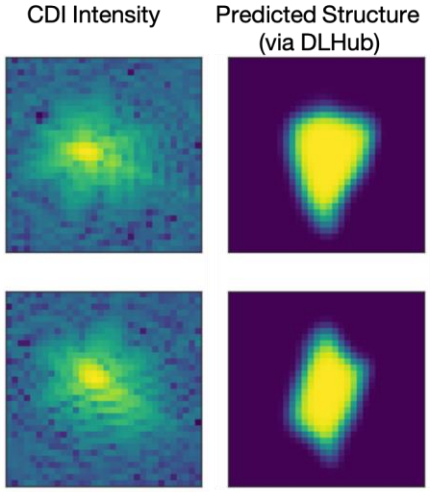
Run

- Make predictions by sending data to DLHub and specifying the servable to use

```
from dlhub_sdk.client import DLHubClient
dl = DLHubClient()
pred = dl.run("candle_p1b1", data)
```

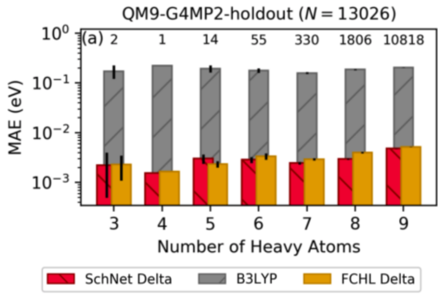
X-Ray Science

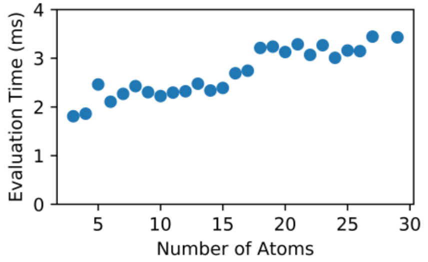
```
from dlhub_sdk.client import DLHubClient
dl = DLHubClient()

struct = dl.run("cherukara_structure", X)
```



Cherukara et al.

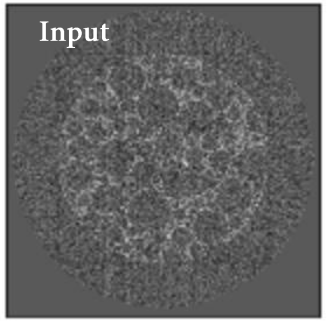
Energy Storage

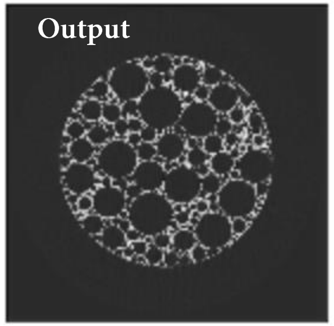




Ward et al.

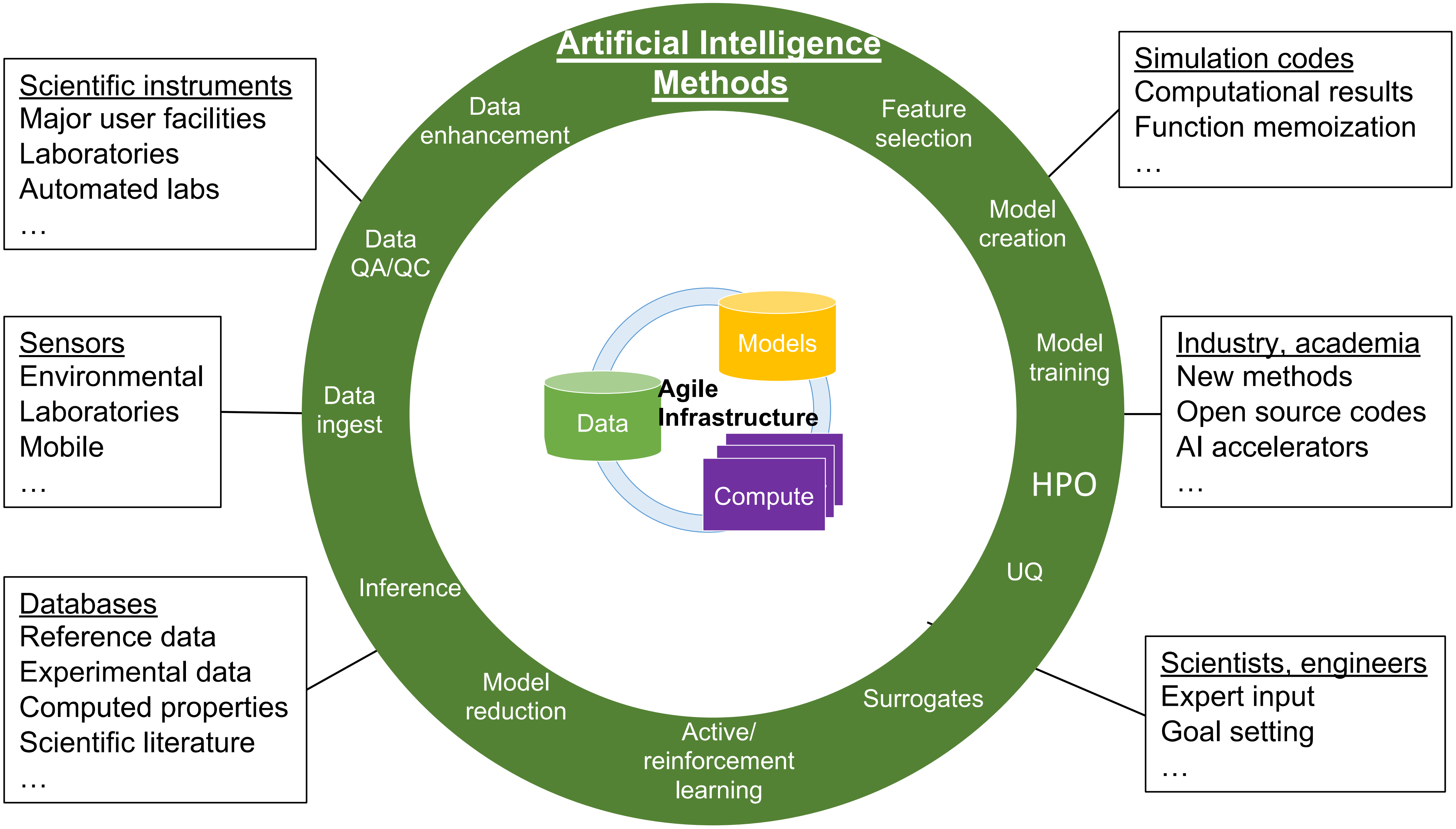
Tomography



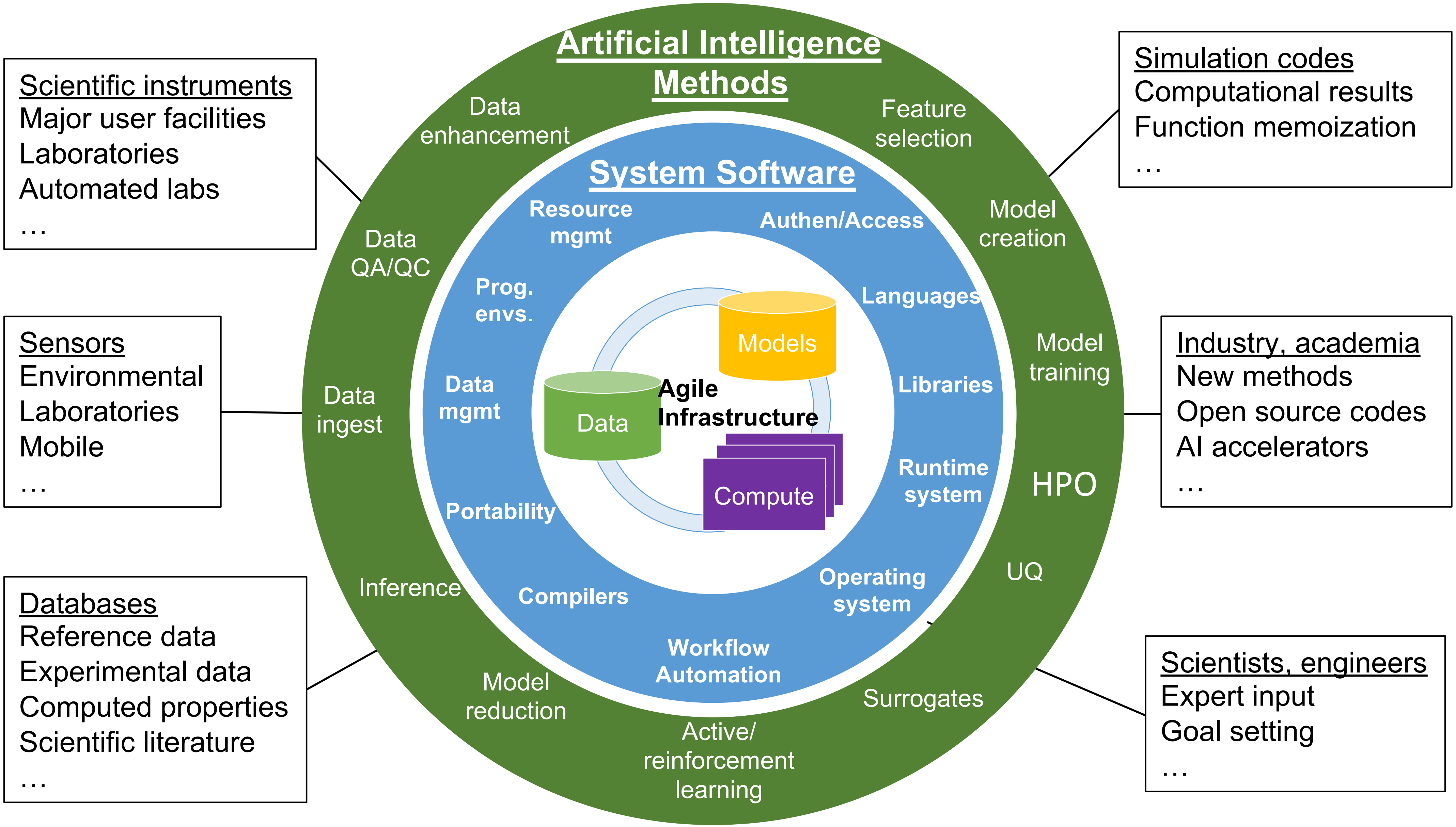


TomoGAN: Liu et al.

New mechanisms enable new methods



They also encourage rethinking existing technologies



Summary

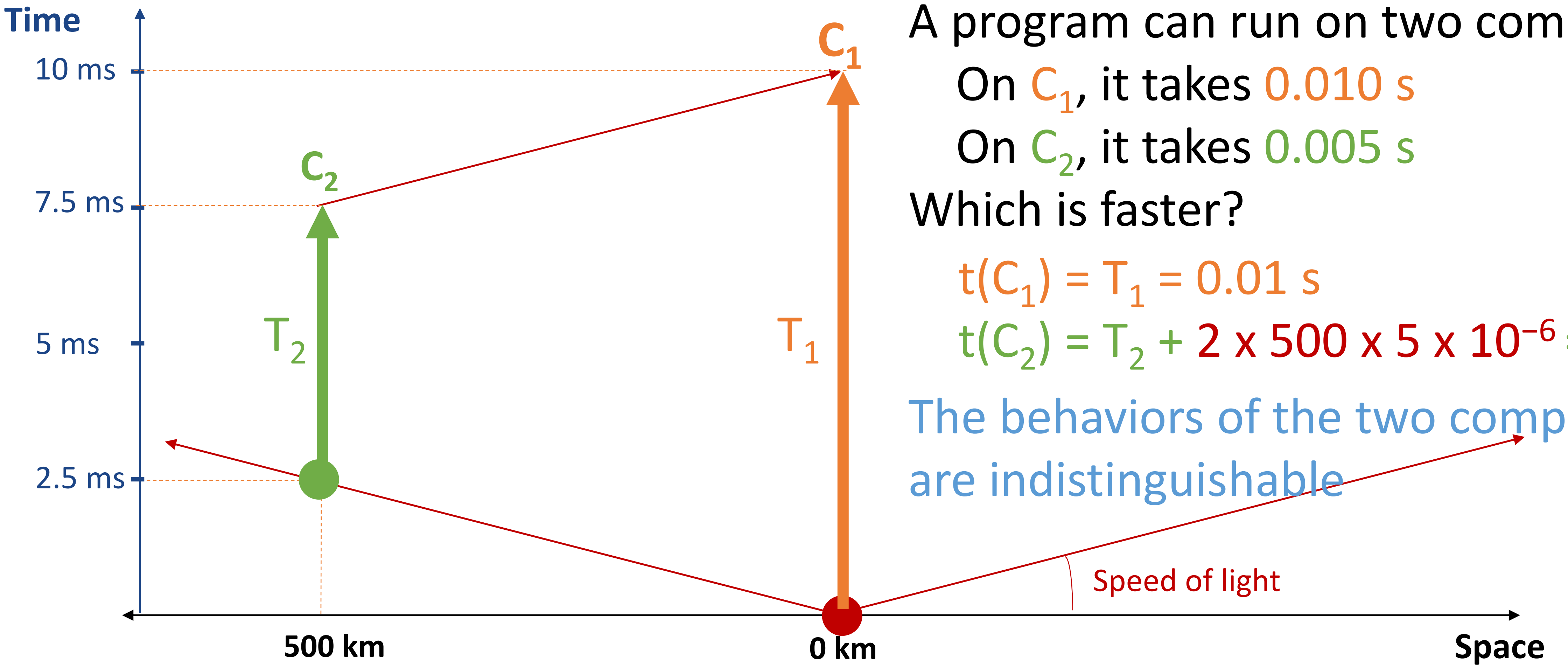
Good mechanisms empower discovery by **bridging the gap from thought to action** – they serve as "bicycles for the mind"

2020s will see an expanding gap in science due to new applications (including AI) and rapidly changing technologies (the "continuum")

What new mechanisms will be needed to enable effective use of the emerging computing continuum for new applications?

Project Celerity aims to take steps towards answering this question

The space-time continuum in computational systems



A program can run on two computers

On C_1, it takes 0.010 s

On C_2, it takes 0.005 s

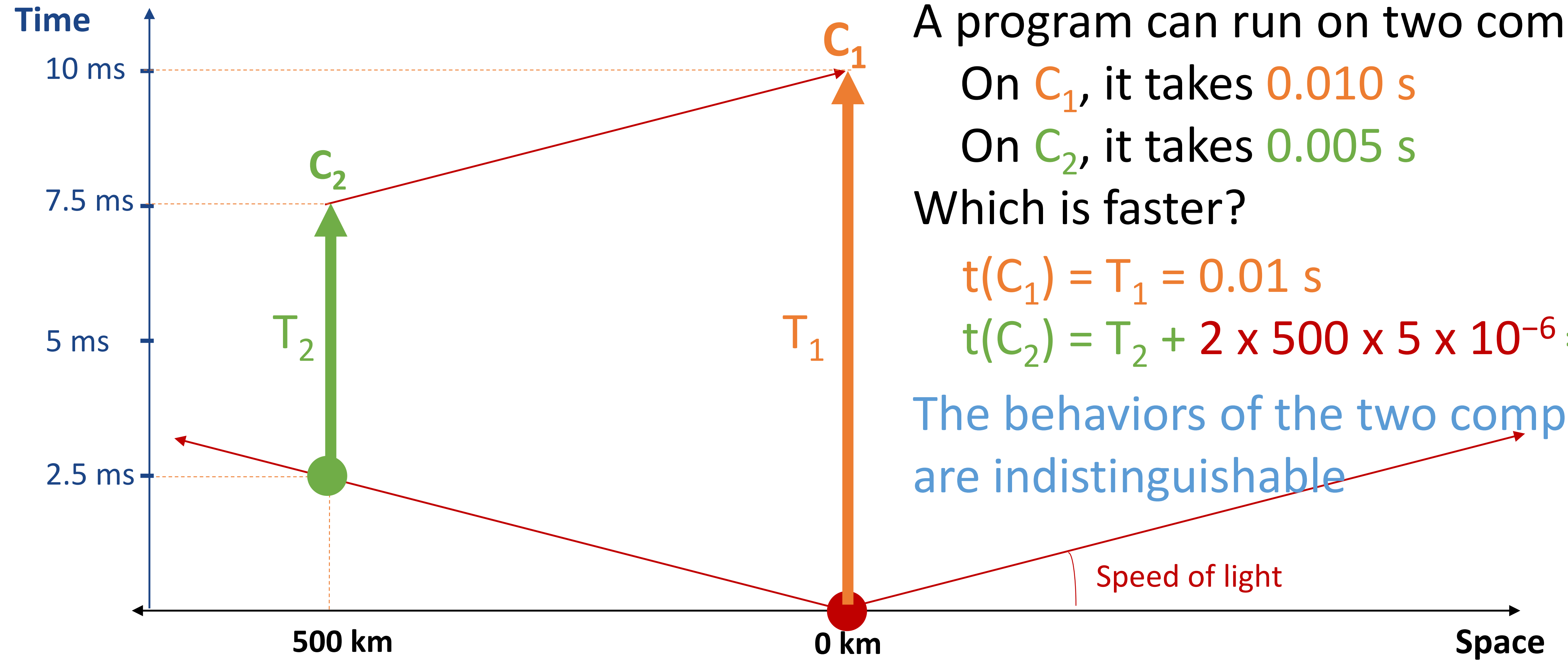
Which is faster?

$t(C_1) = T_1 = 0.01$ s

$t(C_2) = T_2 + 2 \times 500 \times 5 \times 10^{-6} = 0.01$ s

The behaviors of the two computers are indistinguishable

The space-time continuum in computational systems



A program can run on two computers

On C_1, it takes 0.010 s

On C_2, it takes 0.005 s

Which is faster?

$t(C_1) = T_1 = 0.01$ s

$t(C_2) = T_2 + 2 \times 500 \times 5 \times 10^{-6} = 0.01$ s

The behaviors of the two computers are indistinguishable

Misquoting Minkowski: "*Henceforth, **location** for itself, and **speed** for itself shall completely reduce to a mere shadow, and only some sort of union of the two shall preserve independence*."