Efficient Large-Scale Sparse Eigenvalue Computations on Heterogeneous Hardware

Moritz Kreutzer*, Andreas Pieper†, Andreas Alvermann†, Holger Fehske†, Georg Hager*, Gerhard Wellein*, and Alan R. Bishop‡

Density of states

Key step: Computation of Chebyshev polynomials

Kernel Polynomial Method[1] for DOS:

```
for r = 0 to R-1                    ← Loop: random initial states
  |v⟩ = |rand());
  Compute η0, η1
  for m = 1 to M/2                  ← Loop: Chebyshev polynomials
    swap(|w⟩, |v⟩);
    |u⟩ = H|v⟩;                     ← Sparse matrix-vector multiply
    |u⟩ = |u⟩ - b|v⟩;               ← Vector-vector operations
    |w⟩ = -|w⟩;
    |w⟩ = |w⟩ + 2a|u⟩;
    η2m = ⟨v|v⟩;                    ← Compute moments
    η2m+1 = ⟨w|v⟩ ;
  end
end
```

Bandwidth-bound algorithm
→ Increase computational intensity I:

1. Fuse vector-vector operations into SpMV: **augmented SpMV**
2. Apply matrix to block of R random vectors[6]: **augmented SpMMV**

```
|V⟩ := |v⟩0..R-1;
|W⟩ := |w⟩0..R-1;
|V⟩ = |rand());
Compute η0, η1
for m = 1 to M/2
   swap(|W⟩, |V⟩);
   |W⟩ = 2a(H-b)|V⟩−|W⟩ &
       η2m = ⟨V|V⟩ &
       η2m+1 = ⟨W|V⟩
end
```

→ Higher performance

→ Custom implementation

→ Interleaved storage of block vectors

→ New bottlenecks!?

Motivation

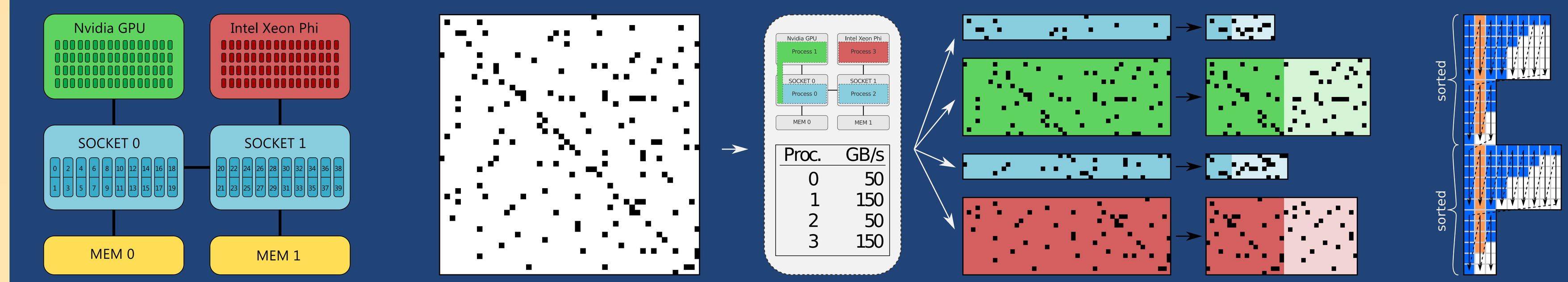
Goal: Determine eigenvalue properties of large, sparse matrix (H)

$$\boldsymbol{Hx=\lambda x}, \quad \lambda_1,\lambda_2,\ldots,\lambda_k,\ldots,\lambda_{D-1},\lambda_D$$

Target features:
- → Density of states (DOS)
  Approximation to the full eigenvalue spectrum
- → Inner eigenvalues ($10^2,\ldots,10^3$)
- → Matrix dimension D ≥ 10^9

→ Efficient and scalable
Fully heterogeneous (CPU+GPU) implementation

Implementation



Fully heterogeneous execution with GHOST[4]:
- → Data-parallel heterogeneous execution
- → Fused and auto-generated kernels
- → SELL-C-σ sparse matrix storage format
- → MPI+OpenMP+CUDA+SIMD+Tasking

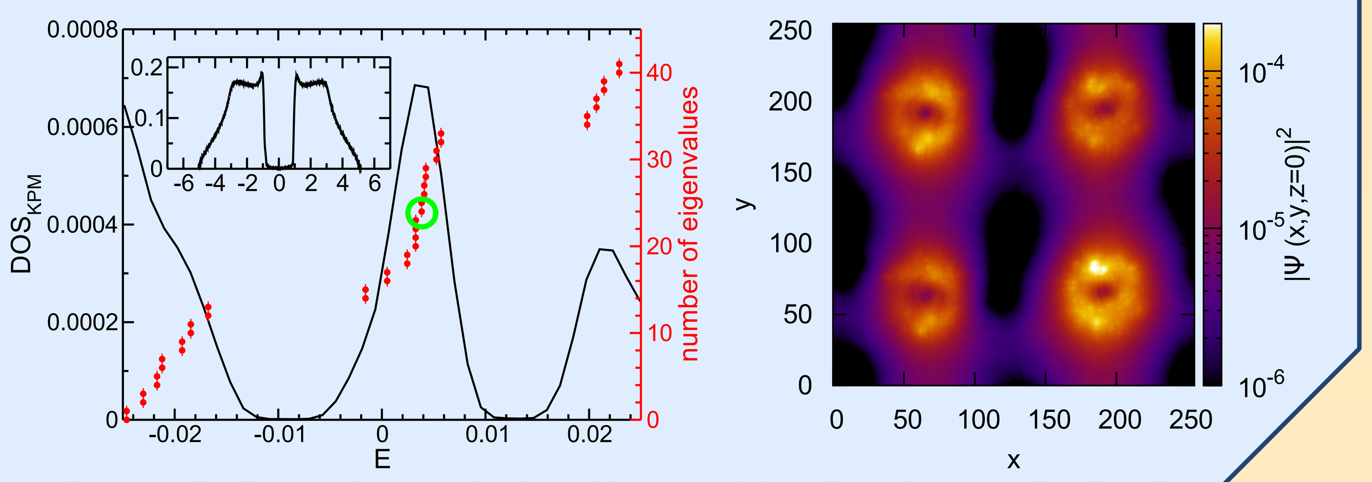
Download

bitbucket.org/essex

Physics application

Gate-defined quantum dots on topological insulator surfaces

$$H = -\sum_{n,j=1}^{3}\left(\Psi^\dagger_{n+\hat{e}_j}\frac{\Gamma^1 - i\Gamma^{j+1}}{2}\Psi_n + \text{H.c.}\right) + \sum_n \Psi^\dagger_n\left(2\Gamma^1 + \epsilon_n \mathbb{1}\right)\Psi_n$$



DOS and eigenvalues calculated by KPM and ChebFD in the band center.

Wavefunction amplitude of the surface eigenstate (green circle).

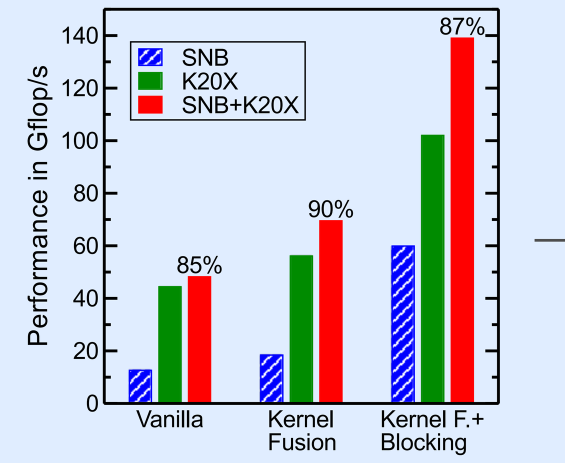
Inner Eigenvalues

Algorithm: Chebyshev Filter Diagonalization (ChebFD [2, 5])

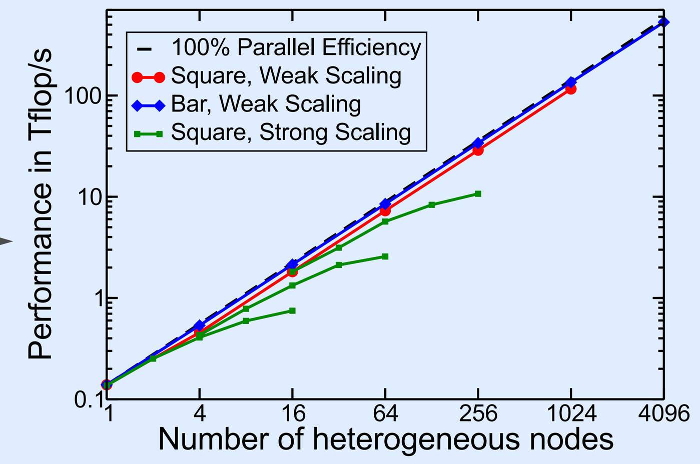
Setup
→ Apply polynomial filter ← Similar to KPM
→ Orthogonalize vectors ← Tall skinny GEMM
→ Convergence check

Performance

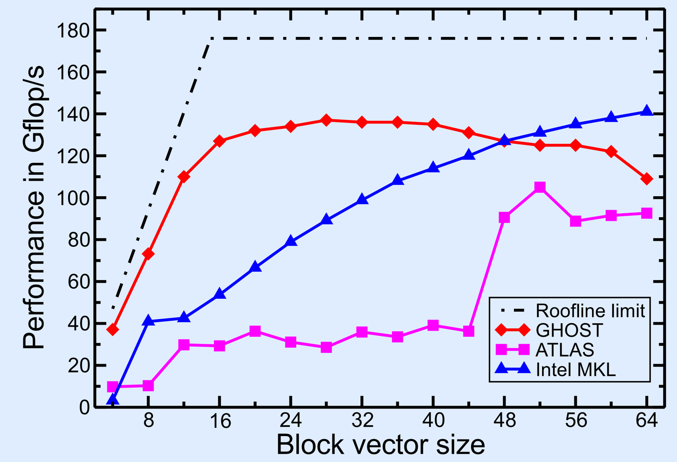
Optimized algorithm + tuned single-device implementations + fully heterogeneous execution



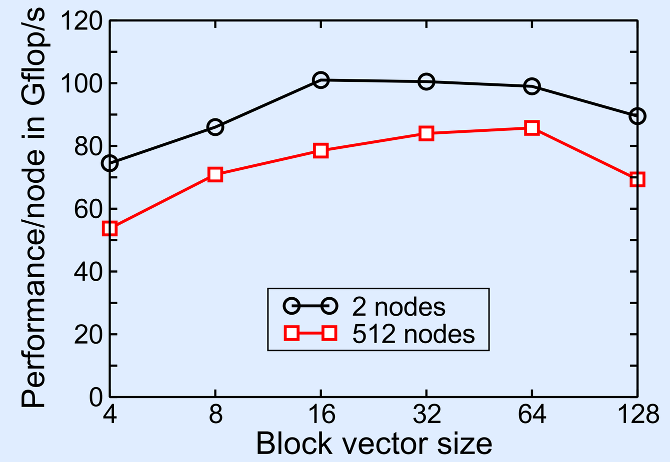
Node-level KPM performance on a single node of Piz Daint (1x Intel Xeon E5-2670, 1x Nvidia Tesla K20X).



Large-scale KPM performance (CPU+GPU) on Piz Daint (largest matrix D=10^{10}).



Block vector times small matrix performance on Intel Xeon E5-2660v2 (tall skinny ZGEMM).



Large-scale ChebFD performance (CPU) on SuperMUC Phase 2 (Intel Xeon E5-2697v3). On 512 nodes, 148 inner eigenvalues were sought of a matrix with D=10^9

Performance: Roofline model

Topological insulator matrix:
non-zeros per row N_{nzr}=13

I = 0.29 F/B
↓ Fused Kernel
I = 0.45 F/B
↓ Blocking (R→∞)
I = 2.88 F/B

Computational intensity (finite R):
$I(R) = 138/[(260/R)+48]$ F/B

Prediction[3] for bandwidth-bound code: $\mathbf{P_{MEM} = I*b_{MEM}}$ (b_{MEM} = max. bandwidth)

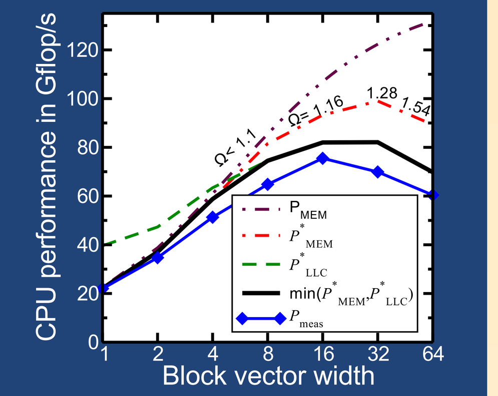
Increasing R ↔ Limited cache size
→ Increased memory traffic

Ω = actual over mimum data volume

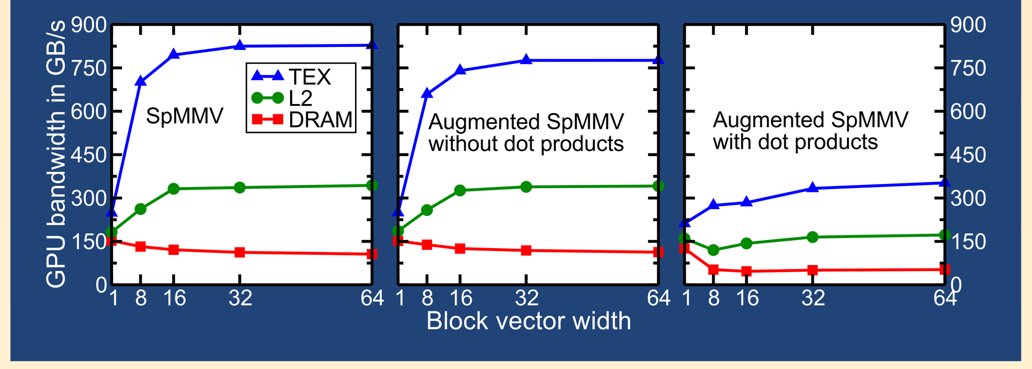
Corrected prediction: $\mathbf{P^*_{MEM}=\Omega P_{MEM}}$

→ Decoupling from main memory

→ New CPU bottleneck: cache bandwidth b_{LLC}



→ New GPU bottleneck: reduction/ synchronization for dot product



Acknowledgment

This work was funded by the German Research Foundation (DFG) through the Priority Program 1648 (Software for Exascale Computing) under project ESSEX. We would like to thank CSCS and LRZ for granting access to their machines Piz Daint and SuperMUC.

References

[1] A. Weiße, G. Wellein, A. Alvermann, H. Fehske. The kernel polynomial method. Rev. Mod. Phys., 78:275-306, Mar 2006
[2] E. Di Napoli, E. Polizzi, Y. Saad. Efficient estimation of eigenvalue counts in an interval. Preprint, 2014.
[3] S. Williams, A. Waterman, D. Patterson. Roofline: An insightful visual performance model for multicore architectures. Commun. ACM, 52(4):65-76, Apr. 2009.
[4] M. Kreutzer, J. Thies, M. Röhrig-Zöllner, A. Pieper, F. Shahzad, M. Galgon, A. Basermann, H. Fehske, G. Hager, G. Wellein. GHOST: Building blocks for high performance sparse linear algebra on heterogeneous systems. Preprint, 2015.
[5] A. Pieper, M. Kreutzer, M. Galgon, A. Alvermann, H. Fehske, G. Hager, B. Lang, G. Wellein. High-performance implementation of Chebyshev filter diagonalization for interior eigenvalue computations. Preprint, 2015
[6] W. D. Gropp, D. K. Kaushik, D. E. Keyes, B. F. Smith. Towards realistic performance bounds for implicit CFD codes. In Proceedings of Parallel CFD'99, pages 233-240. Elsevier, 1999.

*: Erlangen Regional Computing Center, Friedrich-Alexander University of Erlangen-Nuremberg, Erlangen, Germany
†: Institute of Physics, Ernst Moritz Arndt University of Greifswald, Greifswald, Germany
‡: Theory, Simulation and Computation Directorate, Los Alamos National Laboratory, Los Alamos, New Mexico, USA

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
ERNST MORITZ ARNDT UNIVERSITÄT GREIFSWALD
SPPEXA