Accelerating Large-Scale Excited-State GW Calculations in Material Science

Charlene Yang
Application Performance Specialist
NERSC, LBNL

BESC 2021

Recent Developments on BerkeleyGW

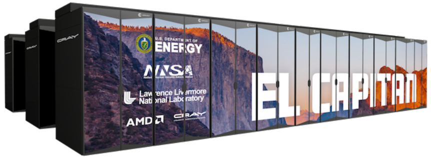
- CPU computing

  → GPU computing

- Four DOE supercomputers
  - Perlmutter - NVIDIA GPU
  - Frontier - AMD GPU
  - El Capitan - AMD GPU
  - Aurora - Intel GPU

NERSC
NVIDIA/Cray

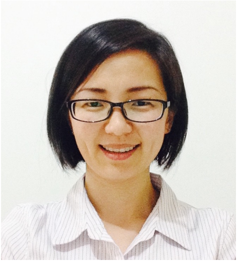
ORNL
AMD/Cray



LLNL
AMD/Cray

ANL
Intel/Cray

ACM Gordon Bell Finalist 2020



Mauro Del Ben	**Charlene Yang**	**Zhenglu Li**	**Felipe da Jornada**	**Steven G. Louie**	**Jack Deslippe**
Materials Science	HPC Performance	Physics	Materials Science	Physics	HPC Performance
CRD, LBNL	NERSC, LBNL	UC Berkeley	Stanford	UC Berkeley	NERSC, LBNL

M. Del Ben, C. Yang, Z. Li, F. H. da Jornada, S. G. Louie, and J. Deslippe, [“Accelerating Large-Scale Excited-State GW Calculations on Leadership HPC Systems”,] *International Conference for High Performance Computing, Networking, Storage and Analysis (SC’20)*, pp. 36-46, November 2020

Three Key Numbers

10,968 electrons

(Ground-Breaking for High Fidelity Excited-State Calculations)

105.9 PFLOP/s

(72% of the LINPACK Peak of Summit)

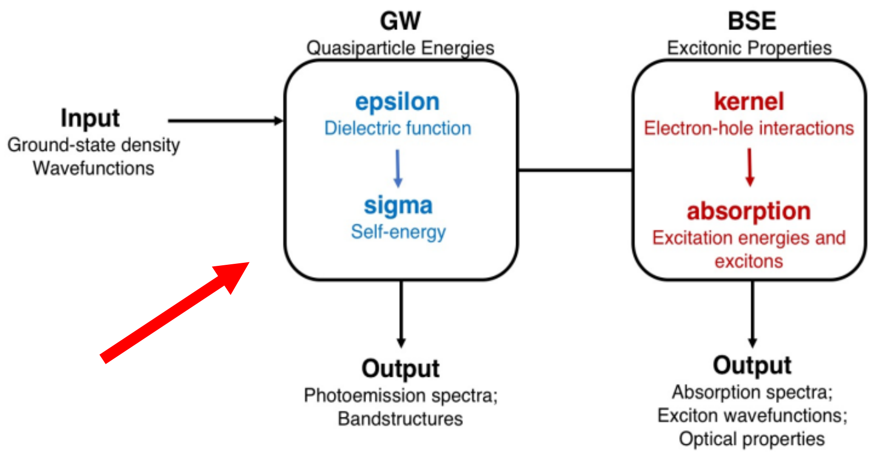
10 minutes

(Same Time to Make a Coffee)

GPU Implementation and Optimization

BerkeleyGW

https://berkeleygw.org



Code Base:

- ~100k LOC; Fortran; MPI/OpenMP on CPU

Computational motifs:

- Large matrix multiplications (100k x 100m!)
- Fourier transforms
- Large low-rank reductions
- Eigen problems
- Matrix inversions

Scaling for computation vs memory:

- Epsilon: $O(N^4)$ vs $O(N^3)$
- Sigma: $O(N^3)$ vs $O(N^2)$

	Kernel	Computation	Memory
Epsilon	MTXEL	$O(N_v N_c N_G^\psi \log N_G^\psi)$	$O(N_v N_c N_G)$
	CHI-0	$O(N_v N_c N_G^2)$	$O(N_v N_c N_G + N_G^2)$
	Inversion	$O(N_G^3)$	$O(N_G^2)$
Sigma	MTXEL	$O(N_\Sigma N_b N_G^\psi \log N_G^\psi)$	$O(N_b N_G)$
	GPP	$O(N_\Sigma N_b N_G^2)$	$O(N_G^2 + N_b N_G)$

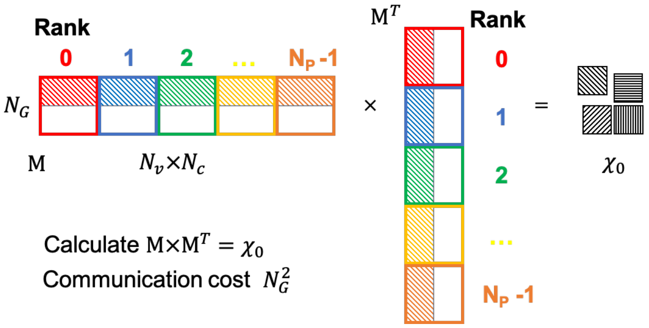
BerkeleyGW https://berkeleygw.org

- **CPU code:**
  - ~100k LOC; Fortran; MPI/OpenMP
- **GPU porting and optimization:**
  - CUDA/C++ and OpenACC branches
  - cuBLAS/cuFFT libraries and custom codes
  - non-blocking cyclic communication scheme
  - CUDA streams
  - batching operation
  - data prestaging
  - Roofline analysis

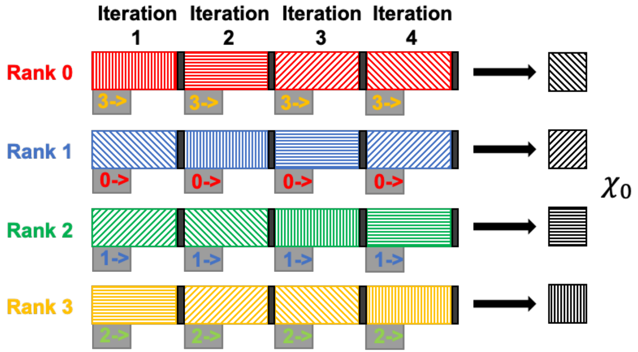
Some optimizations are on the **application level**, and some are on the **kernel level**!

App-level: Epsilon CHI-0

- Non-blocking cyclic communication:
  - overlap GPU computation with MPI communication
  - point-to-point MPI vs. MPI collectives
- Batching mechanism:
  - avoid OOM on GPU and CPU
- Offload data preparation to device
  - D-H is a weak link in accelerated computing
  - utilize asynchronous D-H transfers



data layout for **M** matrix in CHI-0

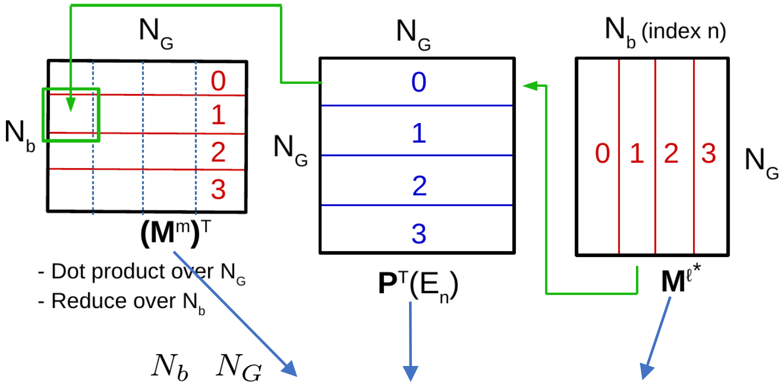


non-blocking cyclic communication scheme

App-level: Sigma GPP

- Tensor contraction
- Abundance of parallelism
  - inter-pool vs. intra-pool
  - MPI ranks, CUDA streams, threadblocks, threads

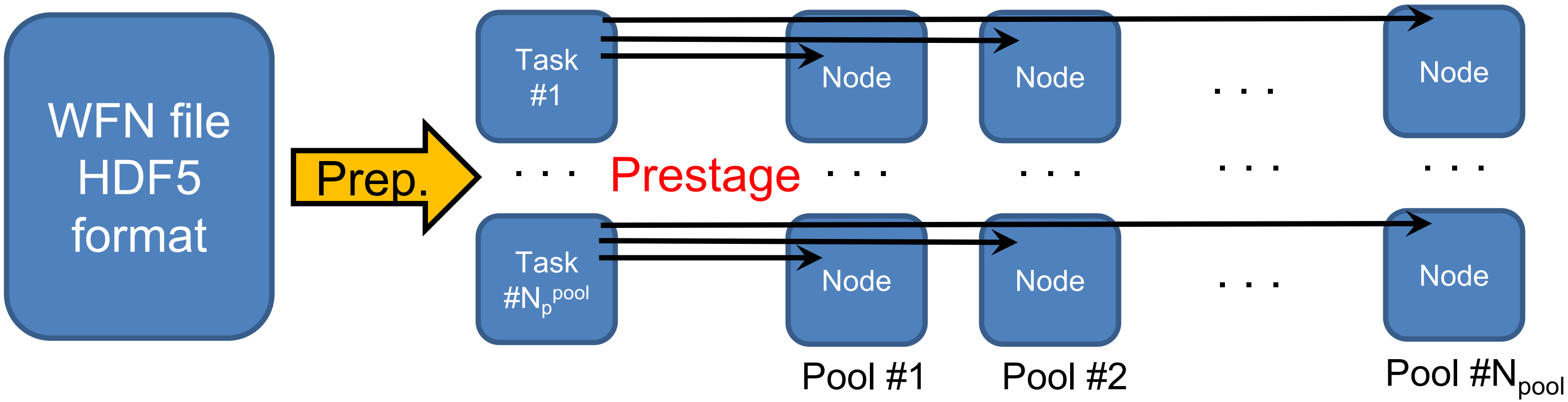
    → large data reduction
- Kernel-level optimization
  - execution latency, memory latency
- Roofline analysis
  - bandwidth bound → compute bound



$$\Sigma_{lm} = \sum_{n}^{N_b} \sum_{GG'}^{N_G} M^{m}_{G'n}[P^{\mathsf{T}}(E_n)]_{G'G} M^{l*}_{Gn}$$

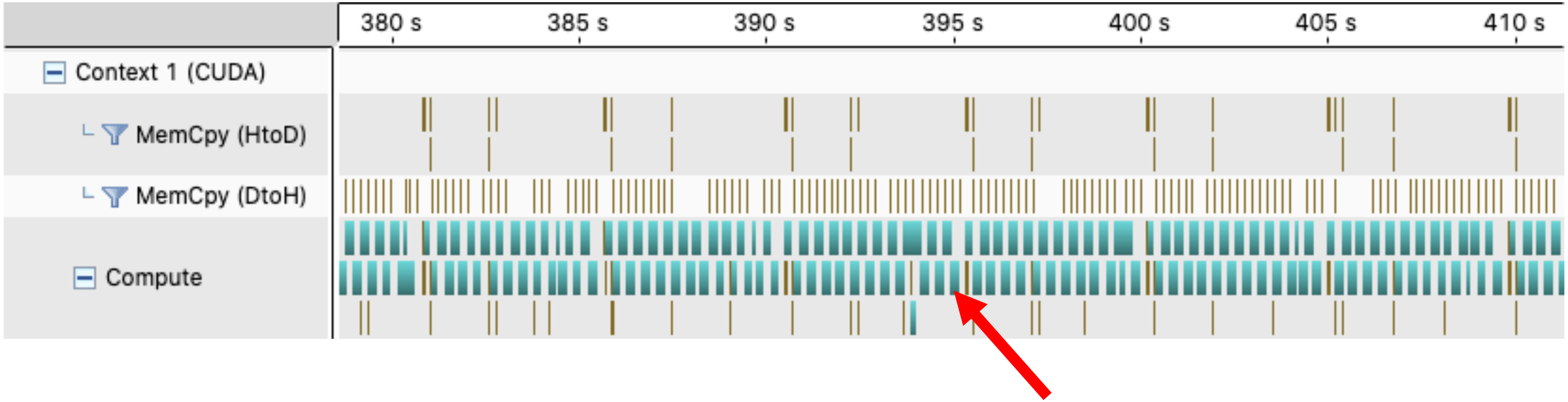
App-level: I/O optimization

- Exploit node-local solid-state memory (SSD) on Summit
  - Prepare data → in a distributed form
  - Prestage data → to node-local SSDs
  - Runtime → each rank reads from its own SSD



Kernel-level: Sigma GPP

- Contributes to >90% of the runtime for Sigma
- Nsight System profile:



One of 3000 kernel invocations on each GPU

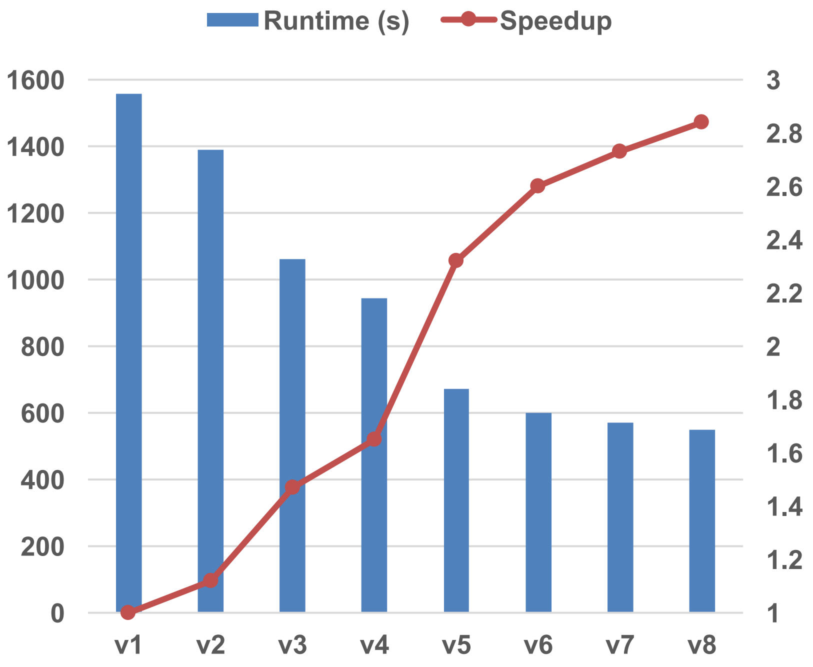
Kernel-level: Sigma GPP

```
# pseudo code per invocation
for band = 1, nbands # O(1k)
 for igp = 1, ngpown # O(10k)
  for ig = 1, ncouls # O(100k)
   for iw = 1, nw     # small
    Computation
Reduce to small arrays
```

- Tensor contraction
  - Bandwidth bound
- Reduction of 10^{12} numbers
  - Shared mem for partial sums
- Double complex numbers
  - High register usage
- Multiple multi-dim arrays
  - Memory access pattern
- Long-latency operations
  - Divisions, square roots

Kernel-level: Sigma GPP

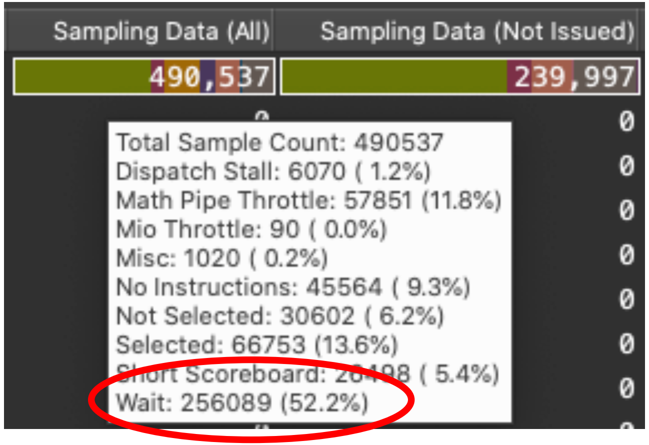
1. Baseline*
2. Replace divides with reciprocals
3. Replace square roots with power of 2
4. Replace divides and square roots
5. Loop re-ordering
6. Further increase occupancy
7. Cache blocking
8. Add more arrays to shared memory



*with certain optimizations included retrospectively

1. Reduce Execution Latency (v4)

- Before optimization



Warps are stalled waiting on a
fixed latency execution dependency

https://docs.nvidia.com/nsight-compute/ProfilingGuide/index.html#statistical-sampler

- Replace complex divides by reciprocals

$$(a+bi)/(c+di)$$

$$1.0/(c^2+d^2) * ((ac+bd)+(bc-ad)i)$$

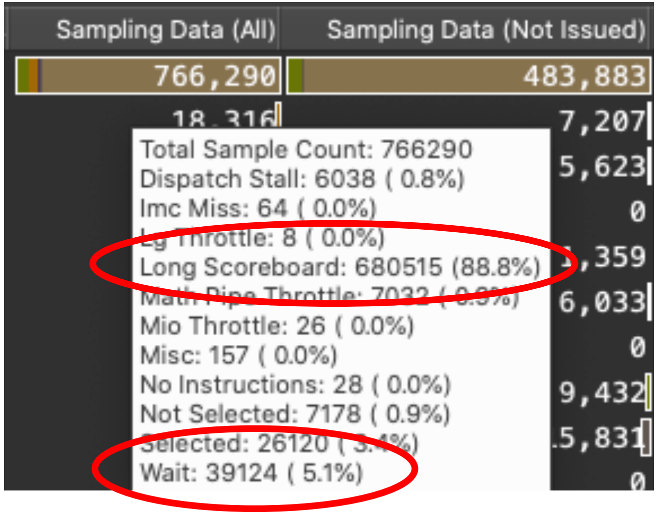
- Replace square roots by power of 2 calculations

$$abs(a+bi)>c$$

$$(a^2+b^2)>c^2$$

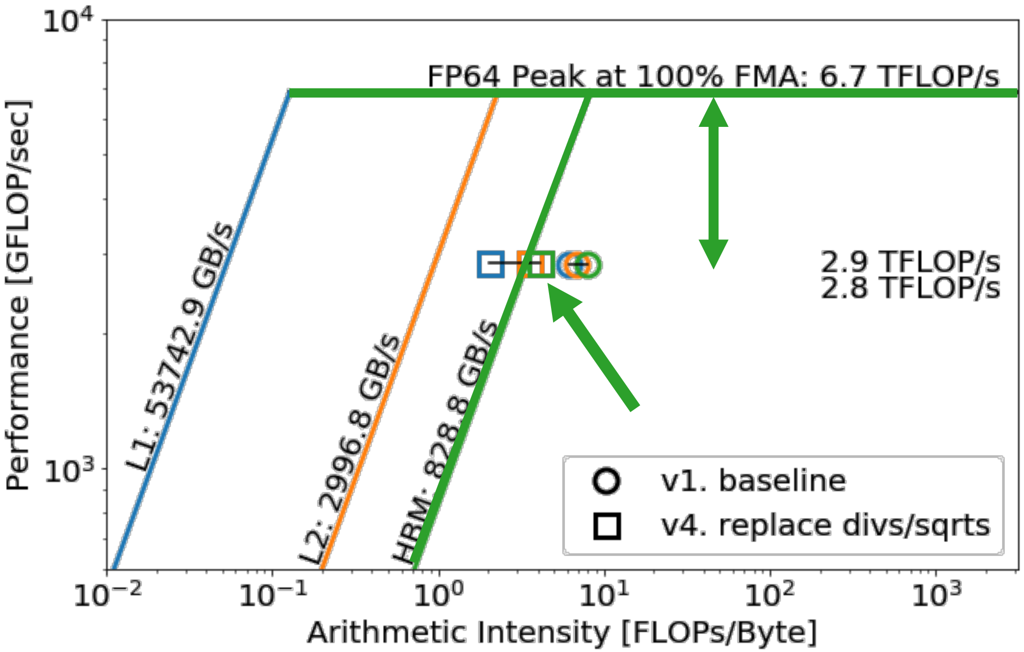
1. Reduce Execution Latency (v4)

- After optimization



Warp ***Waits*** have dropped significantly, even though ***Long Scoreboard*** has become more pronounced

- Hierarchical Roofline chart



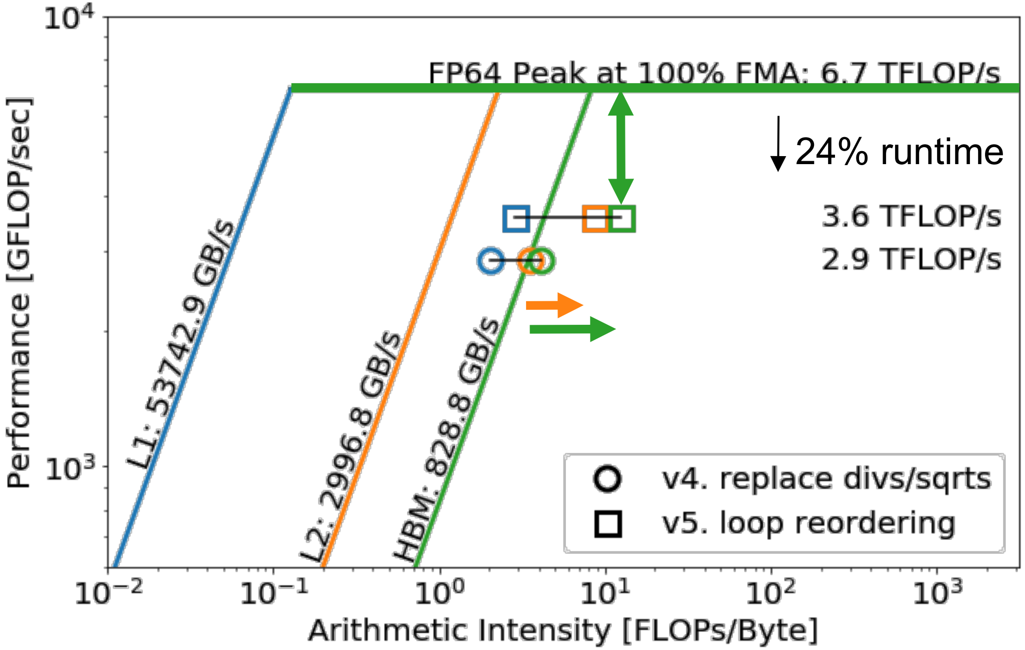
2. Gain Arithmetic Intensity (v5)

```
# Before optimization (v4)
for band = 1, nbands    # O(1,000)
  for igp = 1, ngpown   # O(10,000)
    for ig = 1, ncouls # O(100,000)
    …

# After optimization (v5) (more kernels)
for igp = 1, ngpown        # O(10,000)
  for ig = 1, ncouls       # O(100,000)
    for band = 1, nbands # O(100)
    …
```

- **Parallelism**
- **Memory access**
- **Cache blocking**

2. Gain Arithmetic Intensity (v5)



- Higher HBM arithmetic intensity
- Compute bound now !

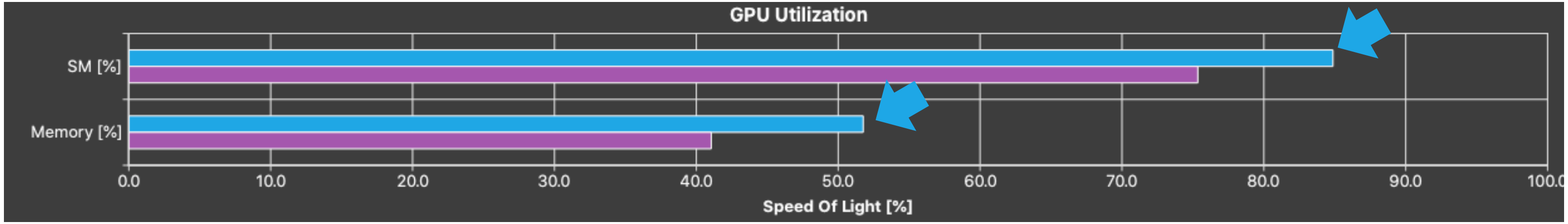
- What's the peak for V100?
  - 6.7 TFLOP/s vs 7.8 TFLOP/s
  - 1312 MHz vs 1530 MHz

$$80\times32\times2\times1312\text{e}6=6.7\ \text{TFLOP}/s$$

Try nvidia-smi or Nsight Compute

3. More Compute Resources (v6)

- GPU computing is all about **latency hiding**!
- Keep an eye on kernel launch parameters
- Experiment with `maxregcount`
  - Trade register spill for higher occupancy
  - Do this when the code is stable (register usage might change)



Performance Results

Benchmark Setup

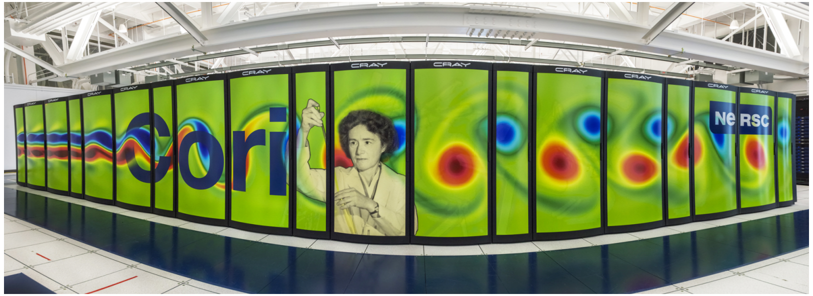
- Summit (OLCF)

  4,608 nodes, each with 2 IBM POWER9 CPUs and 6 NVIDIA V100 GPUs
- Cori-GPU (NERSC)

  18 nodes, each with 2 Intel Xeon Skylake CPUs and 8 NVIDIA V100 GPUs
- Cori-Haswell (NERSC)

  2,688 Haswell nodes, each with 2 Intel Xeon E5-2698v3 CPUs

Summit
OLCF

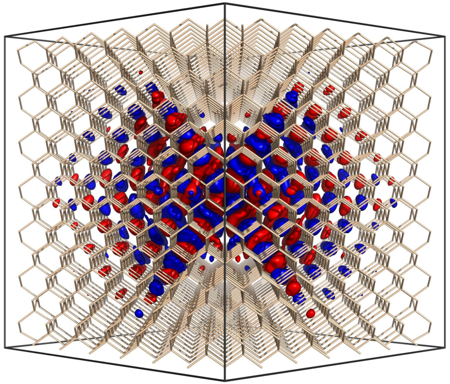


Cori
NERSC

Benchmark Setup

- Point defects in semiconductors
  - silicon / silicon carbide for qubit prototypes
- Up to 2,742 atoms and 10,968 electrons

Parameters	Si-214	Si-510	Si-998	SiC-998	Si-2742
N_{spin}	1	1	1	2 (↑/↓)	1
N_G^{ψ}	31,463	74,653	145,837	422,789	363,477
N_G	11,075	26,529	51,627	149,397	141,505
N_b	6,397	15,045	29,346	16,153	80,694
N_v	428	1,020	1,996	1,997/1,995	5,484
N_c	5,969	14,025	27,350	14,156/14,158	75,210
N_{Σ}	Variable, up to 120 per spin				
Epsilon PFLOPs	2.5	80.5	1164	10,091	66,070
Epsilon Memory (TB)	0.45	6.07	45.1	135	934
Epsilon Comm. Vol. (GB)	3.92	22.5	85.3	1428	640
Sigma PFLOPs	0.127	1.71	12.6	58.2	260.7
Sigma Memory (GB)	6.19	34.3	133.8	791.4	1006
Sigma Comm. Vol. (GB)	2.27	12.8	48.5	77.2	365.4



Isosurface for one of the in-gap states associated with a divacancy defect in Silicon

Large-scale calculation:
100s of TBs memory!
10s of EFLOPs compute!

GPU vs CPU Speedup

Epsilon

Si-214, Skylake CPU vs. V100 GPU on Cori,
2 nodes total (4 CPUs vs. 16 GPUs)

18.6x speedup!

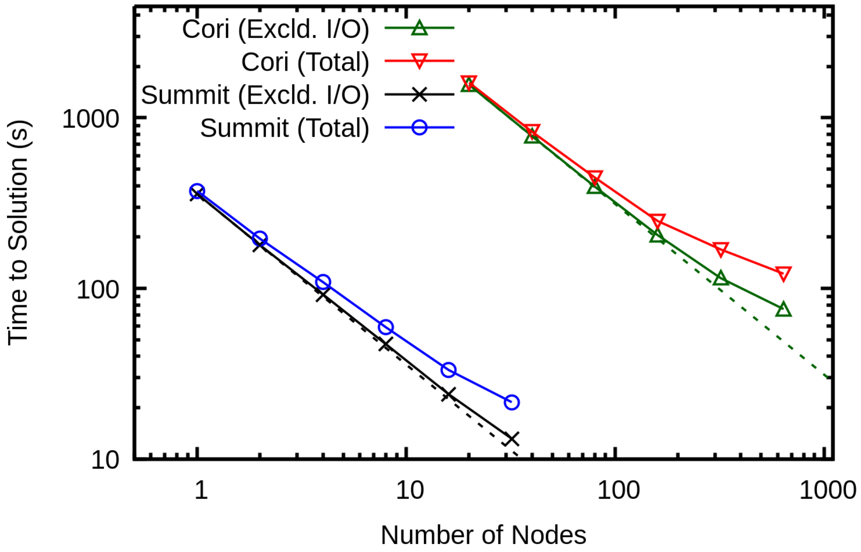
	MTXEL	CHI-0	Invert	Total
CPU Only	616	1120	10.3	1794
GPU Full-Offload	24.4	47.7	9.6	96.3

25x **23.5x** **18.6x**

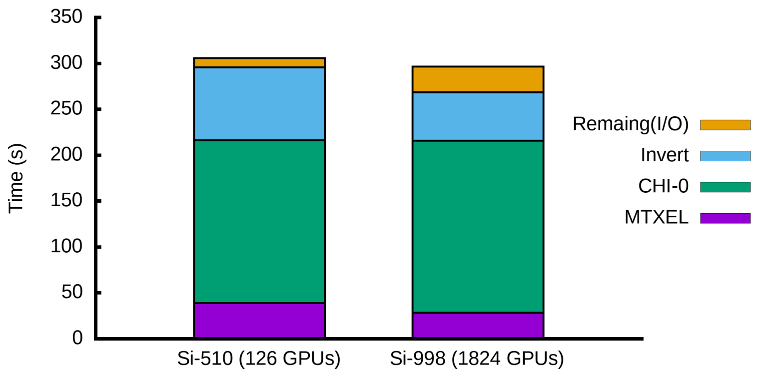
Sigma

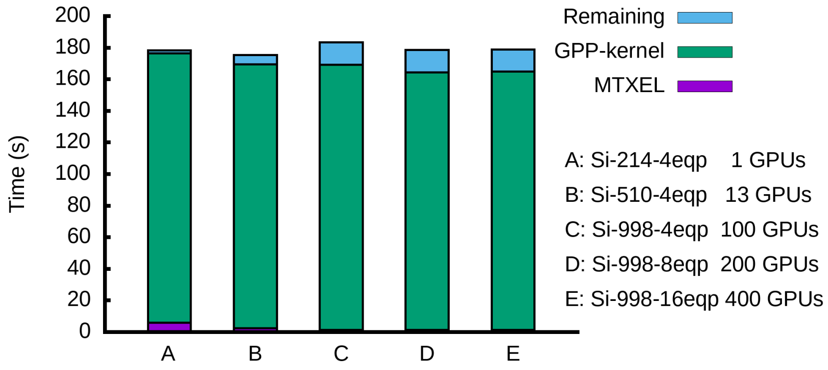
Si-510, Cori Haswell CPU vs. Summit V100 GPU,
node-to-node (2 CPUs vs. 6 GPUs)

86x speedup!



Weak Scaling





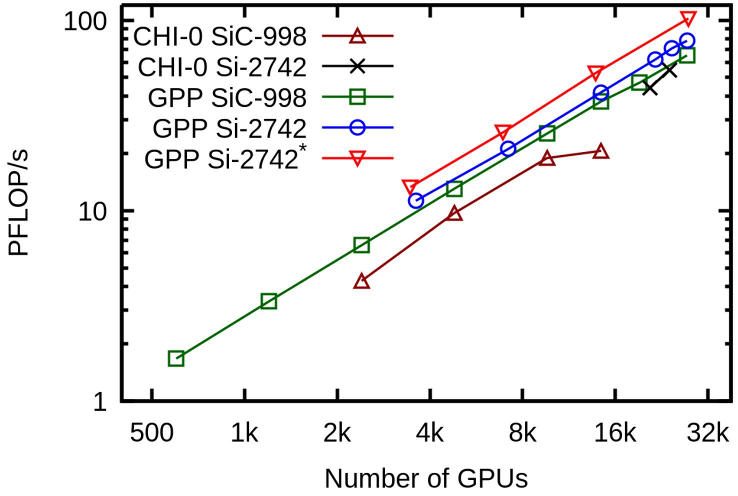
Epsilon on Summit

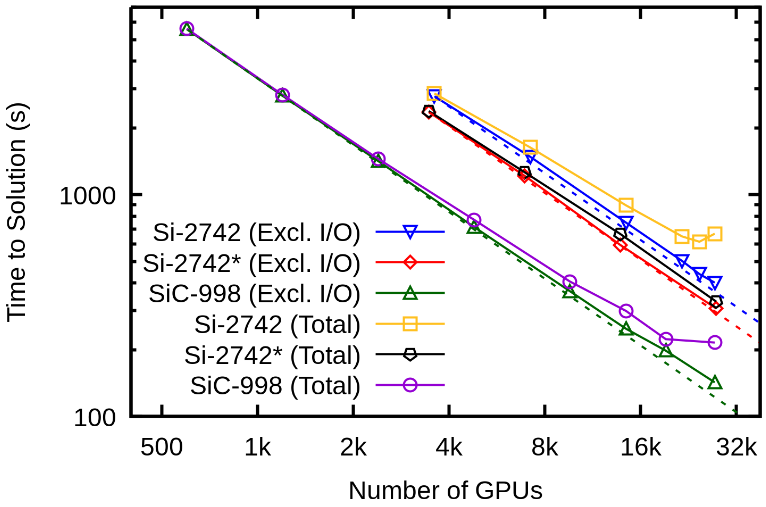
- Most computationally intensive: CHI-0
- The number of GPUs is scaled according to the computational complexity $O(N^4)$.

Sigma on Summit

- The number of GPUs is scaled according to the $O(N^3)$ computational complexity in Cases A, B and C, and to the number of quasiparticles in Cases C, D and E.

Strong Scaling and Best Performance





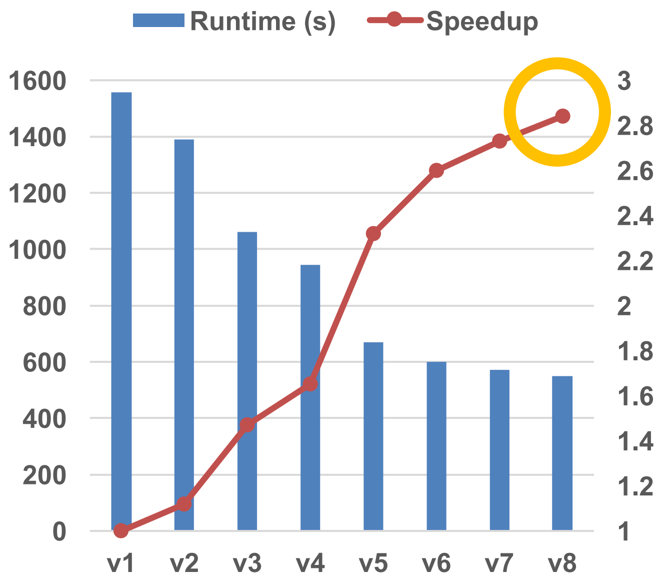
Best Performance from Sigma

	# of GPUs	# of Pools	GPUs per Pool	Compute (s)	IO (s)	Throughput (PFLOP/s)	% of Peak
SiC-998	27,360	80	342	142	71	65.3	32.9
Si-2742	27,360	120	228	401	226	78.0	39.2
Si-2742*	27,648	128	216	307	23	102.1	50.9
Si-2742*	27,648	256	108	592	39	105.9	52.7

- **[Top left]** Throughput of Epsilon CHI-0 and Sigma GPP for SiC-998 and Si-2742 on Summit
- **[Top right and Bottom]** Strong scaling and best performance (PFLOP/s) of Sigma on Summit

Best Performance

GPP per GPU: 3.9 TFLOP/s



Sigma on Full Summit: 105.9 PFLOP/s

Application	BerkeleyGW
Benchmark	Si-2742
# of GPUs	27,648 (full Summit)
Compute Time	592 s
I/O Time	39 s
Throughput	105.9 PFLOP/s (FP64)
% of R_{peak}	52.7% of 200.79 PFLOP/s
% of R_{max}	71.3% of 148.60 PFLOP/s

Three Key Numbers

10,968 electrons

(Ground-Breaking for High Fidelity Excited-State Calculations)

105.9 PFLOP/s

(72% of the LINPACK Peak of Summit)

10 minutes

(Same Time to Make a Coffee)

Acknowledgement

NERSC

BERKELEY LAB
Bringing Science Solutions to the World

CRD COMPUTATIONAL RESEARCH DIVISION

C2SEPEM
Center for Computational Study of Excited-State Phenomena in Energy Materials

Berkeley
UNIVERSITY OF CALIFORNIA

Stanford University

OAK RIDGE National Laboratory | LEADERSHIP COMPUTING FACILITY

U.S. DEPARTMENT OF ENERGY | Office of Science

Acknowledgement

- This research used resources at the National Energy Research Scientific Computing Center (NERSC), which is supported by the U.S. Department of Energy Office of Science under contract DE-AC02-05CH11231.

- This research used resources at the Oak Ridge Leadership Computing Facility (OLCF) through the Innovative and Novel Computational Impact on Theory and Experiment (INCITE) program, which is supported by the U.S. Department of Energy Office of Science under Contract No. DE-AC05-00OR22725.

- This work was supported by the Center for Computational Study of Excited-State Phenomena in Energy Materials (C2SEPEM), funded by the U.S. Department of Energy Office of Science under Contract No. DEAC02-05CH11231.