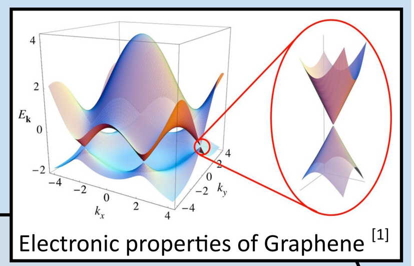
Recursive Algebraic Coloring Engine

Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), Erlangen Regional Computing Centre (RRZE)
Christie Louis Alappat <christie.alappat@fau.de>, Georg Hager, Gerhard Wellein, Olaf Schenk

RACE Hardware Friendly Coloring

MOTIVATION

Many iterative numerical methods for sparse systems and building blocks of sparse linear algebra are difficult to parallelize due to strong data dependencies. For example these may be loop carried dependency as they appear in solvers and preconditioners like Gauss Seidel (GS) or more complex write conflicts as seen in Kaczmarz (KACZ) iterative solver or Symmetric Sparse Matrix Vector multipliation (SymmSpMV).



Electronic properties of Graphene [1]

Gauss-Seidel

```
for(row=0:nrows)
{
    temp=b[row];

    for (j=rowptr[row]+1:rowptr[row+1])
        temp -= A[j] * x[col[j]];

    diag = A[rowptr[row]];
    x[row]  = temp/diag;
}
```

Preconditioner in many solvers, example [2] HPCG

FEAST sparse eigenvalue solver

Internal solver

KACZMARZ

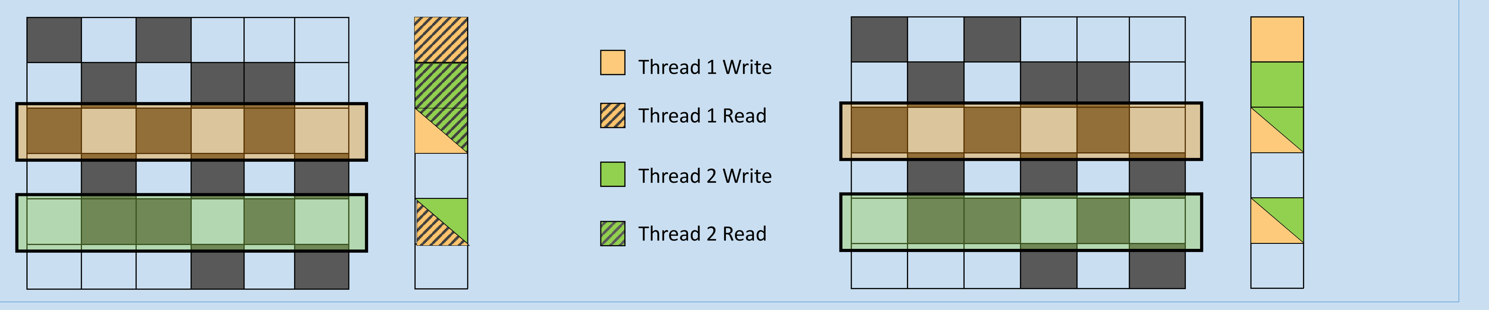
```
for(row=0:nrows)
{
   scale = b[row];

   for (j=rowptr[row]:rowptr[row+1])
       scale -= A[j] * x[col[j]];

   for (j=rowptr[row]:rowptr[row+1])
       x[col[j]] += scale * A[j];
}
```

DISTANCE-1 dependency

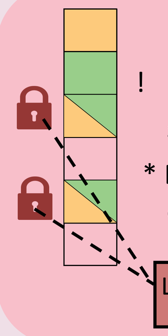
- Write-read conflicts
- Common examples : Preconditioners like GS, incomplete Cholesky decomposition.

DISTANCE-2 dependency

- Write-write conflicts, false sharing.
- Common examples : Sparse Matrix Transpose Vector (SpMTV), SymmSpMV, solvers like KACZ.



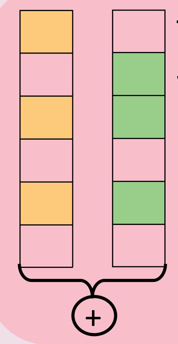
EXISTING APPROACHES



LOCKING

- ! Danger of serialization.
- * Easy to implement

Lock elements



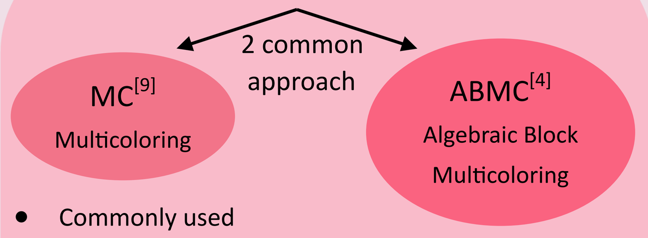
THREAD PRIVATE VECTOR

- ! Extra data traffic
- ! Worse for skinny matrix
- * Commonly used strategy for SpMTV like kernels

MATRIX REORDERING

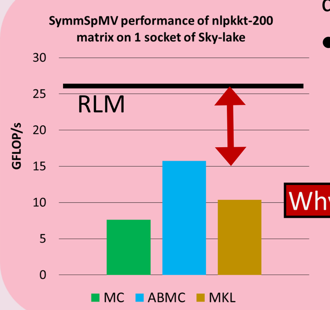
MC[9] Multicoloring — 2 common approach — ABMC[4] Algebraic Block Multicoloring

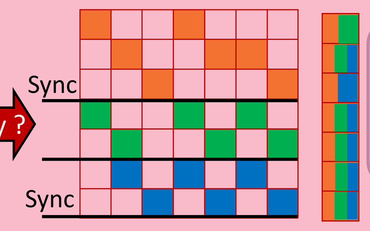
- Commonly used strategy for Distance-1 coloring.
- No explicit disadvantage stated in other methods, and minimal change to kernel(s).
- Extension of Multicoloring for Distance-2 has been done[3].
- We extend ABMC for Distance-2 coloring



DIFFERENT STORAGE FORMAT

- ! Need code rewriting.
- ! Require lot of tuning
- * Example CSR, RSB[5]





Problem increases
↑ nrows
↑ k

RAC OBJECTIVES

- Data Locality $I_{SymmSpMV} = \frac{2}{8+4+16*\alpha+\frac{8+4}{N_{nzr}}}\left[\frac{Flops}{Bytes}\right]$
- Sufficient Parallelism
- HARDWARE EFFICIENCY
- Reduced synchronization
- Simple Data format like CRS

RAC METHOD

- RAC—Recursive Algebraic Coloring
- Recursive level based method
- Four steps
- General method applicable to distance-k dependencies.
- Currently limited to matrices having symmetric structure or undirected graph.

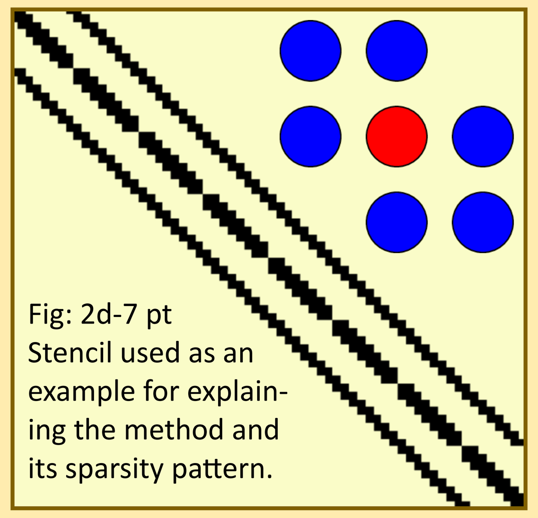
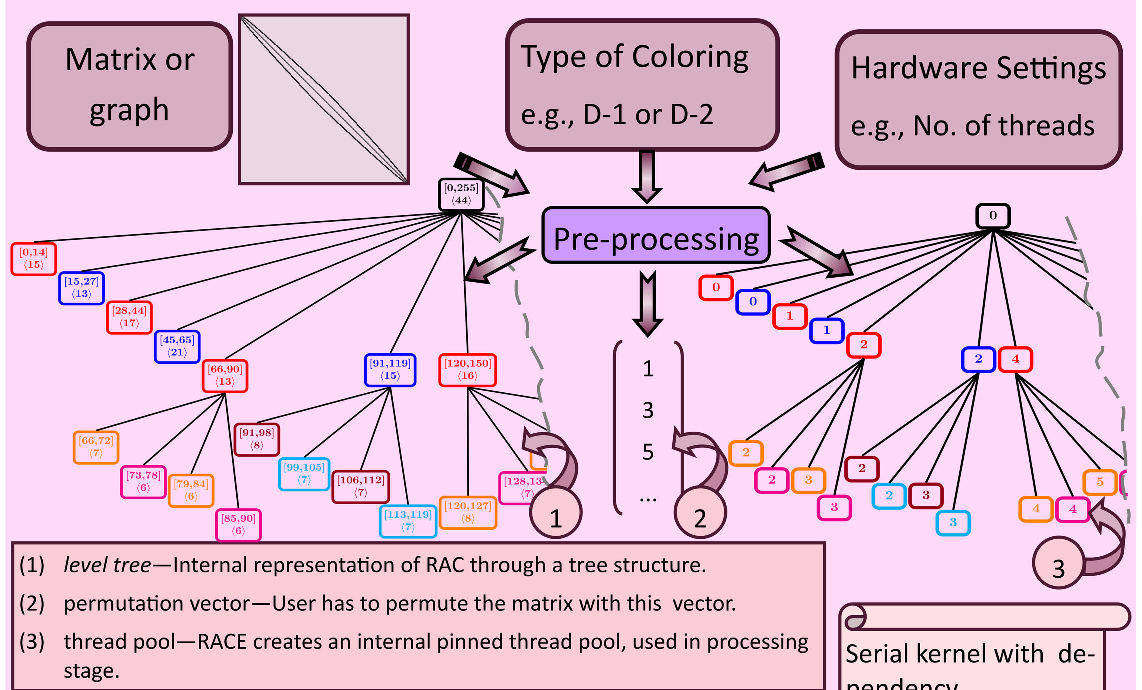


Fig: 2d-7 pt Stencil used as an example for explaining the method and its sparsity pattern.

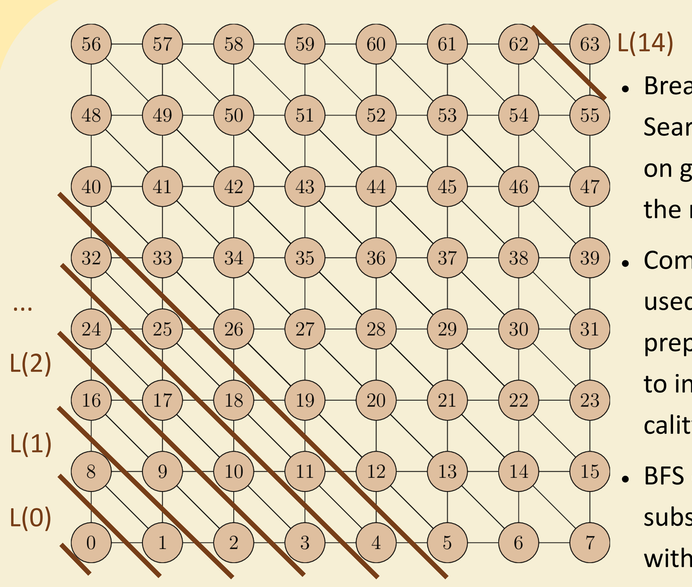
RACE library

Library supports in 2 phases namely pre-processing and processing.



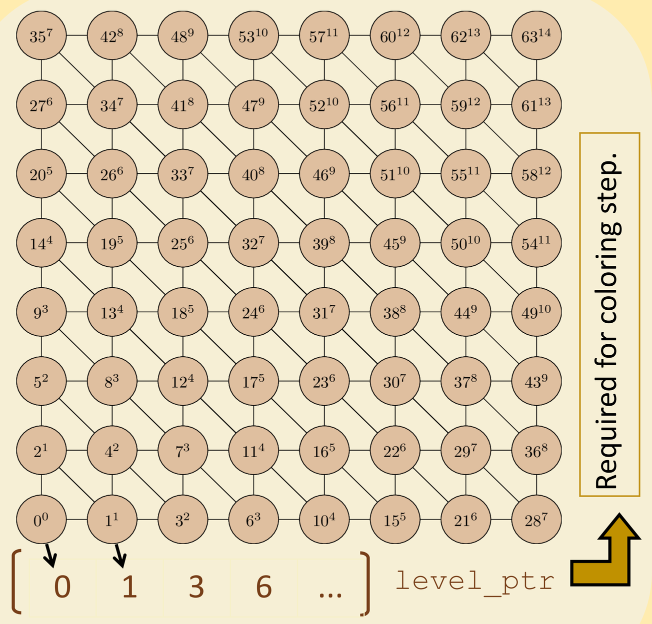
(1) *level tree*—Internal representation of RAC through a tree structure.
(2) permutation vector—User has to permute the matrix with this vector.
(3) thread pool—RACE creates an internal pinned thread pool, used in processing stage.

RACE executes the registered function in parallel with the help of created thread pool and taking care of dependencies specified by *level tree*.

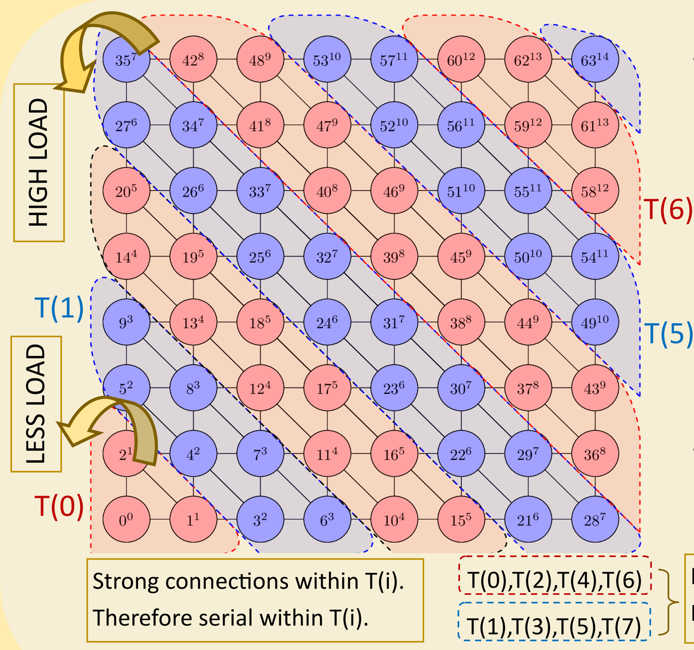


- Breadth First Search (BFS)[7] on graph of the matrix.
- Commonly used in SpMV preprocessing to increase locality
- BFS could be substituted with RCM[8]

Step 1: Level Construction



Step 2: Permutation



- L(i) and L(i±(k+j)) are distance-k independent for all j ≥ 1 .
- Here example of distance-2 (D-2) coloring. Therefore we leave at least 2 levels between.
- As such this is not efficient due to load imbalances.

Step 3: Distance-k coloring

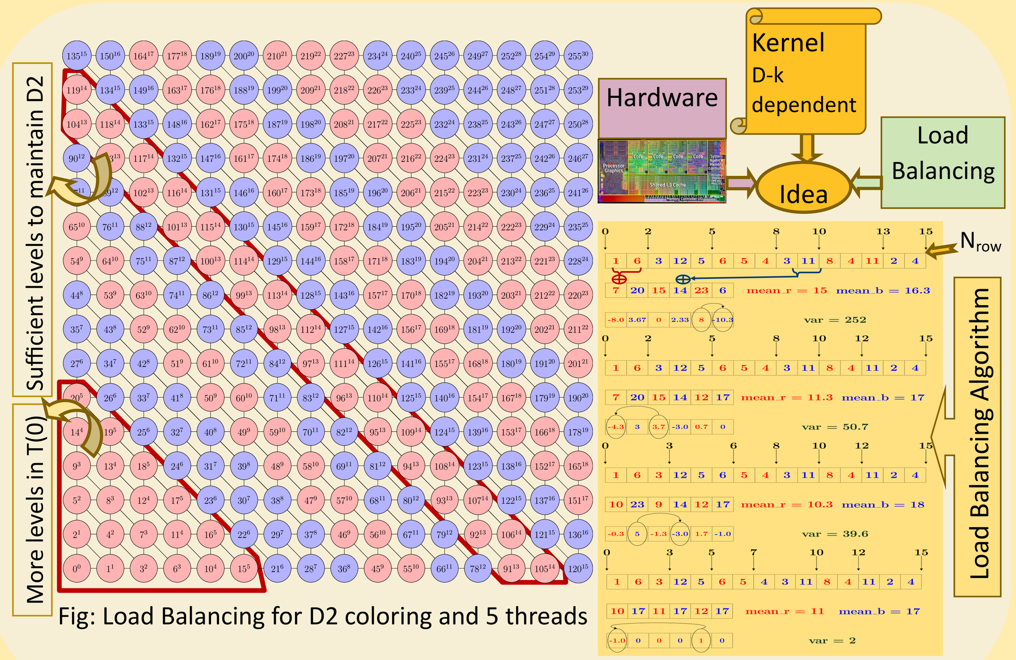
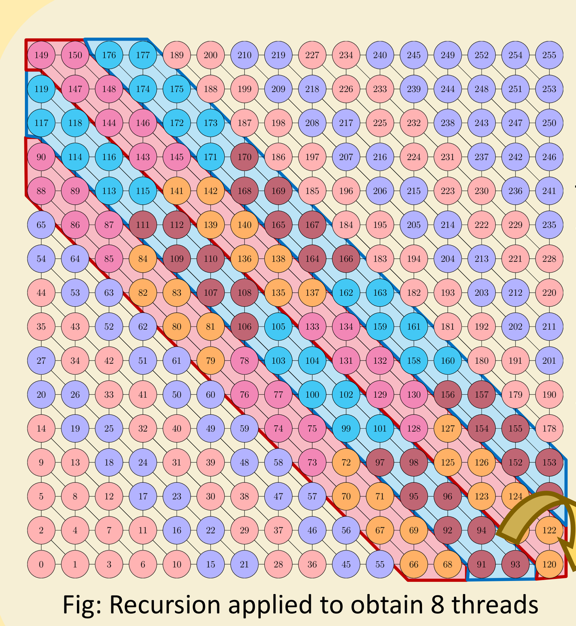


Fig: Load Balancing for D2 coloring and 5 threads

Step 4: Load Balancing



Require more parallelism ? Use recursion.

Procedure

- Select sub-graph based on a global load balancing algorithm.
- Apply all 4 steps recursively on this sub-graph.

Sub-graph recursed for 2 more threads each.

Fig: Recursion applied to obtain 8 threads

Recursion

PERFORMANCE

SymmSpMV

Only upper triangle part of matrix A

```
for(row=0:nrows){
    diag_idx = rowPtr[row];
    b[row] += A[diag_idx]*x[row];

    for (j=rowPtr[row]:rowPtr[row+1])
    {
        b[row] += A[j] * x[col[j]];
        b[col[j]] += A[j] * x[row];
    }
}
```

Data traffic

$$= 8+4+16*\alpha+8*\alpha+\frac{4}{N_{nzr}} \quad \left[\frac{Bytes}{N_{nz}}\right]$$

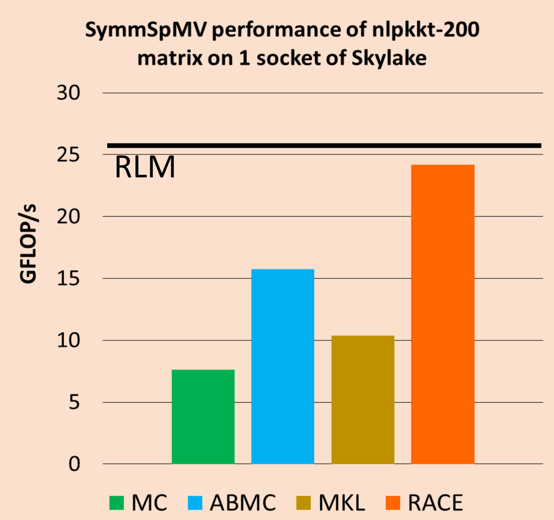
$$= 12+24*\alpha+\frac{4}{N_{nzr}} \; (\alpha = 0.22, \text{from SpMV})$$

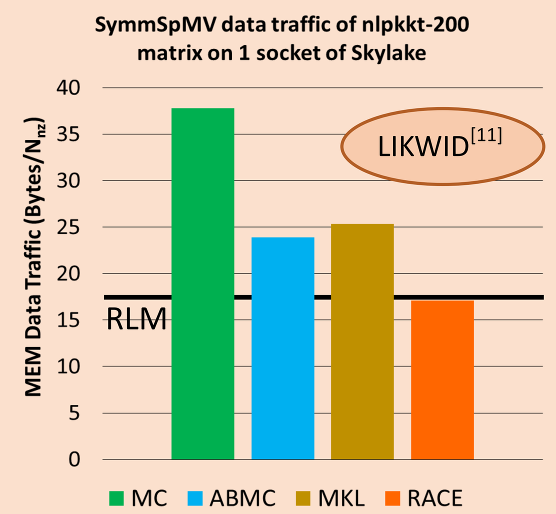
$$= 17.56 \left[\frac{Bytes}{N_{nz}}\right]$$

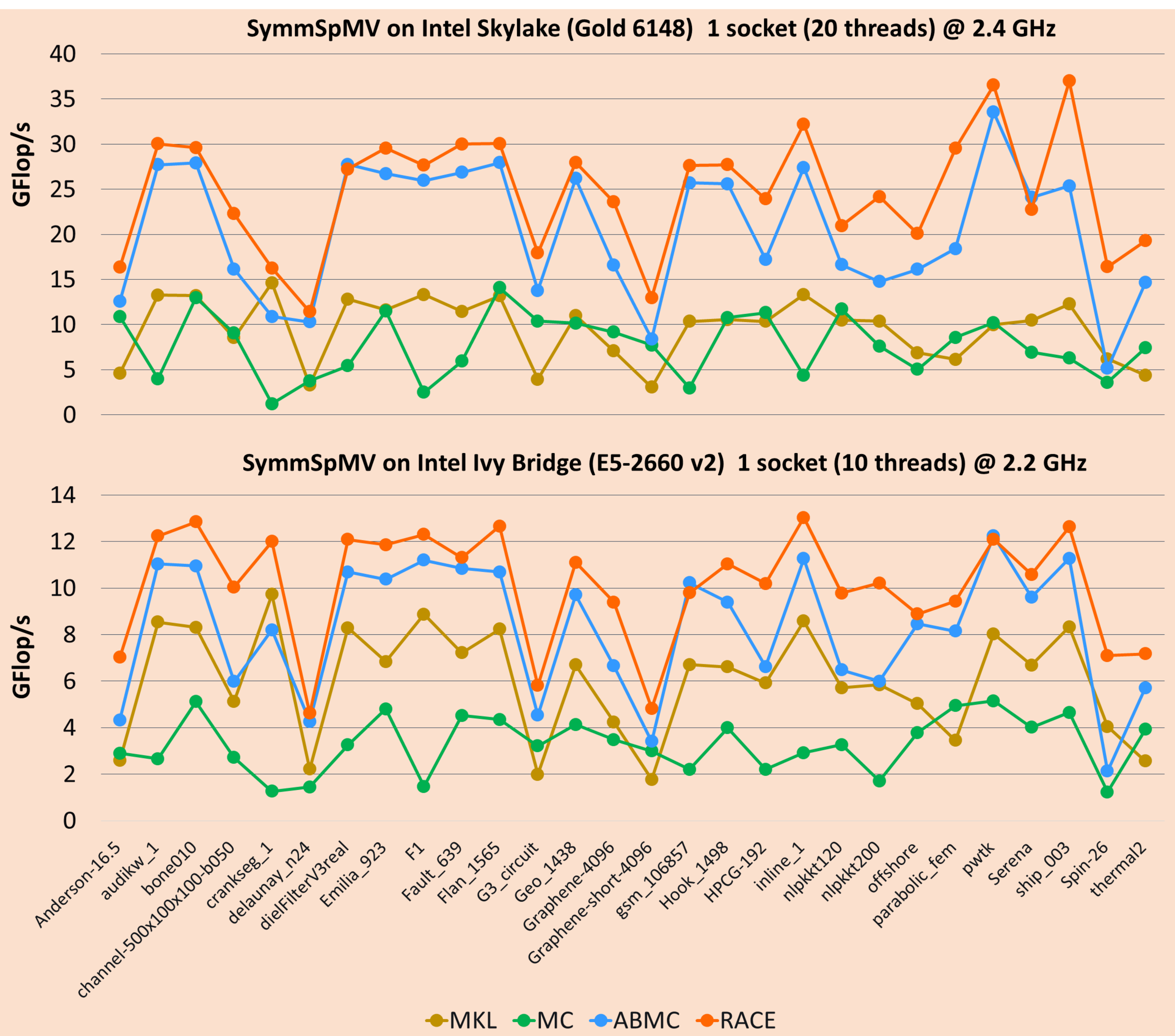
RLM Performance

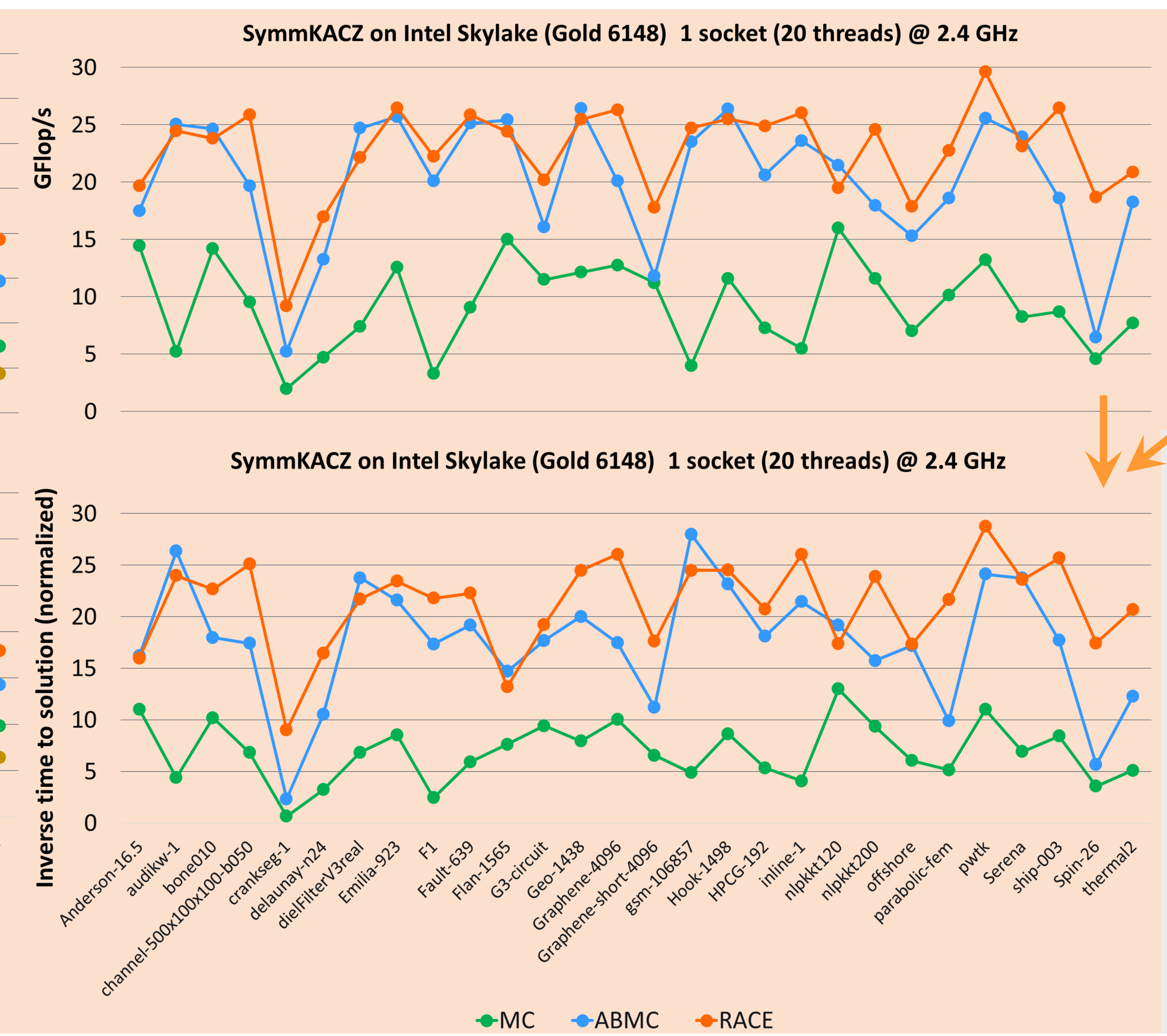
$$= I * b_s \left[\frac{GFlop}{s}\right] \quad \text{bandwidth}, b_s = 115 \left[\frac{GByte}{s}\right] \text{(load)}$$

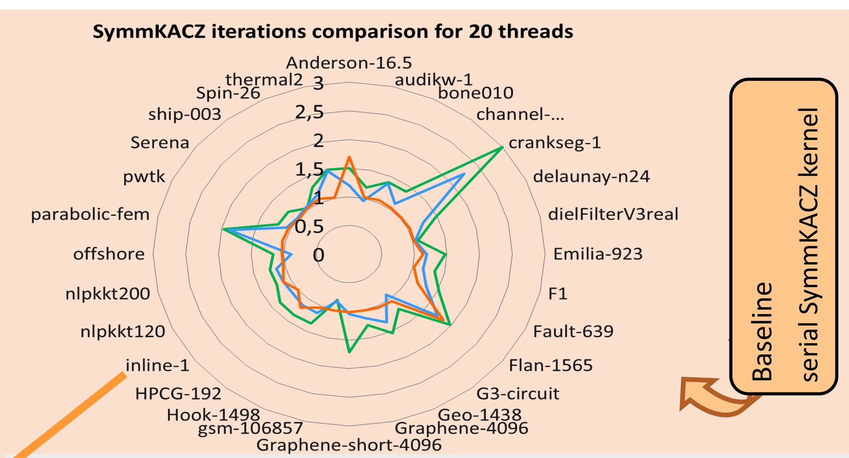
$$= \left(\frac{4}{17.56}\right) * 115 = 26.2 \left[\frac{GFlop}{s}\right]$$











Future Work

- RACE and distributed parallelism (MPI). The ability to derive exactly the amount of parallelism in recursive stages enables efficient distributed parallelism of dependent kernels. Initial calculations show promising results.
- RACE allows to completely overlap computation and communication for kernels like SpMV, benefit of this this has to be tested.
- Efficient vectorization using sparse matrix formats like SELL-C-σ[6].
- Loose synchronization of processing part.
- Embedding KACZ with RACE in FEAST eigenvalue solver[3], enabling large runs for interesting problems like Anderson localization.
- Parallelize pre-processing part of RACE.

REFERENCES

1. A. H. Castro Neto, F. Guinea, N. M. R. Peres, K. S. Novoselov, and A. K. Geim, The electronic properties of graphene, Rev. Mod. Phys., 81 (2009),pp.109-163, https://doi.org/10.1103/RevModPhys.81.109
2. J. Dongarra and M. Heroux, Toward a new metric for ranking high performance computing systems, Tech. Report SAND2013-4744, Sandia National Laboratories, 2013. www.hpcg-benchmark.org
3. M. Galgon, L. Krämer, J. Thies, A. Basermann, and B. Lang, On the parallel iterative solution of linear systems arising in the feast algorithm for computing inner eigenvalues, Parallel Comput., (2015), pp. 153-163, https://doi.org/10.1016/j.parco.2015.06.005.
4. T. Iwashita, H. Nakashima, and Y. Takahashi, Algebraic block multi-color ordering method for parallel multi-threaded sparse triangular solver in iccg method, IPDPS '12, pp. 474-483, https://doi.org/10.1109/IPDPS.2012.51.
5. M.Martone, Efficient Multithreaded Untransposed, Transposed or Symmetric Sparse Matrix-vector Multiplication with the Recursive Sparse Blocks Format, Parallel Comput. 2104, pp. 251-270,https://doi.org/10.1016/j.parco.2014.03.008
6. M. Kreutzer, G. Hager, G. Wellein, H. Fehske, and A. R. Bishop, A Unified Sparse Matrix Data Format for Efficient General Sparse Matrix-Vector Multiplication on Modern Processors with Wide SIMD Units, SIAM Journal on Scientific Computing, 2014,pp. C401-C423, https://doi.org/10.1137/130930352.
7. C. Y. Lee, An algorithm for path connections and its applications, IRE Transactions on Electronic Computers, EC-10 (1961), pp. 346-365, https://doi.org/10.1109/TEC.1961.5219222.
8. E. Cuthill, Several Strategies for Reducing the Bandwidth of Matrices, Springer US, Boston,1972, pp. 157-166, https://doi.org/10.1007/978-1-4615-8675-3_14
9. A. H. Gebremedhin, D. Nguyen, M. M. A. Patwary, and A. Pothen, Colpack: Software for graph coloring and related problems in scientific computing ACM Trans. Math. Softw.,(2013), pp. 1:1-1:31, https://doi.org/10.1145/2513109.2513110.
10. G. Karypis and V. Kumar, A fast and high quality multilevel scheme for partitioning irregular graphs, SIAM Journal on Scientic Computing (1998), pp. 359-392, https://doi.org/10.1137/S1064827595287997.
11. J. Treibig, G. Hager, and G. Wellein, Likwid: A lightweight performance-oriented tool suite for x86 multicore environments, 2010.
12. Equipping Sparse Solvers for Exascale - ESSEX. https://blogs.fau.de/essex/activities.
13. Intel, Intel math kernel library, https://software.intel.com/en-us/mkl
14. SuiteSparse Matrix Collection. https://sparse.tamu.edu/.

RACE Library (AD available)
Multimedia

Thanks for the funding by DFG SPP1648 (SPPEXA) through ESSEX-II project .

SPPEXA