Genomic Analysis at Scale: Mapping Irregular Computations to Advanced Architectures

Kathy Yelick
Robert S. Pepper Distinguished Professor of EECS
Associate Dean for Research
Division of Computing, Data Science, and Society
UC Berkeley
Senior Advisor on Computing
Lawrence Berkeley National Laboratory
Genomic Analysis at Scale

- Big Data
- Big Machines
- Scalable Algorithms
Understanding the microbiome
Understanding the microbiome

Who, what, why, how?
NMDC launched in 2019 by DOE to collect, analyze, and serve microbial data.

Courtesy of Emiley A. Eloe-Fadrosh, JGI, LBNL
Metagenome Complexity

Species per metagenome

Acid mine

Cow rumen

Soil
Microbiome analysis: metagenome

- Reads
- Microbial community
- Contigs
- Species Bins
- Proteins

Pathways

function
Big Science Questions

What happens to microbes after a wildfire? (1.5TB)

What are the microbial dynamics of soil carbon cycling? (3.3 TB)

What at the seasonal fluctuations in a wetland mangrove? (1.6 TB)

How do microbes affect disease and growth of switchgrass for biofuels (4TB)

Combine genomics with isotope tracing methods for improved functional understanding (8TB)
Complex metagenomes require terascale data

Coverage analysis done by Kelly Cobaugh using Nonpareil 3 (Rodriguez-R, et al. mSystems 2018)
Multiassembly: assembling many samples separately

Coassembly: assembling many samples together
Clustering Huge Protein Datasets

• Protein families via clustering
• Functional diversity
  – Oceans vs human microbiome
• New genes and proteins, e.g.
  – Novel CRISPR/Cas genes
  – Gene clusters encoding antibiotics

Moore’s Law

It’s hard to think exponentially

But it’s also hard to stop
Dennard Scaling is Dead; Moore’s Law Will Follow

Science implication: Atlas computing estimate off by $1B
Exascale Architecture Plans (2008)

- 100x Faster clocks
- 100x more cores
- Accelerators (GPUs)
Exascale Architecture Plans (2021)

Pre-exascale
HPE AMD+NVIDIA

Exascale
HPE AMD+AMD

Exascale
HPE Intel+Intel

US DOE Office of Science Systems
Wider SIMD

SIMD extensions on top of x86/x87

64b SIMD

128b SIMD

256b SIMD

512b SIMD

Top 500
#1 Fugaku
ARM w/ 512b

Image from Dezso Sima presentation
Specialization: End Game for Moore’s Law

Ops/Joule

General Purpose  GPUs  Reconfigurable  Special Purpose
Is deep learning the only application?

Cautionary tale from HPL
Specialization, Yes

Accelerators, No!

- More cores
- More data parallelism
- Narrow data types
- More memory spaces
- CPUs in control
- CPUs communicate
Data Movement is Expensive

Hierarchical energy costs.

- 6 pJ: Cost to move data 1 mm on-chip
- 100 pJ: Typical cost of a single floating point operation
- 120 pJ: Cost to move data 20 mm on chip
- 250 pJ: Cost to move off-chip, but stay within the package (SMP)
- 2000 pJ: Cost to move data off chip into DRAM
- ~2500 pJ: Cost to move data off chip to a neighboring node

Image: http://slideplayer.com/slide/7541288/
Communication Dominates: Dennard was too good

\[
\text{Time} = \text{# flops} \times \gamma + \text{# message} \times \alpha + \text{# bytes comm} \times \beta + \text{# diff memory locs} \times \alpha^2 + \text{# memory words} \times \beta^2
\]

Data from Hennessy / Patterson, Graph from Demmel
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

More Parallelism
- Data Parallelism
- Cores
- GPUs
- Wider SIMD

Costly Communication
- Latency
- BW
- Bisection
- Levels
- SW Controlled

Specialization

Tradeoffs in integration (faster communication) vs scale (amount of fast memory) and flexibility
Algorithms and Software
### ExaBiome project overview

**Exascale algorithms & systems for previously intractable problems**

<table>
<thead>
<tr>
<th>Science Need</th>
<th>Metagenome Assembly</th>
<th>Protein Clustering and Annotation</th>
<th>Comparative Metagenome Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find species, genes and relative abundance in microbial communities</td>
<td>Improve understanding of tree of life for microbes; aid in identifying gene function</td>
<td>Track microbiome over time or space, changes in environment, climate, etc.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Computing Technique</th>
<th>Direct tables, alignment, k-mer counts, sparse matrices, ML (clustering, GNNs)</th>
<th>hash tables, alignment, k-mer counts, ML (dimensionality reduction)</th>
</tr>
</thead>
</table>

**Science**

- Need: Find species, genes and relative abundance in microbial communities
- Improve understanding of tree of life for microbes; aid in identifying gene function
- Track microbiome over time or space, changes in environment, climate, etc.

**Computing Technique**

- Metagenome Assembly: hash tables, alignment, k-mer counts, graph walks
- Protein Clustering and Annotation: direct tables, alignment, k-mer counts, sparse matrices, ML (clustering, GNNs)
- Comparative Metagenome Analysis: hash tables, alignment, k-mer counts, ML (dimensionality reduction)

[http://exabiome.org](http://exabiome.org)
Motifs of Genomic Data Analysis

These computational patterns dominate ExaBiome Project experience

Application problems
- Assemble genomes
- Compute distances
- Cluster (contigs, proteins, ...)
- Annotate

Distributed memory platforms open up new approaches and science questions

### Analytics vs. Simulation Kernels:

<table>
<thead>
<tr>
<th>7 Giants of Data</th>
<th>7 Dwarfs of Simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic statistics</td>
<td>Monte Carlo methods</td>
</tr>
<tr>
<td>Generalized N-Body</td>
<td>Particle methods</td>
</tr>
<tr>
<td><strong>Graph-theory</strong></td>
<td>Unstructured meshes</td>
</tr>
<tr>
<td>Linear algebra</td>
<td>Dense Linear Algebra</td>
</tr>
<tr>
<td>Hashing</td>
<td>Sparse Linear Algebra</td>
</tr>
<tr>
<td>Sorting</td>
<td>Spectral methods</td>
</tr>
<tr>
<td>Alignment</td>
<td>Structured Meshes</td>
</tr>
</tbody>
</table>

- NRC Report + our paper
- Phil Colella

Hashing
Common Technique: Analyze K-mers

reads

k-mers (e.g. k=4)
**Distributed Hash Tables of K-Mers**

**Make hash table of k-mers**

Keys are fixed-length strings:
- Count and remove errors (singletons)
- Find strings with matching k-mer
- Connected components
- Use histogram as approximation

**1-sided comm or irregular all-to-all + memory**
Distributed Hashing / Histogramming

Repeat while more to read and/or exchange

$P_0$

Input read partition → load & parse block → Outgoing k-mers & metadata → Alltoallv

Local k-mer set → store → Hash table partition

$P_1$

$P_{N-1}$

Input read partition → load & parse block → Outgoing k-mers & metadata → Alltoallv

Local k-mer set → store → Hash table partition

implicit barrier
K-mer counting: All the Wires All the Time

- Used to be bulk-synchronous MPI
- UPC++ communication is asynchronous and 1-sided
- UPC++ version is faster, avoids barriers, saves memory (one runtime)
- And it’s simpler!

Steve Hofmeyr, Rob Egan, Evangelos Gerganas, leads on MetaHipMer software
**Communication-Avoiding hash table lookup**

**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!
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 Speedup on 64 Summit nodes

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

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)

Read: ACTG ACTG ACTG CTG CGA GTG A

Minimizer: ACTG

- ACTG
- ACTG
- ACTG
- ACTG
- CTG
- TG
- GCTG
- CTG

Minimizer: CTGC

- CTGC
- CTGC
- CTGC
- CTGC
- CTGC
- CTGC

Minimizer: AGTG

- AGTG
- AGTG
- AGTG
- AGTG
- AGTG
- AGTG

Supermer: ACTG ACTG ACTG CTG CGA GTG A

- Supermer: ACTG ACTG ACTG CTG CGA GTG A
- Supermer: CTG ACTG ACTG ACTG ACTG
- Supermer: CTG ACTG ACTG ACTG ACTG

Speedup on 64 Summit nodes

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

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

<table>
<thead>
<tr>
<th></th>
<th>_</th>
<th>G</th>
<th>A</th>
<th>T</th>
<th>C</th>
<th>A</th>
<th>G</th>
<th>C</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>_</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>G</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>A</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>T</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>A</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>G</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

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

Options to search matrix
- Full search (Smith-Waterman)
- Banded (only search near diagonal)
- X-Drop stop search when the score drops by more than X

Variations for local / global alignment, per character penalties, seeding, etc.
ADEPT: Batch Alignment on GPUs

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.
ADEPT: Impact on ExaBiome Applications

Soil assembly SW time: 2.8 node hours on Cori, 0.1 node hours on Summit (hidden behind CPU work)
Generalized N-Body
diBELLA: Towards a Long Read Assembler

Long reads (PacBio, etc.)
- Longer alignments
- More compute-intensive
- More GPU friendly
No need for De Bruijn graph
- Pairwise alignments
- Filtered k-mers
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 $c^2$
• #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

 Execution Time vs. Replication Factor

Down is good

96% reduction in shift time (red)
1D vs 2D Algorithm on DNA “overlap”

Strong Scaling (C. elegans)

- diBELLA 1D
- diBELLA 2D
- Linear

Strong Scaling (H. sapiens)

- diBELLA 1D
- diBELLA 2D
- Linear

Lower is Better

G. Guidi, O. Selvitopi†, M. Ellis, L. Oliker, Y, A. Buluc (IPDP ’21 to appear)
Graphs and Sparse Matrices (unsupervised learning)
Protein Clustering with Sparse Matrices

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

**Input**: Adjacency matrix $A$ (sparse)

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

**PASTIS + HipMCL**
Sparse Matrix Algorithms

Distributed memory enabled new science

12.4× faster with GPUs!
Graphs and Sparse and Dense Matrices (supervised learning)
Bottleneck in GNN Training

- $\mathbf{A}^T \mathbf{H}^{l-1}$ sparse-dense matmul (SpMM)
- $(\mathbf{A}^T \mathbf{H}^{l-1}) \mathbf{W}^l$ dense-dense matmul (DGEMM)
- SpMM is the bottleneck, not DGEMM!
Communication-Avoiding Matrix Multiply

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

Logistic Regression, Support Vector Machines
Dimensionality Reduction (e.g., NMF, CX/CUR, PCA)
Clustering (e.g., MCL, Spectral Clustering)
Graphical Model Structure Learning (e.g., CONCORD)
Deep Learning (Convolutional Neural Nets)

Sparse Matrix-Sparse Vector (SpMSpv)
Sparse Matrix-Dense Vector (SpMV)
Sparse Matrix Times Multiple Dense Vectors (SpMM)
Sparse - Sparse Matrix Product (SpGEMM)
Dense Matrix Vector (BLAS2)
Sparse - Dense Matrix Product (SpDM3)
Dense Matrix Matrix (BLAS3)

Increasing arithmetic intensity

Aydin Buluc, Sang Oh, John Gilbert, Kathy Yelick
Take-Aways

• **Applications**
  — More data, more computing can reveal new insights
  — Genomics problems dominated by ~7 motifs

• **Architectures**
  — Specialization and data parallelism will be increasingly important
  — Communication will (still) dominate
  — Need better integration, lower overheads mechanism

• **Algorithms**
  — Irregular, fine-grained problems
  — Can map to distributed memory and data parallelism
  — Avoid communication:
    • Hide latency or aggregating messages (can trade off)
    • Reduce bandwidth (volume), and
    • Use all the wires all the time