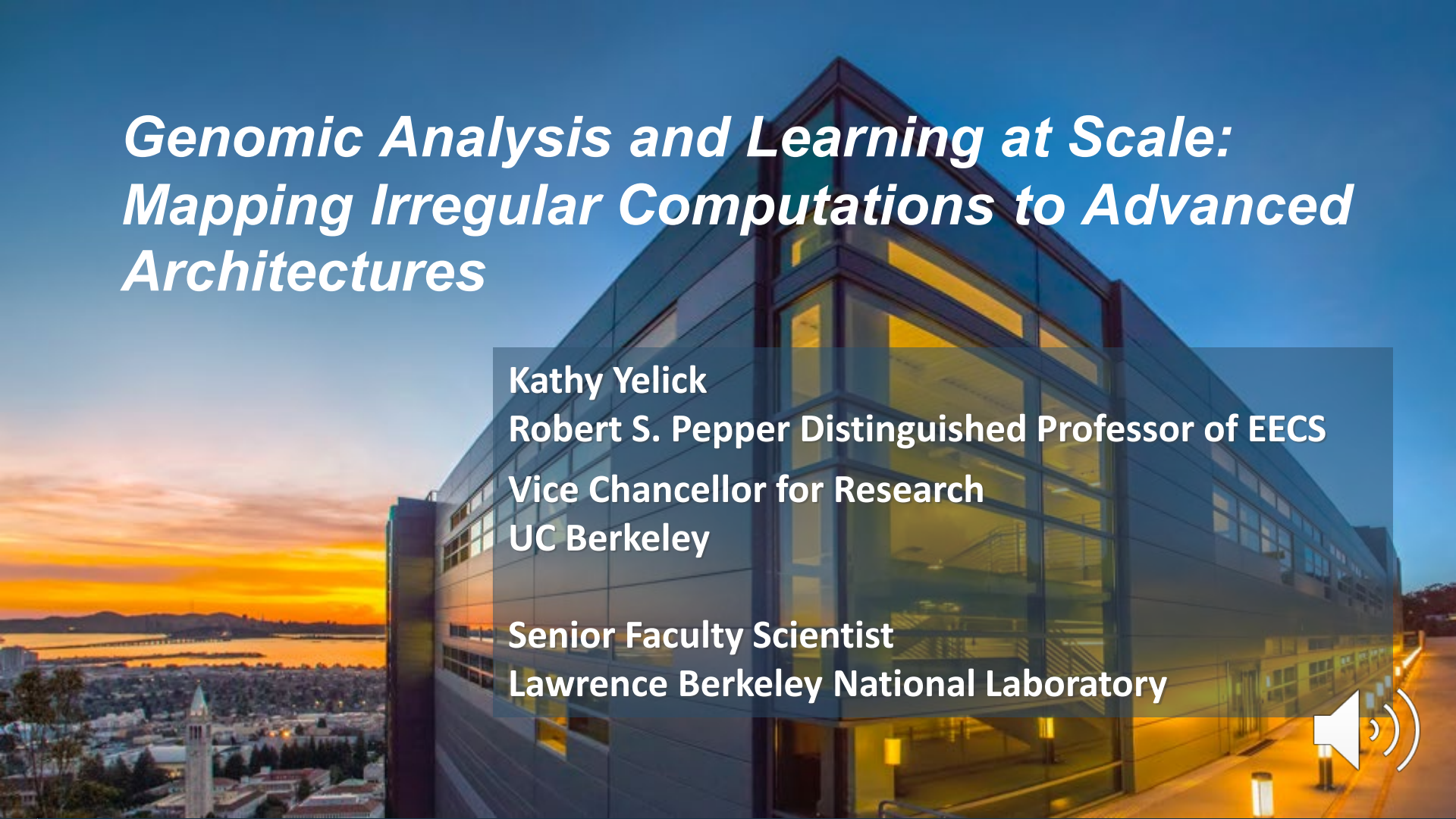


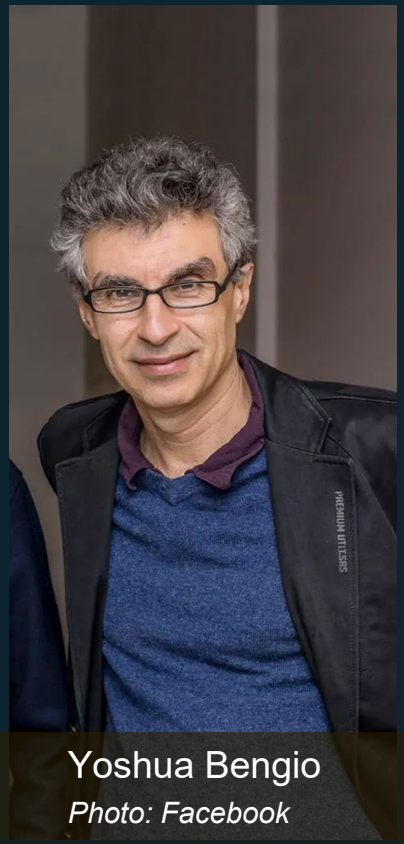
Genomic Analysis and Learning at Scale: Mapping Irregular Computations to Advanced Architectures

Kathy Yelick
Robert S. Pepper Distinguished Professor of EECS
Vice Chancellor for Research
UC Berkeley

Senior Faculty Scientist
Lawrence Berkeley National Laboratory


ECP
EXASCALE
COMPUTING
PROJECT

2018 ACM Turing Award for Deep Learning

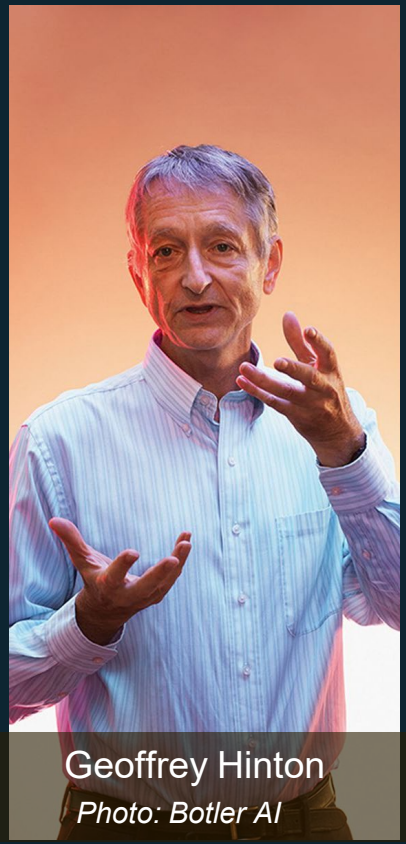


Yoshua Bengio

Photo: Facebook

Yann LeCun

Photo: Google



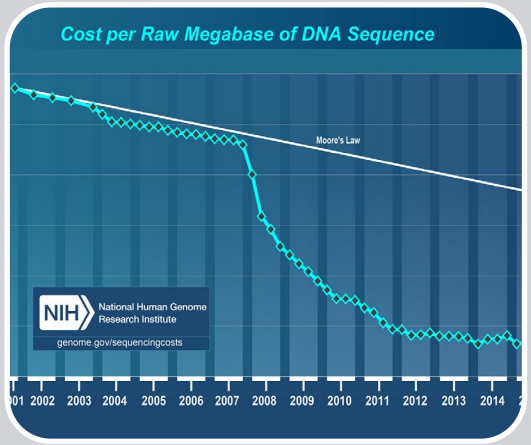
Geoffrey Hinton

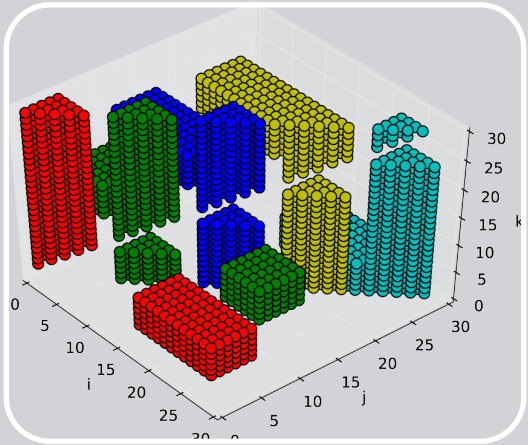
Photo: Botler AI

Hinton's Turing Lecture:

"So I think a lot of the credit for deep learning really goes to the people who collected the big databases like Fei Fei Li and the people who made the computers go fast like David Patterson and others."

Other problems where data + machines win?

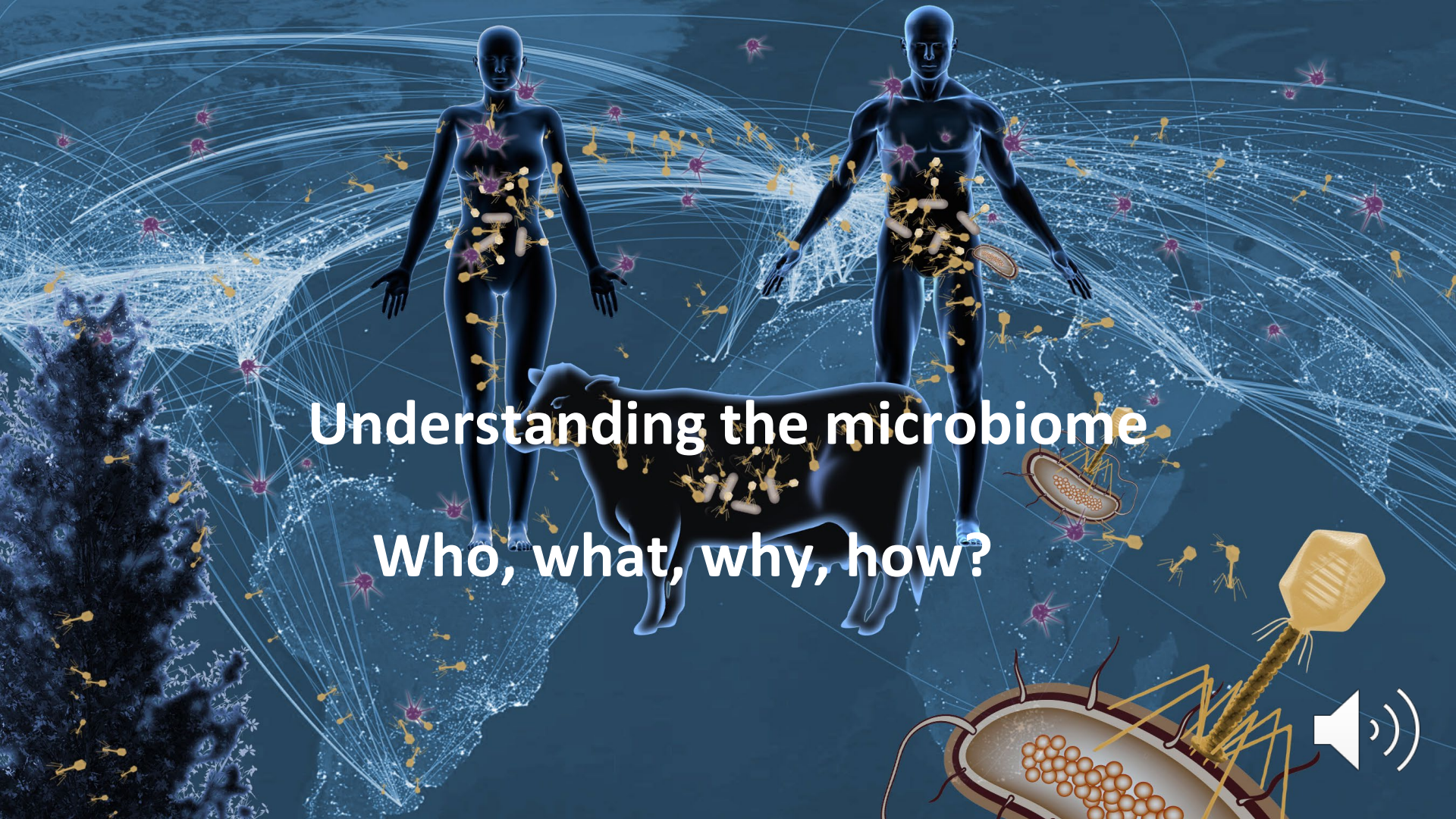




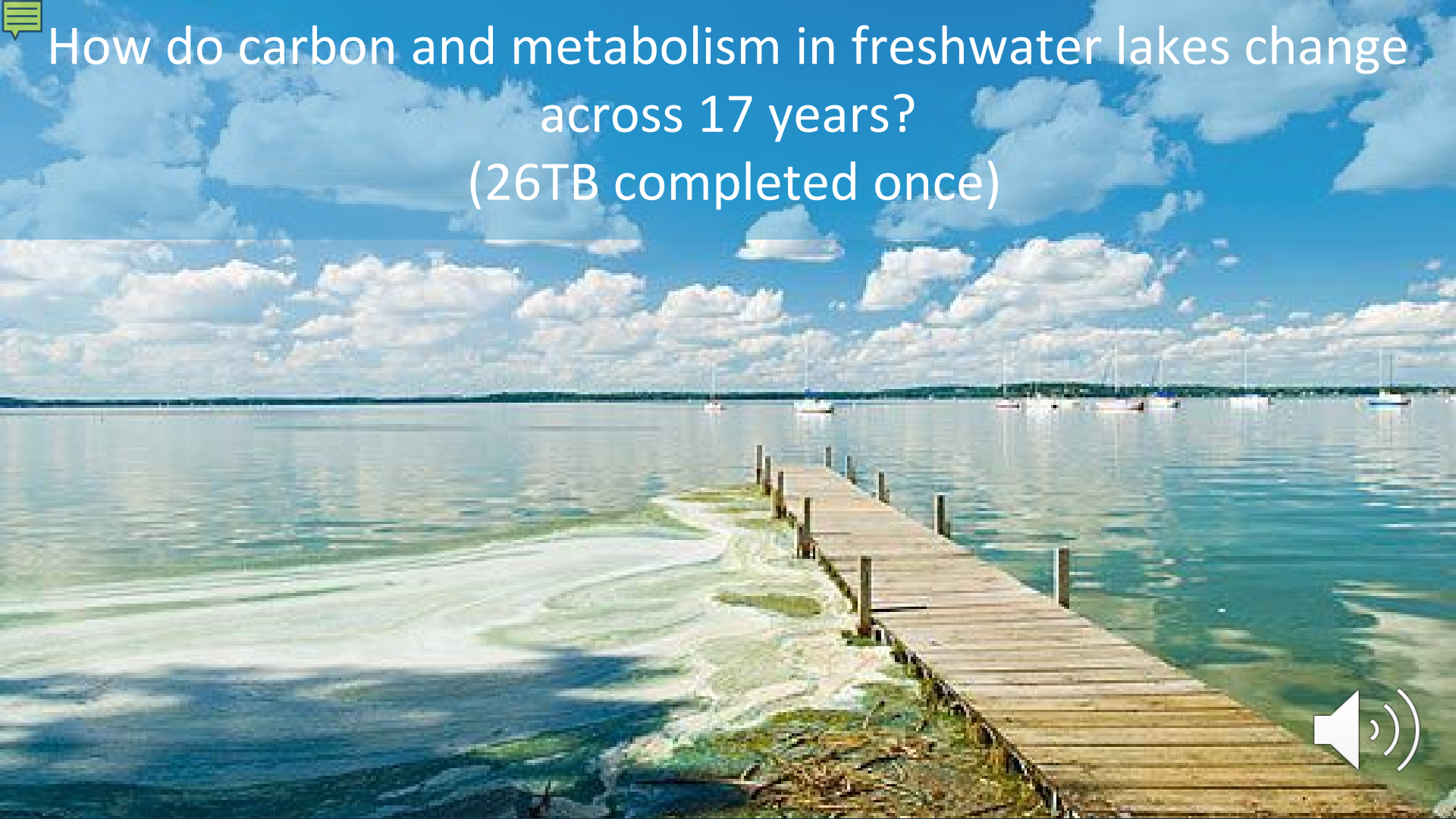
Big Data

Big Machines

Scalable Algorithms


Understanding the microbiome
Who, what, why, how?


What happens to microbes after a wildfire?
(1.5Terabtyes - completed)
JGI
JOINT GENOME INSTITUTE
NERSC


How do carbon and metabolism in freshwater lakes change across 17 years?
(26TB completed once)

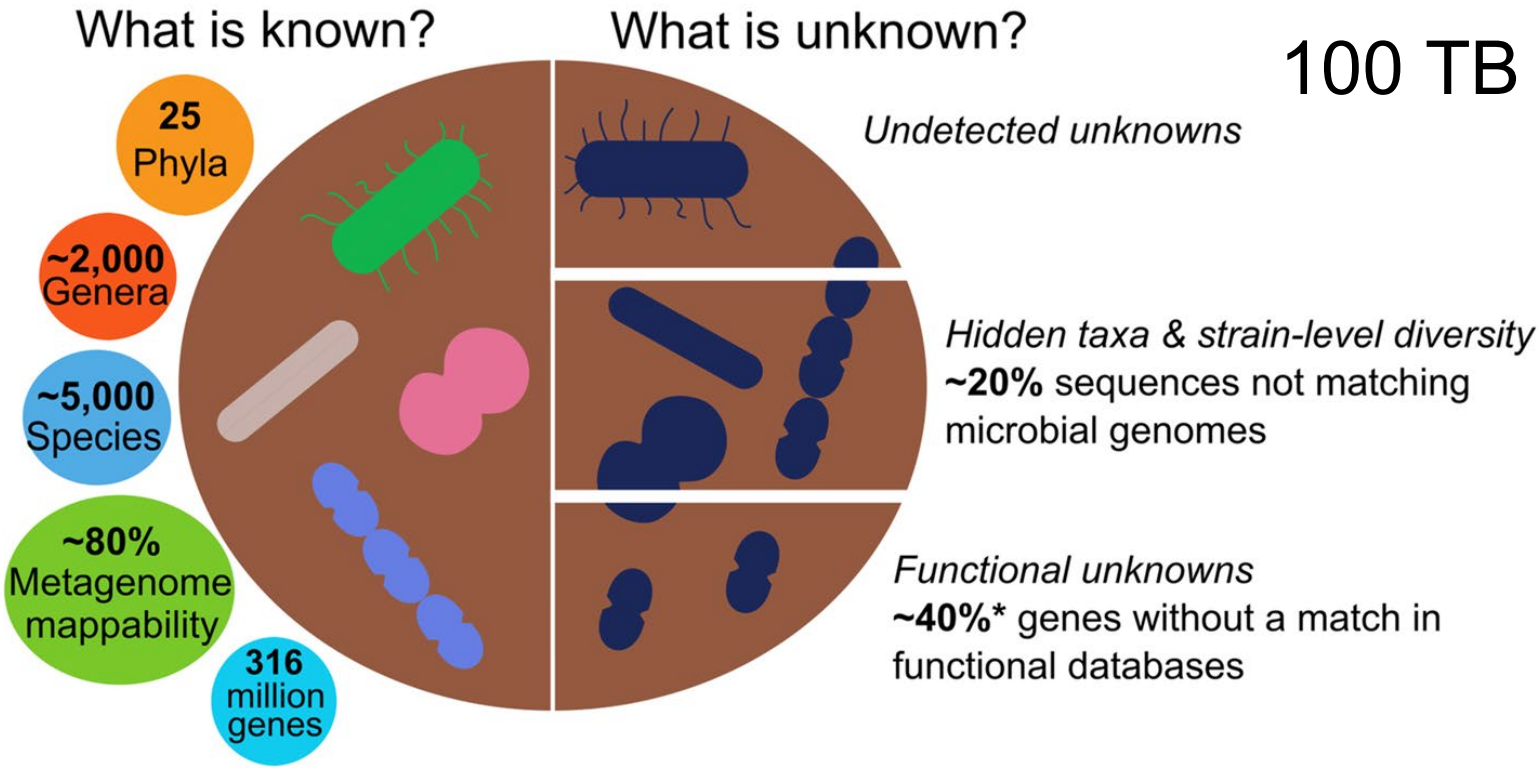
Tara Oceans

Showing the invisible life of the ocean

- 2009–2013 expeditions
- 35000 samples from all oceans

- 84 TB of data!

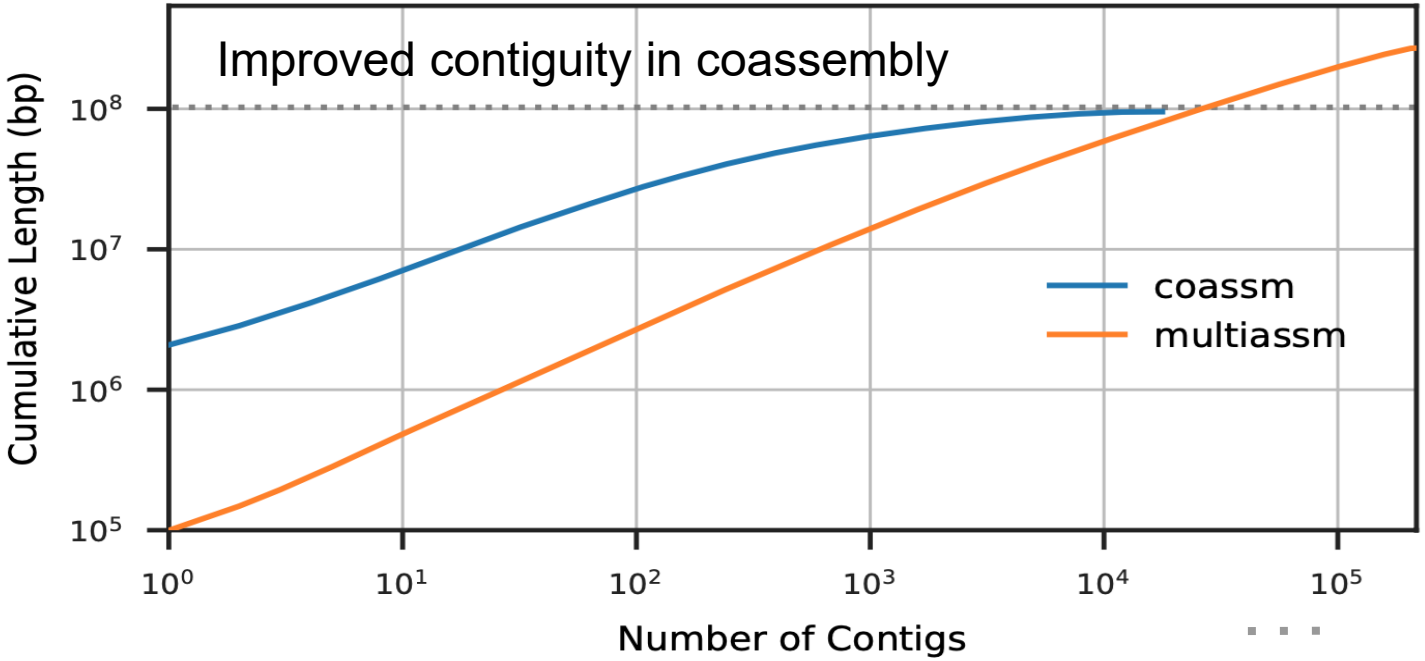
The Human Microbiome



100 TB of data!

Thomas, A.M., Segata, N. Multiple levels of the unknown in microbiome research. *BMC Biol* **17,** 48 (2019).

More Data Yields Better Science

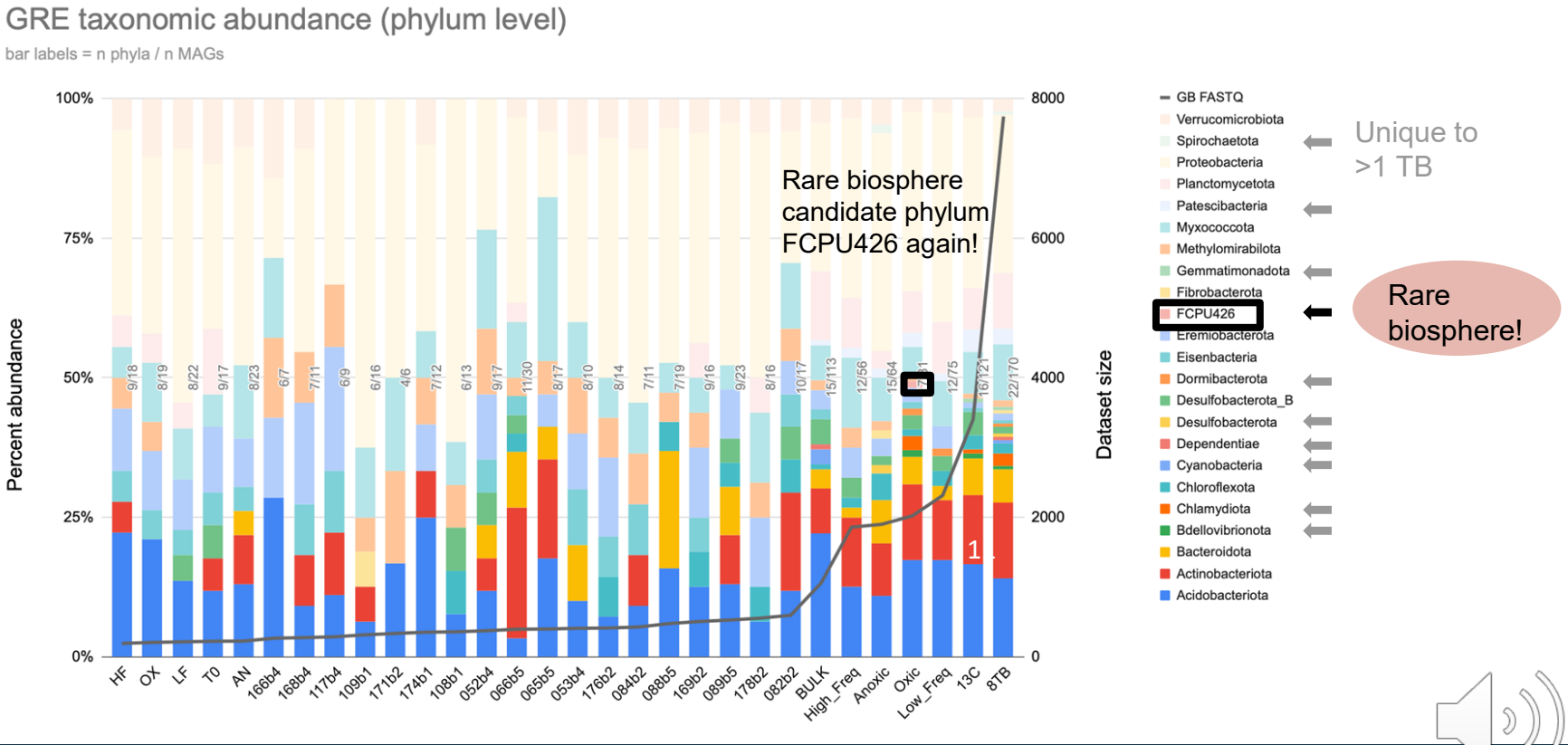


Comparable to best known assemblers on small datasets
Unique science results on large ones, co-assembled

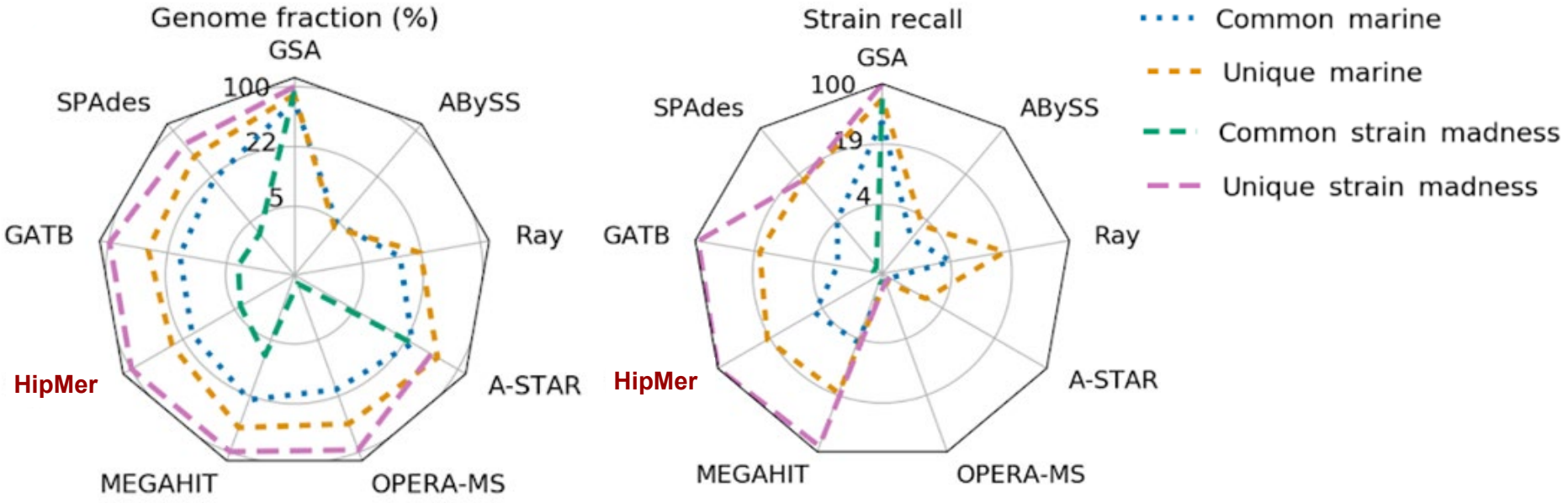
800 GB of soil (Western Arctic, 12) data plus synthetic data from 64 reference genomes

Hofmeyr et al. (2020). Terabase-scale metagenome coassembly with MetaHipMer. *Nature Sci Rep*, 10(1)

More taxonomic diversity

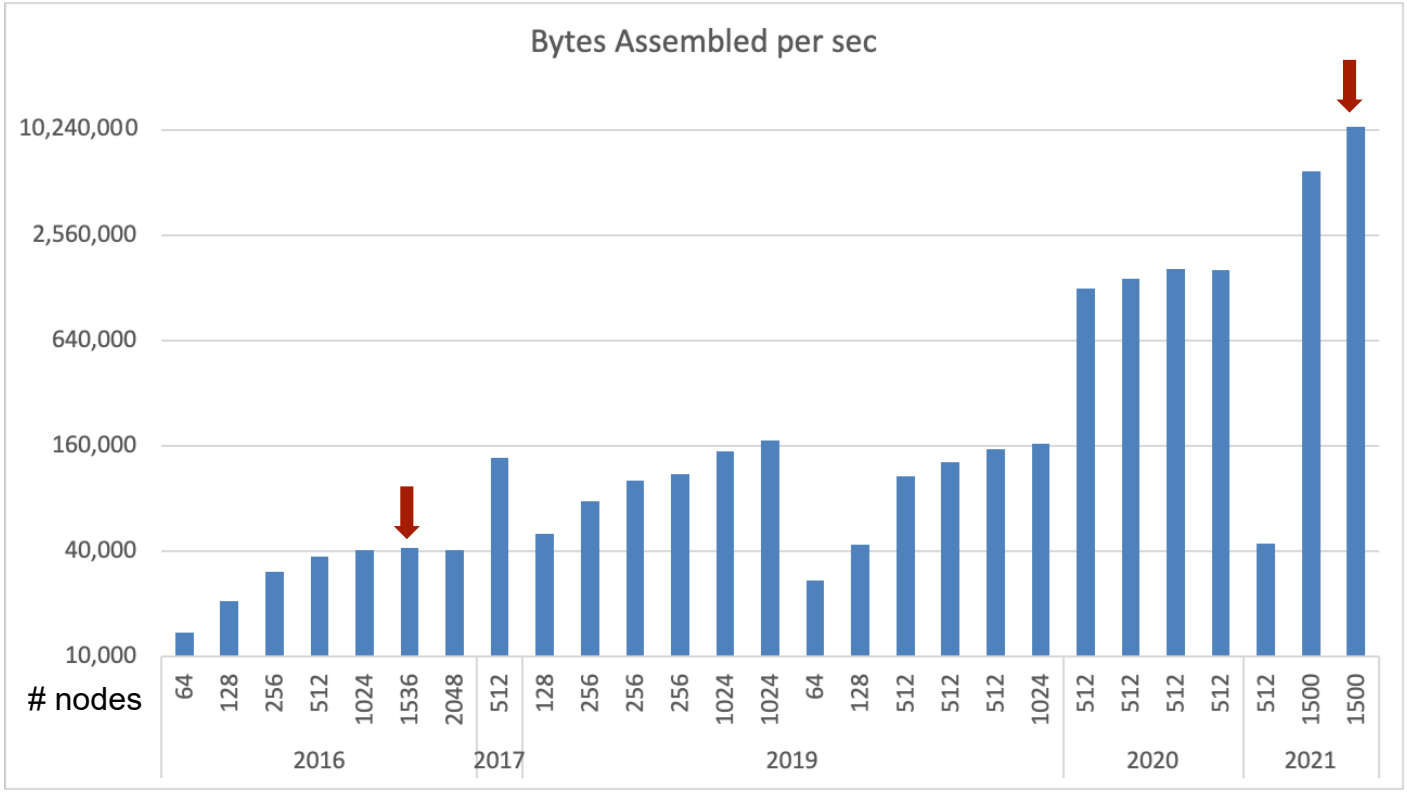


Ensuring High Quality Assemblies



"The **best ranking method across metrics and all datasets was HipMer**...."

Assembly Rate on Science Data



Not just data size

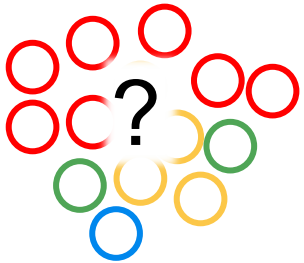
Effective use of HPC increased assembly rate

Over 250x on ~equal node counts!

- better algorithms
- less software
- use of GPUs

ExaBiome: Exascale Solutions for the Microbiome

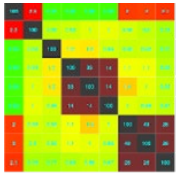
Sample



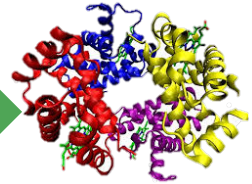
Function

Sequence

Compare



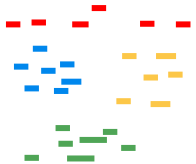
Structure



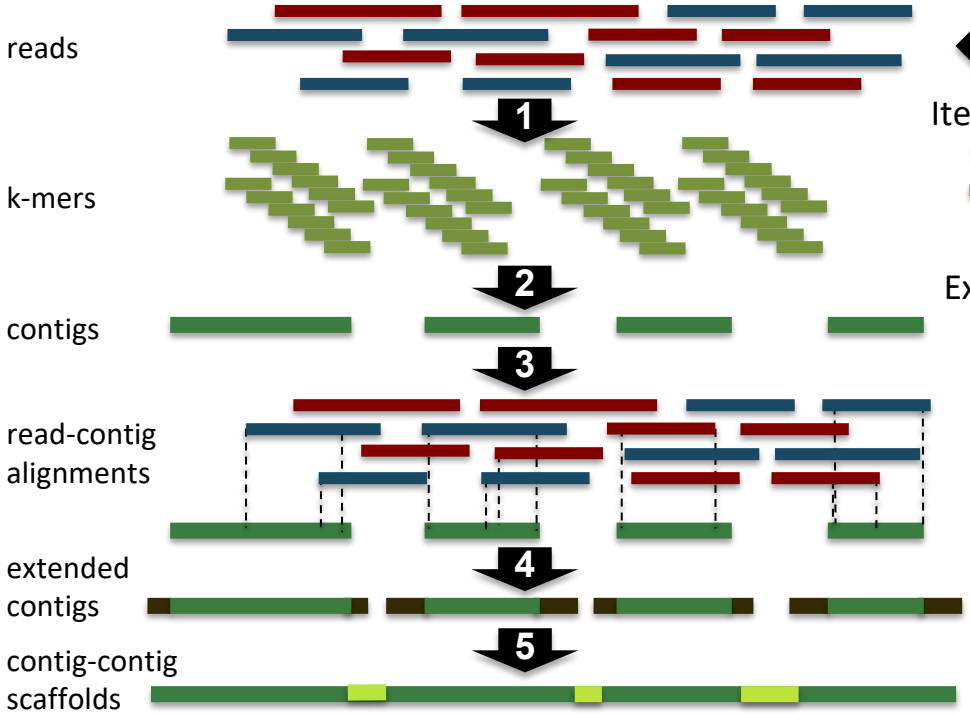
Assemble

Bin

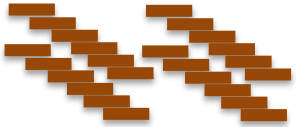
Find Genes



MetaHipMer Assembly Pipeline (UPC++)



Iterate for k+s



Extract k+s-mers

1) K-mer Analysis
K-mer histogram

2) Contig Generation
Walk k-mer graph

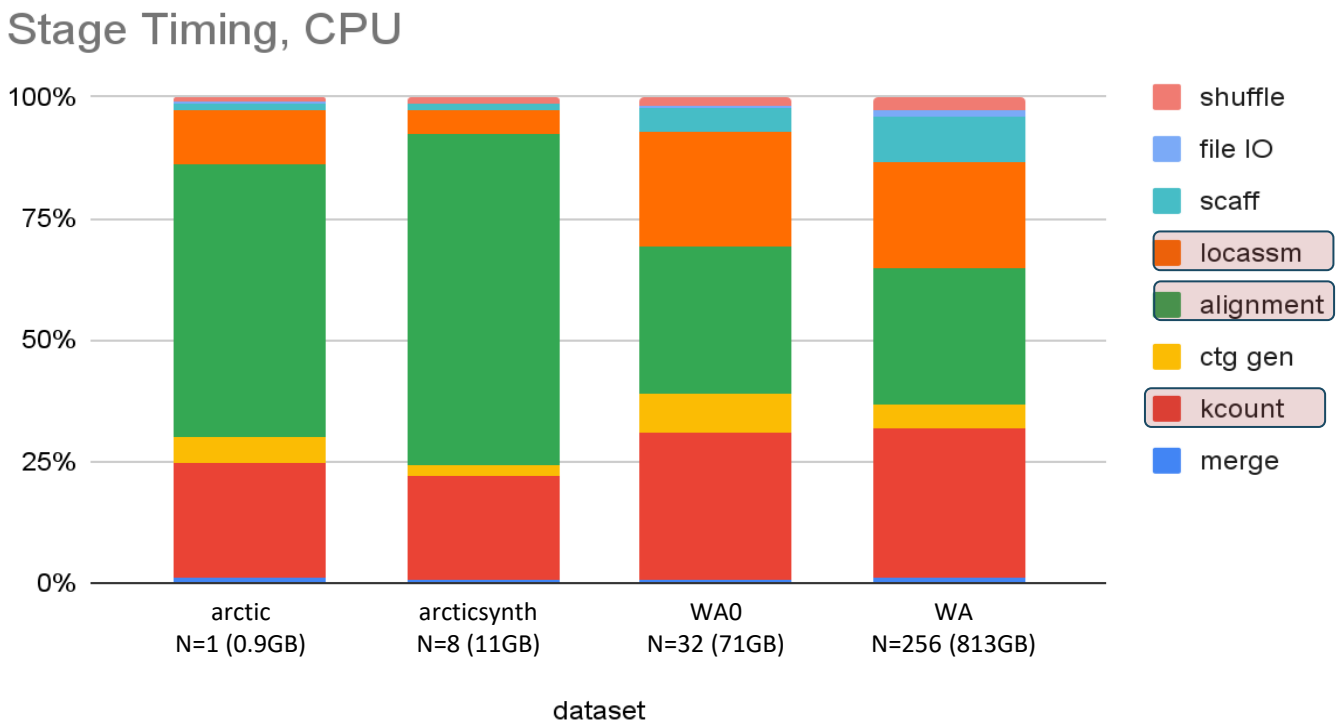
3) Alignment
Align reads to contigs

4) Local Assembly
Extend ends of contigs

5) Scaffolding
Walk contig graph (iterate)

Actual pipeline is more complex, simplified for purpose of presentation

MetaHipMer Time Breakdown



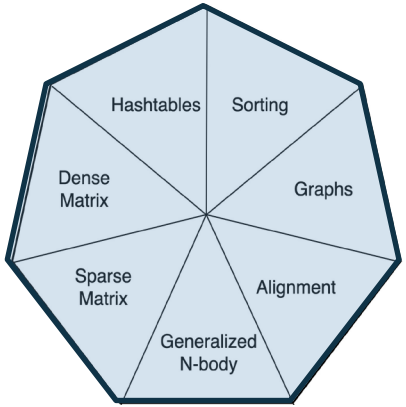
Weak-ish scaling

CPU time for alignment slower than “normal” due to SIMD Power9 issues

Simulation Vs. Data Motifs

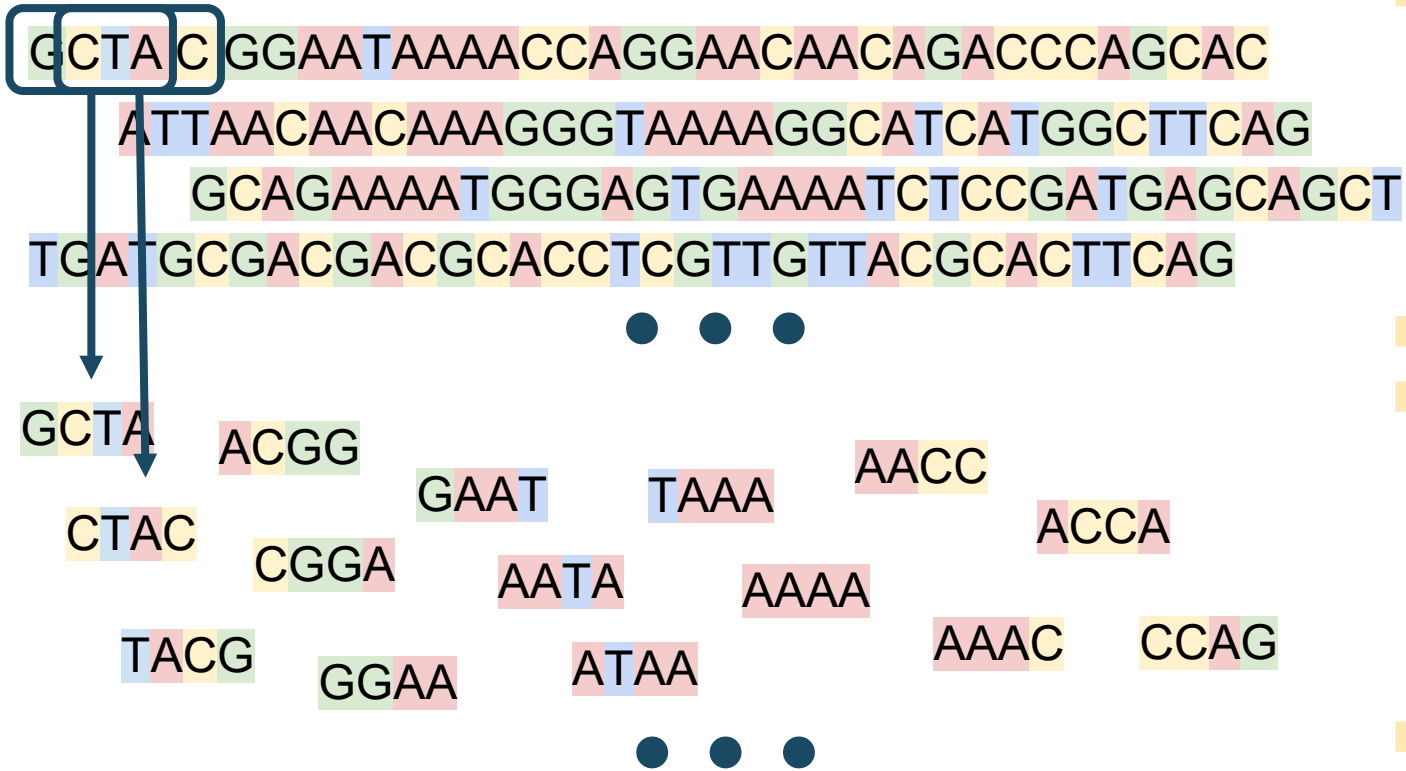
7 Dwarfs of Simulation	7 Giants of Big Data
Particle methods	Generalized N-Body
Unstructured meshes	Graph-theory
Dense Linear Algebra	Linear algebra
Sparse Linear Algebra	Hashing
Spectral methods	Sorting
Structured Meshes	Alignment
Monte Carlo methods	Basic Statistics
Phil Colella	NRC Report + our paper

Yelick, et al. “The Parallelism Motifs of Genomic Data Analysis”, Philosophical Transactions A, 2020


Hashtables
Sorting
Dense Matrix
Graphs
Sparse Matrix
Alignment
Generalized N-body

Hashing

Counting K-mers to Remove Errors



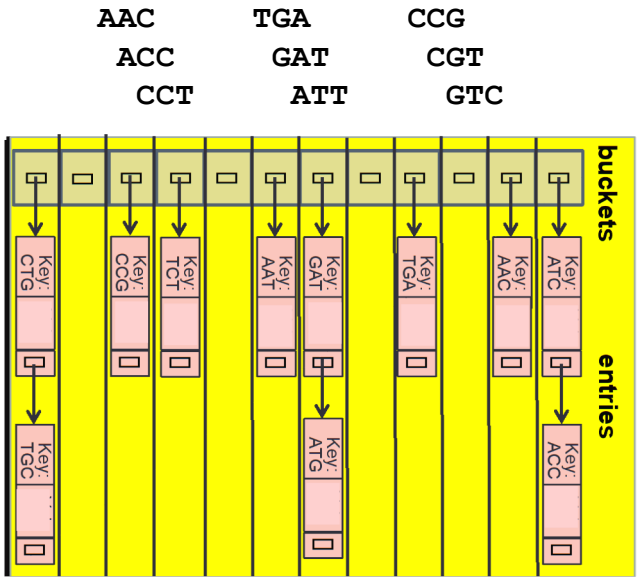
reads

k-mers

(e.g. k=4)

Distributed Hash Tables of K-Mers

Make hash table of k-mers



1-sided communication to insert / lookup

Keys are fixed-length strings:

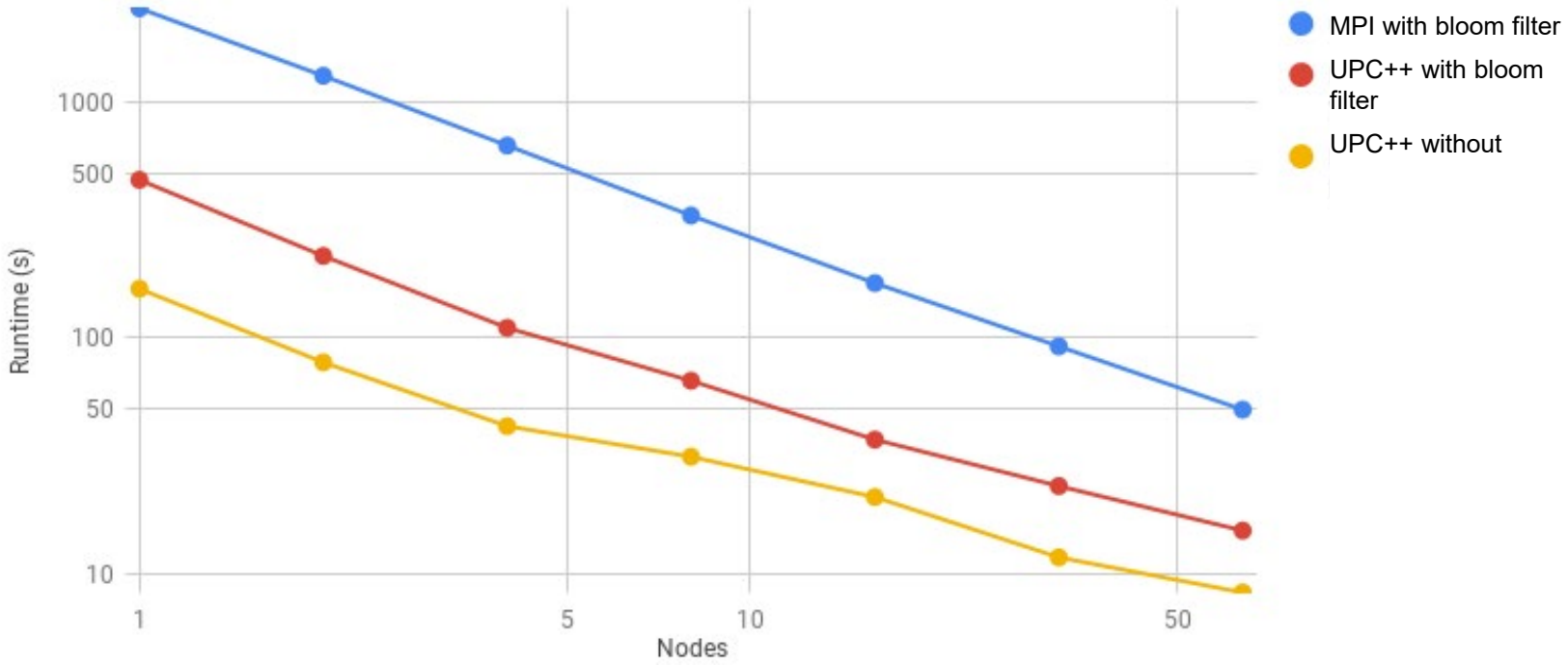
Values depend on application:

- A count to remove singletons

Close to k-times memory blowup

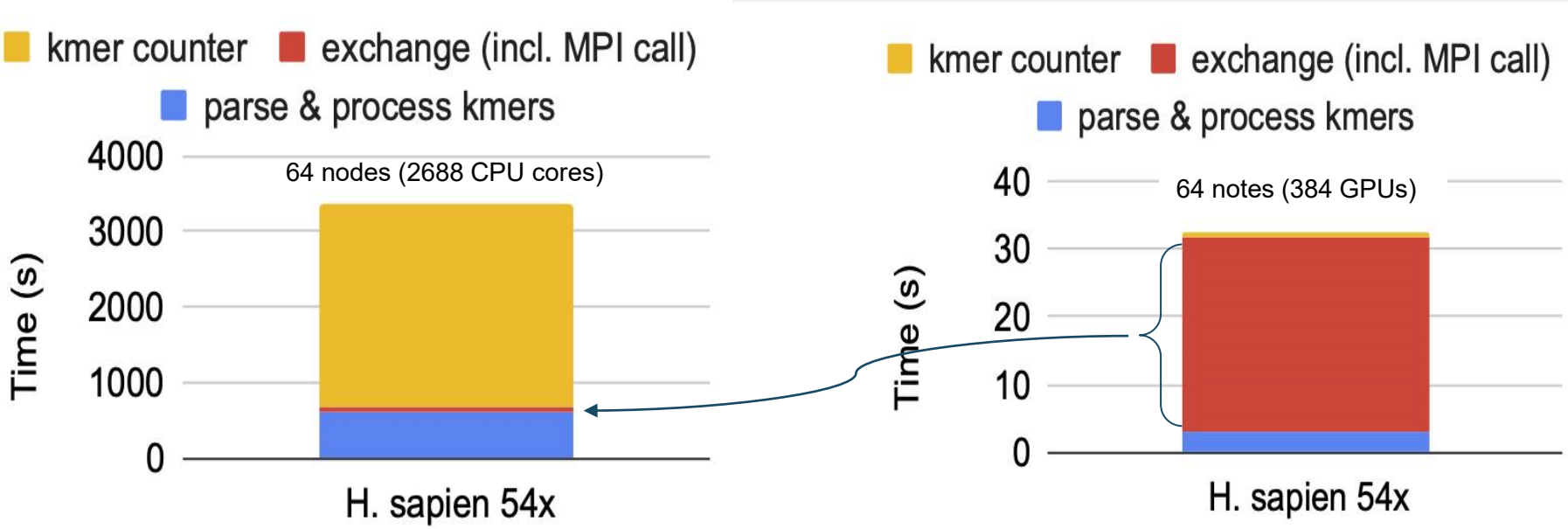
- Use Bloom filter to reduce space
- Asynchronous insert with UPC++

K-mer counting: All the Wires All the Time



Bulk-synchronous MPI vs Asynchronous 1-sided UPC++ (w/ and w/out Bloom Filter)

K-mer Counting: Finding Data Parallelism

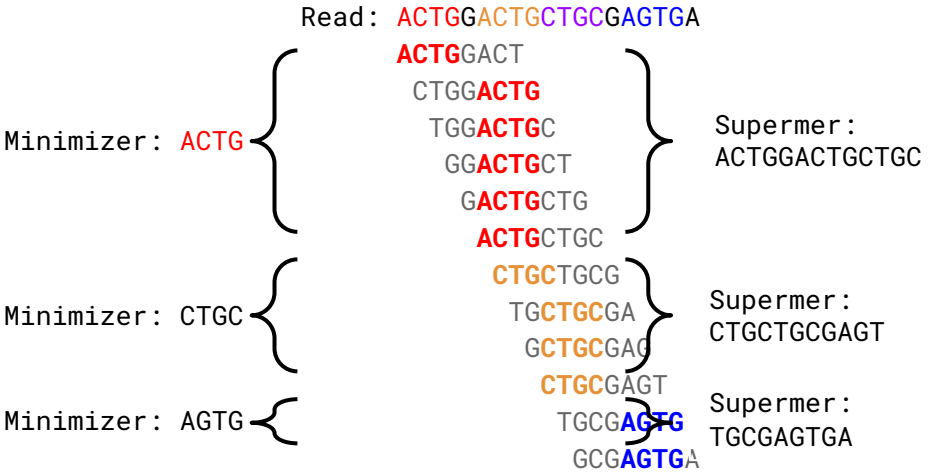


- K-mer counter on Summit. (Note scales -- red k-mer exchange time is roughly equal.)
- Reduce CPU/GPU communication by parsing as well as processing on GPU

Over 100x speedup!!

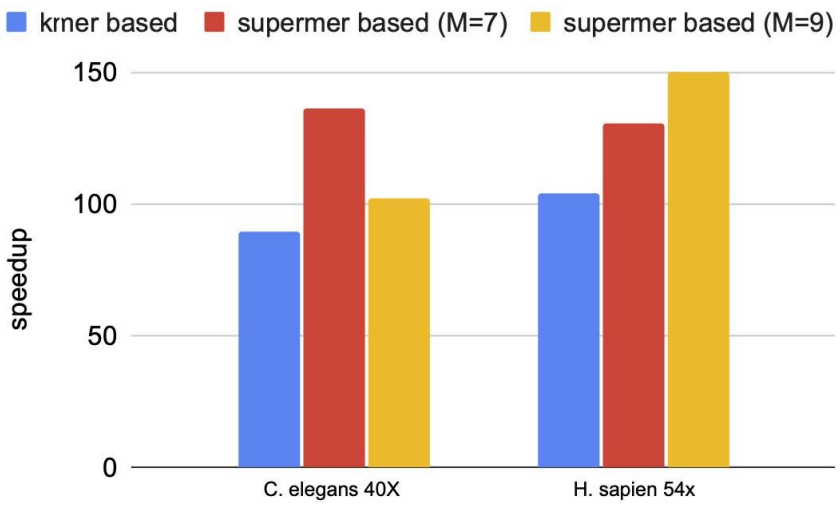
Israt Nisa, P. Pandey, M. Ellis, L. Oliker, A. Buluç, K. Yelick. Distributed-Memory k-mer Counting on GPUs. *IPDPS '21 (to appear)*

K-mer Counting: Reducing Communication



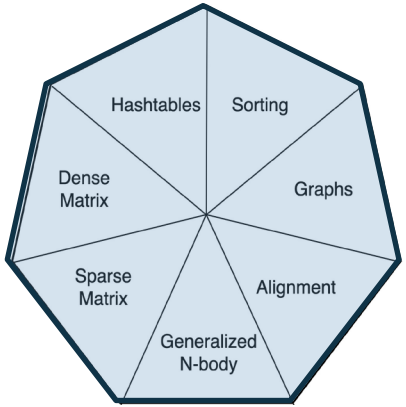
Reduce communication with "Supermers"

- Multiple contiguous k-mer
- map to the same process ID with minimizer-based hashing
- Saves volume (bandwidth) and number of messages (latency)



Speedup on 64 Summit nodes

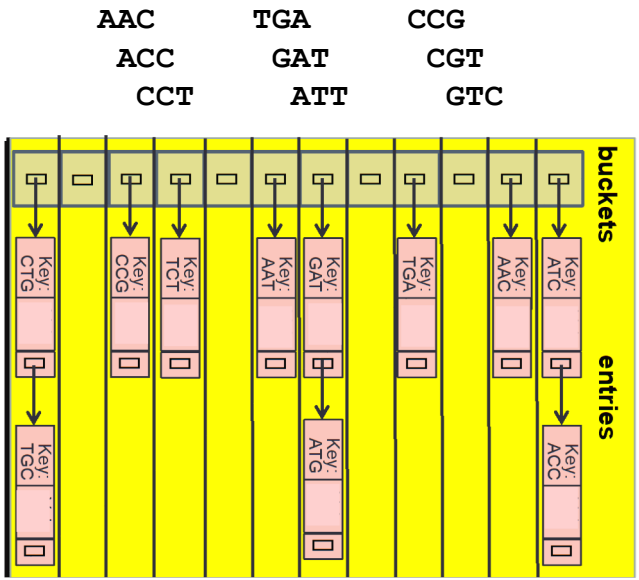
- 6 GPUs / node
- baseline: 42 cores / node


Hashtables
Sorting
Dense Matrix
Graphs
Sparse Matrix
Alignment
Generalized N-body

Graphs

K-Mer Hash Tables Viewed as a Graph

Make hash table of k-mers



1-sided communication to insert / lookup

Keys are fixed-length strings

Values

- Remove branches
- Find connected component “contigs”

Graph walk with poor locality

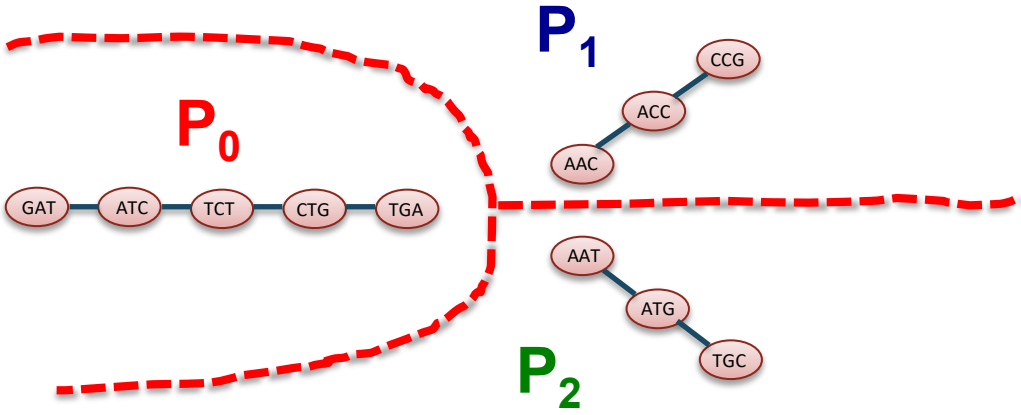
- Asynchronous lookup with UPC++

Avoiding Communication in Graph Walk (DFS)

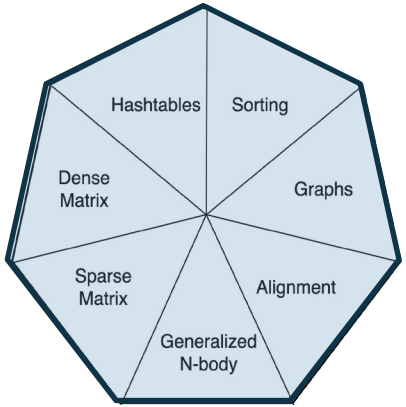
Next step in this assembler is a DFS on the k-mer graph (edges are k-1 overlaps)

Caching for temporal locality (reuse): if few large items, so lookups will repeat

Layout for spatial locality: if we have an "oracle" that approximate final genome



Traversal is up to 2.8x faster! Up to 76% reduction of off-node communication !


Hashtables
Sorting
Dense Matrix
Graphs
Sparse Matrix
Alignment
Generalized N-body

Alignment

Smith-Waterman: Dynamic Programming

	_	G	A	T	C	A	G	C	T
_	0	0	0	0	0	0	0	0	0
G	0	1	0	0	0	0	0	0	0
A	0	0	2	0	0	1	0	0	0
T	0	0	0	3	1	0	0	0	1
A	0	0	1	1	2	2	0	0	0
G	0	0	0	0	2	1	3	1	0
C	0	0	0	0	1	1	2	4	2
C	0	0	0	0	1	0	2	3	3

GATCACCT
GAT_ACCC

Scoring
insert/delete = -2
match = 1
mismatch = -1.

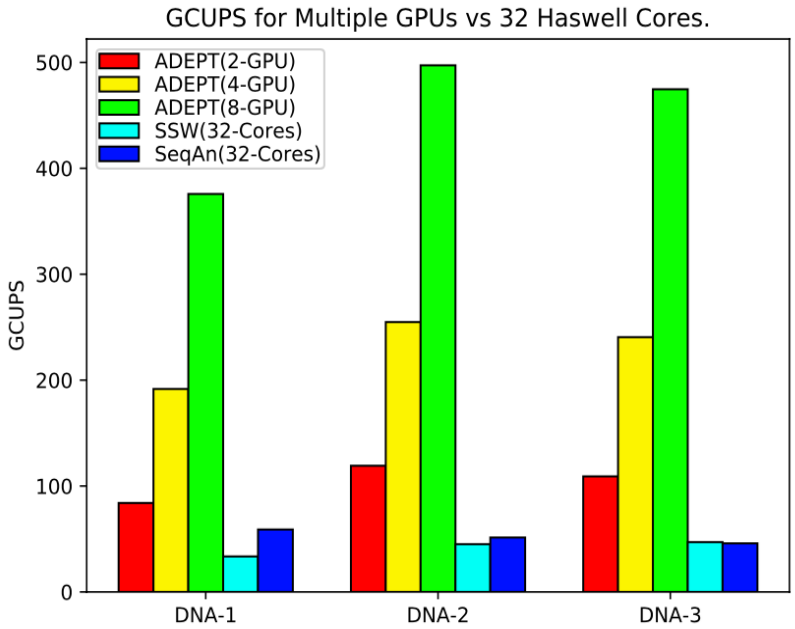
Options to search matrix

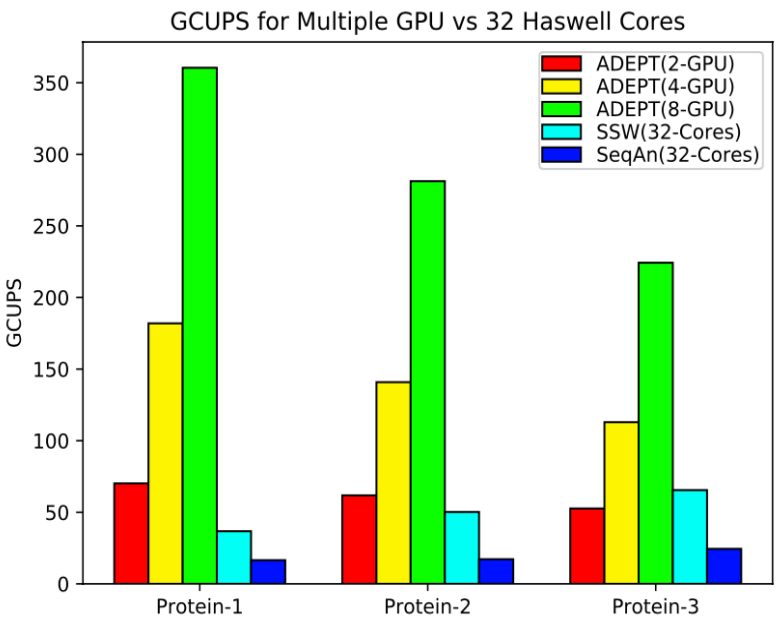
- Full search (Smith-Waterman)
- Banded (only search near diagonal)
- X-Drop stop poor searches early

Many variations

ADEPT: Batch Alignment on GPUs

GCUPS = Giga-Cell Updates per second



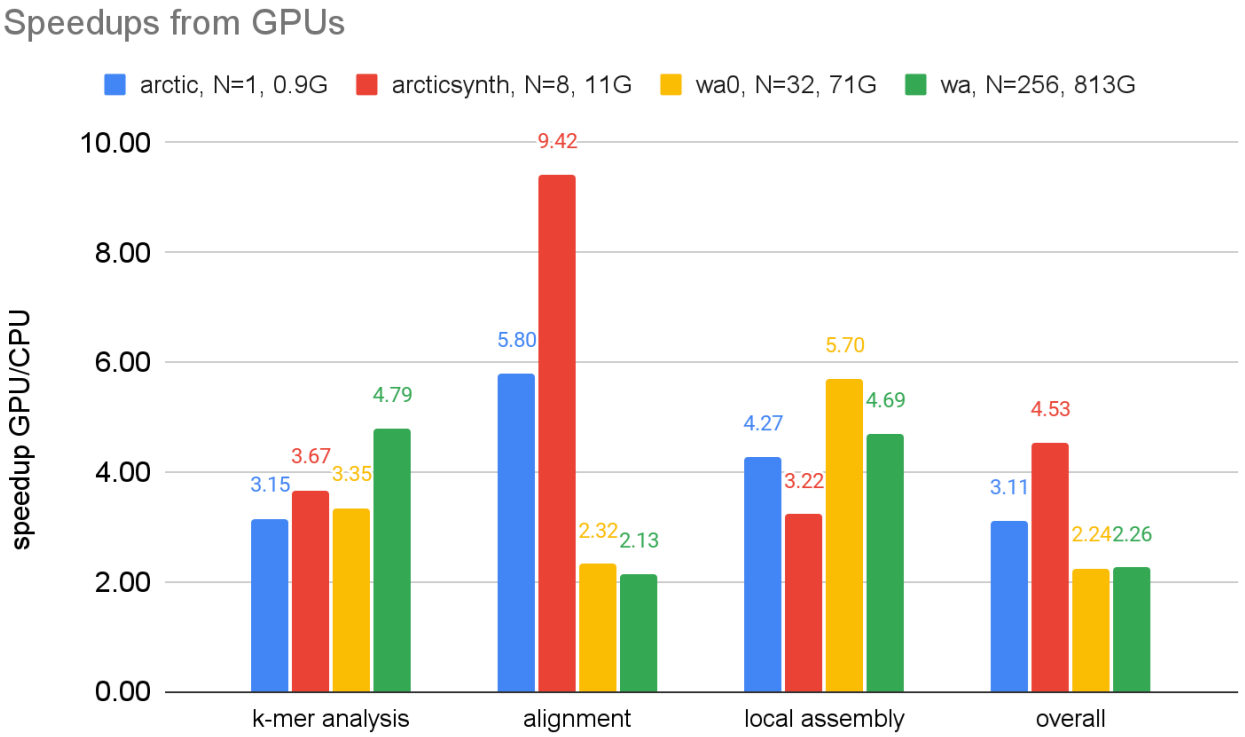


Adept is designed for relatively short, low-error sequences, both DNA (left) and proteins (right)
SSW and SeqAn are vectorized implementations of Smith-Waterman Algorithm on CPU.

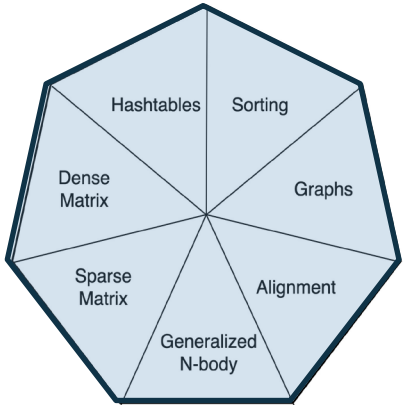
BERKELEY LAB
Lawrence Berkeley National Laboratory

Muaaz G. Awan, Jack Deslippe, Aydin Buluc, Oguz Selvitopi, Steven Hofmeyr, Leonid Oliker & Katherine Yelick

GPU Optimizations



GPU optimizations are complex (hash tables, graphs, etc.)



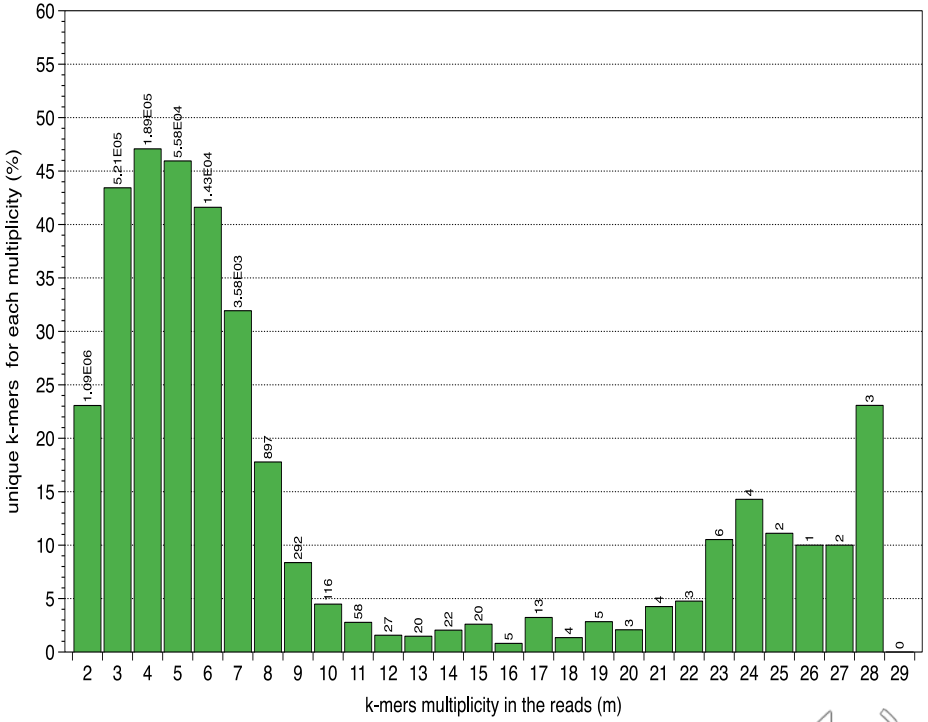
Generalized N-Body

diBELLA: Towards a Long Read Assembler

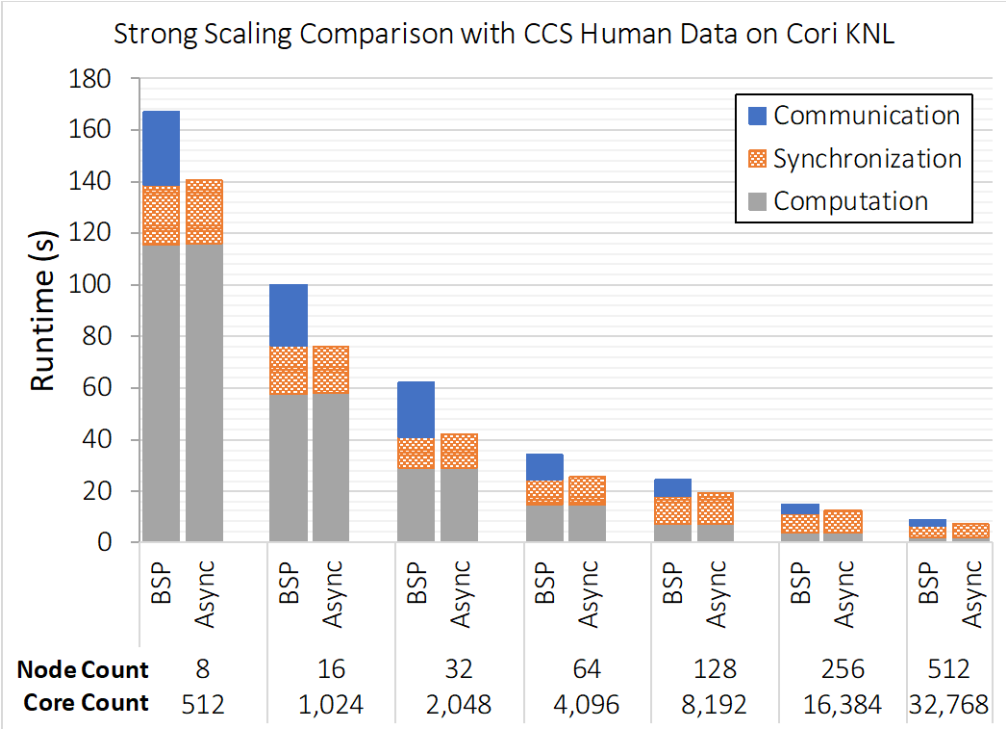
Long reads (PacBio, etc.)

- **Longer alignments**
- **More compute-intensive**
- **More GPU friendly**

Only align pairs of reads that have a common k-mer



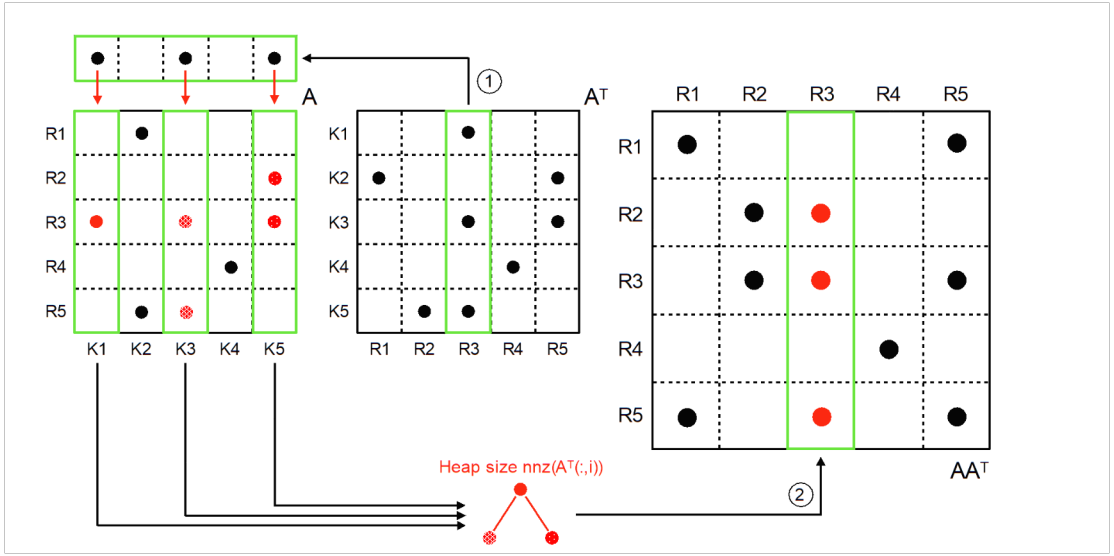
Bulk-Synchronous vs 1-sided Asynchronous



Asynchronous communication hides latency and uses less memory in general

Set Alignment is a Sparse All-to-All

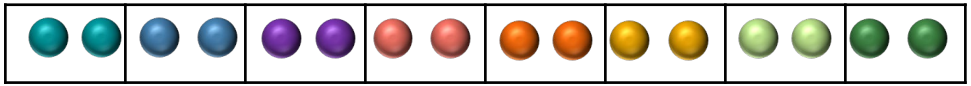
Run expensive alignment on all pairs with a common k-mer



Avoid Communication, Maximize Parallelism

Compute on all pairs of particles or strings, or…

Obvious solution

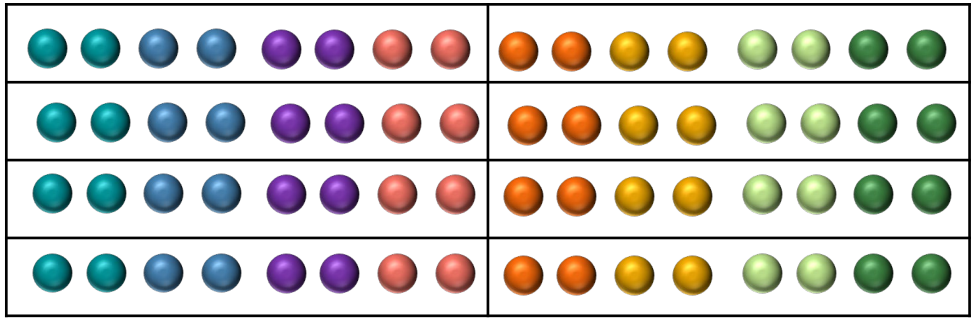


16 particles on 8 processors
Pass all particles around (p steps)

Decreases

- #messages by factor $\mathbf{c^2}$
- #volume sent by factor **c**

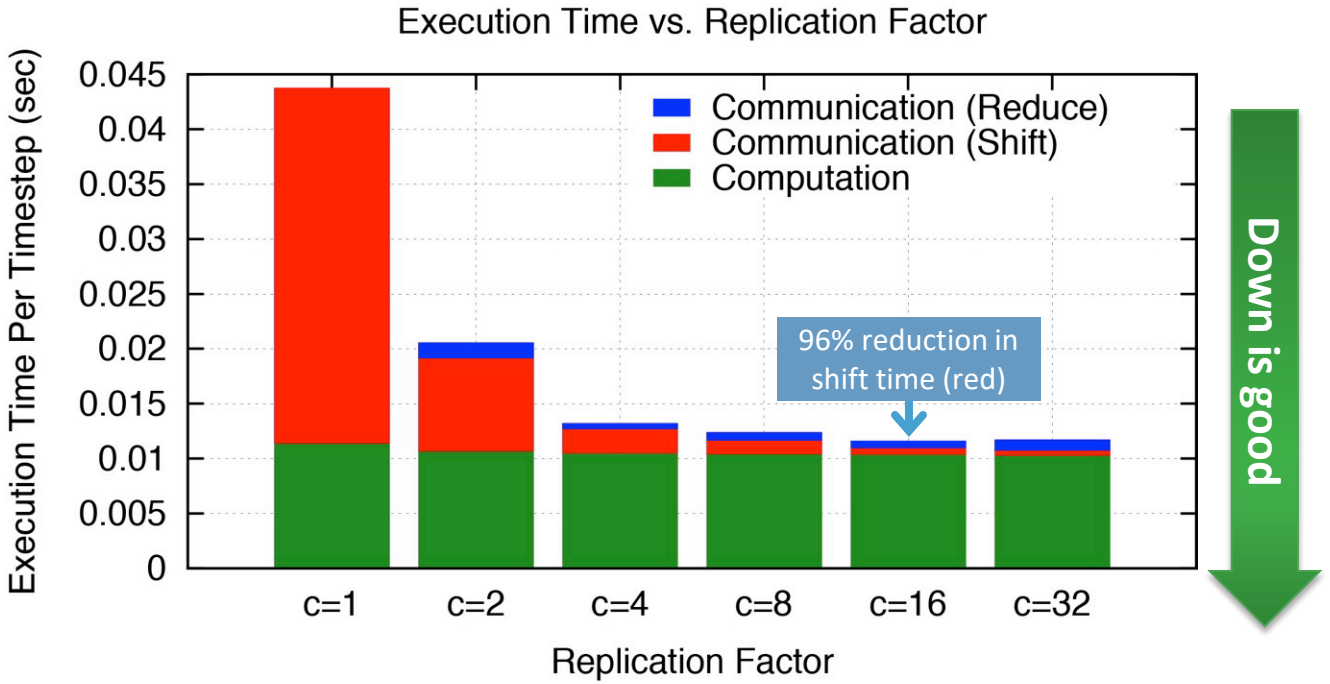
Better solution



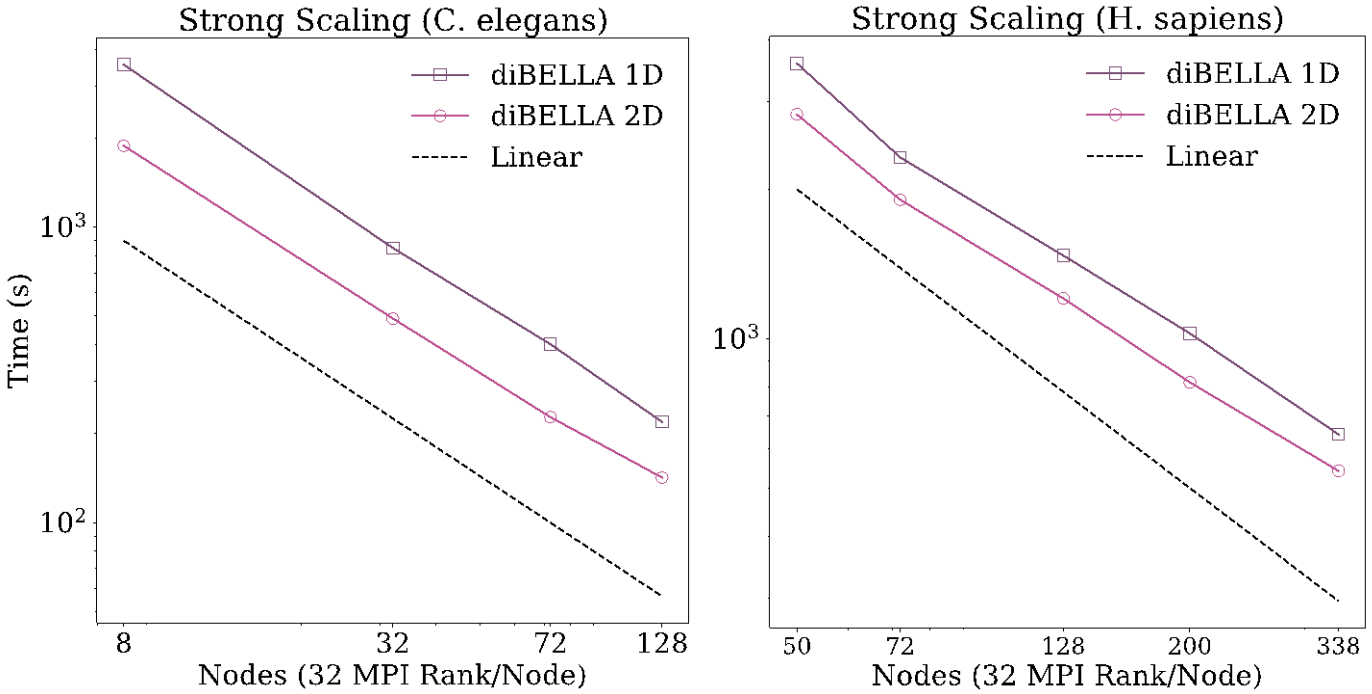
c = 4 copies of particles
8 particles on each

Less Communication..

Cray XE6; n=24K particles, p=6K cores



1D vs 2D Algorithm on DNA “overlap”

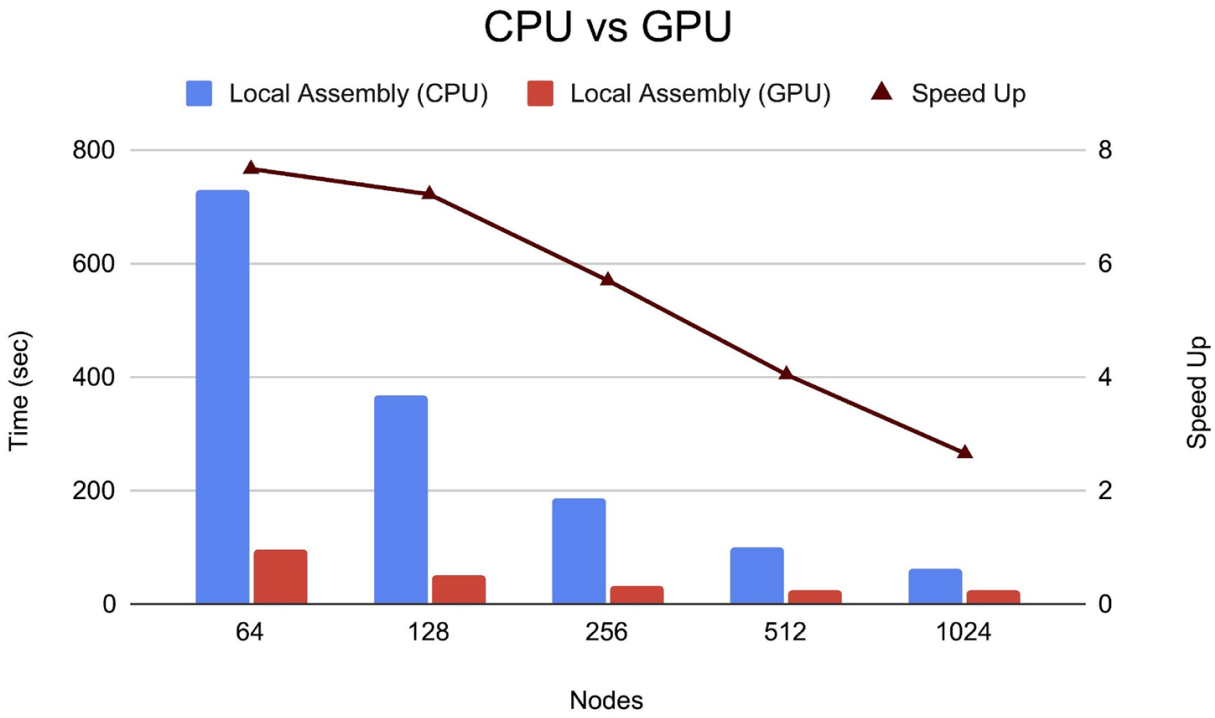


Lower is Better

Seven Take-Aways

- **Applications**
  - More data, more compute → more insights
  - ~7 motifs of genomic analysis (analytics)
- **Programming models**
  - Use of PGAS for irregular, fine-grained problems
  - Can still map GPUs
- **Algorithms**
  - Hide latency or aggregating messages (can trade off)
  - Use memory to reduce data (volume)
  - Use all the wires all the time

Local Assembly on Summit



- Speedup of 7x on 64 Summit nodes.
- Lower as expected as machine scales (strong scaling)

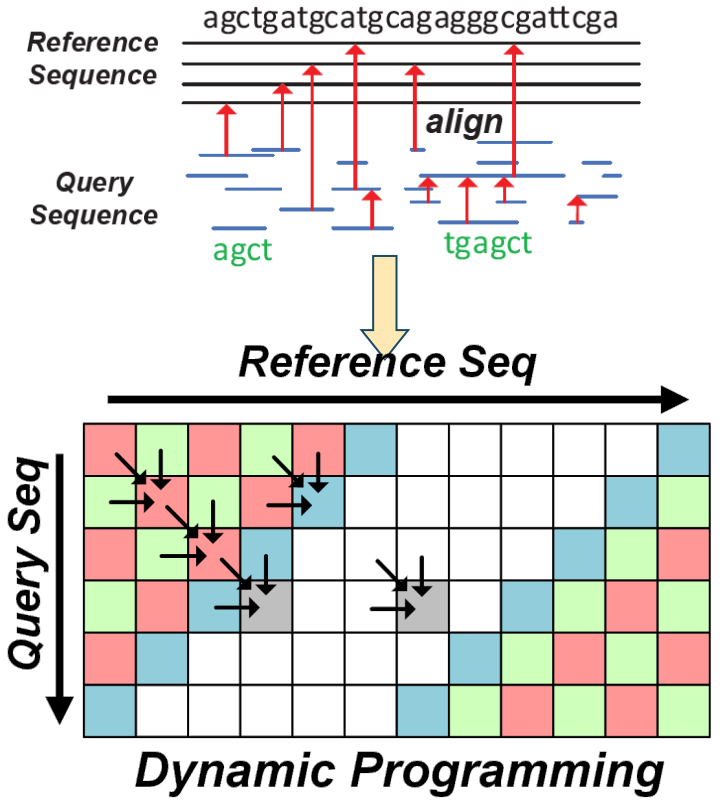
Sequence Alignment

Dynamic Programming

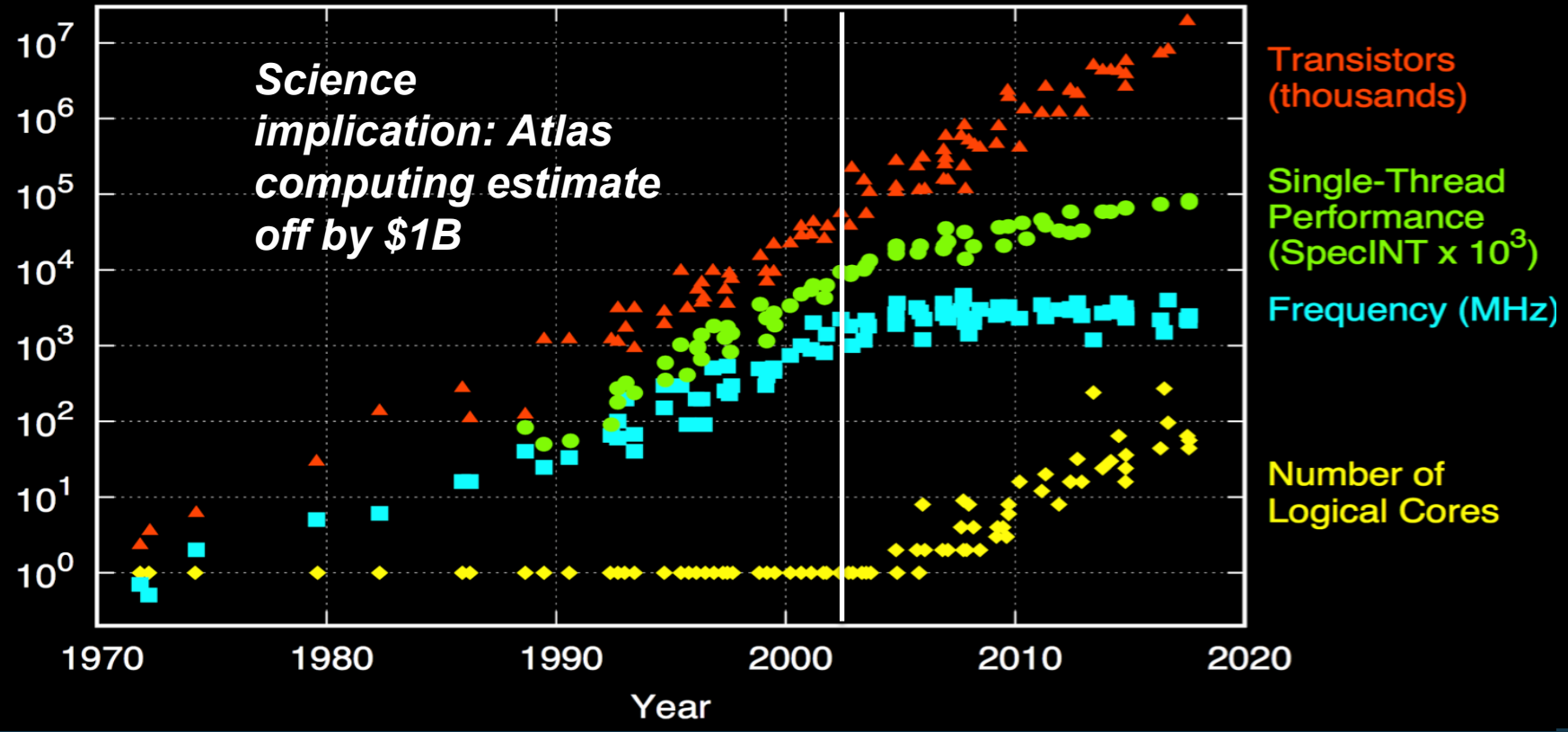
- Low Arithmetic Intensity
- Irregular memory access patterns
- Complex parallelism
- Integer only computations

ADEPT Sequencing Library

- Also working on code generator



Dennard Scaling is Dead; Moore's Law Will Follow



M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, C. Batten, and K. Rupp

Exascale Architecture Plans (2008)

Exascale Architecture Plans (2021)

US DOE Office of Science Systems

Exascale
HPE AMD+AMD

Exascale
HPE Intel+Intel

Pre-exascale
HPE AMD+NVIDIA

Accelerators

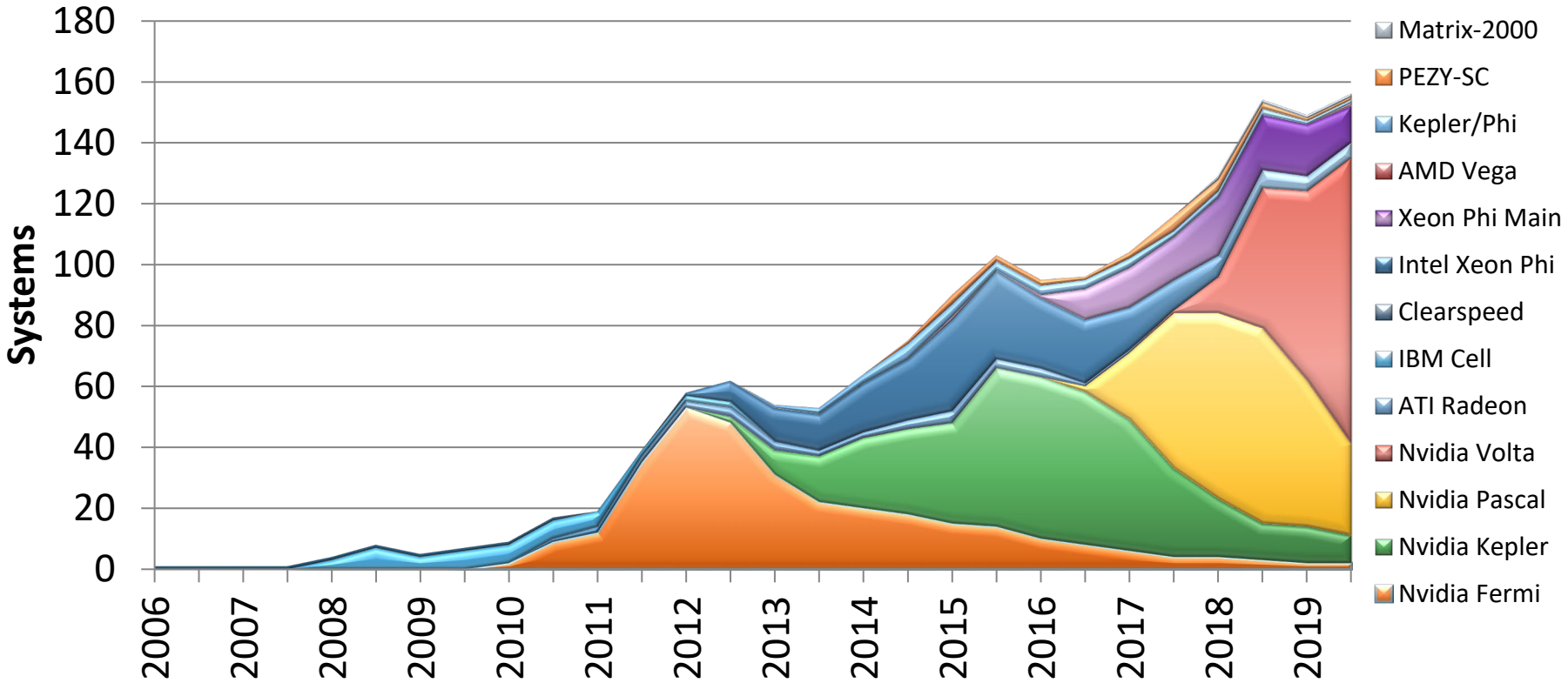
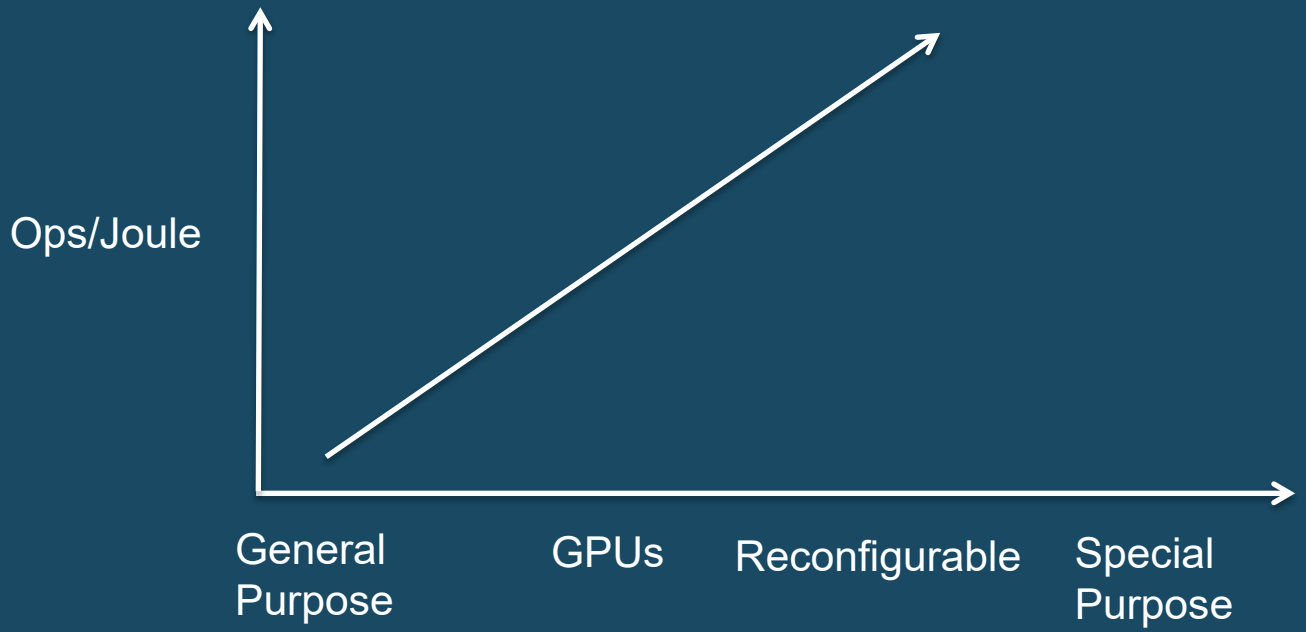
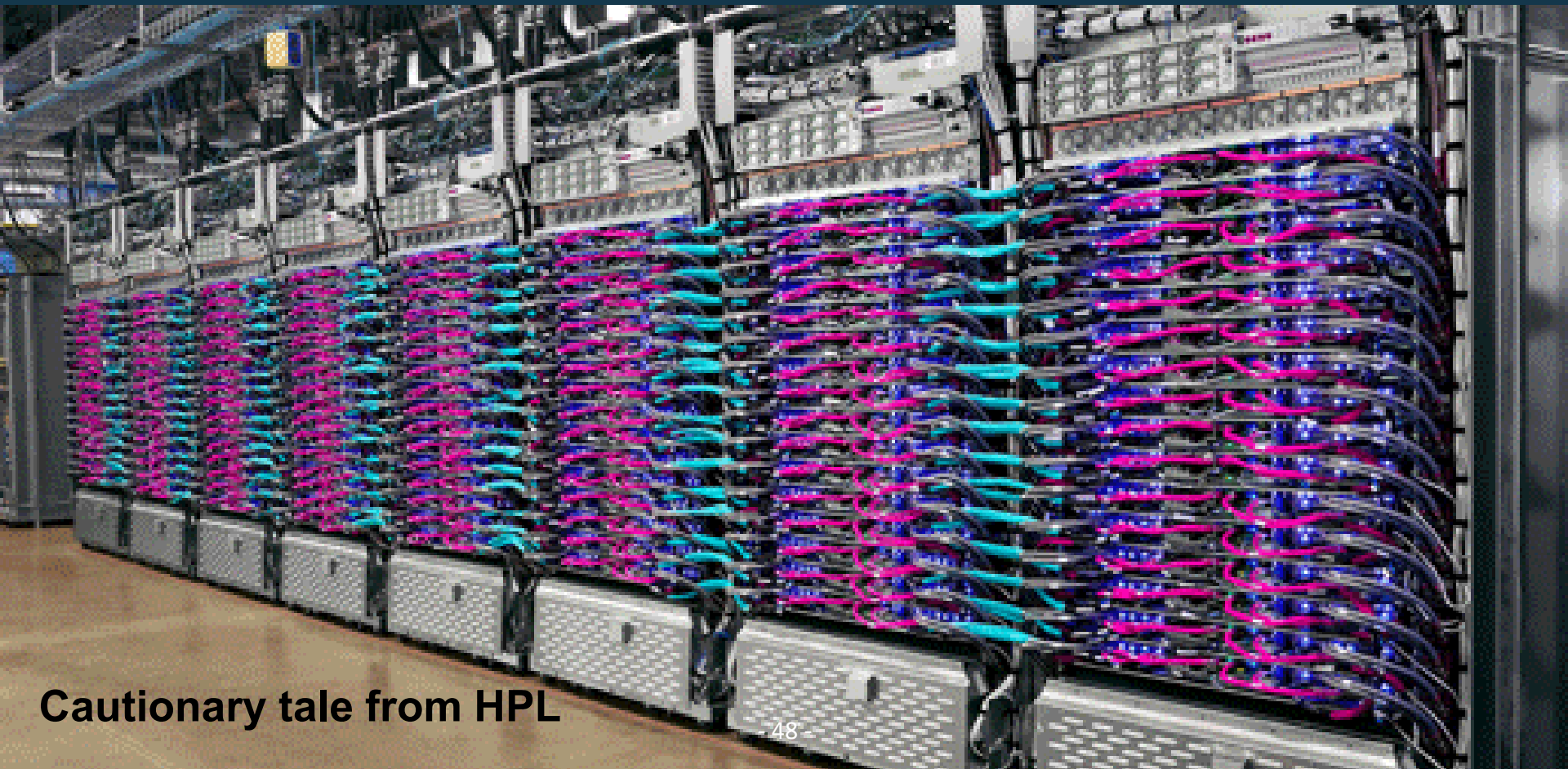


Image: http://slideplayer.com/slide/7541288/

Specialization: End Game for Moore's Law



Is deep learning the only application?



Cautionary tale from HPL

Data Movement is Expensive

Hierarchical energy costs.

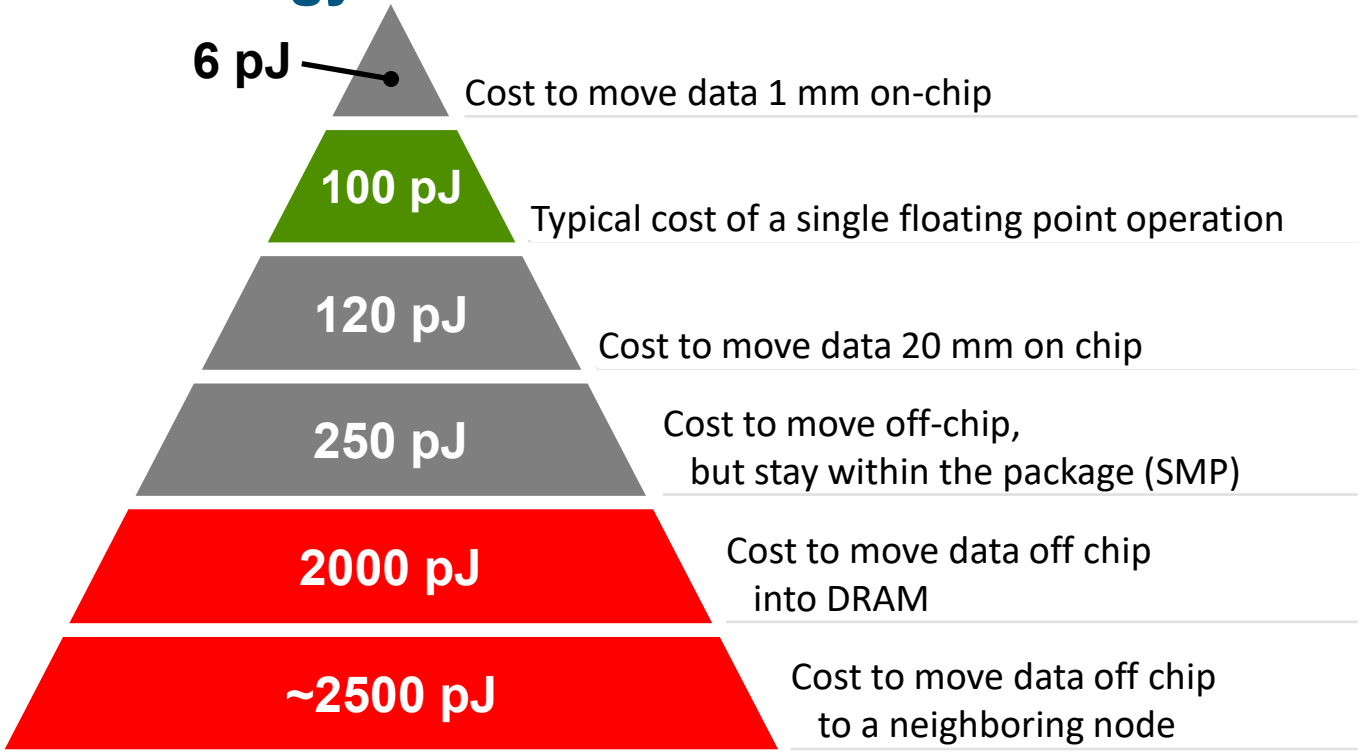
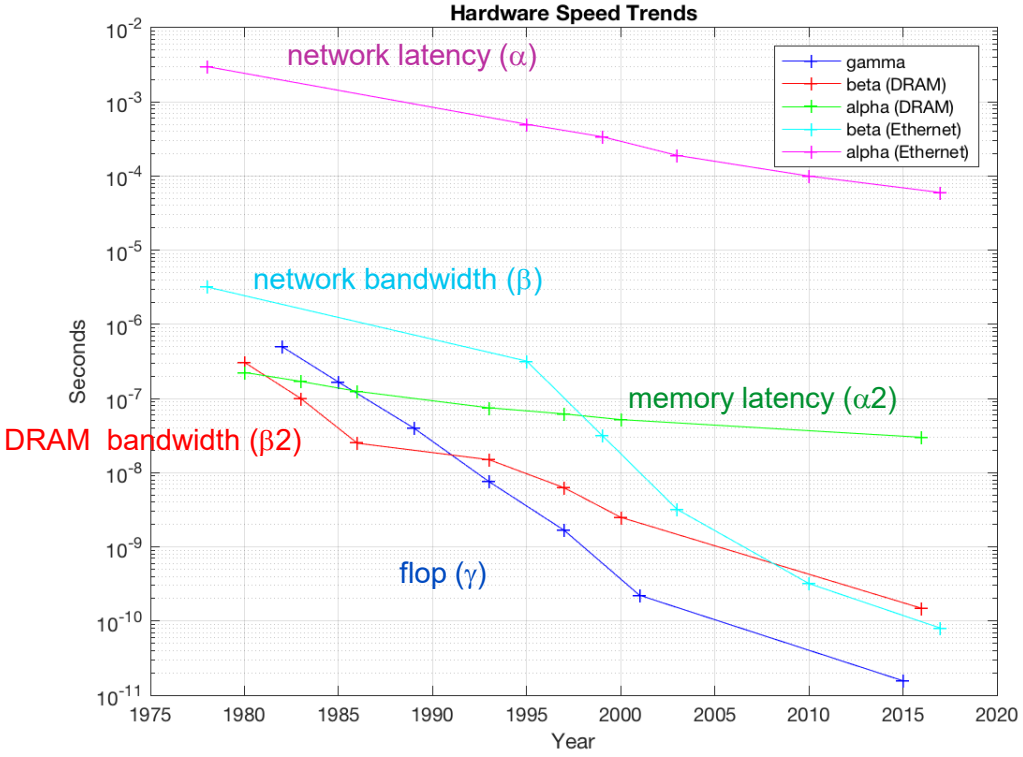


Image: http://slideplayer.com/slide/7541288/

Communication Dominates: Dennard was too good



Time =

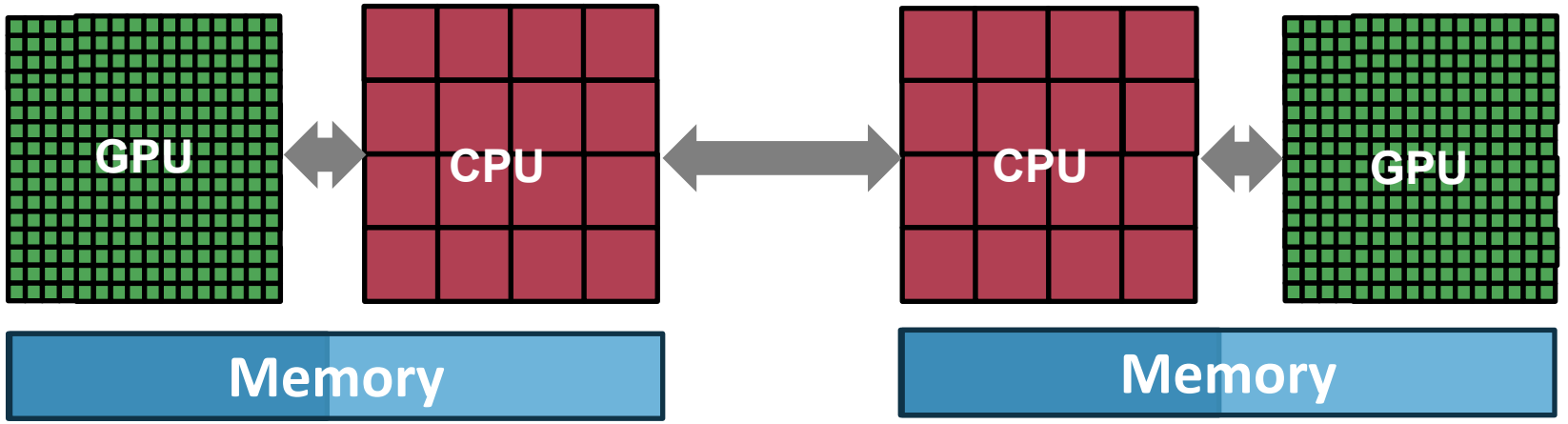
flops * γ +

message * α +
bytes comm * β +

diff memory locs * α2 +
memory words * β2

Data from Hennessy / Patterson, Graph from Demmel

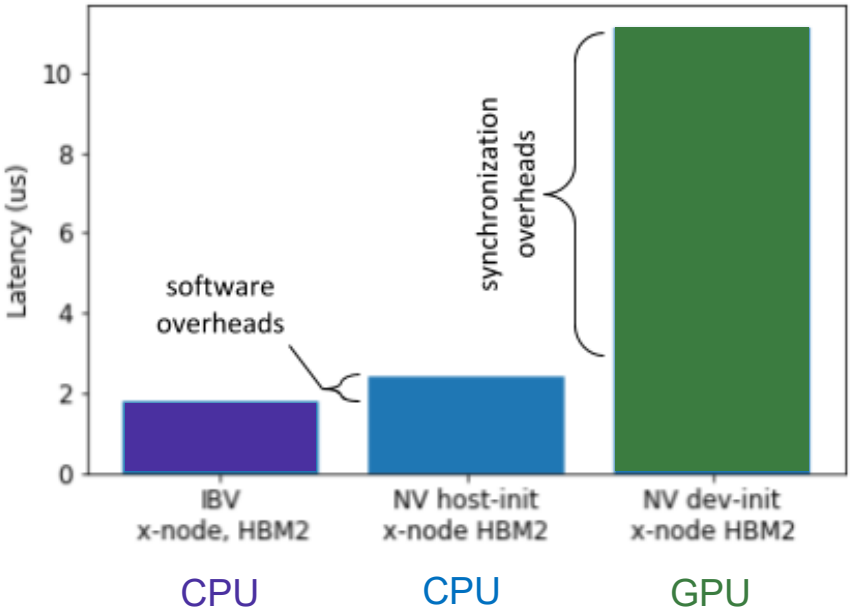
Specialization, Yes Accelerators, No!

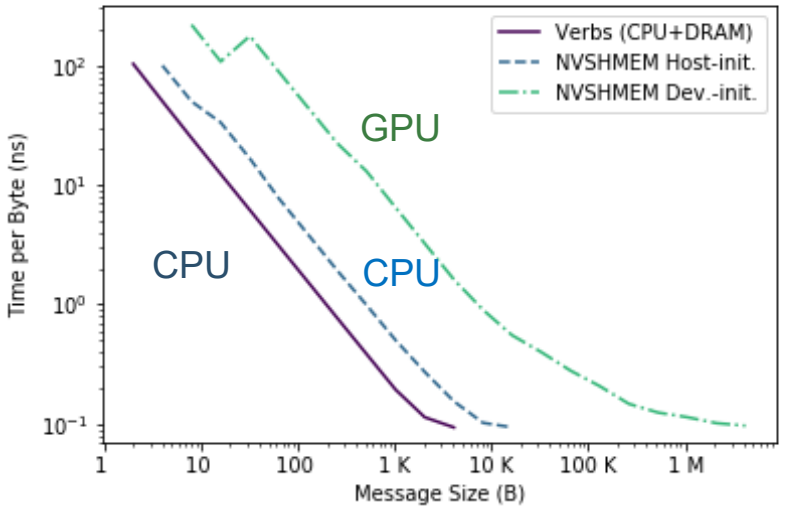


More cores	More data parallelism	Narrow data types	More memory spaces	CPUs in control	CPUs communicate

Put Accelerators in Charge of Communication

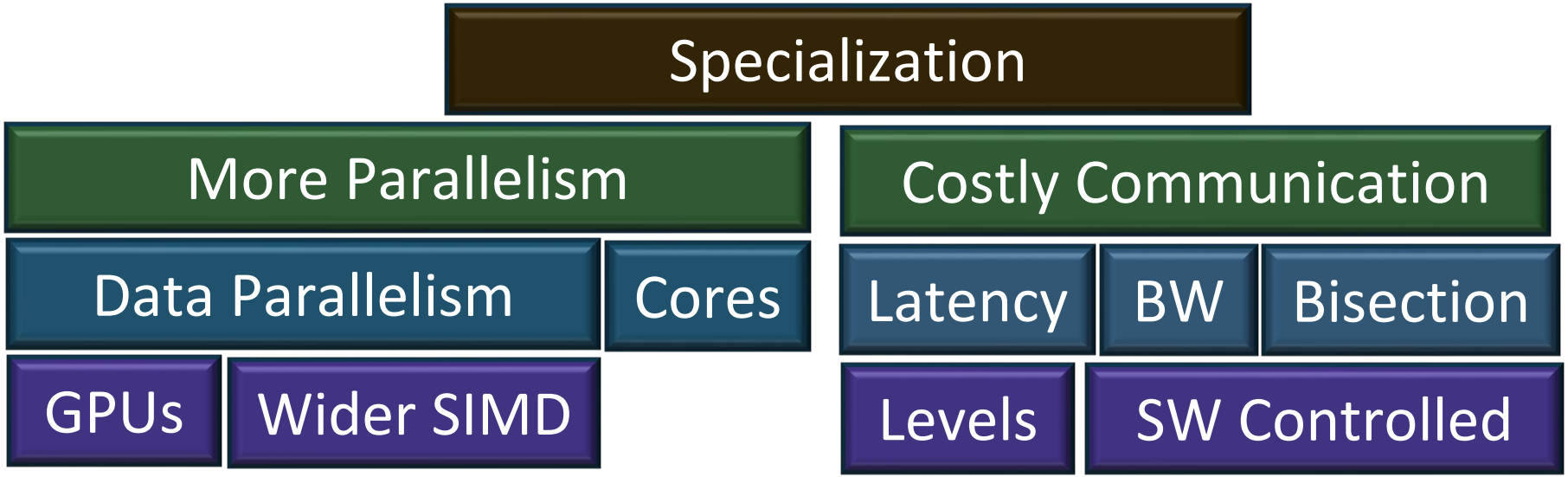
Architecture and software are not yet structured for accelerated-initiated communication (Summit with NVLink between Power9 CPUs and NVIDIA GPUs)





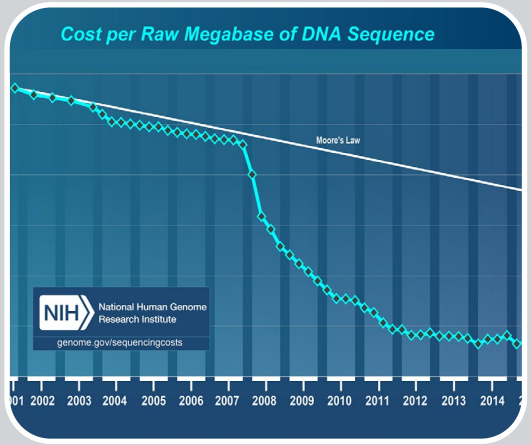
Taylor Groves et al

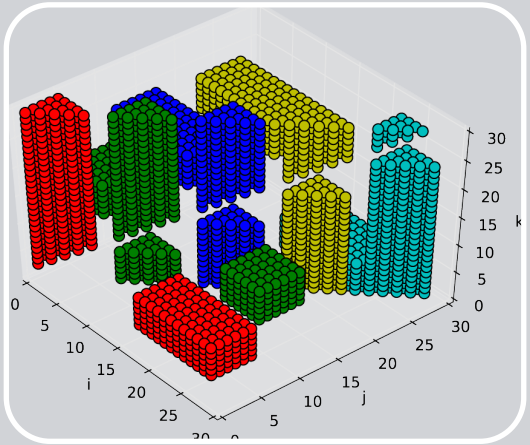
Hardware Trends



Tradeoffs in integration (faster communication) vs scale (amount of fast memory) and flexibility

Genomic Analysis at Scale



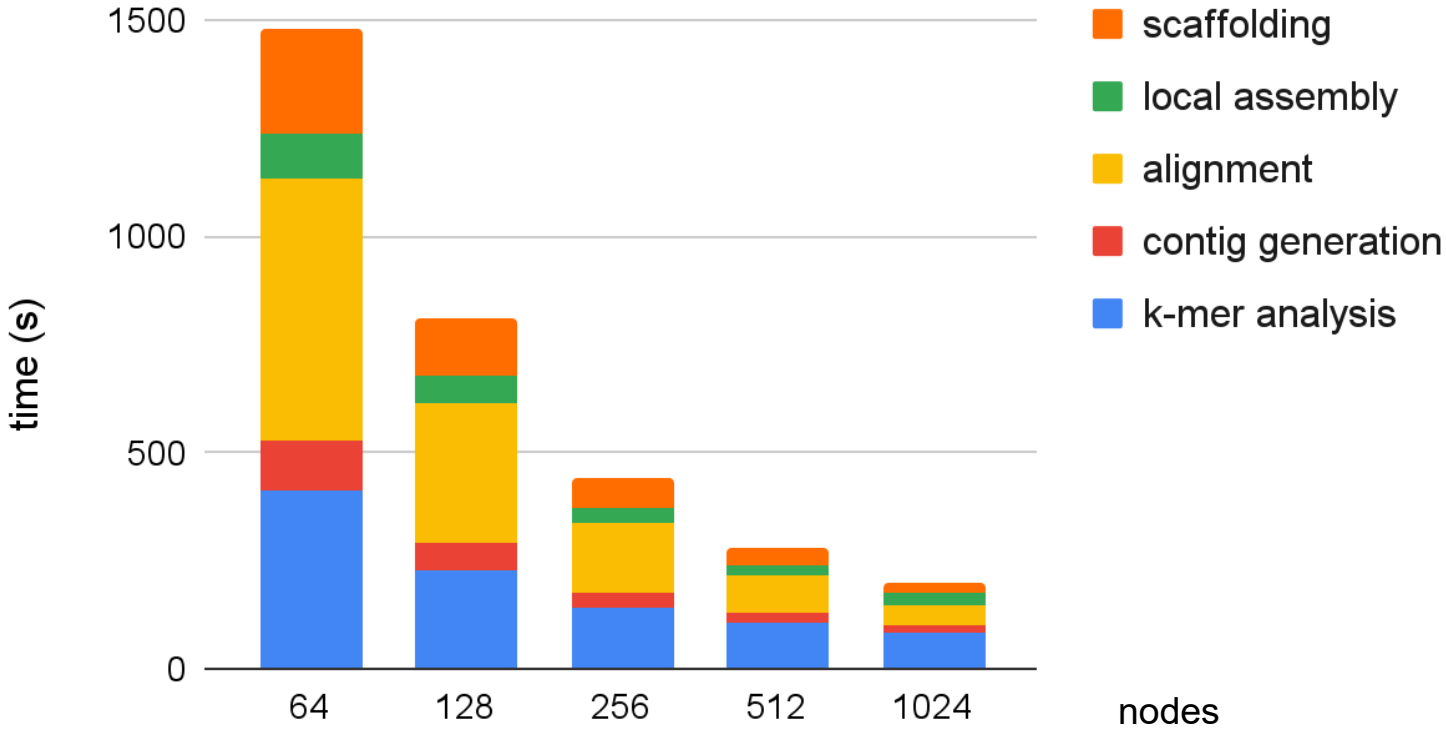


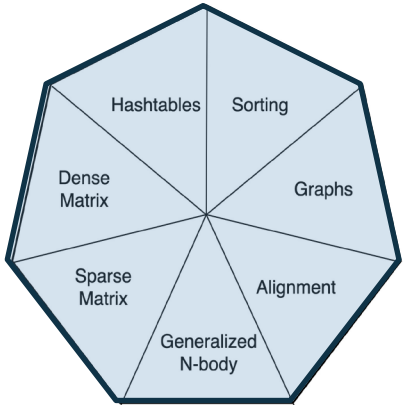
Big Data

Big Machines

Scalable Algorithms

Strong Scaling on Summit



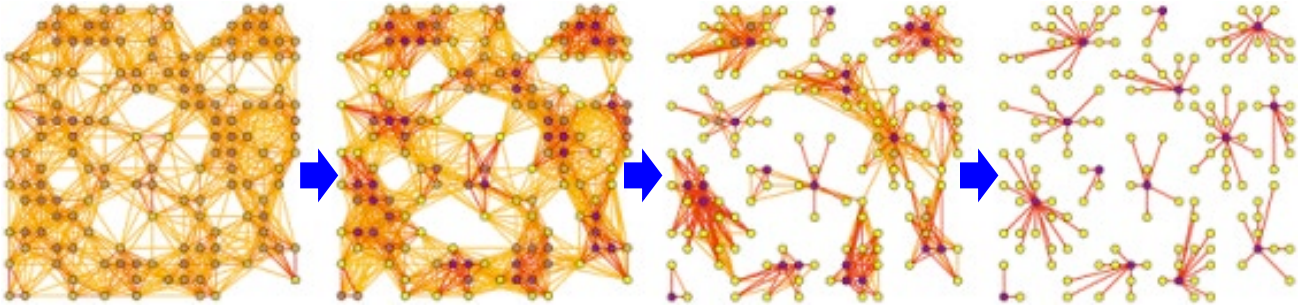


Sparse Matrices (unsupervised learning)

Protein Clustering with Sparse Matrices

Input: Adjacency matrix A (sparse)

Image source: http://micans.org/mcl/

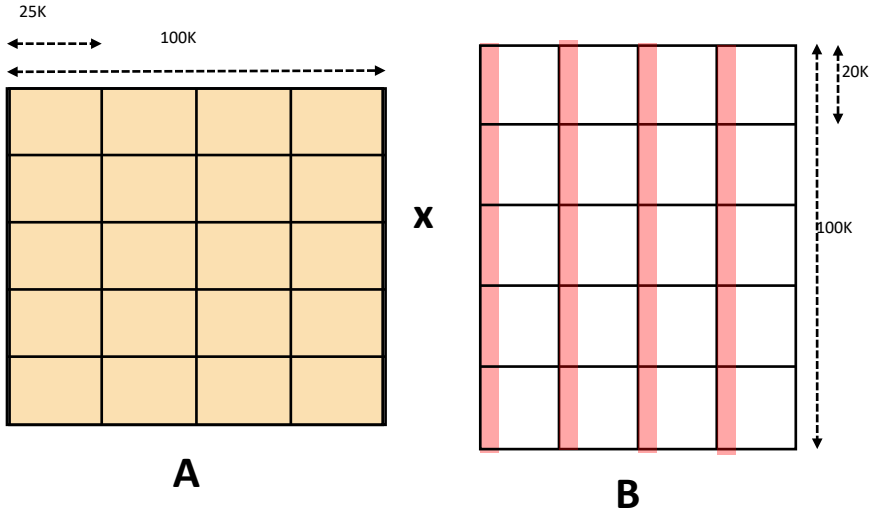


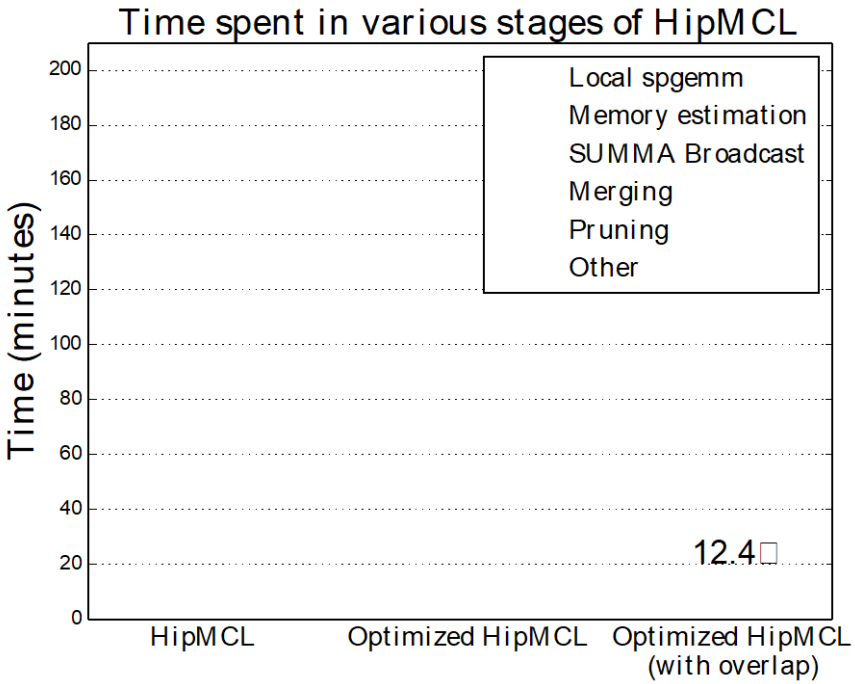
- **Similarity Matrix**: “Many-to-many” protein alignment
- **Expansion**: Square matrix, pruning small entries, dense columns
- **Inflation**: element-wise powers

PASTIS + HipMCL

Oguz Selvitopi; Md Taufique Hussain; Ariful Azad; Aydın Buluç

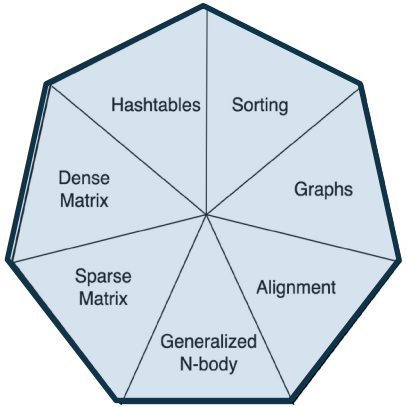
Sparse Matrix Algorithms





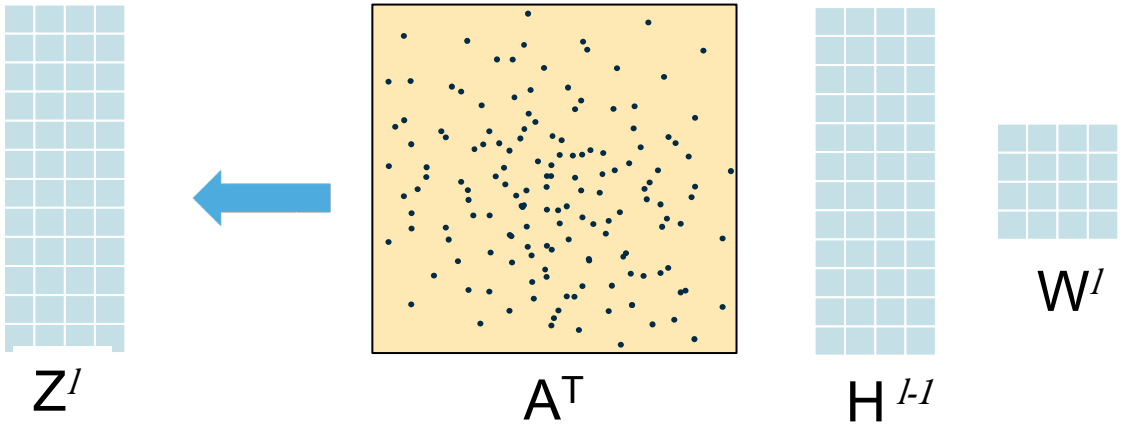
Distributed memory enabled new science

12.4× faster with GPUs!



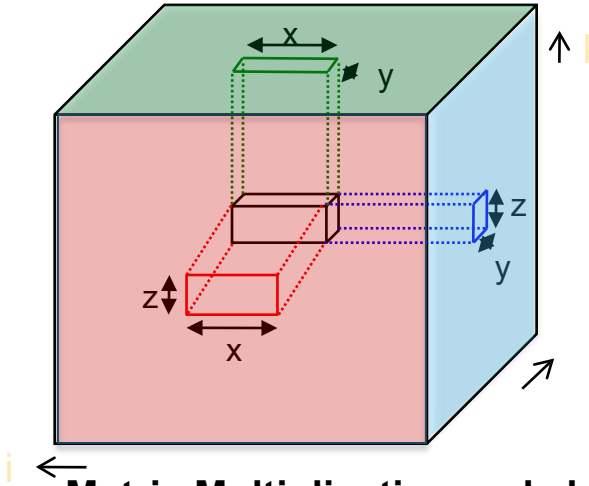
Sparse and Dense Matrices (supervised learning)

Bottleneck in GNN Training



- $A^T H^{l-1}$ sparse-dense matmul (SpMM)
- $(A^T H^{l-1}) W^l$ dense-dense matmul (DGEMM)
- **SpMM is the bottleneck, not DGEMM!**

Communication-Avoiding Matrix Multiply

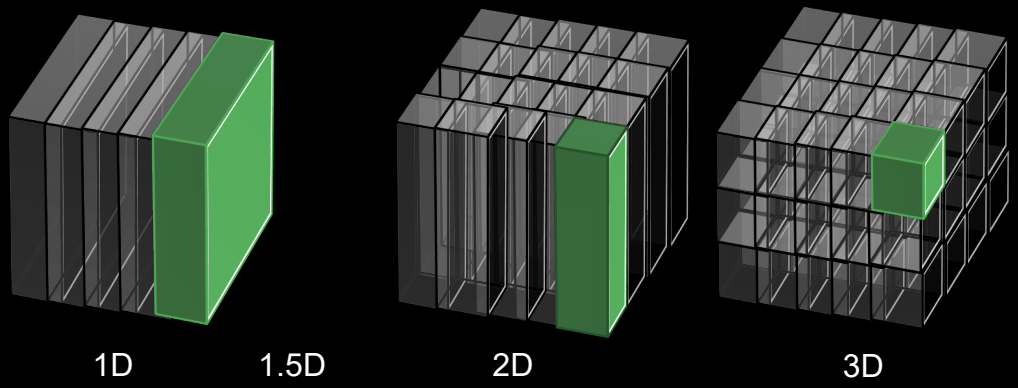


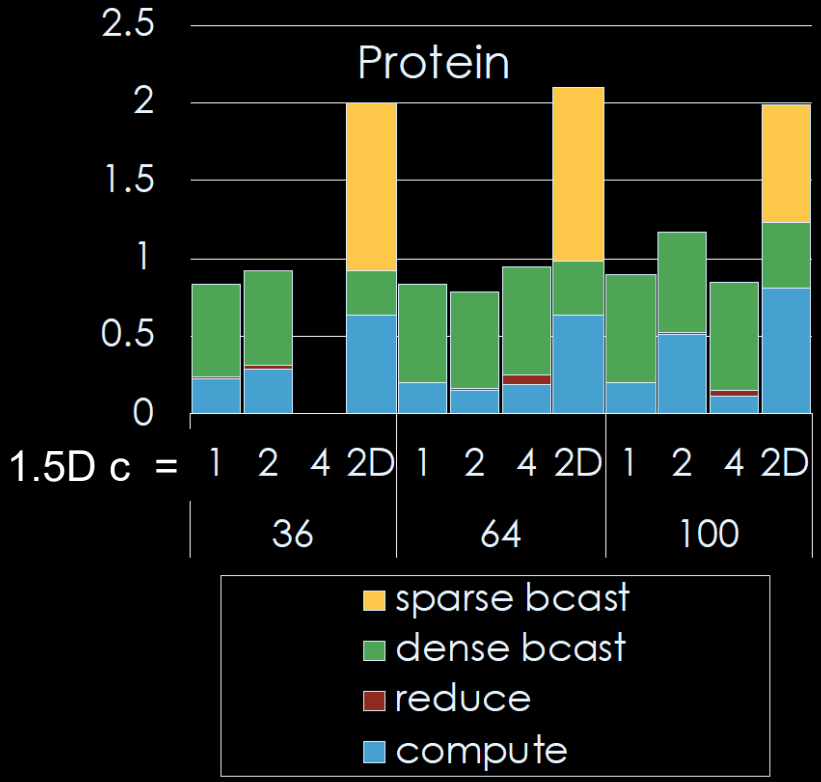
- 2D algorithm: never chop k dim
- 3D: Assume + is associative; chop k, which is → replication of C matrix

Matrix Multiplication code has a 3D iteration space
Each point in the space is a constant computation (*/+)

```
for i
  for j
    for k
          C[i,j] …  A[i,k] …   B[k,j]  …
```

Avoiding Communication in GNNs





Tripathy, Yelick, Buluc, Reducing Communication in Graph Neural Network Training, SC'20

Machine Learning Mapping to Linear Algebra