
Convergence of Machine Learning, Big Data, and Supercomputing

**Dr. Jeremy Kepner
MIT Lincoln Laboratory Fellow**

July 2017



This material is based upon work supported by the Assistant Secretary of Defense for Research and Engineering under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Assistant Secretary of Defense for Research and Engineering.

Delivered to the U.S. Government with Unlimited Rights, as defined in DFARS Part 252.227-7013 or 7014 (Feb 2014). Notwithstanding any copyright notice, U.S. Government rights in this work are defined by DFARS 252.227-7013 or DFARS 252.227-7014 as detailed above. Use of this work other than as specifically authorized by the U.S. Government may violate any copyrights that exist in this work.

The Mathematical Convergence and Architectural Divergence of Machine Learning, Big Data, and Supercomputing

Dr. Jeremy Kepner
MIT Lincoln Laboratory Fellow

June 2017



This material is based upon work supported by the Assistant Secretary of Defense for Research and Engineering under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Assistant Secretary of Defense for Research and Engineering.

© 2017 Massachusetts Institute of Technology.

Delivered to the U.S. Government with Unlimited Rights, as defined in DFARS Part 252.227-7013 or 7014 (Feb 2014). Notwithstanding any copyright notice, U.S. Government rights in this work are defined by DFARS 252.227-7013 or DFARS 252.227-7014 as detailed above. Use of this work other than as specifically authorized by the U.S. Government may violate any copyrights that exist in this work.



Outline

- **Introduction**

- **Big Data (Scale Out)**
- **Supercomputing (Scale Up)**
- **Machine Learning (Scale Deep)**
- **Summary**



Lincoln Laboratory Supercomputing Center (LLSC)



Mission Areas

Air and Missile Defense

Homeland Protection

Air Traffic Control

Communication Systems

Cyber Security

Advanced Technology

Space Control

ISR Systems and Technology

Tactical Systems

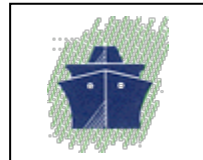
Engineering



Vast Data Sources



OSINT



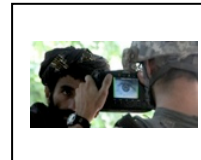
Weather



HUMINT



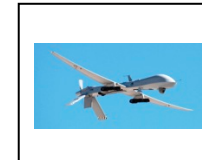
C2



Ground



Maritime



Air



Space



Cyber



LLSC develops & deploys unique, energy-efficient supercomputing that provides cross-mission

- Data centers, hardware, software, user support, and pioneering research
- Thousands of users
- Interactive data analysis, simulations, and machine learning

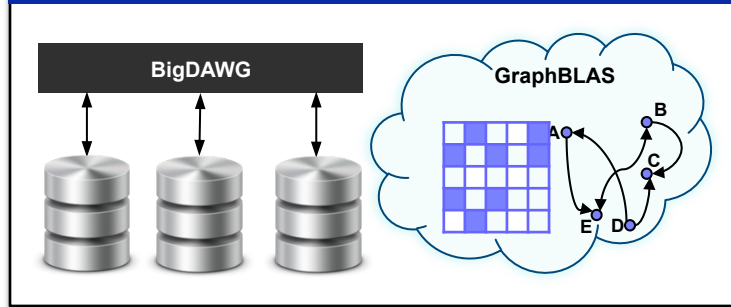
Supercomputing Center Focus Areas

Interactive Supercomputing



- High performance storage and compute
- Application programming interface design, modeling and benchmarking
- Supercomputing infrastructure and scalable software environments

Big Data Technologies



- Big data architectures, databases, and graph processing
- Architecture analysis, benchmarking, and advanced mathematics
- Database infrastructure, federated interfaces, and processing standards

Education and Outreach



- Online education and knowledge transfer
- Expert content, course development, and production platform
- Online courses, virtualized environments, in-person classes, workshops, conferences, books

Mathematically rigorous approach to computational challenges



Large Scale Computing: Challenges

Volume

- **Challenge:** Scale of data beyond what current approaches can handle
Social media, cyber networks, internet-of-things, bioinformatics, ...

Velocity

- **Challenge:** Analytics beyond what current approaches can handle
Engineering simulation, drug discovery, autonomous systems, ...

Variety

- **Challenge:** Diversity beyond what current approaches can handle
Computer vision, language processing, decision making, ...



Large Scale Computing: Hardware

Volume

- **Challenge:** Scale of data beyond what current approaches can handle
- **Hardware:** Scale-out, more servers per data center (hyperscale)



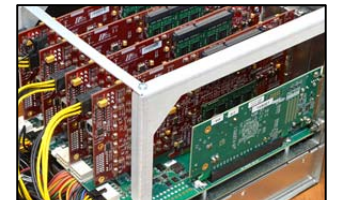
Velocity

- **Challenge:** Analytics beyond what current approaches can handle
- **Hardware:** Scale-up, more transistors per server (accelerators)



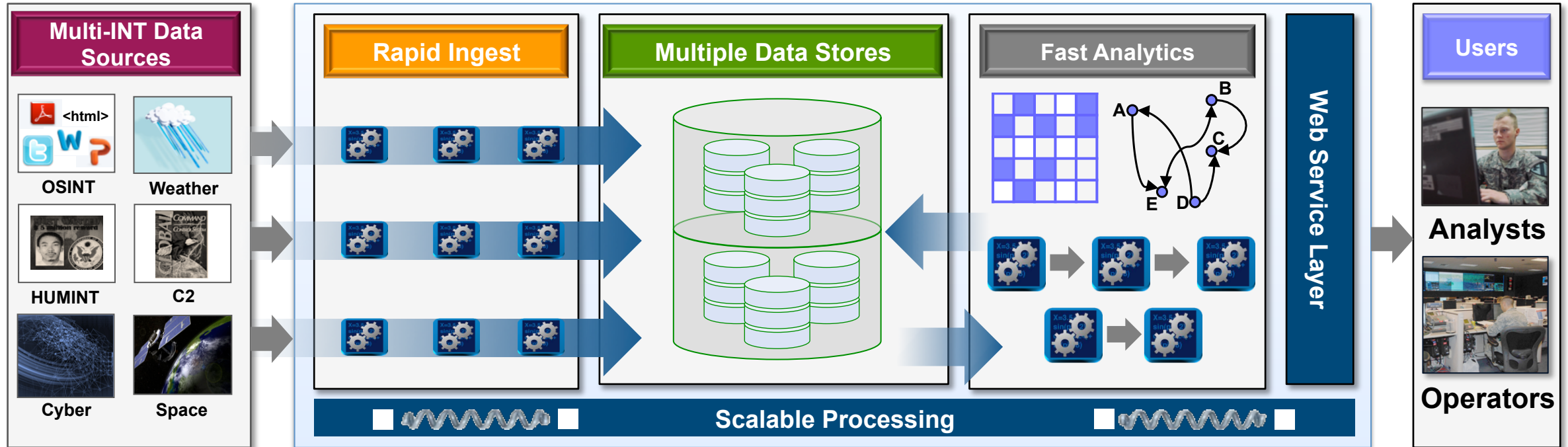
Variety

- **Challenge:** Diversity beyond what current approaches can handle
- **Hardware:** Scale-deep, more customizable processors (FPGAs, ...)



Requires mathematically rigorous approaches to insulate users from scaling

Standard Processing Architecture



High Performance Requirements

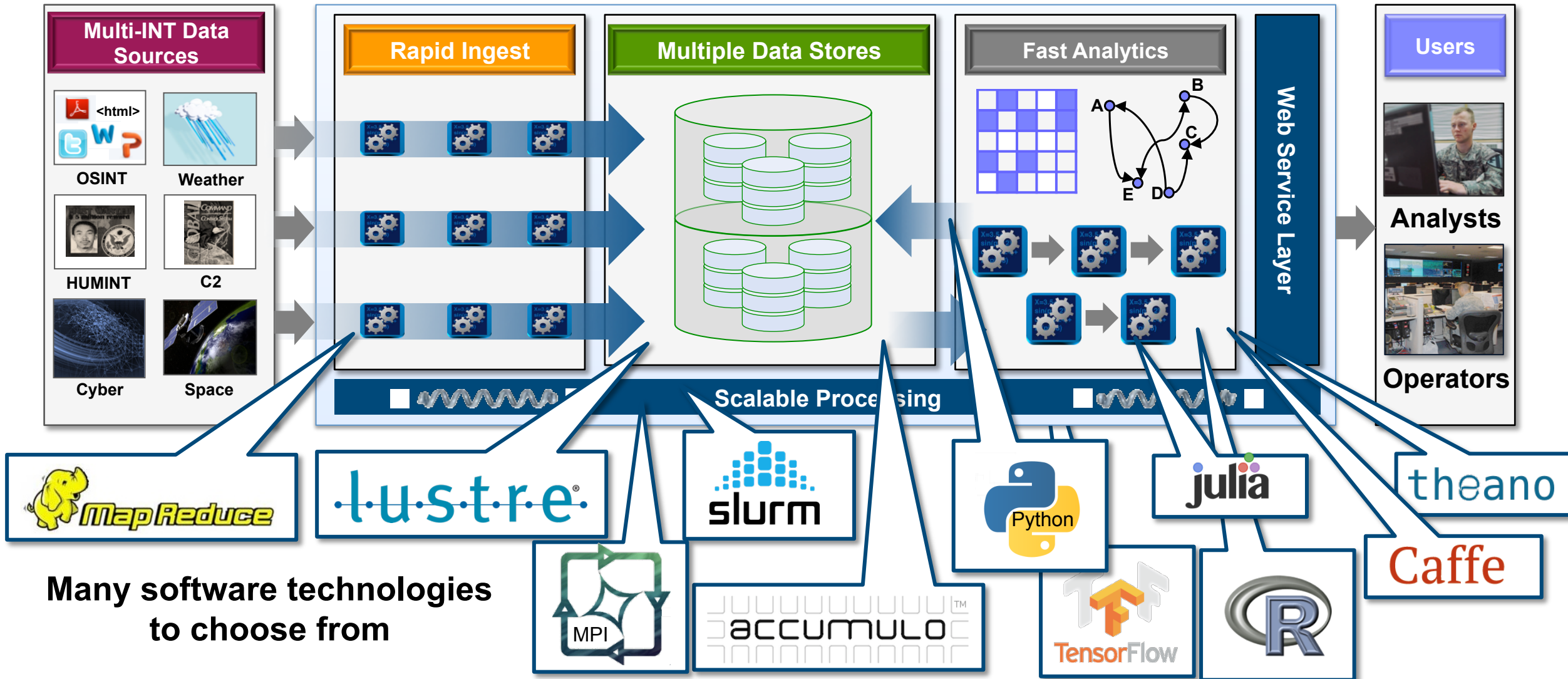
- Sustaining rapid ingest
- Fast analytics
- Integrating diverse data

Analysts Preferred Environments

- Familiar
- High Level
- Mission Focused

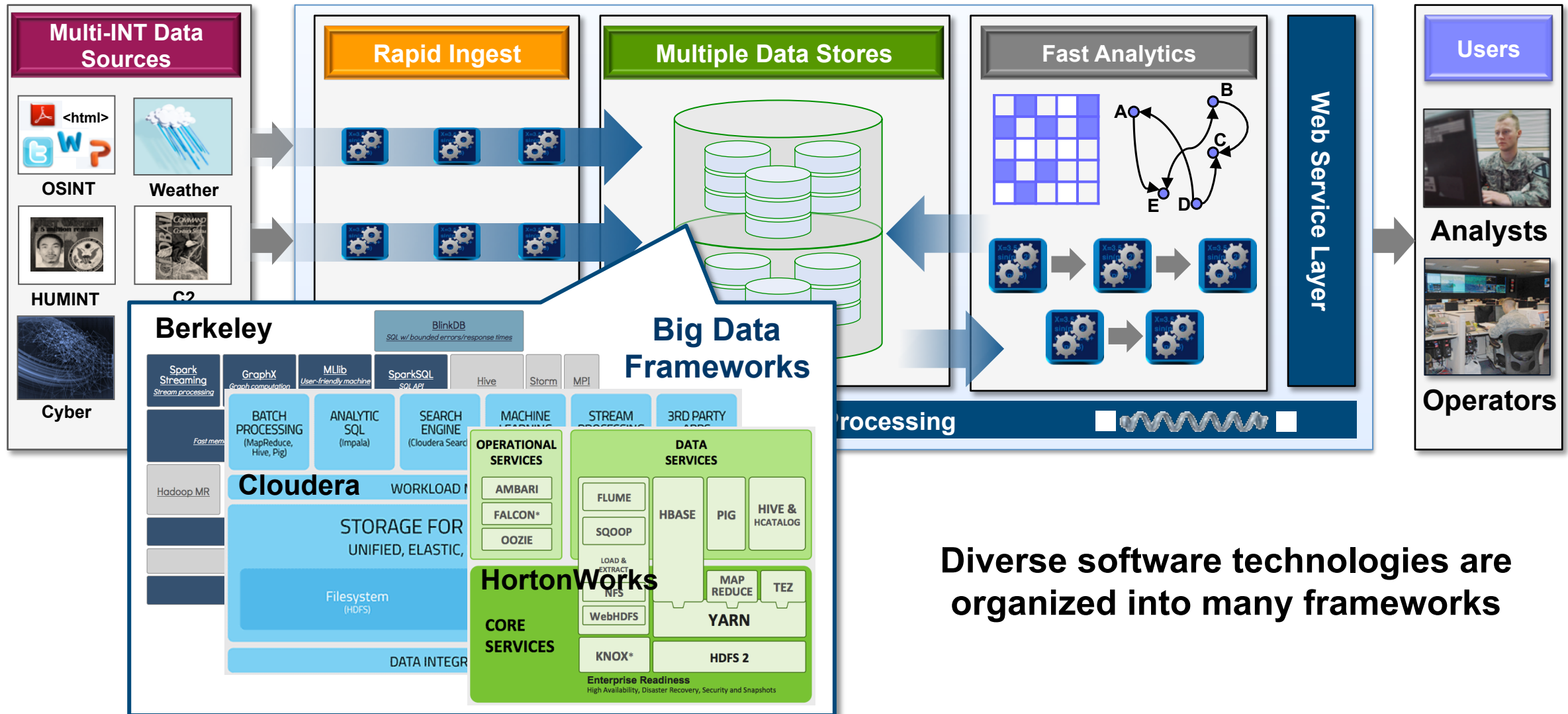


Standard Processing Architecture: Software





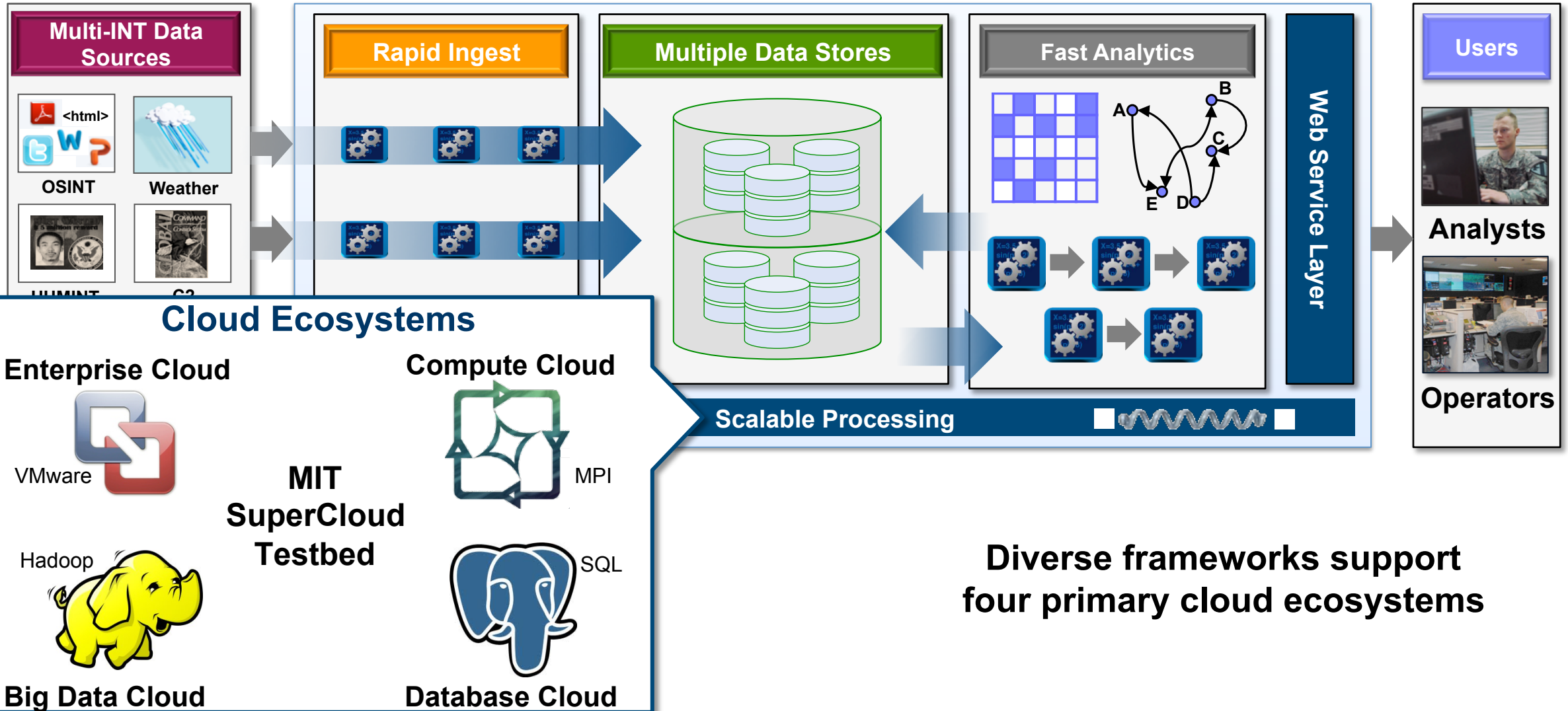
Standard Processing Architecture: Frameworks



Diverse software technologies are organized into many frameworks



Standard Processing Architecture: Ecosystems



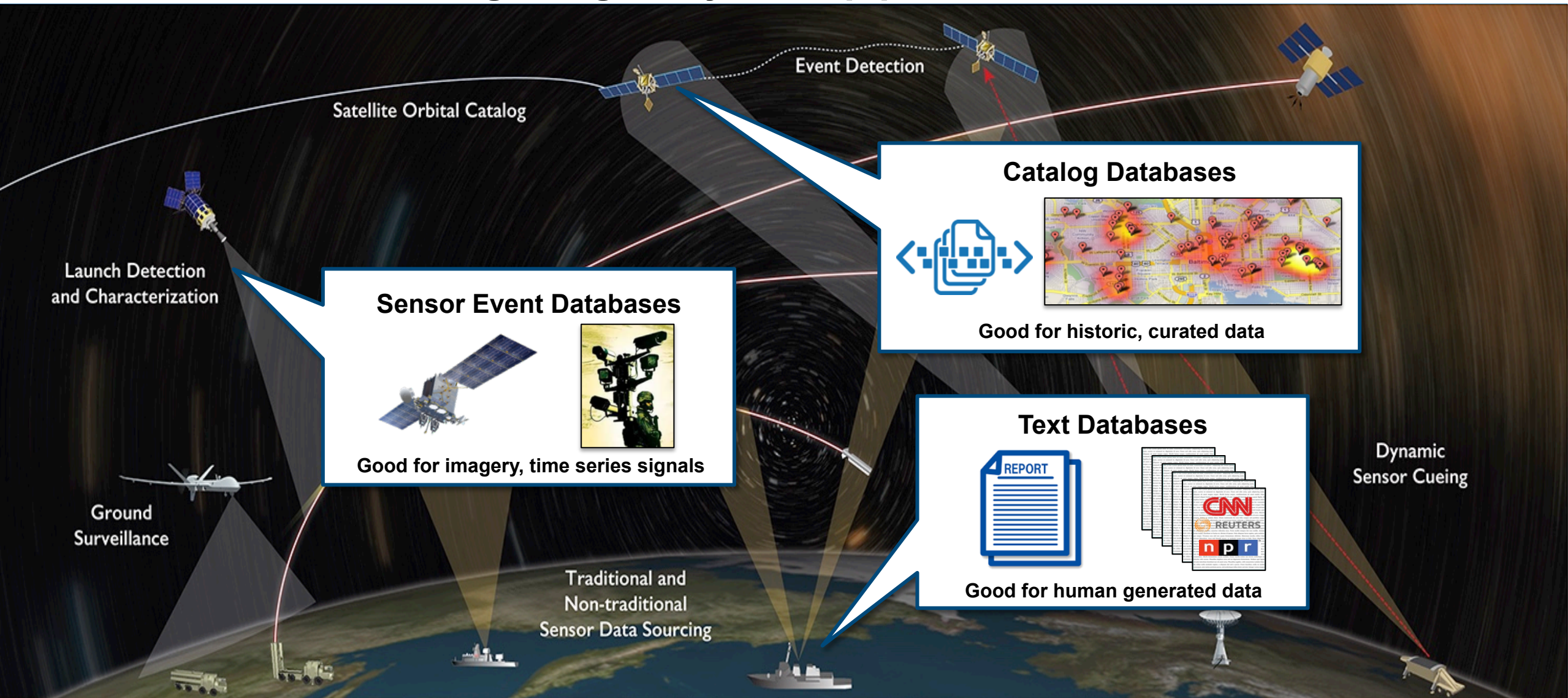


Outline

- Introduction
- **Big Data (Scale Out)**
- Supercomputing (Scale Up)
- Machine Learning (Scale Deep)
- Summary

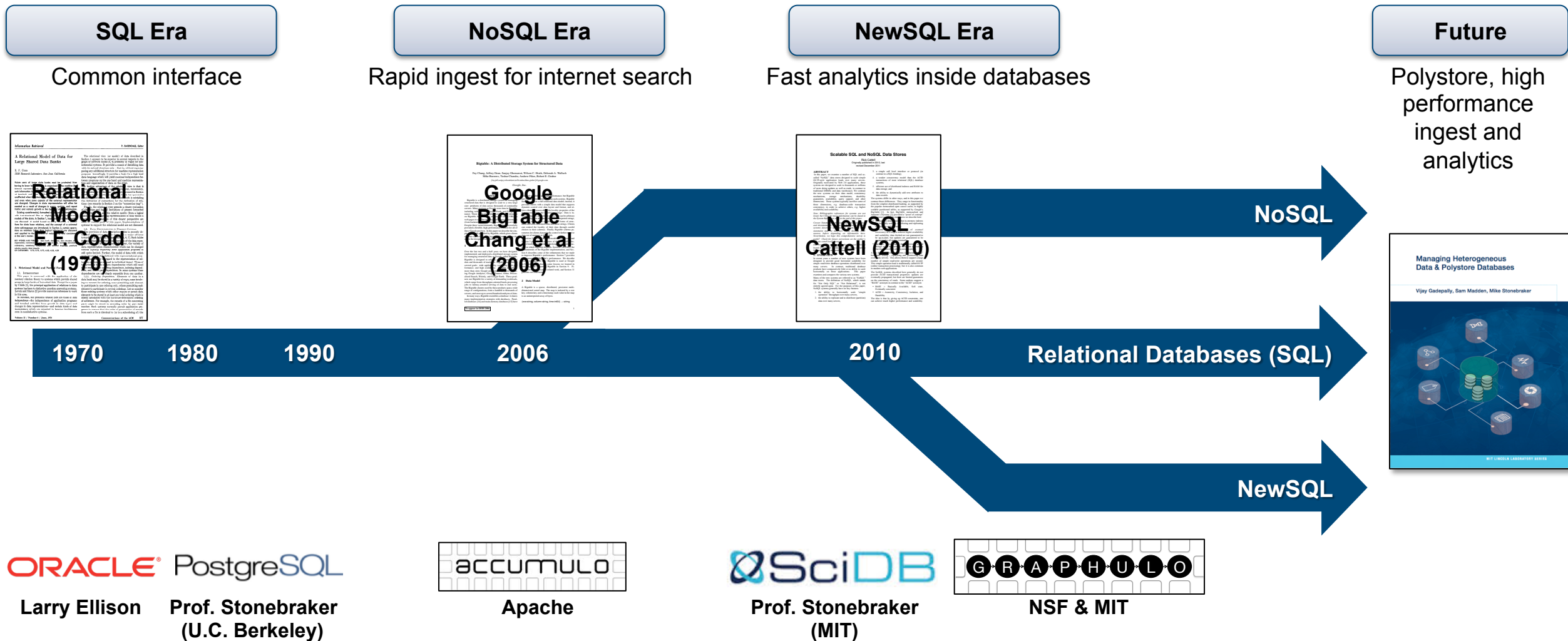
Example Big Data Application

- Integrating Many Stovepiped Databases -





Modern Database Paradigm Shifts





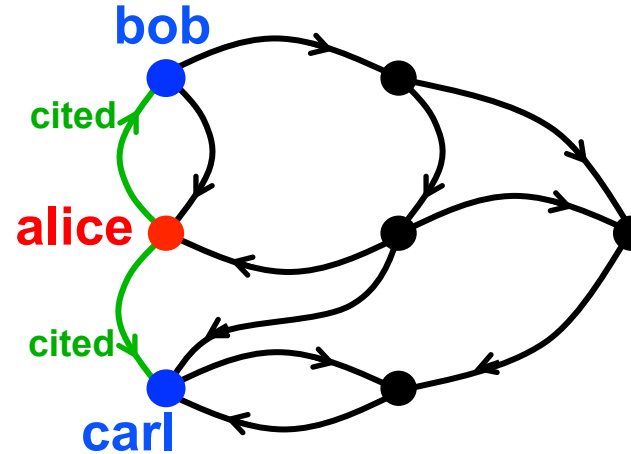
Declarative, Mathematically Rigorous Interfaces

SQL
Set Operations

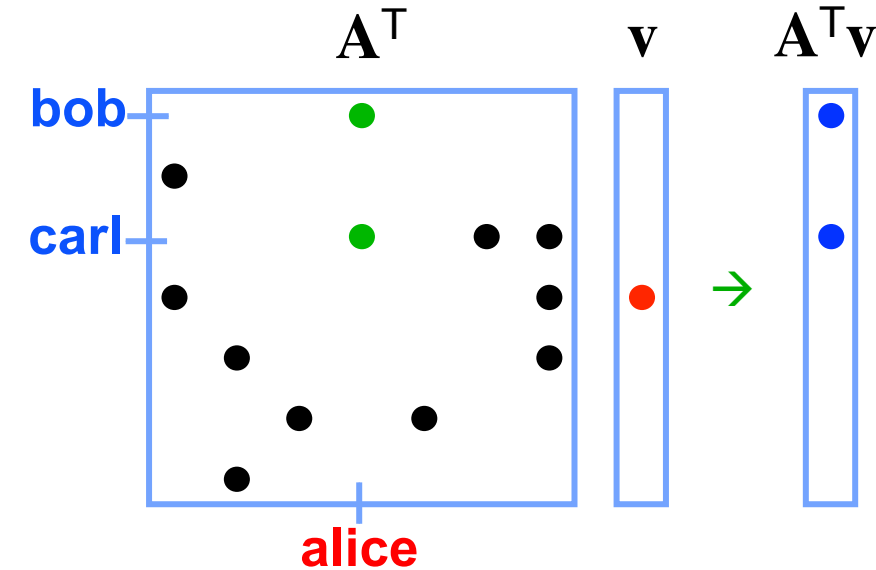
	from	link	to
001	alice	cited	bob
002	bob	cited	alice
003	alice	cited	carl

SELECT 'to' FROM T
WHERE 'from=alice'

NoSQL
Graph Operations



NewSQL
Linear Algebra



Operation: finding Alice's nearest neighbors

Associative Array Algebra Provides a Unified Mathematics for SQL, NoSQL, NewSQL

$$\mathbf{A} = \mathbb{S}^{N \times M}(\mathbf{k}_1, \mathbf{k}_2, \mathbf{v}, \oplus) \quad (\mathbf{k}_1, \mathbf{k}_2, \mathbf{v}) = \mathbf{A} \quad \mathbf{C} = \mathbf{A}^T \quad \mathbf{C} = \mathbf{A} \oplus \mathbf{B} \quad \mathbf{C} = \mathbf{A} \otimes \mathbf{C} \quad \mathbf{C} = \mathbf{A} \mathbf{B} = \mathbf{A} \oplus . \otimes \mathbf{B}$$

Operations in all representations are equivalent and are linear systems



GraphBLAS.org Standard for Sparse Matrix Math

- **Six key operations**

$$A = \mathbb{S}^{N \times M}(i, j, v) \quad (i, j, v) = A \quad C = A^T \quad C = A \oplus B \quad C = A \otimes C \quad C = A B = A \oplus \cdot \otimes B$$

- **That are composable**

$$A \oplus B = B \oplus A \quad (A \oplus B) \oplus C = A \oplus (B \oplus C) \quad A \otimes (B \oplus C) = (A \otimes B) \oplus (A \otimes C)$$

$$A \otimes B = B \otimes A \quad (A \otimes B) \otimes C = A \otimes (B \otimes C) \quad A (B \oplus C) = (A B) \oplus (A C)$$

- **Can be used to build standard GraphBLAS functions**

buildMatrix, extractTuples, Transpose, mXm, mXv, vXm, extract, assign, eWiseAdd, ...

- **Can be used to build a variety of graph utility functions**

Tril(), Triu(), Degreed Filtered BFS, ...

- **Can be used to build a variety of graph algorithms**

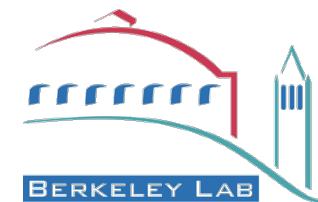
Triangle Counting, K-Truss, Jaccard Coefficient, Non-Negative Matrix Factorization, ...

- **That work on a wide range of graphs**

Hyper, multi-directed, multi-weighted, multi-partite, multi-edge



NVIDIA



UCDAVIS



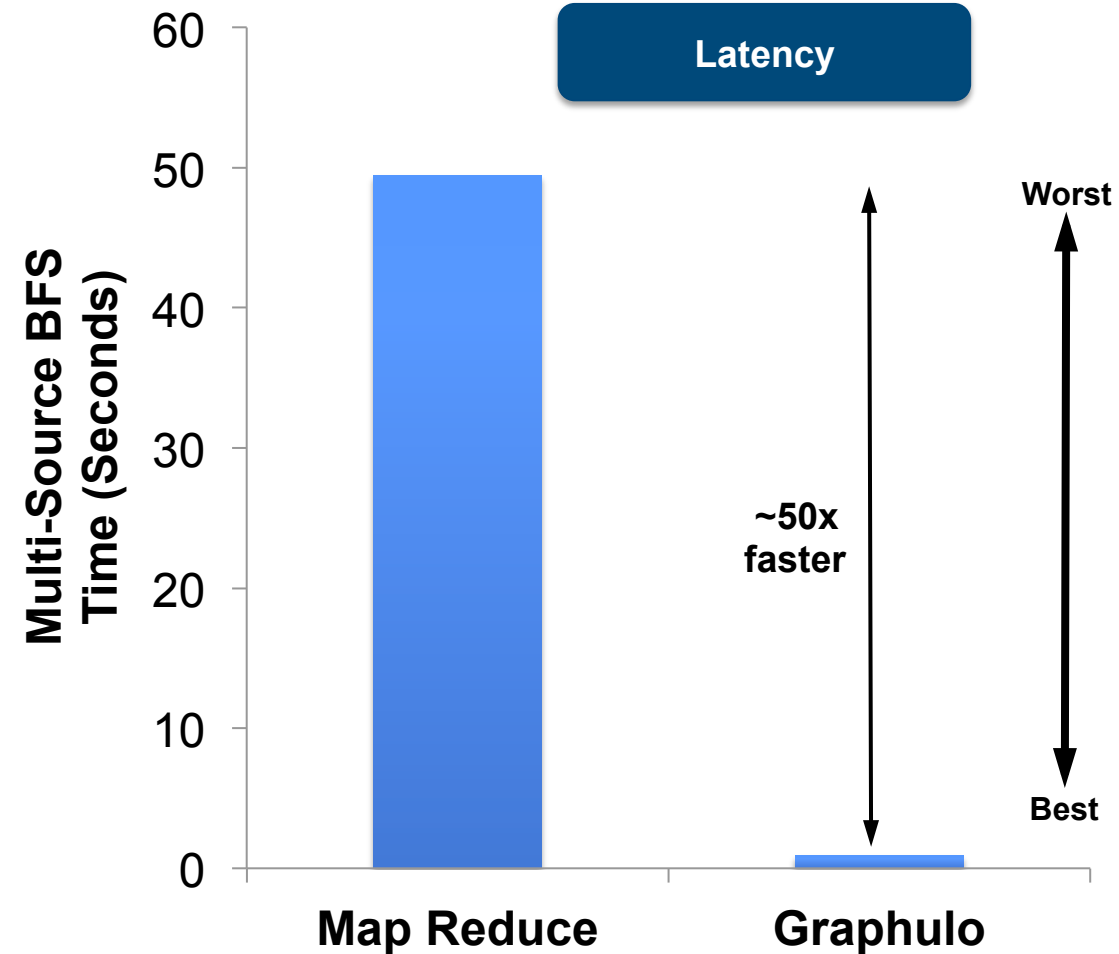


Graphulo High Performance Database Library

```
% Do BFS
v = G.AdjBFS(Atable, v0, k, Rtable, RtableTranspose, Adegtable,

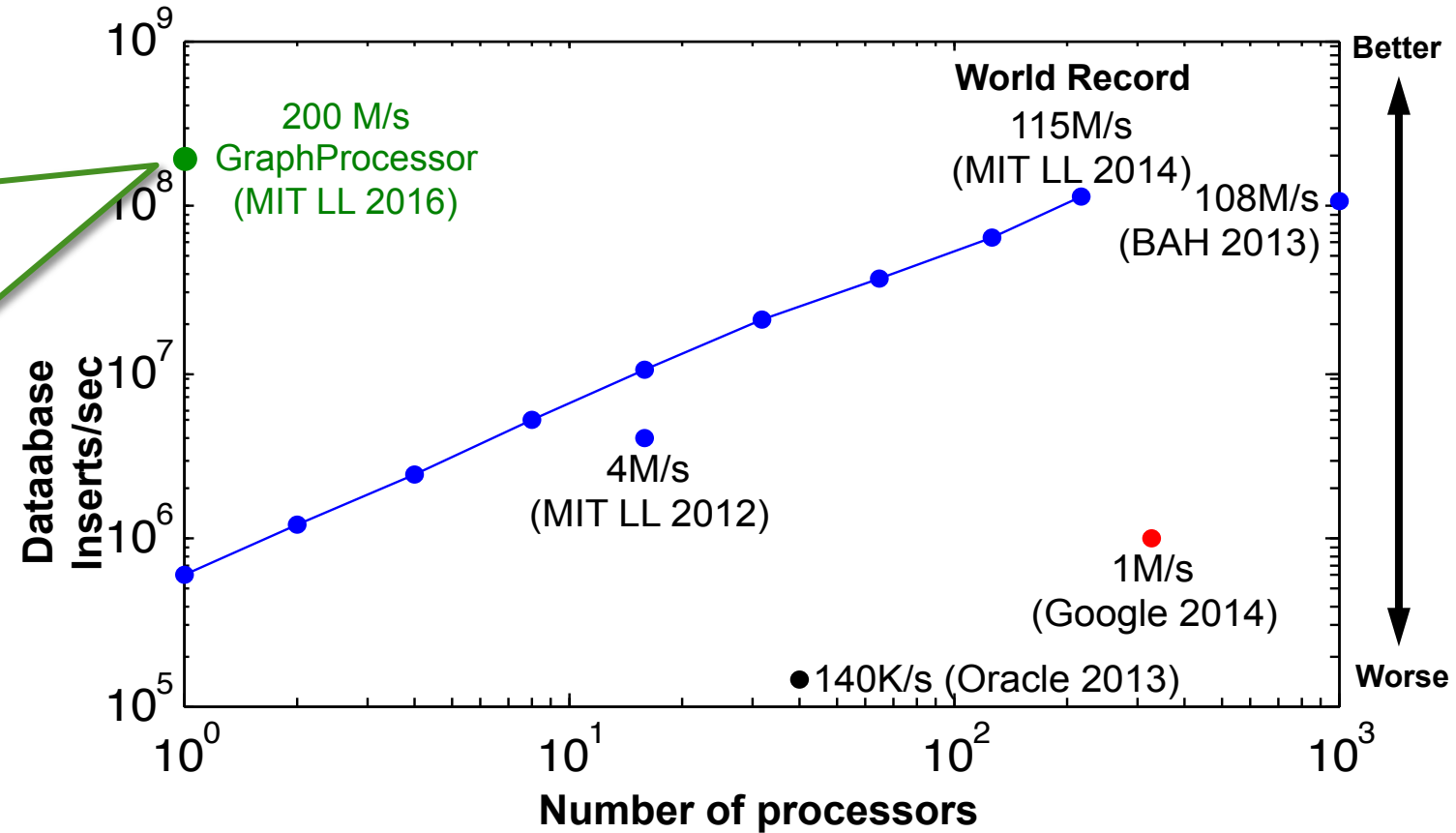
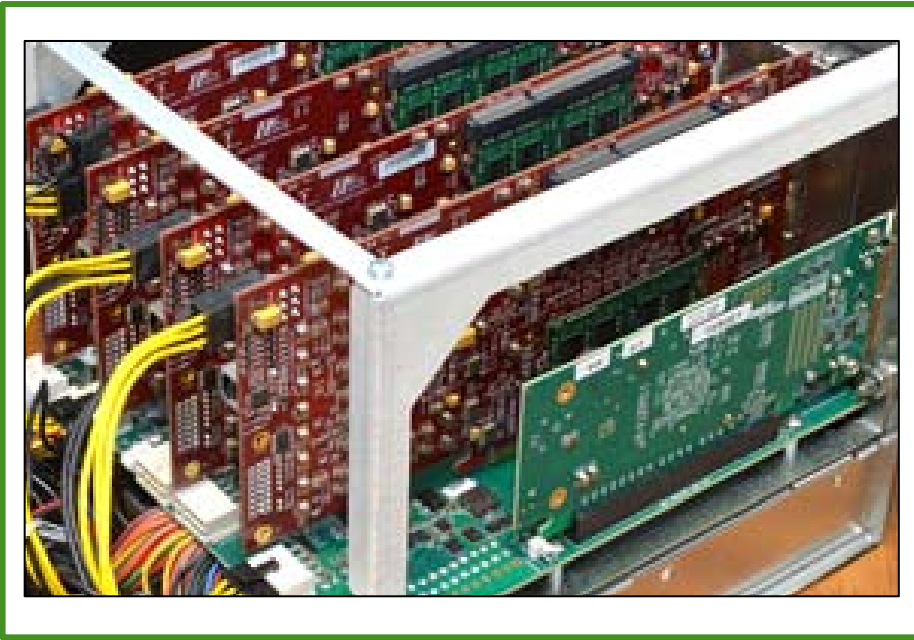
% Vertices reached in 3 hops from v4 and v5
disp('Vertices reached in 3 hops from v4 and v5')
v
```

- GraphBLAS library for Accumulo
- High performance graph analytics
- 50x faster than industry standard
- Jupyter interactive portal interface
 - Similar to Mathematica notebooks



Graph Processing Hardware

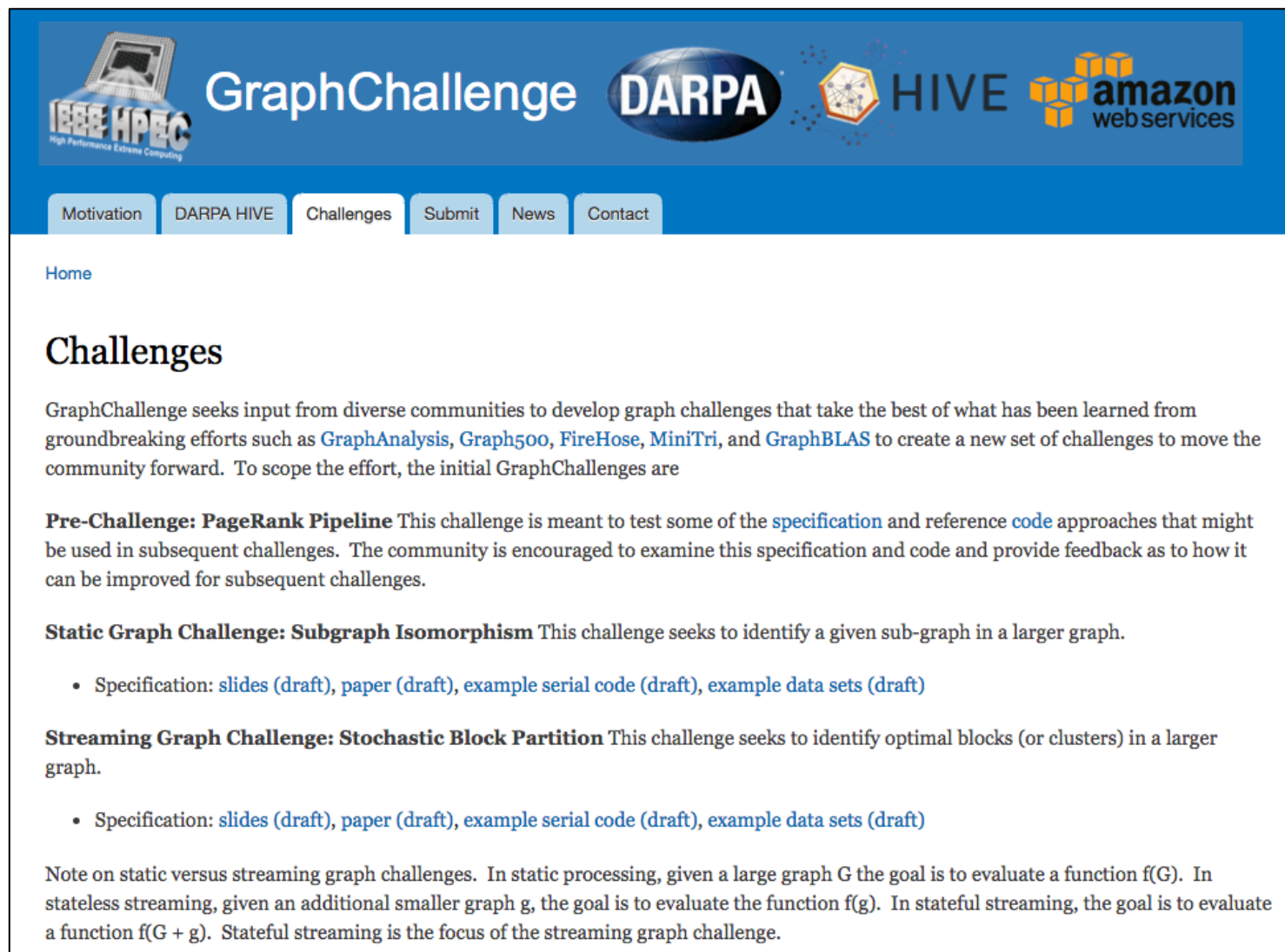
Lincoln GraphProcessor faster than 200+ node database cluster





- Parallel processing
- Parallel memory access
- Fastest (TB/s) to memory
- Higher scalability (TB/s)
- Optimized for Graphs

Hierarchical Identify Verify Exploit



The screenshot shows the GraphChallenge.org website. At the top, there are logos for IEEE HPEC, GraphChallenge, DARPA, HIVE, and Amazon Web Services. Below the logos is a navigation bar with links: Motivation, DARPA HIVE, Challenges, Submit, News, and Contact. The main content area is titled 'Challenges' and contains the following text:

GraphChallenge seeks input from diverse communities to develop graph challenges that take the best of what has been learned from groundbreaking efforts such as [GraphAnalysis](#), [Graph500](#), [FireHose](#), [MiniTri](#), and [GraphBLAS](#) to create a new set of challenges to move the community forward. To scope the effort, the initial GraphChallenges are

Pre-Challenge: PageRank Pipeline This challenge is meant to test some of the [specification](#) and reference [code](#) approaches that might be used in subsequent challenges. The community is encouraged to examine this specification and code and provide feedback as to how it can be improved for subsequent challenges.

Static Graph Challenge: Subgraph Isomorphism This challenge seeks to identify a given sub-graph in a larger graph.

- Specification: [slides \(draft\)](#), [paper \(draft\)](#), [example serial code \(draft\)](#), [example data sets \(draft\)](#)

Streaming Graph Challenge: Stochastic Block Partition This challenge seeks to identify optimal blocks (or clusters) in a larger graph.

- Specification: [slides \(draft\)](#), [paper \(draft\)](#), [example serial code \(draft\)](#), [example data sets \(draft\)](#)

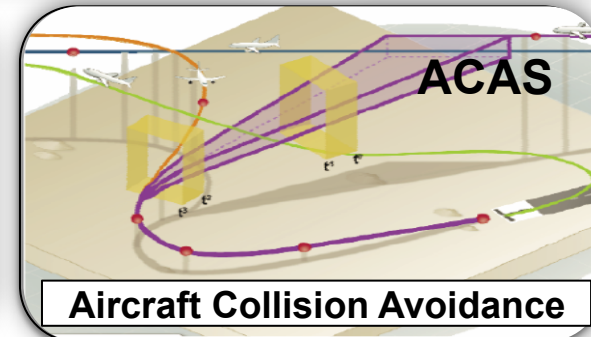
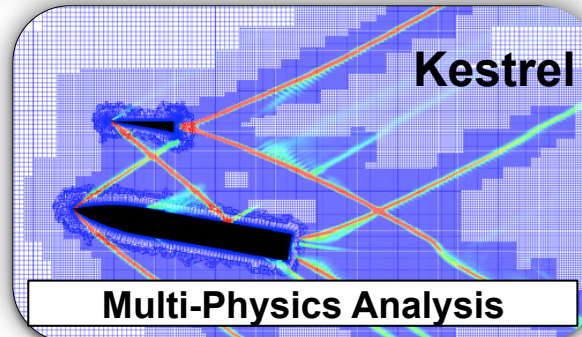
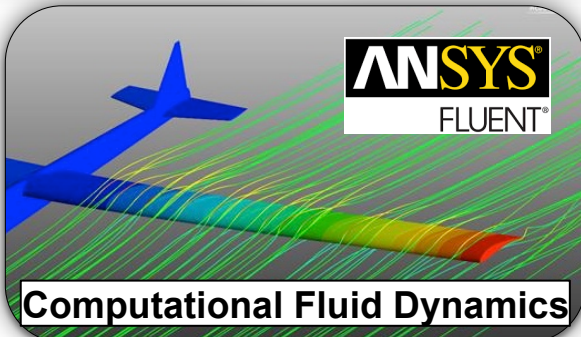
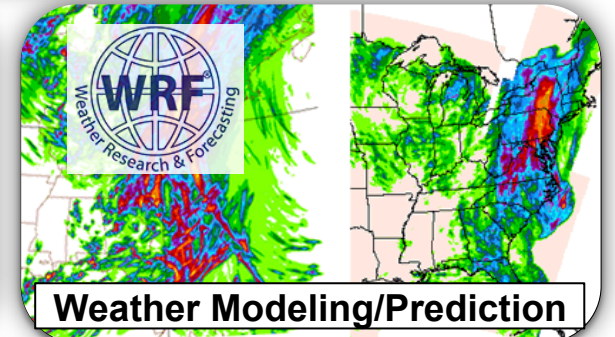
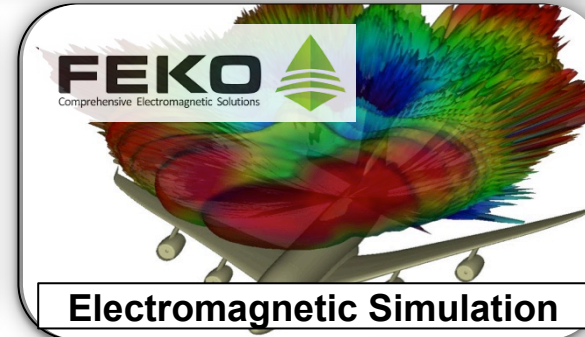
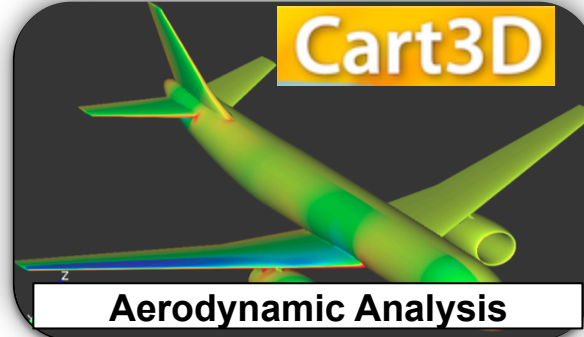
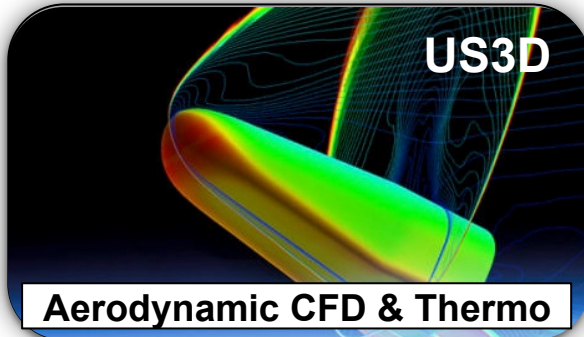
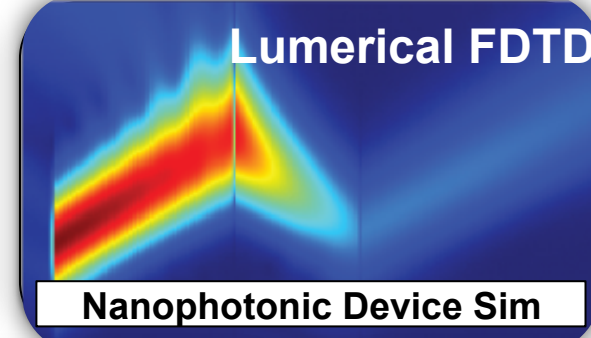
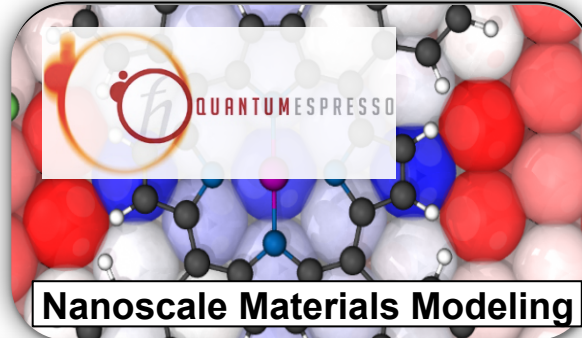
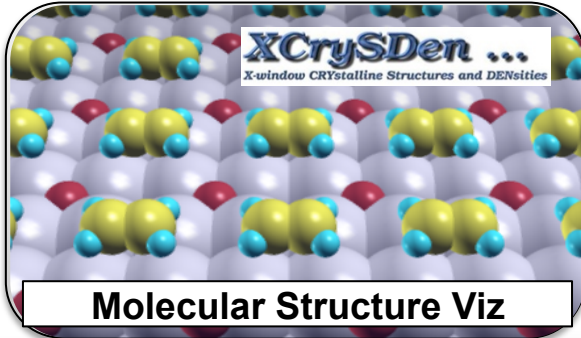
Note on static versus streaming graph challenges. In static processing, given a large graph G the goal is to evaluate a function $f(G)$. In stateless streaming, given an additional smaller graph g , the goal is to evaluate the function $f(g)$. In stateful streaming, the goal is to evaluate a function $f(G + g)$. Stateful streaming is the focus of the streaming graph challenge.



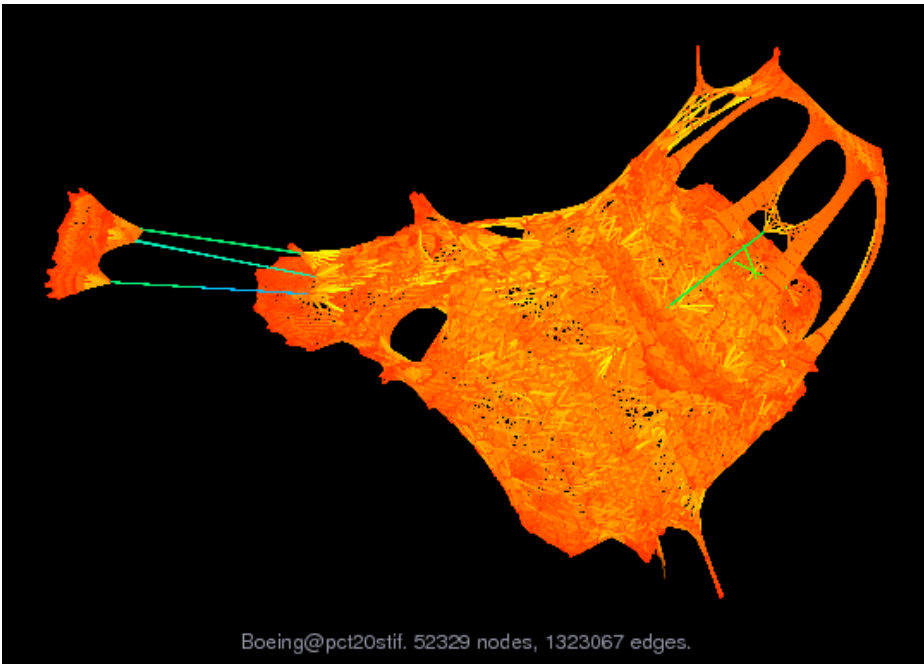
Outline

- Introduction
- Big Data (Scale Out)
- **Supercomputing (Scale Up)**
- Machine Learning (Scale Deep)
- Summary

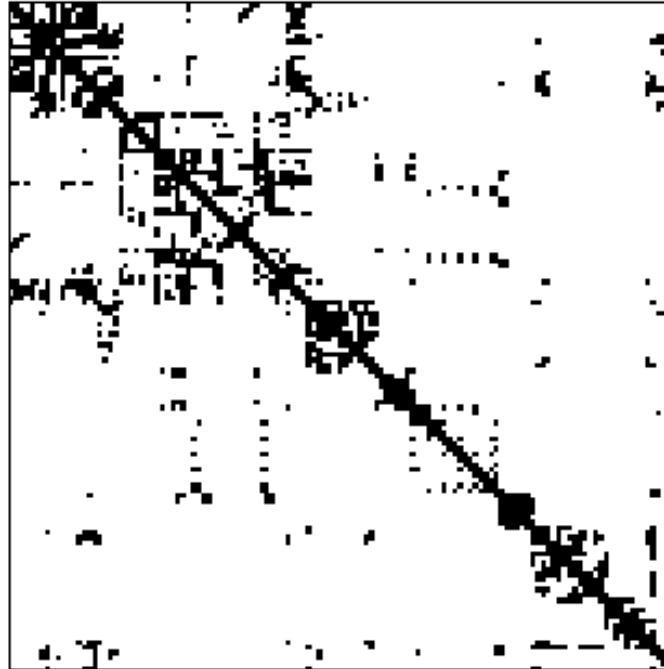
Example Supercomputing Applications



Example Algorithm: Finite Element Method



Mesh of engine block



Corresponding stiffness matrix K

$$Ku = f$$

displacements u forces f

Finite element equation

- Standard approach for many engineering problems
- Iteratively solves large sparse matrix equations (as many small dense matrices)



Example Matrix Math Software Stacks

BLAS (Basic Linear Algebra Subprograms)

Menu

[Presentation:](#)

[Acknowledgments:](#)

[History](#)

[Software:](#)

[Licensing:](#)

[REFERENCE BLAS Version 3.7.0](#)

[CBLAS](#)

[Level 3 BLAS tuned for single processors with caches](#)

[Extended precision Level 2 BLAS routines](#)

[BLAS for windows](#)

[GIT Access](#)

[The netlib family and its cousins](#)

[Support](#)

[Documentation](#)

[BLAS Technical Forum](#)

[Optimized BLAS Library](#)

[BLAS Routines](#)

[LEVEL 1](#)

[LEVEL 2](#)

[LEVEL 3](#)

[Extended precision Level 2 BLAS routines](#)


Questions/comments? lapack@icl.utk.edu

Portable, Extensible Toolkit for Scientific Computation

The current version of PETSc is 3.7; released April 25, 2016.

PETSc, pronounced PET-see (the S is silent), is a suite of data structures and routines for the scalable (parallel) solution of scientific applications modeled by partial differential equations. It supports MPI, and [GPU's through CUDA or OpenCL](#), as well as hybrid MPI-GPU parallelism.

- [Scientific applications](#) that use PETSc
- [Features](#) of the PETSc libraries (and a recent [podcast](#))
- [Linear system solvers](#) accessible from PETSc
- Related packages that use PETSc
 - [MOOSE - Multiphysics Object-Oriented Simulation Environment](#) finite element framework, built on top of libMesh and PETSc
 - [SLEPc - Scalable Library for Eigenvalue Problems](#)
 - [COOLFluid - CFD, plasma and multi-physics simulation package](#)
 - [Fluidity - a finite element/volume fluids code](#)
 - [OpenFVM - finite volume based CFD solver](#)
 - [OOFEM - object oriented finite element library](#)
 - [libMesh - adaptive finite element library](#)
 - [FEniCS - sophisticated Python based finite element simulation package](#)
 - [Firedrake - sophisticated Python based finite element simulation package](#)
 - [DEAL.II - sophisticated C++ based finite element simulation package](#)
 - [PHAML - The Parallel Hierarchical Adaptive MultiLevel Project](#)
 - [Chaste - Cancer, Heart and Soft Tissue Environment](#)
 - [PyClaw - A massively parallel, high order accurate, hyperbolic PDE solver](#)
 - [PetIGA - A framework for high performance Isogeometric Analysis](#)
 - [MFEM - lightweight, scalable C++ library for finite element methods](#)
 - [Python Bindings](#)
 - [petsc4py](#) from Lisandro Dalcin at CIMEC
 - [Elefant](#) from the SML group at NICTA
 - [Java Bindings](#)



Home	Getting Started	Capabilities	About
C++11 Requirement		User Experience	
		Framework and Tools	
		Linear Algebra Services	
		Parallel Programming Environments	
		Software Engineering Technologies and Integration	
		Meshes, Geometry and Load Balancing	
		Discretizations	
		Linear and Eigen Solvers	
		Embedded Nonlinear Analysis Tools	

[Introduction](#)
[Trilinos User Doc](#)
[Trilinos Developer Package List](#)

High performance matrix math for parallel computers and accelerators



Selected Supercomputing Processors and Systems

Sunway TaihuLight (2016)

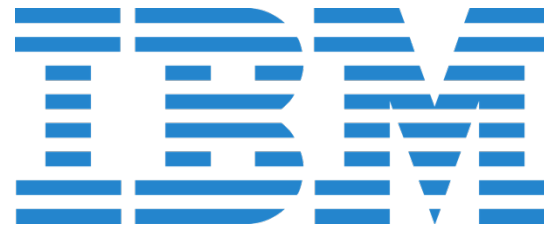
国家超级计算无锡中心
National Supercomputing Center in Wuxi

上海集成电路技术与产业促进中心
National High-Performance IC Design Center

Sunway Processor
260 Cores
260 256 bit vector units



Summit (2017)



DGX-1 Node Specs (2016)

SYSTEM SPECIFICATIONS

GPUs	8x Tesla GP100
TFLOPS (GPU FP16 / CPU FP32)	170/3

Aurora (2019)

Argonne 
NATIONAL LABORATORY



Processor Specs (2016)

64 Cores
128 512 bit vector units

All deliver maximum performance on dense matrix math



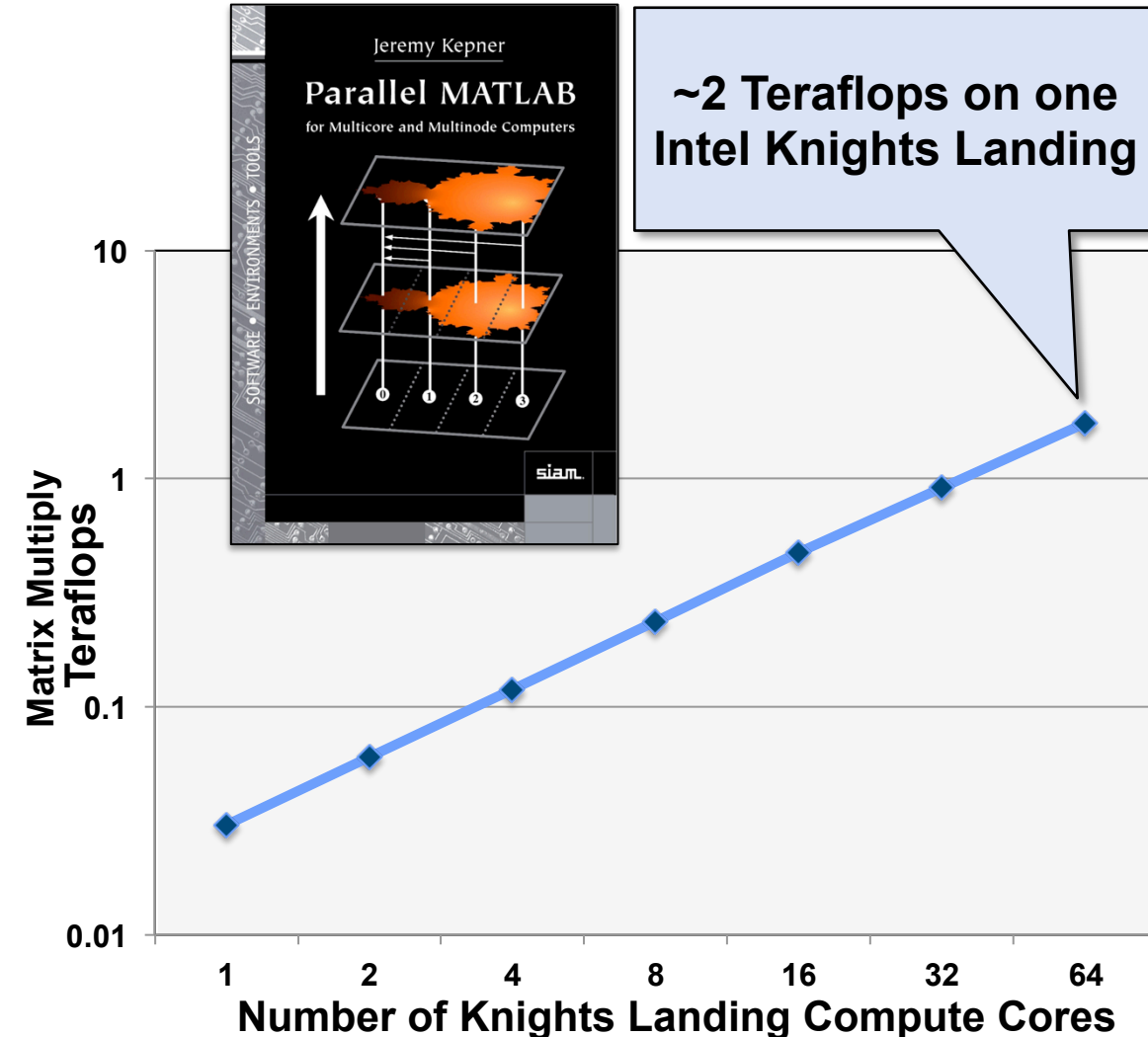
Interactive Parallel Matrix Math

```
jupyter ParallelComputingInJupyter Last Checkpoint: a minute ago (
File Edit View Insert Cell Kernel Navigate Widgets Help
[Icons] [Code] [CellToolbar]

In [6]: eval(pRUN(pBlurimage_v2, 4, grid));

Submitting pBlurimage_v2 on 4 processor(
-- SLURM Grid job has completed or has a
Launching MPI rank: 3 on: grid_slurm_03
Launching MPI rank: 2 on: grid_slurm_02
Launching MPI rank: 1 on: grid_slurm_01
Launching MPI rank: 0 on: node-050.local
```

- Parallel Matlab library
- High performance dense matrix math
- Linear speedup on Intel Knights Landing
- Jupyter interactive portal interface
 - Similar to Mathematica notebooks





Lincoln Laboratory Petascale System

	TX-Green Upgrade
Processor	Intel Knights Landing
Total Cores	41,472
Peak Petaflops	1.724
Top500 Petaflops	1.025 (measured)
Total Terabytes	124
Network Link	Intel OmniPath 25 GB/s



Based on Nov 2016 Top500.org list

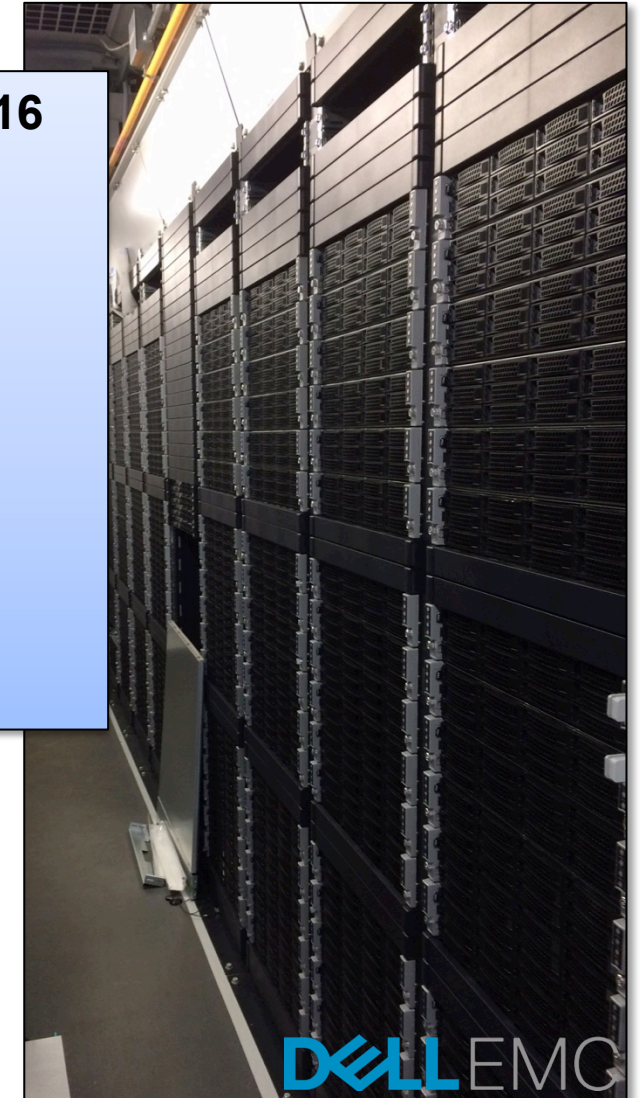
#1 at Lincoln
#1 at MIT
#1 in Massachusetts
#1 in New England
#2 in the Northeast
#3 at a US University
#3 at a University in the
Western Hemisphere
#43 in the United States
#106 in the World



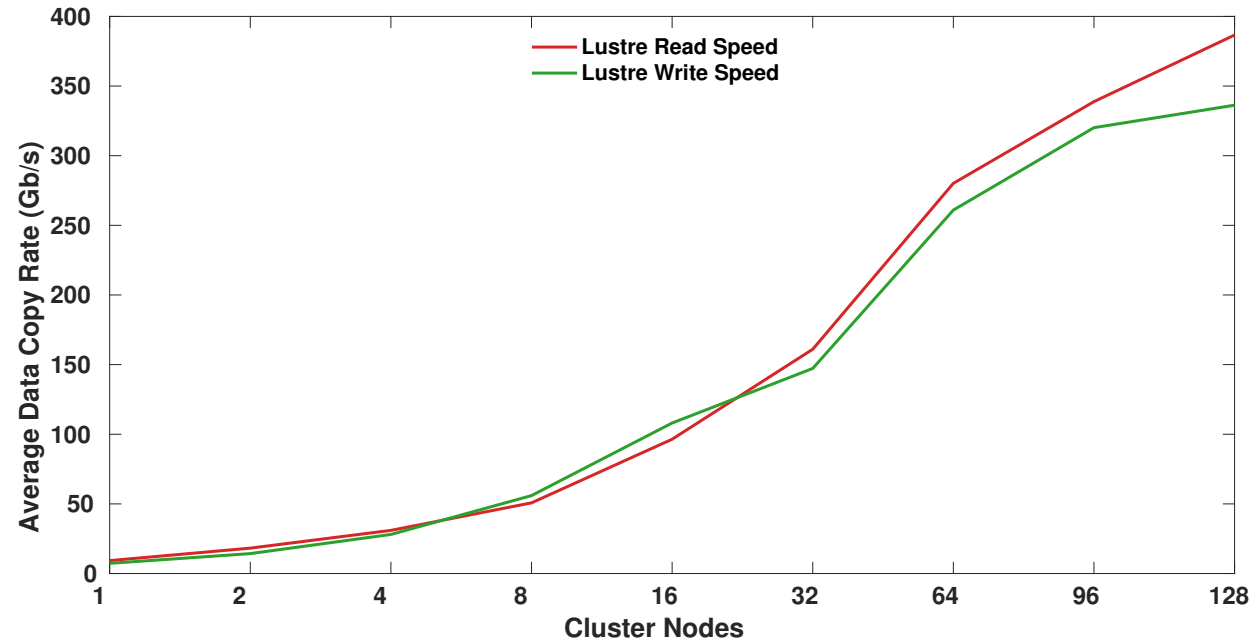
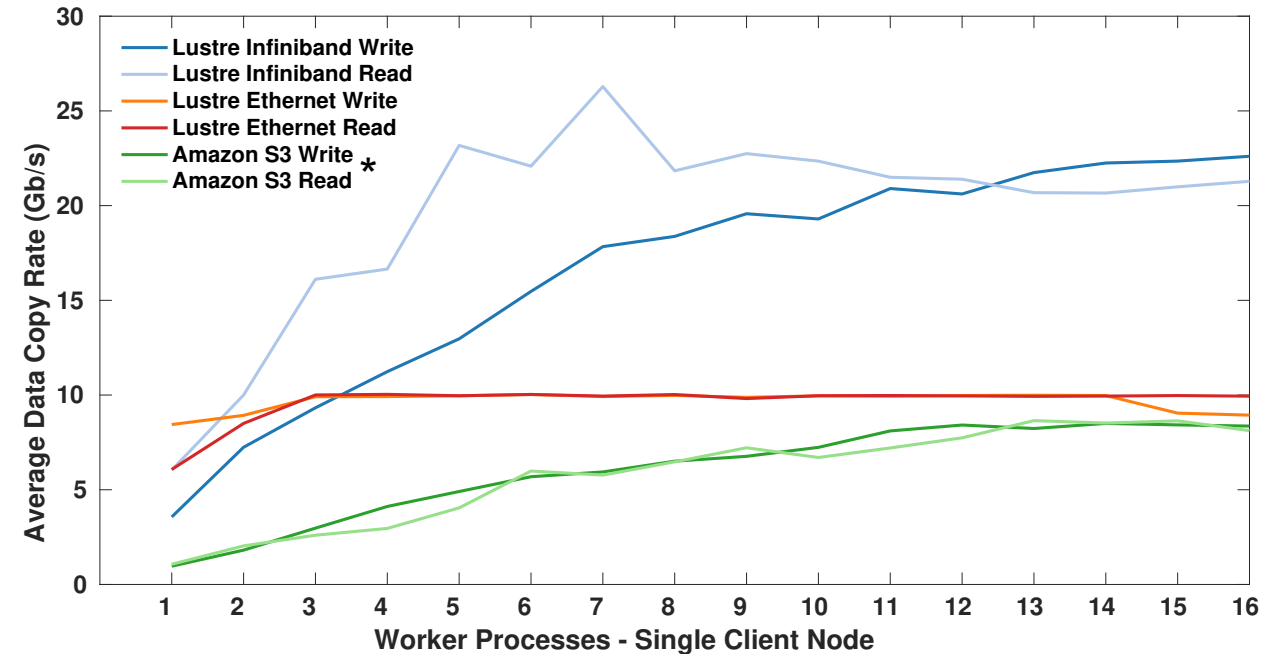
Only zero
carbon
emission
system
in Top500

Manycore system sustains Lincoln's leadership position in interactive supercomputing

- Compatible with all existing LLSC software
- Provides processing (6x) and bandwidth (20x) for physical simulation and machine learning applications



Supercomputing I/O

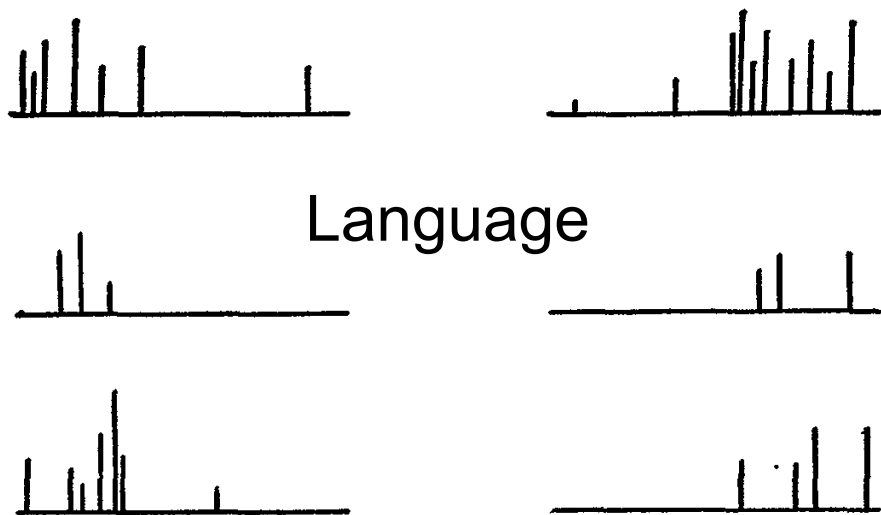
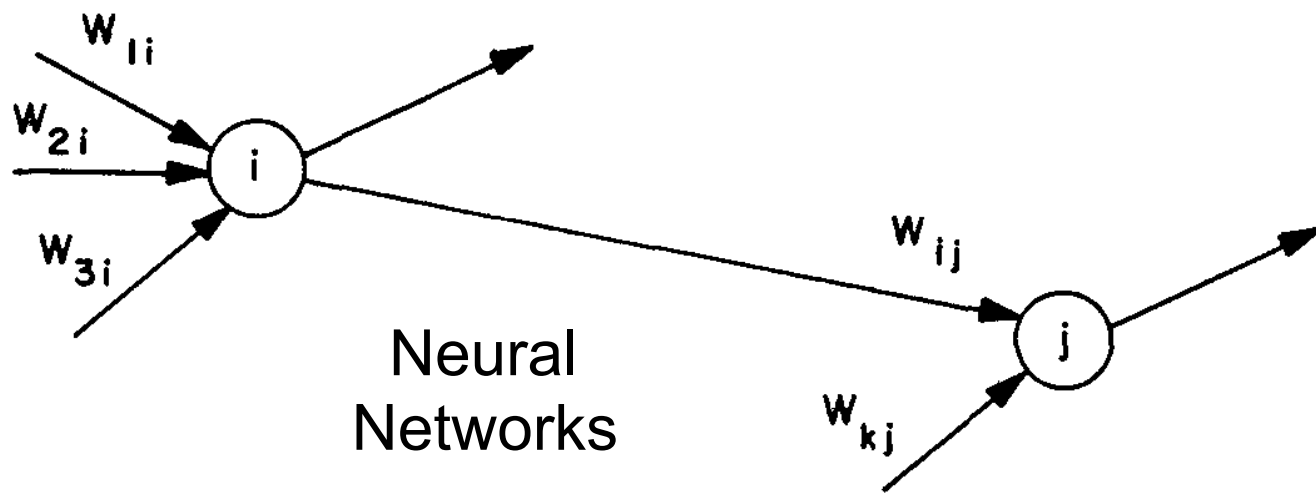


- Parallel simulations produce enormous amounts of data
- Parallel file systems designed to meet these requirements

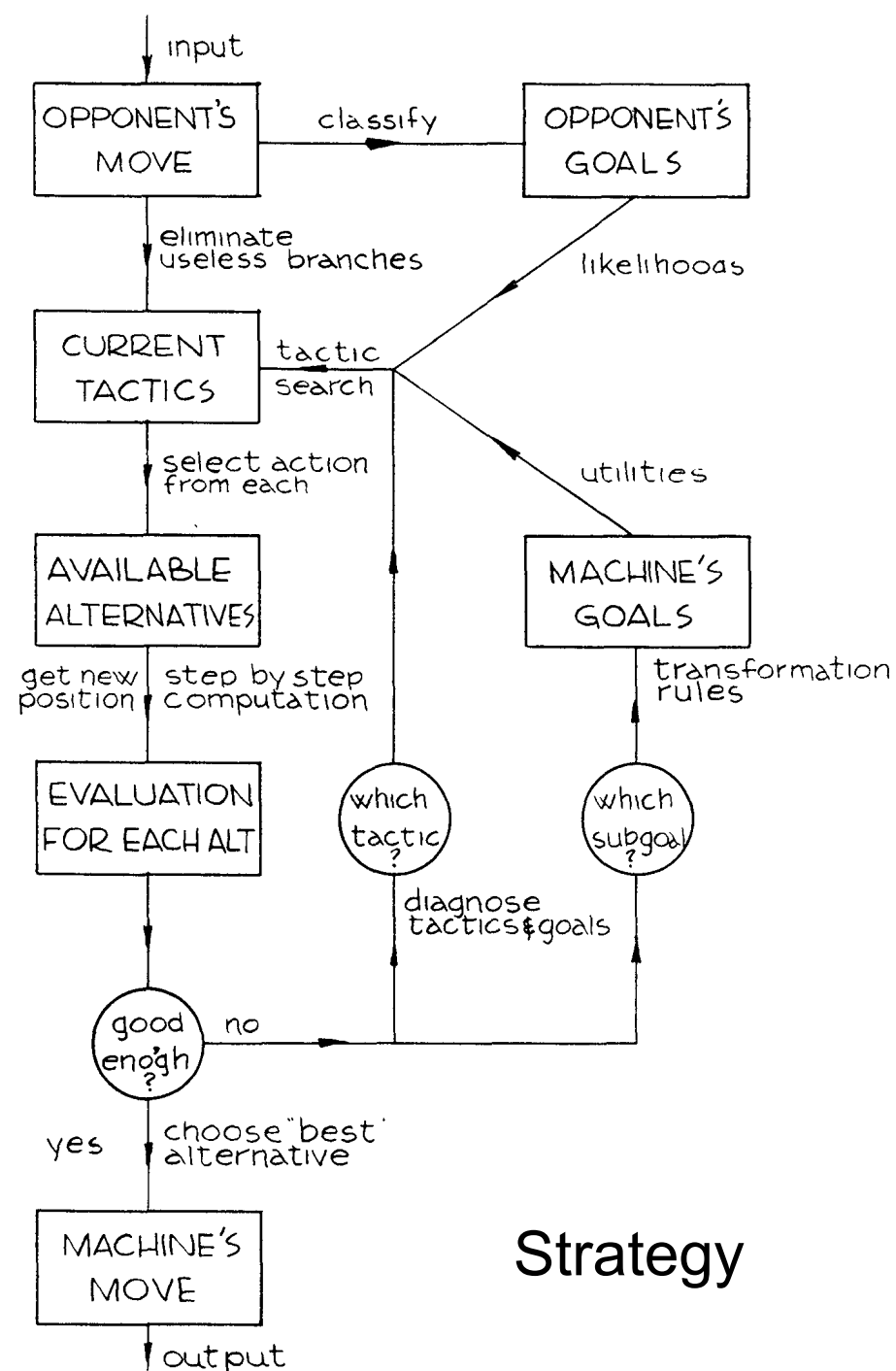
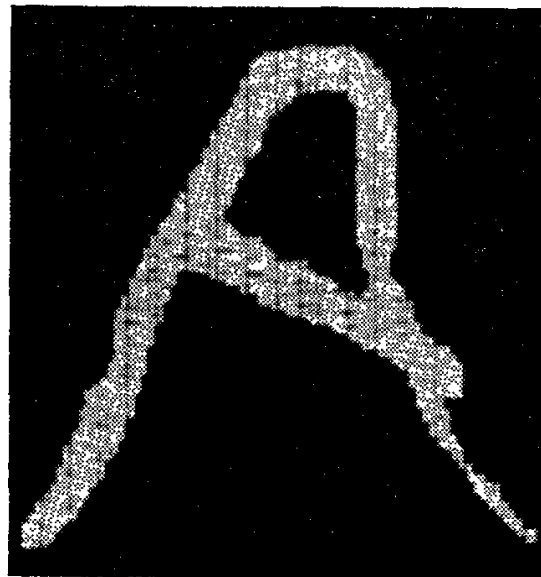


Outline

- Introduction
- Big Data (Scale Out)
- Supercomputing (Scale Up)
- **Machine Learning (Scale Deep)**
- Summary



Vision



Deep Neural Networks (DNNs) for Machine Learning

- Increased abstraction at deeper layers

$$y_{i+1} = h(W_i y_i + b_i)$$

requires a non-linear function, such as

$$h(y) = \max(y, 0)$$

- Matrix multiply $W_i y_i$ dominates compute

Remark: can rewrite using GraphBLAS as

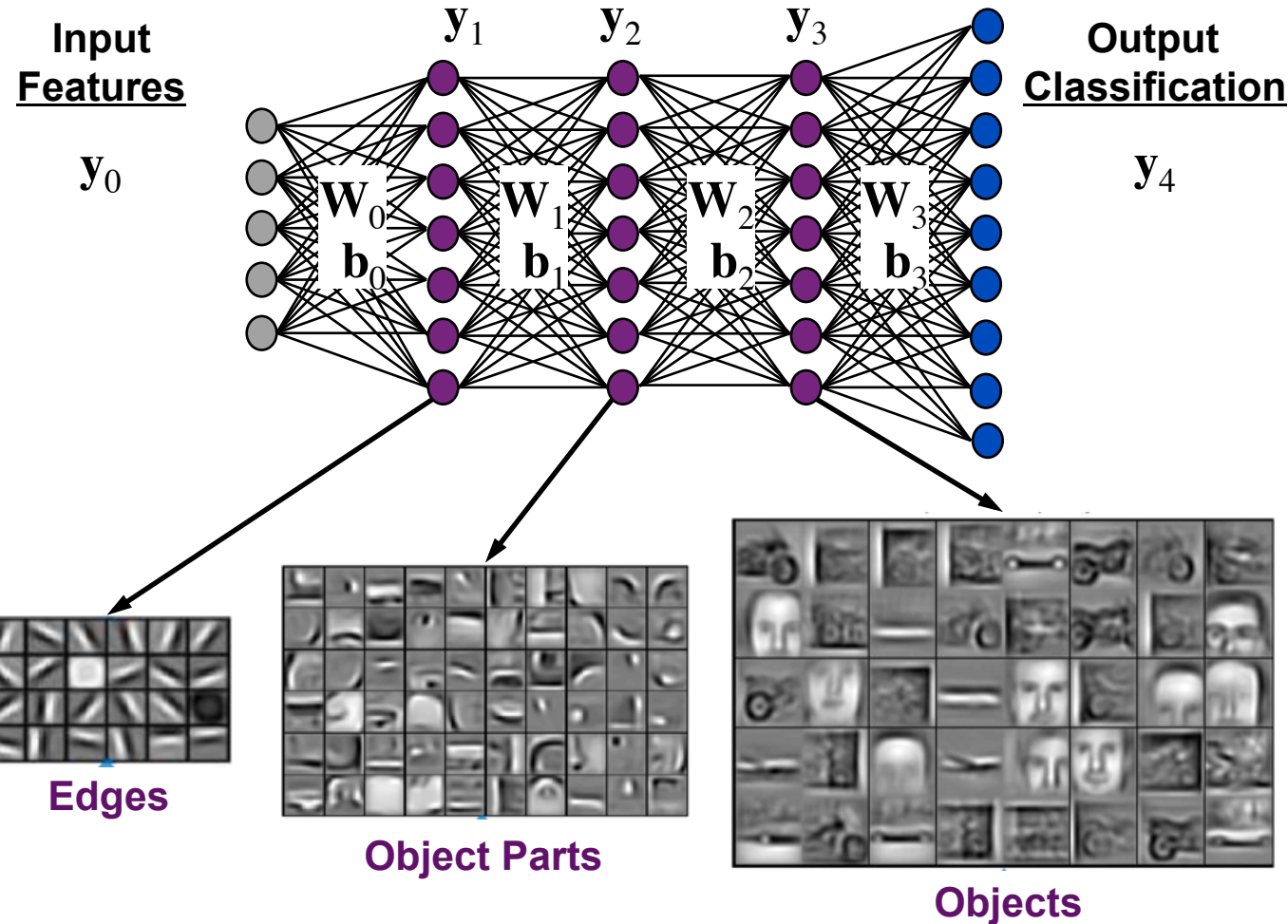
$$y_{i+1} = W_i y_i \otimes b_i \oplus 0$$

where $\oplus = \max()$ and $\otimes = +$

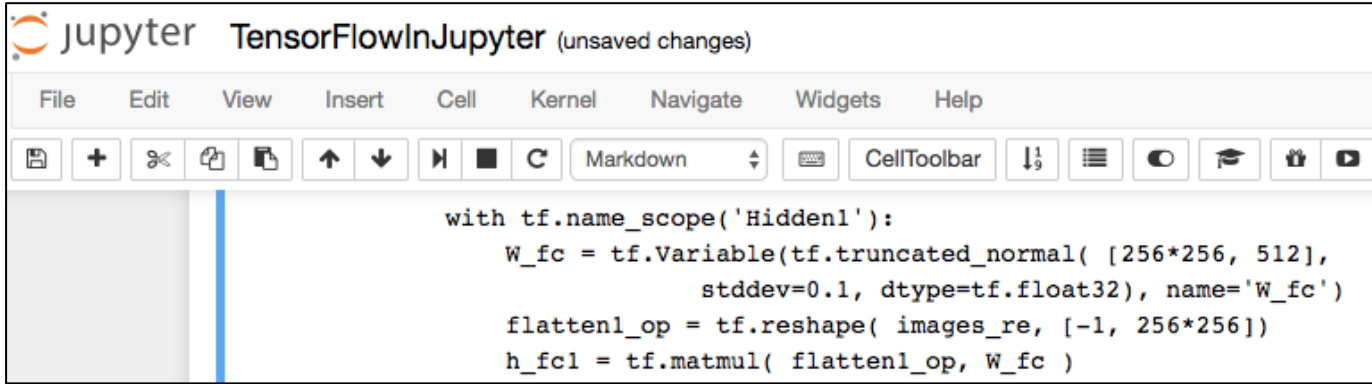
DNN oscillates over two linear semirings

$$S_1 = (\mathbb{R}, +, \cdot, 0, 1)$$

$$S_2 = (\{-\infty \cup \mathbb{R}\}, \max, +, -\infty, 0)$$



Example Machine Learning Software

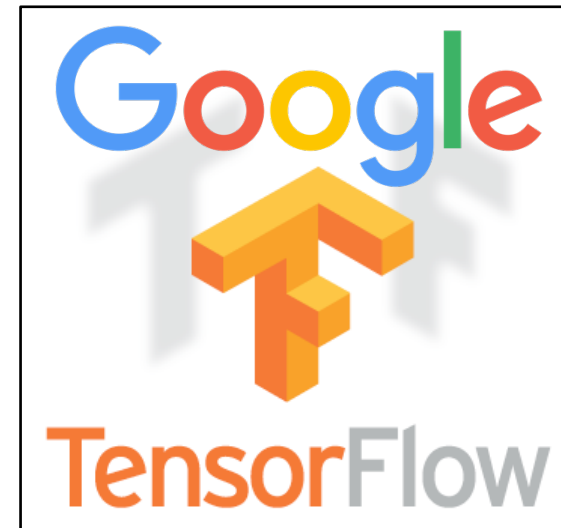


```

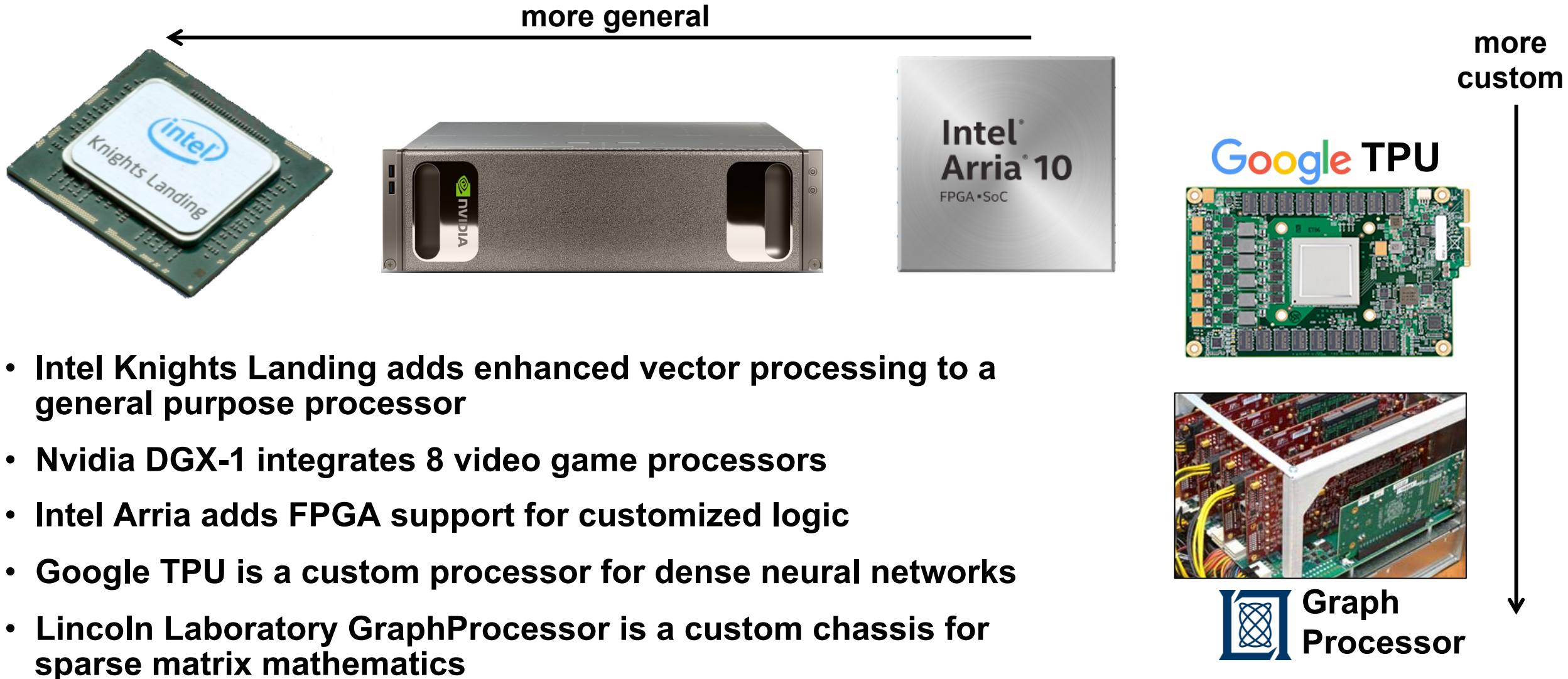
jupyter TensorFlowInJupyter (unsaved changes)
File Edit View Insert Cell Kernel Navigate Widgets Help
[Icons] [Markdown] [CellToolbar] [Icons]

with tf.name_scope('Hidden1'):
    W_fc = tf.Variable(tf.truncated_normal( [256*256, 512],
                                           stddev=0.1, dtype=tf.float32), name='W_fc')
    flatten1_op = tf.reshape( images_re, [-1, 256*256])
    h_fc1 = tf.matmul( flatten1_op, W_fc )
    
```

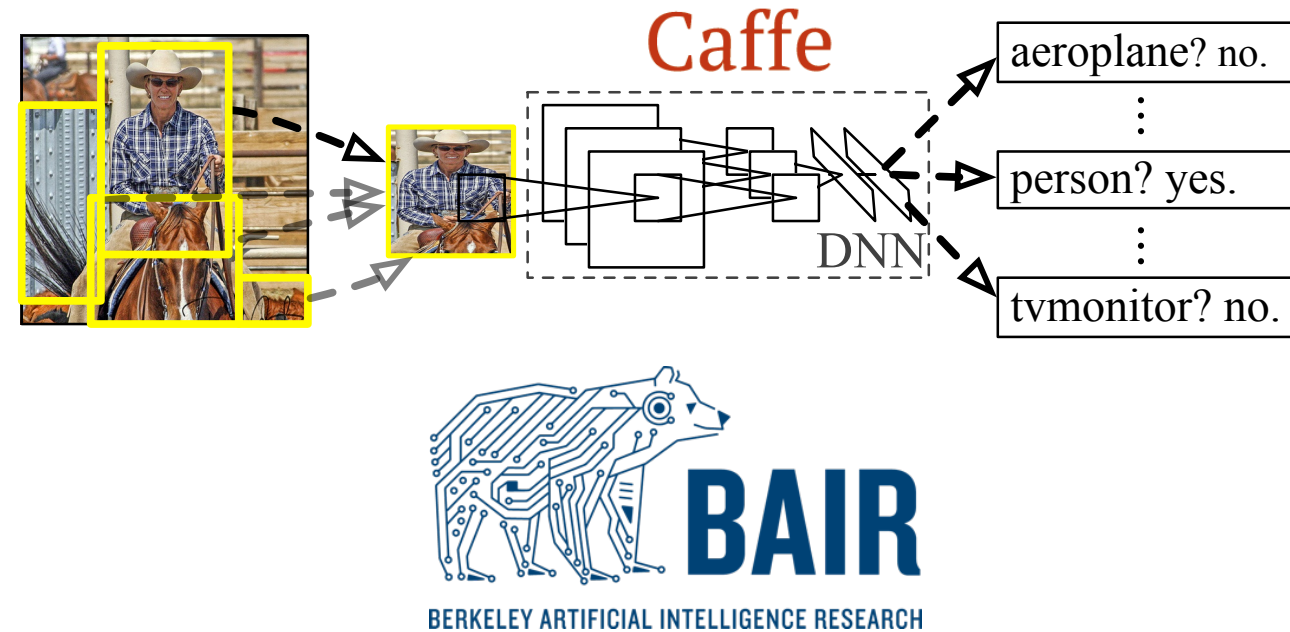
- Lots of machine learning software
- Designed for diverse data
- Jupyter interactive portal interface
 - Similar to Mathematica notebooks



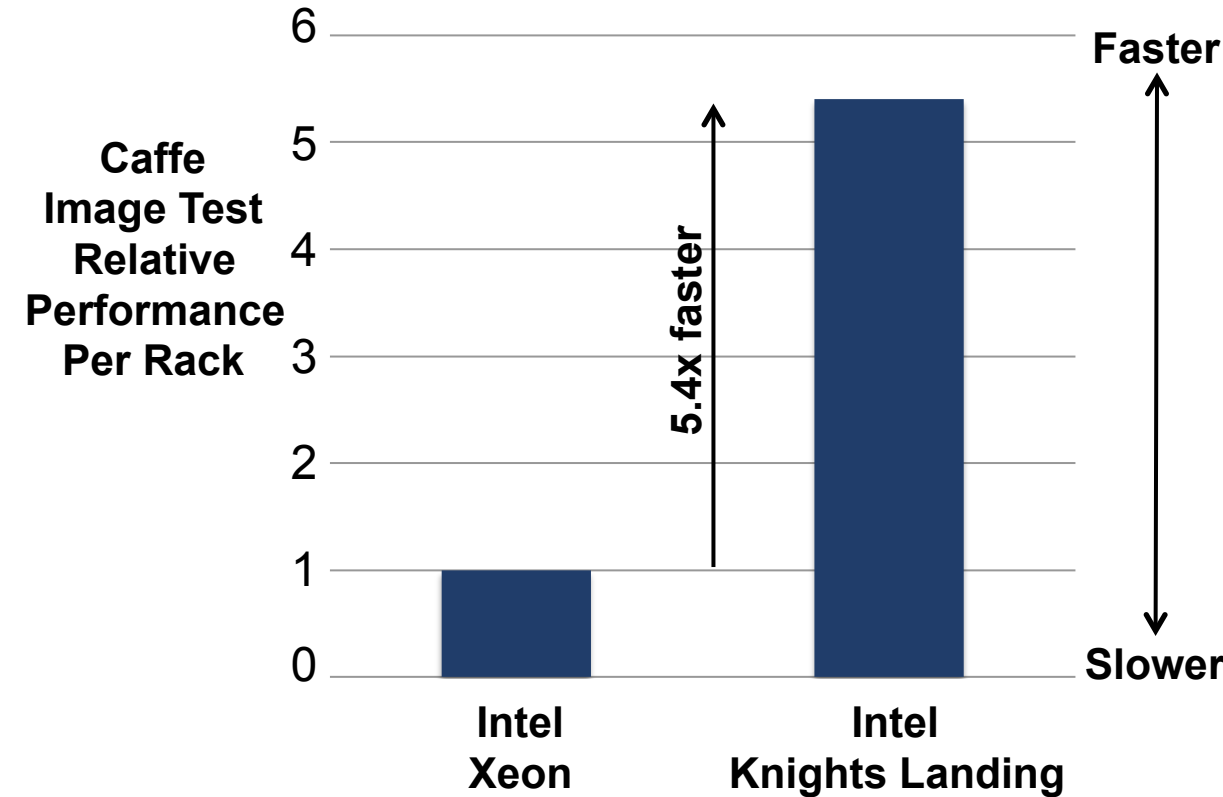
Example Machine Learning Hardware



Performance: Caffe Deep Learning Framework



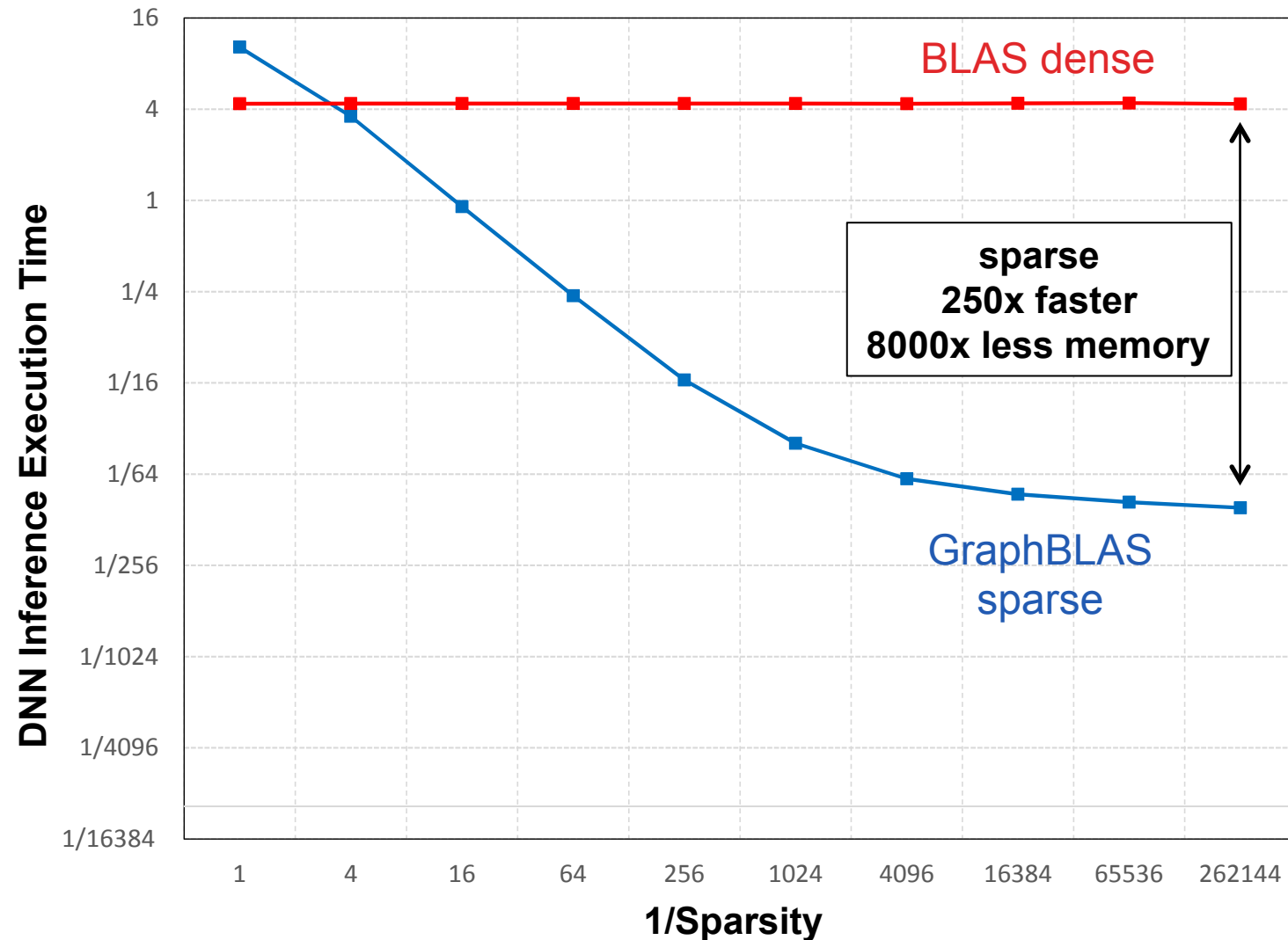
- Caffe is a widely used machine learning package developed at UC Berkeley
- Intel Knights Landing processor delivers 5.4x more performance per rack over standard Intel Xeon processor





Next Generation: Sparse Neural Networks

- Large neural networks drive machine learning performance
 - 100,000s features
 - 10s of layers
 - 100,000s of categories
- Larger networks need memory
 - Requires sparse implementation
- Natural fit for GraphBLAS



Summary

Volume

- **Challenge:** Scale of data beyond what current approaches can handle
- **Hardware:** Scale-out, more servers per data center (hyperscale)



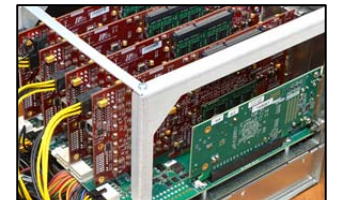
Velocity

- **Challenge:** Analytics beyond what current approaches can handle
- **Hardware:** Scale-up, more transistors per server (accelerators)



Variety

- **Challenge:** Diversity beyond what current approaches can handle
- **Hardware:** Scale-deep, more customizable processors (FPGAs, ...)

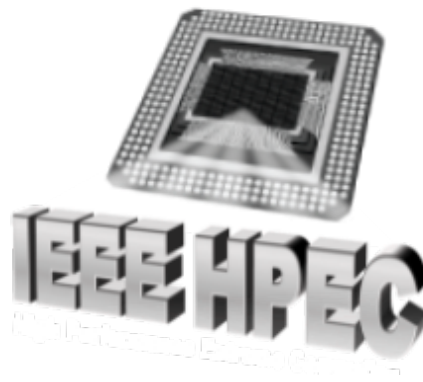


Requires mathematically rigorous approaches to insulate users from scaling

21st IEEE HPEC Conference

September 12-14, 2017 (ieee-hpec.org)

- **Premiere conference on High Performance Extreme Computing**
 - Largest computing conference in New England (280 people)
- **Invited Speakers**
 - Prof. Ivan Sutherland (Turing Award)
 - Trung Tran (DARPA MTO)
 - Andreas Olofsson (DARPA MTO)
 - Prof. Barry Shoop (IEEE President)
- **Special sessions on**
 - DARPA Graph Challenge
 - Resilient systems
 - Big Data
 - GPU & FPGA Computing



Platinum Co-Sponsors



Silver Sponsor

MITRE

Cooperating Society

siam

Media Sponsor



Technical Organizer



- Sustains gov't leadership position
- Keeps gov't users ahead of the technology curve