Electrical & Computer ENGINEERING

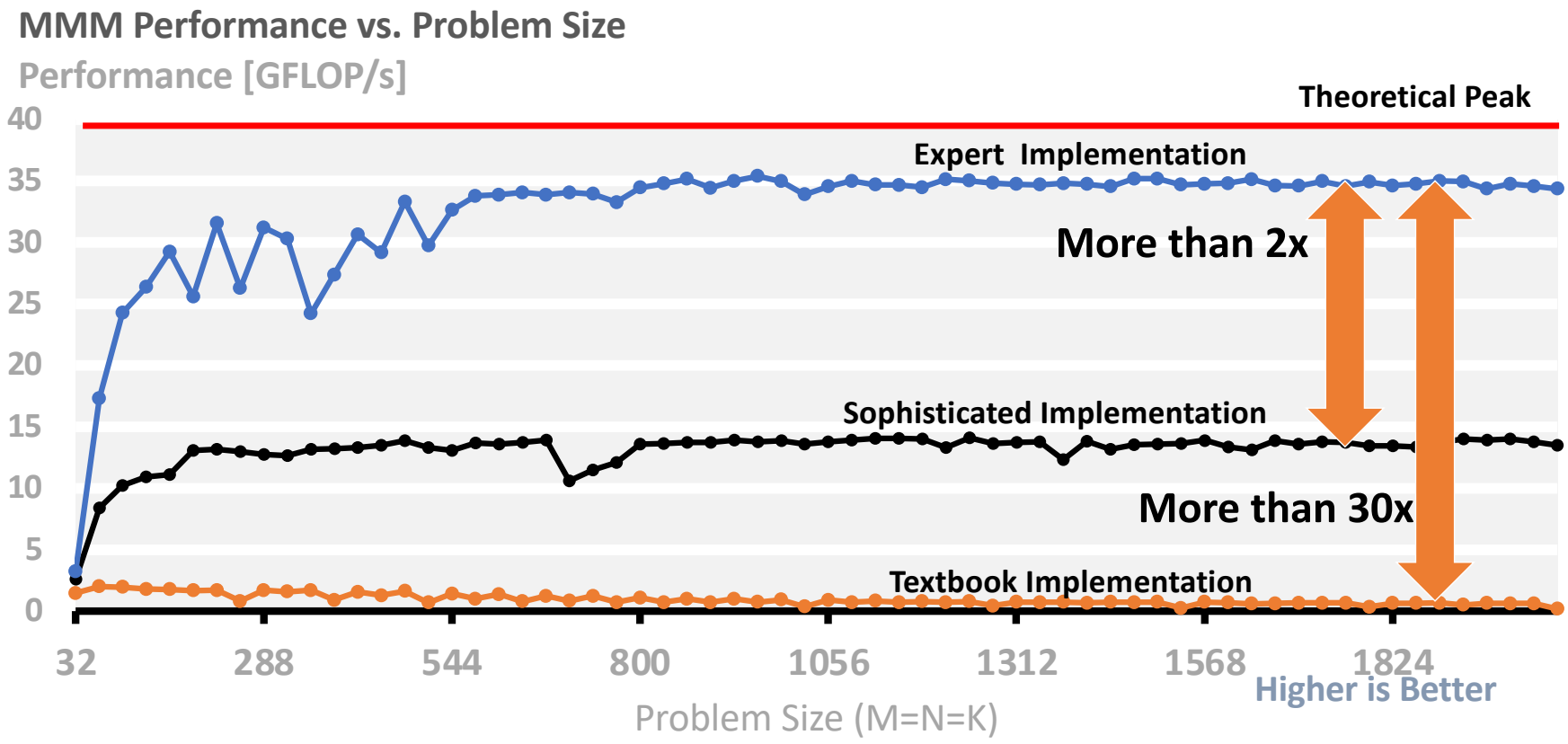
Designing for Portable, Efficient and Explainable Performance

Tze Meng Low
Assistant Research Professor
Carnegie Mellon University

10 May 2022

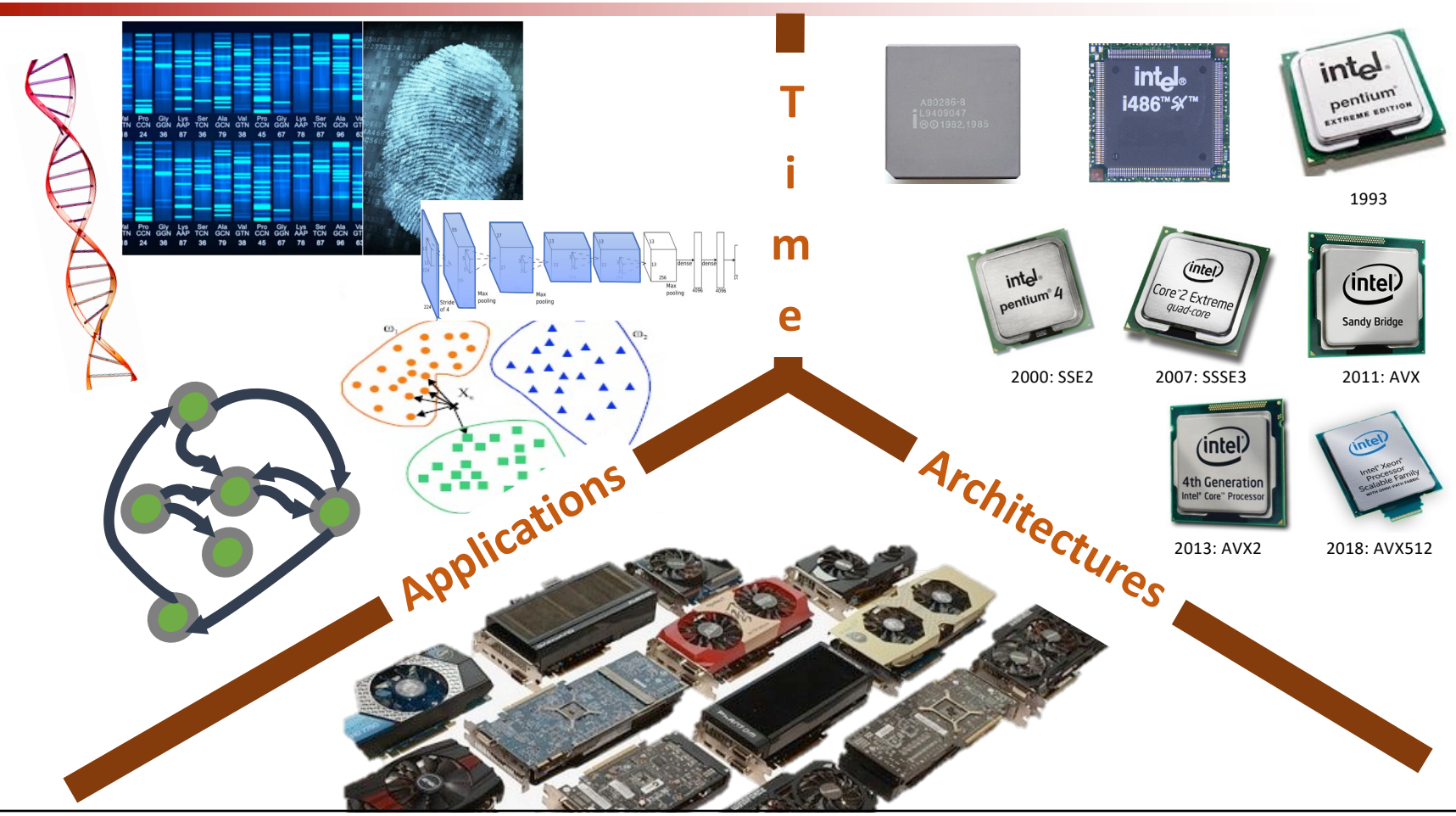
Electrical & Computer ENGINEERING

Writing for Efficient Code is Hard

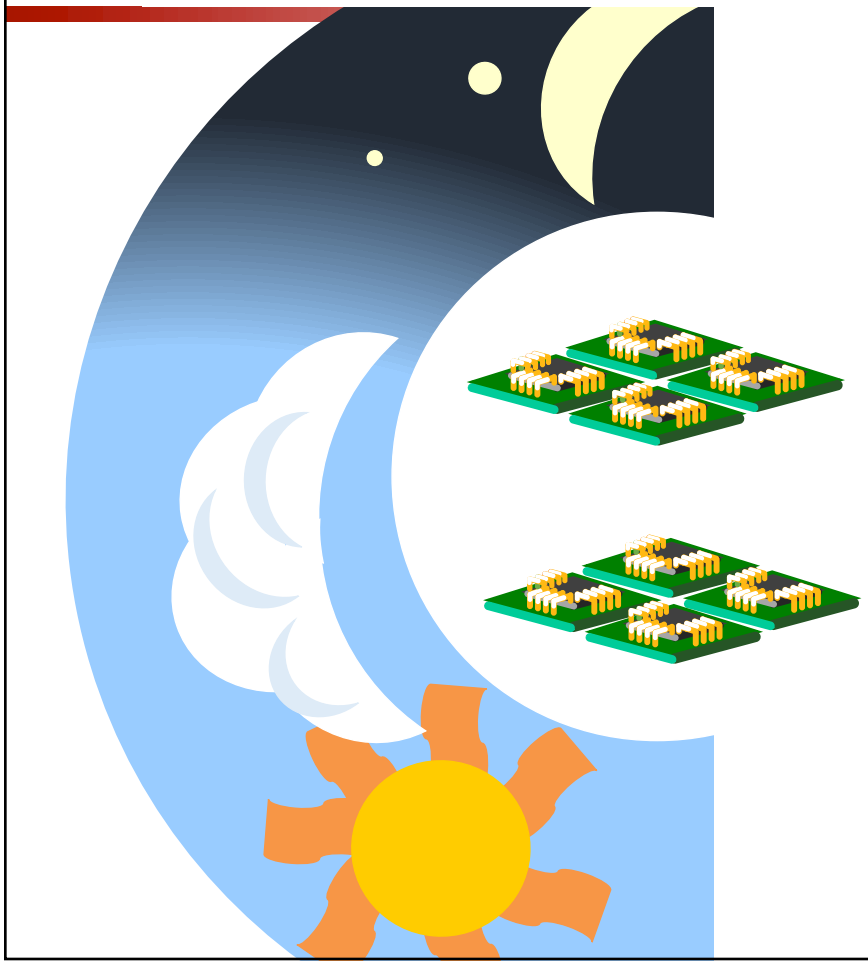


Electrical & Computer ENGINEERING

Portable performance is more complex!



Performance Portability is Necessary



- Manual heroic optimization effort is not scalable
- Repeated optimization with changes
  - Computing platform
  - Different applications
  - Different ML networks
- Blackbox tools and libraries provide little insights into why an implementation is bad/good

Key Questions

How do we capture knowledge about performance?

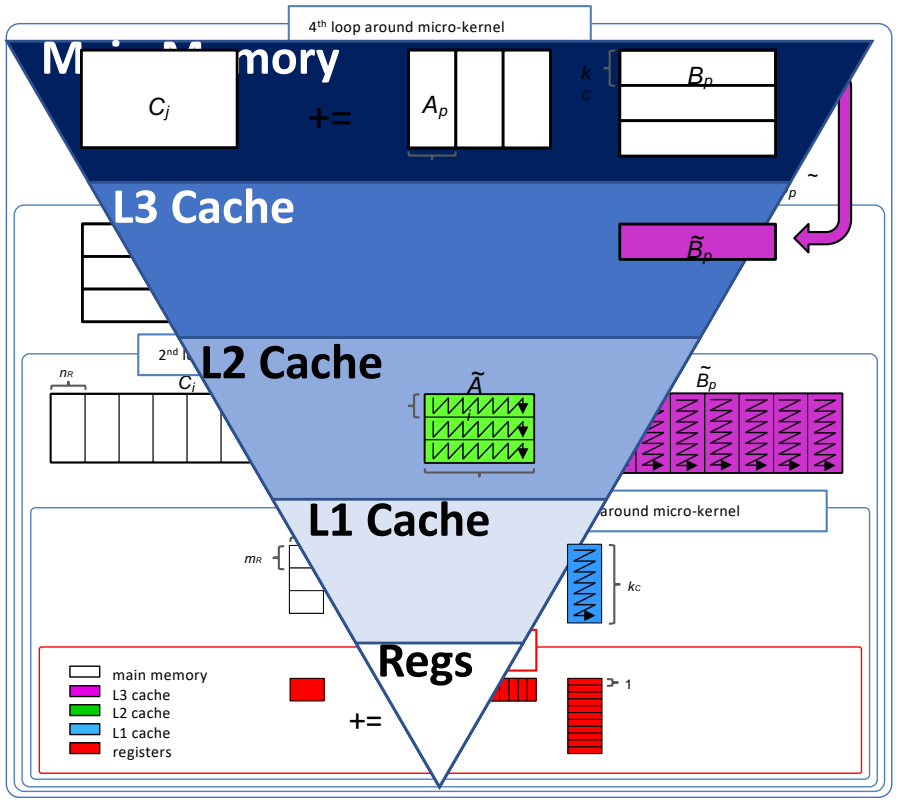
How do we apply it to different applications and architectures?

Explainable Designs though Analytical Models

Electrical & Computer ENGINEERING

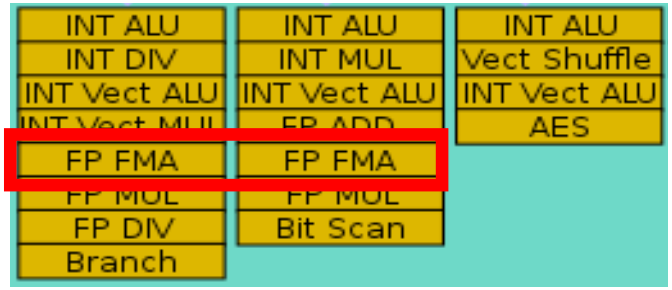
Capture HW-SW interactions with Models

- SW optimizations target specific HW features
- Analytical models map how SW is tuned based on available HW features
- Models inform changes in SW as HW and application requirements change

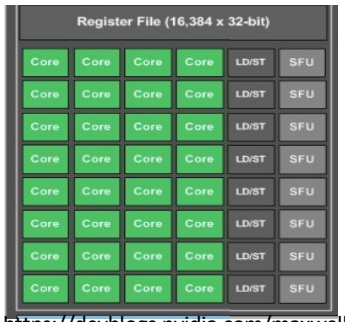


Hardware-driven constraints

- HW must be fully utilized for good performance
- Computational units characterized by
  - Number of Units,
  - Number of output computed
  - Time to compute each output (Latency)



https://en.wikichip.org/wiki/intel/microarchitectures/haswell



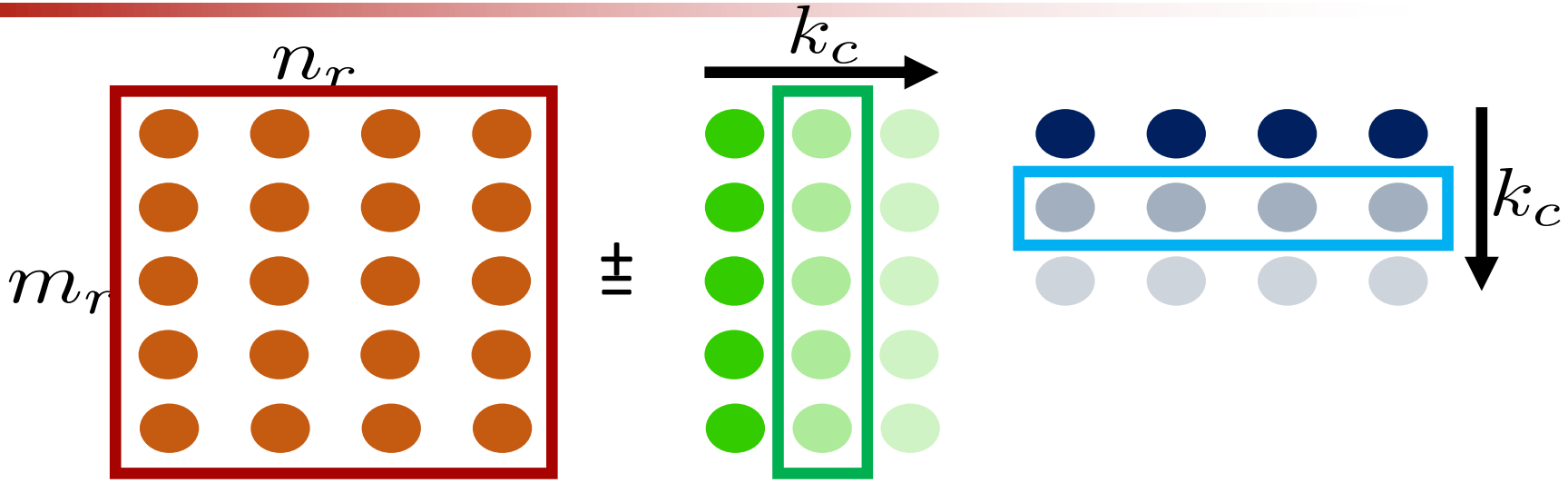
https://devblogs.nvidia.com/maxwell-most-advanced-cuda-gpu-ever-made/

$$\underbrace{L_{\mathrm{FMA}} N_{\mathrm{FMA}} N_{vec}}$$

Hardware Constraints

Tze Meng Low, Francisco D. Igual, Tyler M. Smith, and Enrique S. Quintana-Orti. 2016. Analytical Modeling Is Enough for High-Performance BLIS. *ACM Trans. Math. Softw.* 43, 2, Article 12

Software implementation of MMM



$$\underbrace{m_r n_r}_{\text{Software Parameters}} \geq \underbrace{L_{\text{FMA}} N_{\text{FMA}} N_{vec}}_{\text{Hardware Constraints}}$$

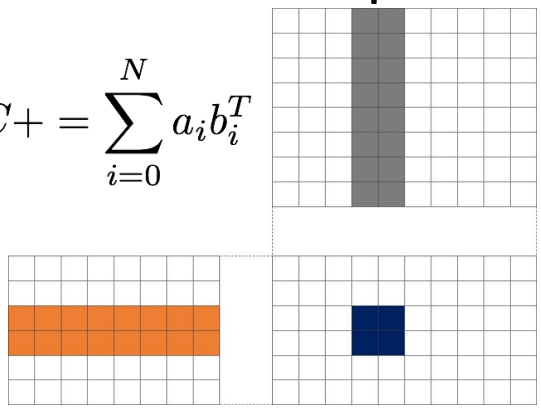
Tze Meng Low, Francisco D. Igual, Tyler M. Smith, and Enrique S. Quintana-Orti. 2016. Analytical Modeling Is Enough for High-Performance BLIS. *ACM Trans. Math. Softw.* 43, 2, Article 12

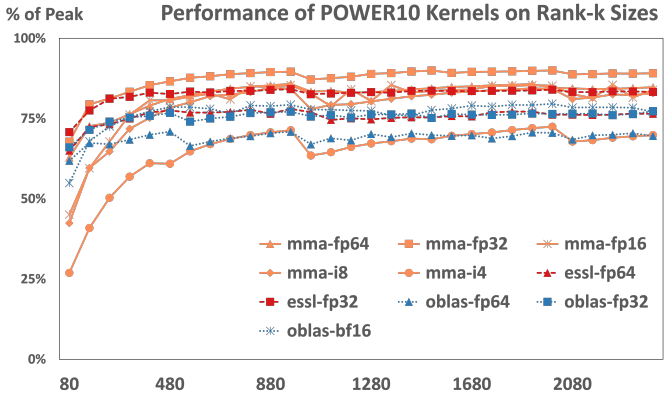
Portable to new hardware capabilities

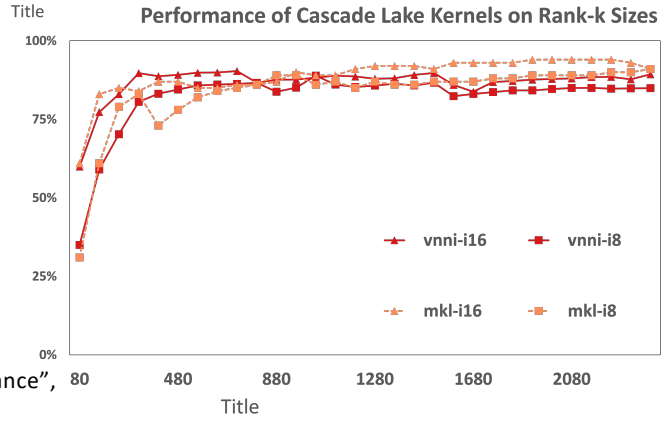
New Accelerator - Matrix Engines

- Single model for different architectures and data types
- Implementation specialized based on HW parameters

$$C+ = \sum_{i=0}^{N} a_i b_i^T$$

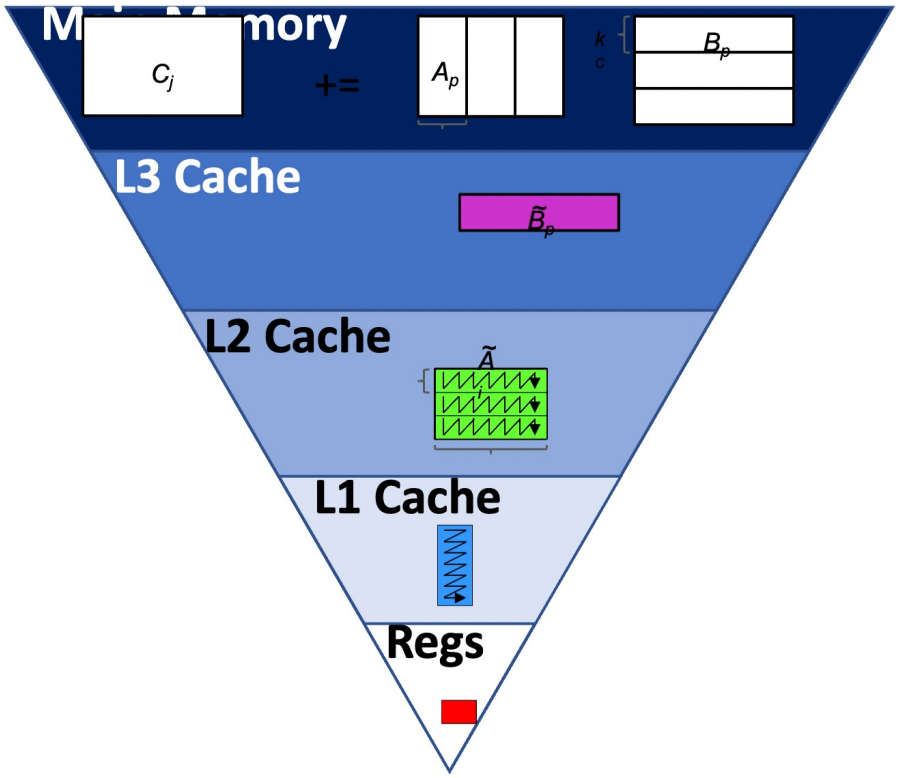






N. Tukanov, R. Srinivasaraghavan, J. E. Moreira, T.M. Low, "Modeling Matrix Engines for Portability and Performance", *IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2022.

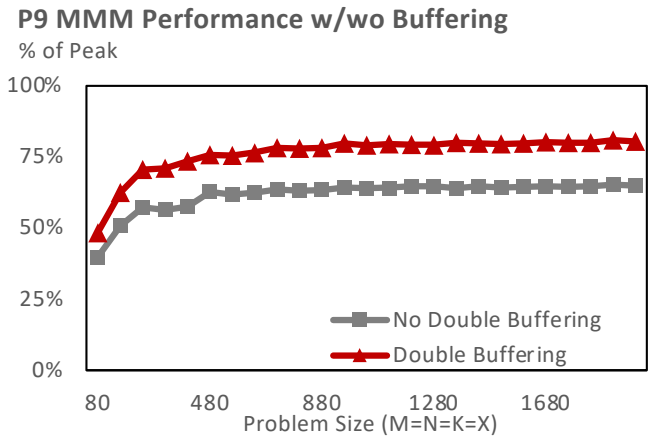
Data Orchestration is Key

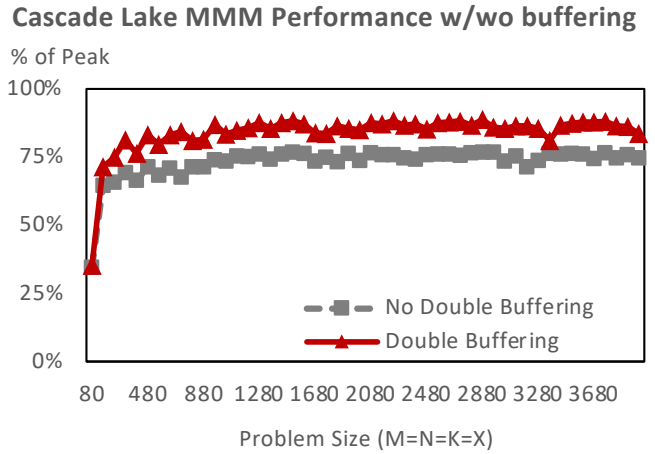


- Data orchestration is about keeping functional units busy
  - Performing data movement
  - Managing the caches

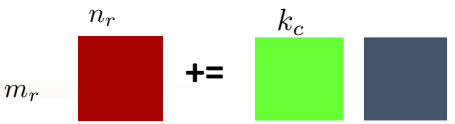
Using ``Double Buffering'' Liberally

- Bringing in data ahead of time
- Amount of buffering depends on
  - Location of data
  - Amount of data required
  - Computation time
- Need to ensure data in certain level of the cache





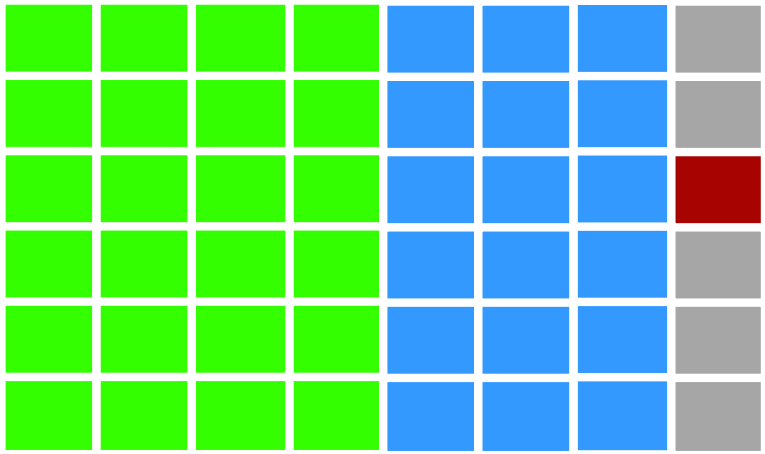
Modeling Data Layouts in Caches



- Hardware features: Set Associativity, Size of cache, Cache line size
- Software parameters: Block/Tile size

Current Architecture

8 ways

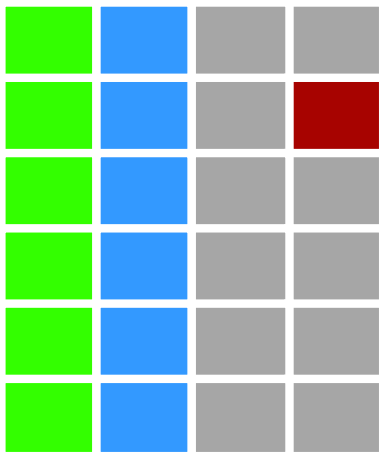


$$C_{A_r} \leq \left\lfloor \frac{m_r}{m_r + n_r}(W_{L1} - 1) \right\rfloor$$

$$k_c = \frac{C_{A_r} N_{L1} C_{L1}}{m_r S_{\text{data}}}$$

New Architecture

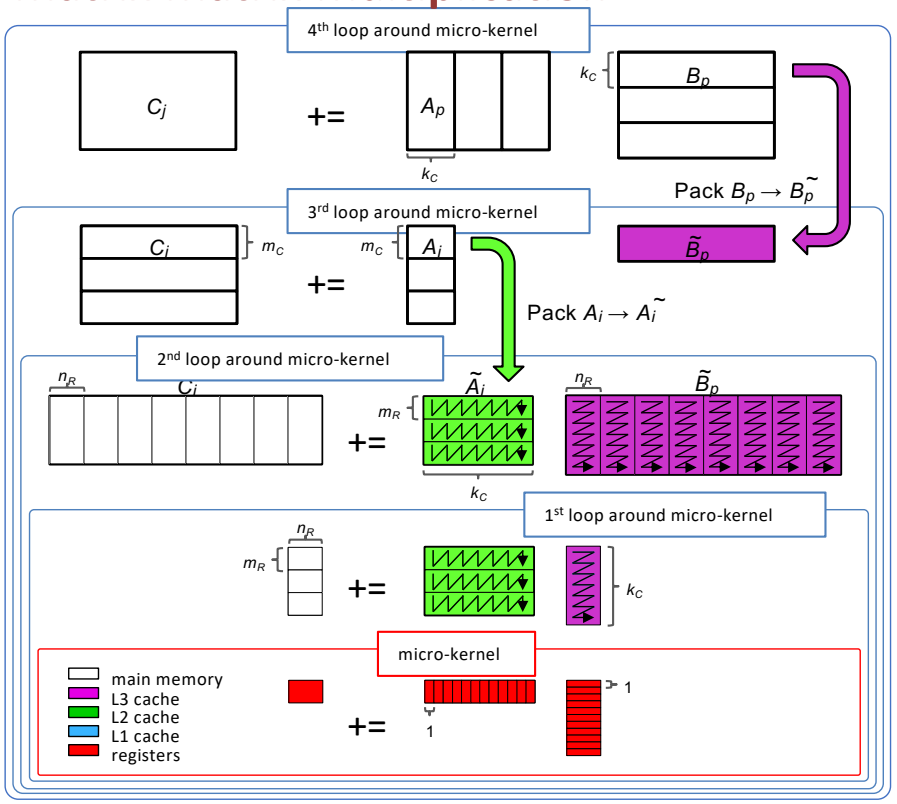
4 ways



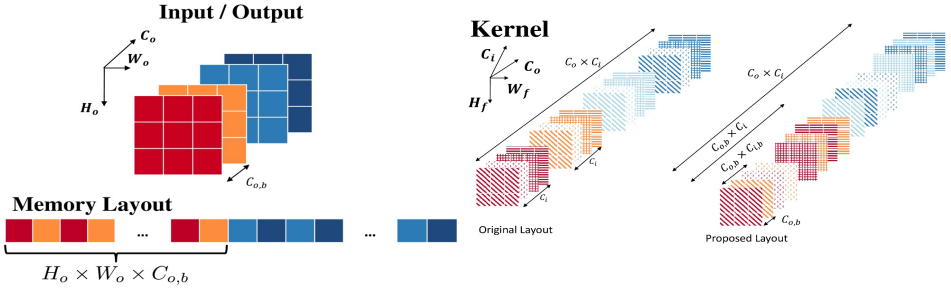
Tze Meng Low, Francisco D. Igual, Tyler M. Smith, and Enrique S. Quintana-Orti. 2016. Analytical Modeling Is Enough for High-Performance BLIS. *ACM Trans. Math. Softw.* 43, 2, Article 12

Intermediate Layouts Is Necessary

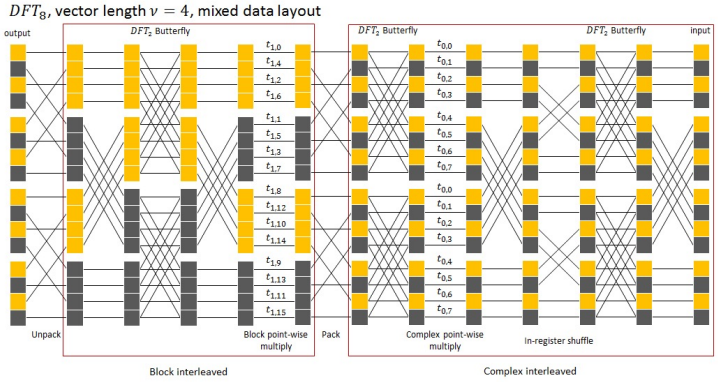
Matrix Matrix Multiplication



Direct Convolution



Fast Fourier Transforms

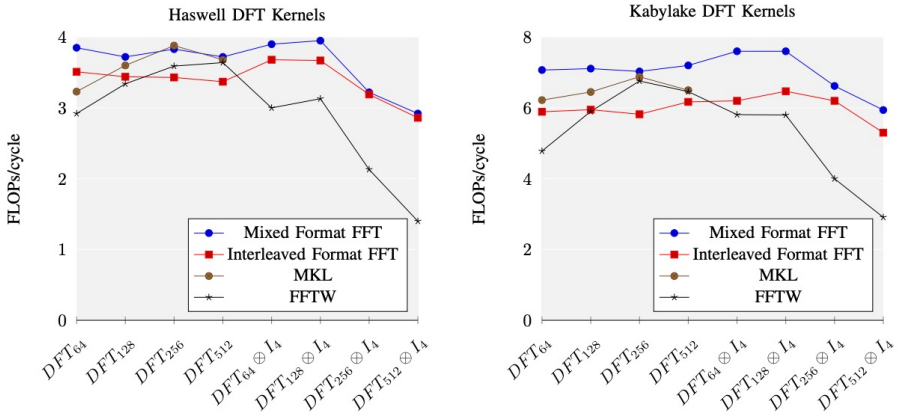


Further questions

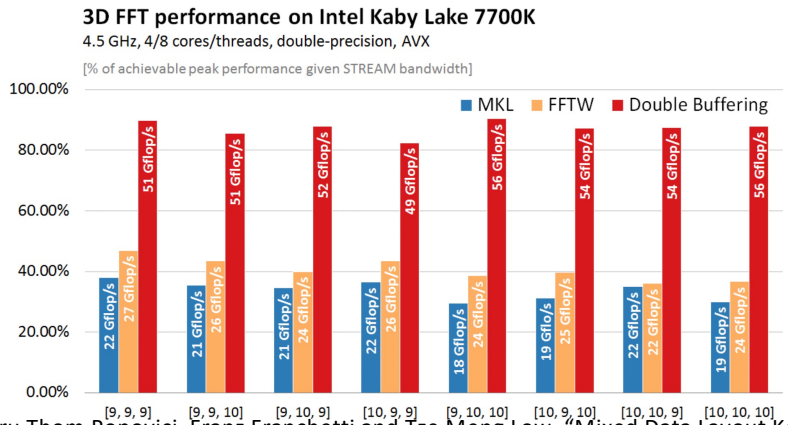
How general is the approach?

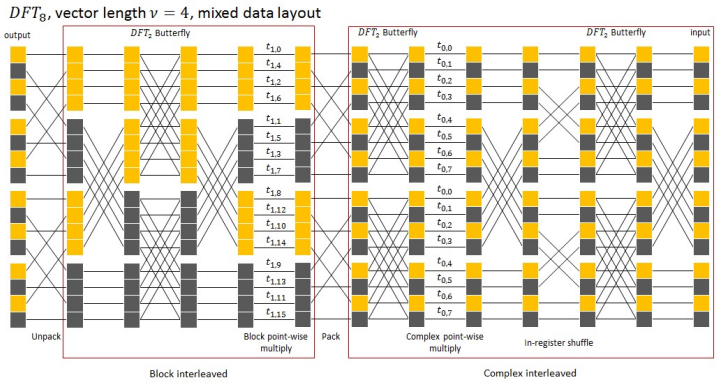
How can models for one application be used for another?

Rethinking FFT algorithms



- FFTs usually considered memory-bound
- Caches are fast enough to sustain FFT computation
  - Use different layouts at different stages of the computation



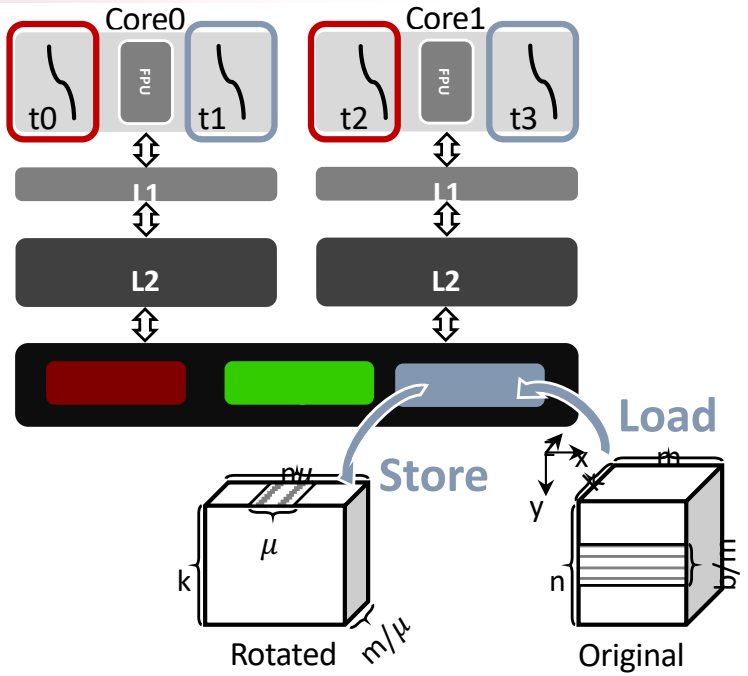


Doru Thom Popovici, Franz Franchetti and Tze Meng Low. "Mixed Data Layout Kernels for Vectorized Complex Arithmetic", 2017, *HPEC*

Doru Thom Popovici, Tze Meng Low and Franz Franchetti. "Large Bandwidth-efficient FFTs on Multicore and Multi-Socket Systems", 2018, *IPDPS*

Sharing caches with SMT Threads

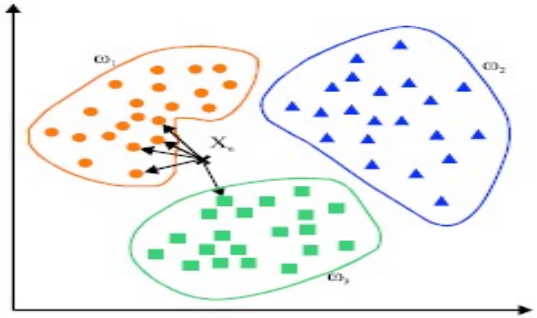
- SMT Threads share “private” caches
- Controlled data movement across different layers of the cache
  - Use of temporal loads/stores
  - Need locks to ensure correctness
  - Split caches by assigning ways to caches
- Sharing of functional units
  - NOPs introduced to allow both threads to proceed with their tasks



Many problems are “MMM”

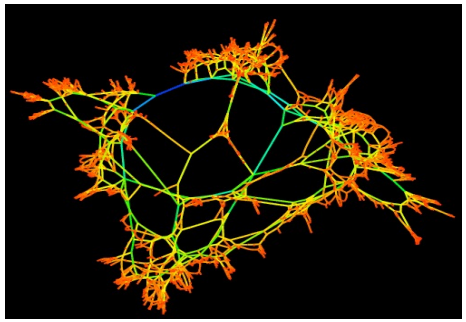
Population Genomics

k-Nearest Neighbours



DNA Fingerprinting

All-Pairs Shortest Path



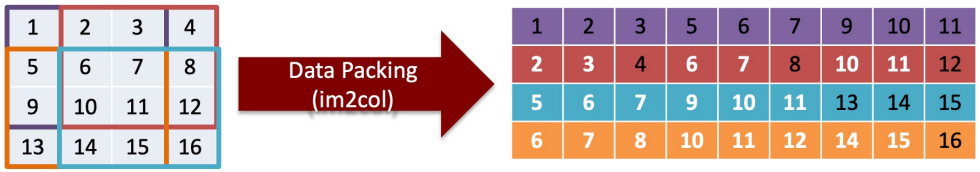
Charalampos Theodoris, Nikolaos Alachiotis, Tze Meng Low and Pavlos Pavlidis, “qLD: High-performance Computation of Linkage Disequilibrium on CPU and GPU” 2020 IEEE International Conference on BioInformatics And BioEngineering (BIBE)

Elliott Binder, Tze Meng Low, and Doru Thom Popovici. "A Portable GPU Framework for SNP Comparisons.” 2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)

High Performance Machine Learning

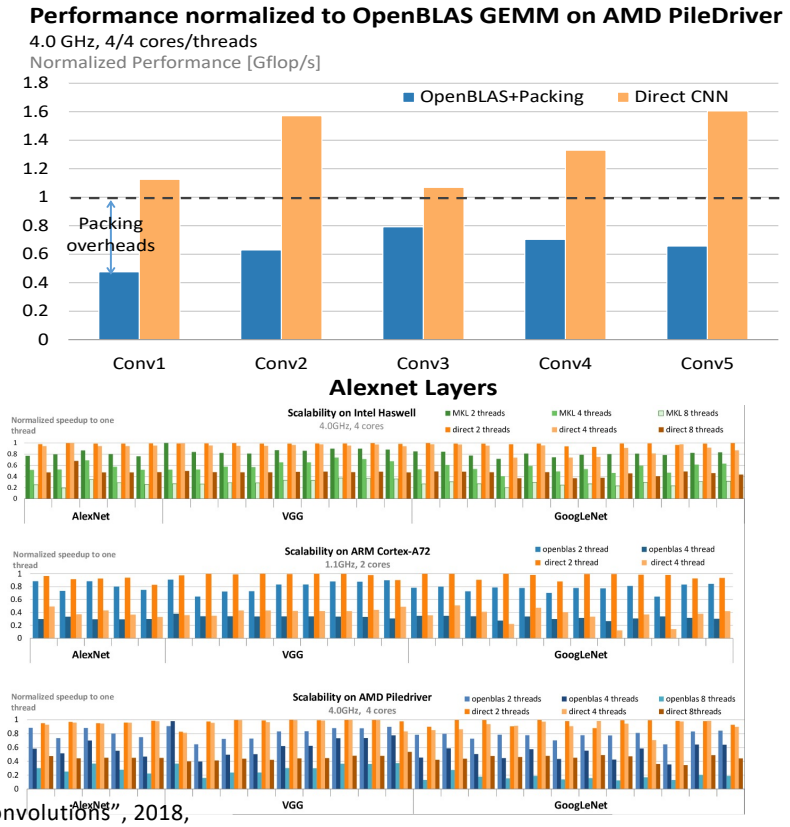
High Performance Direct Convolution

Leverage model-based data layout




```
for jj = 1 to C_o/Cob do
  for ii = 1 to C_i/C_ib do
    for l = 1 to H_o do
      for kk = 1 to W_o/W_ob do
        for n = 1 to H_f do
          for m = 1 to W_f do
            for i = 1 to C_ib do
              for k = 1 to W_ob do
                for j = 1 to C_ob do
                  O[...] += I[...]* F[...]
```

Compute different output blocks
- Compute across different inputs blocks
  - L2 / L3 Size
  - Associativity & Cache Size
- Reuse Kernel across outputs
  - L1 / L2 Size
  - Associativity & Cache Size
- Satisfy ILP requirement
  - Register Limit
  - SIMD Length
  - Cache Length Size



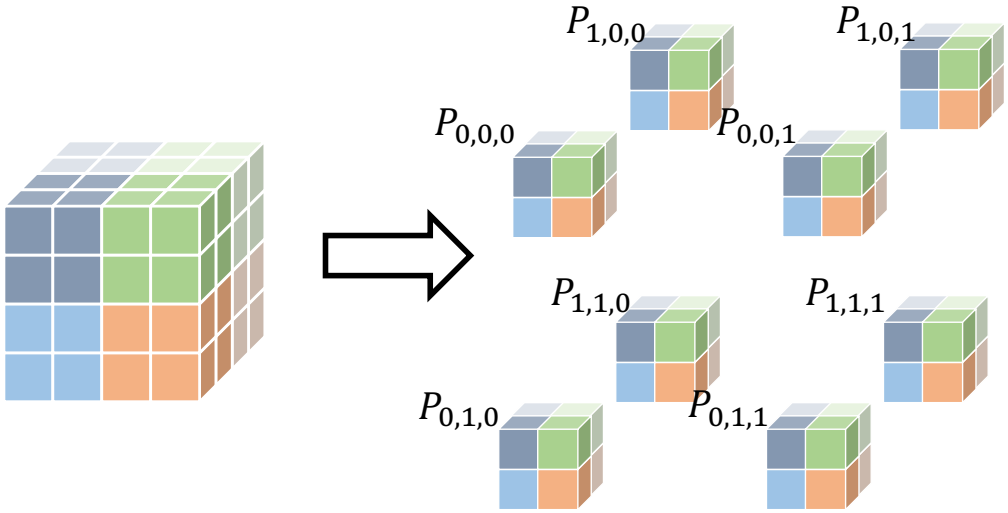
Jiyuan Zhang, Franz Franchetti, Tze Meng Low, "High Performance Zero-Memory Overhead Direct Convolutions", 2018, International Conference of Machine Learning (ICML)

Modeling extends to large machines

Larger Exascale Machines

3D parallelism + New Layout

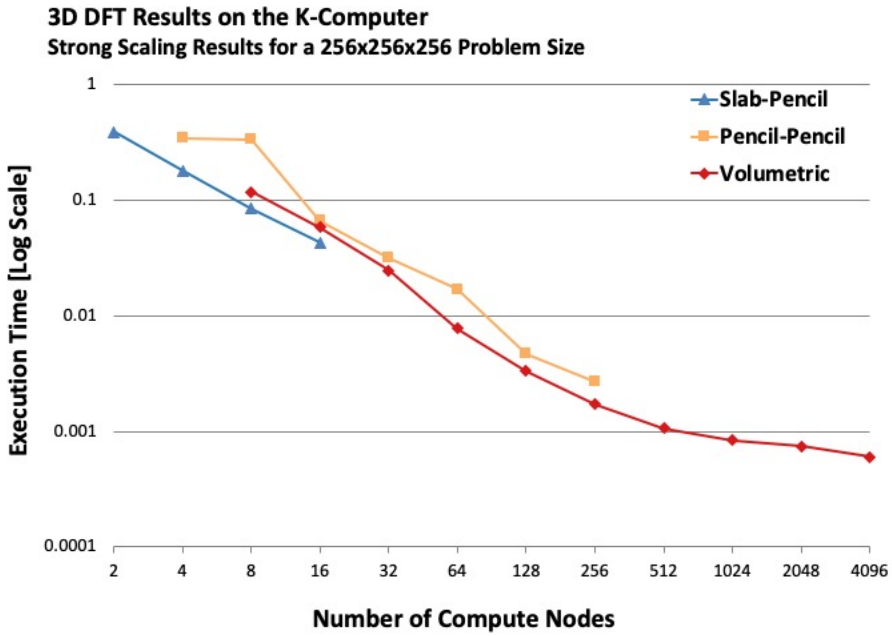
Computes partial FFT on single node (More but cheaper comms)

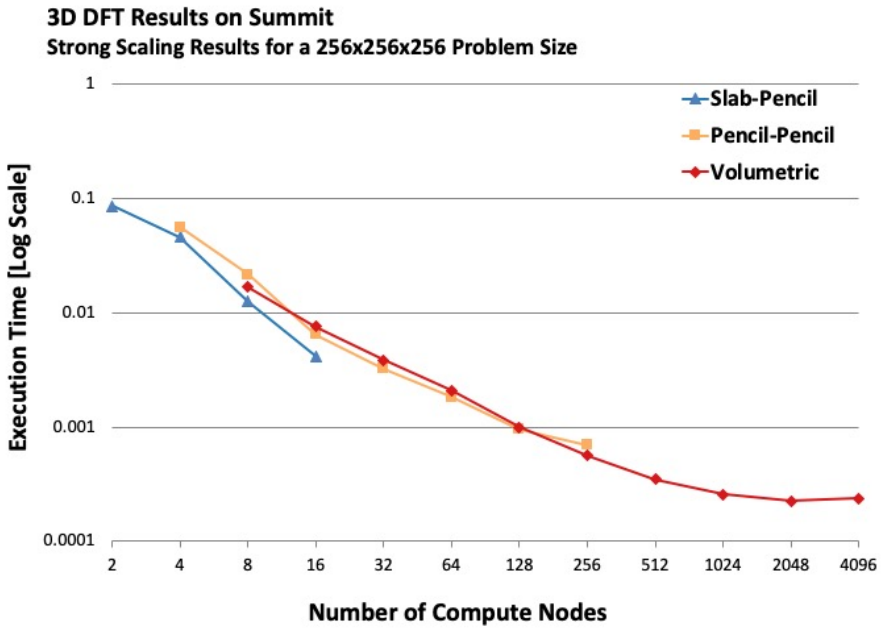


Doru Thom Popovici, Martin D. Schatz, Franz Franchetti, Tze Meng Low, "A Flexible Framework for Multidimensional DFTs", *SIAM Journal on Scientific Computing, 42(5), C245- C264*, Sep 2020.

Elemental 3D-FFT

Strong scaling to more nodes





Doru Thom Popovici, Martin D. Schatz, Franz Franchetti, Tze Meng Low, "A Flexible Framework for Multidimensional DFTs", *SIAM Journal on Scientific Computing, 42(5), C245- C264*, Sep 2020.

Towards Graphs & Sparse Linear Algebra

- Graph algorithms in the language of linear algebra
- New algorithms for
  - finding patterns/subgraphs
  - propagating information
- Modeling to find out HW bottlenecks

SuiteSparse

Galois GB

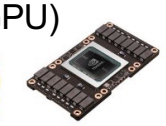
TEXAS
The University of Texas at Austin

IBM-GraphBLAS

redisgraph
redislabs

GraphBLAST (GPU)

UCDAVIS
UNIVERSITY OF CALIFORNIA



LAGraph
C algorithms repository

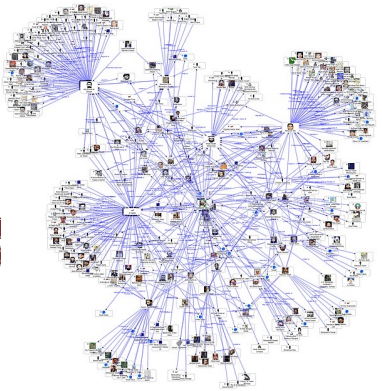
gbtl

Carnegie Mellon University

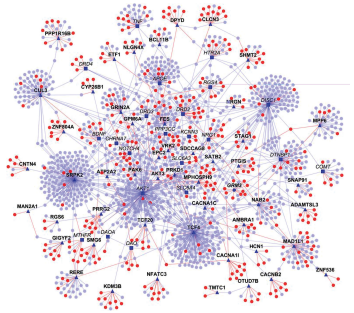
Pacific Northwest NATIONAL LABORATORY
Proudly Operated by Battelle Since 1965

pygraphblas
Python Wrapper

Distributed gbtl
Lawrence Livermore National Laboratory



Social Network



Protein Interactions

Mark P. Blanco, Tze Meng Low, and Kyungjoo Kim, "Exploration of Fine-Grained Parallelism for Load Balancing Eager K-truss on GPU and CPU", IEEE HPEC 2019 , **Graph Challenge Champion**

Tze Meng Low, Daniele G. Spampinato, Anurag Kutuluru, Upasana Sridhar, Doru Thom Popovici, Franz Franchetti, Scott McMillan (CMU) , "Linear Algebraic Formulation of Edge-centric K-truss Algorithms with Adjacency Matrices ", IEEE HPEC 2018 **Graph Challenge Finalist**

Electrical & Computer ENGINEERING

Summary

- Analytical models capture key SW-HW interactions
  - Flexibility and portability across architectures and applications
  - Rethink of many current algorithms & implementations
  - Tools integration (e.g. Polly-LLVM) ➔ programmer productivity

Questions?

lowt@cmu.edu