Scalable Algorithms for Tensor Computations

Edgar Solomonik

L P N A @CS@Illinois

Department of Computer Science
University of Illinois at Urbana-Champaign

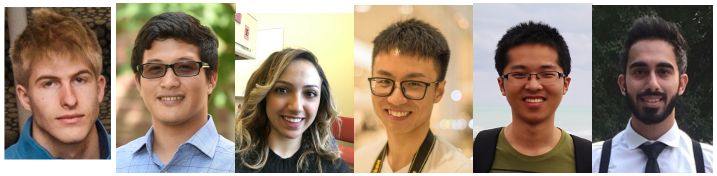
SIAM PP, Seattle WA

Laboratory for Parallel Numerical Algorithms

Recent/ongoing research topics

- parallel matrix computations
  - QR factorization
  - triangular solve
  - eigenvalue problems
- tensor computations
  - tensor decomposition
  - sparse tensor kernels
  - tensor completion
- simulation of quantum systems
  - tensor networks
  - quantum chemistry
  - quantum circuits
- fast bilinear algorithms
  - convolution algorithms
  - tensor symmetry
  - fast matrix multiplication

L P N A @CS@Illinois



DOE CSGF

NSF

http://lpna.cs.illinois.edu

Outline

3D algorithms for matrix computations

For Cholesky factorization with p processors, BSP (critical path) costs are

$$F = \Theta(n^3/p), \quad W = \Theta(n^2/\sqrt{cp}), \quad S = \Theta(\sqrt{cp})$$

using c matrix copies (processor grid is 2D for $c = 1$, 3D for $c = p^{1/3}$).

Achieving similar costs for LU, QR, and the symmetric eigenvalue problem requires algorithmic changes.

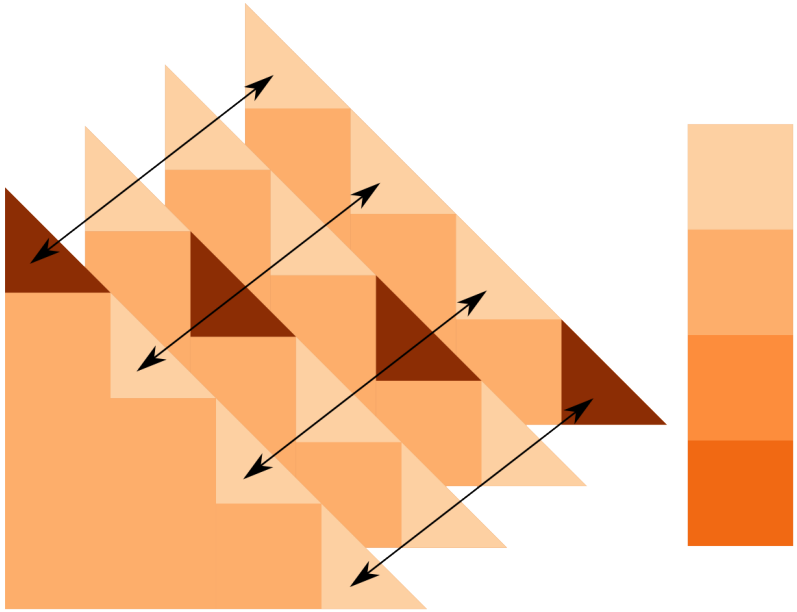
triangular solve	square TRSM ✓[1]	rectangular TRSM ✓[2]
LU with pivoting	pairwise pivoting ✓[3]	tournament pivoting ✓[4]
QR factorization	Givens on square ✓[3]	Householder on rect. ✓[5]
sym. eig.	eigenvalues only ✓[5]	eigenvectors X

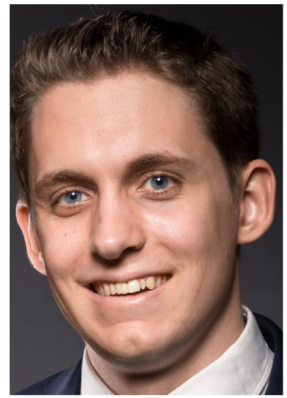
✓means costs attained (synchronization within polylog factors).

[1] B. Lipshitz, MS thesis 2013
[2] T. Wicky, E.S., T. Hoefler, IPDPS 2017
[3] A. Tiskin, FGCS 2007
[4] E.S., J. Demmel, EuroPar 2011
[5] E.S., G. Ballard, T. Hoefler, J. Demmel, SPAA 2017

New algorithms can circumvent lower bounds

For TRSM, we can achieve a lower synchronization/communication cost by performing triangular inversion on diagonal blocks





Tobias Wicky

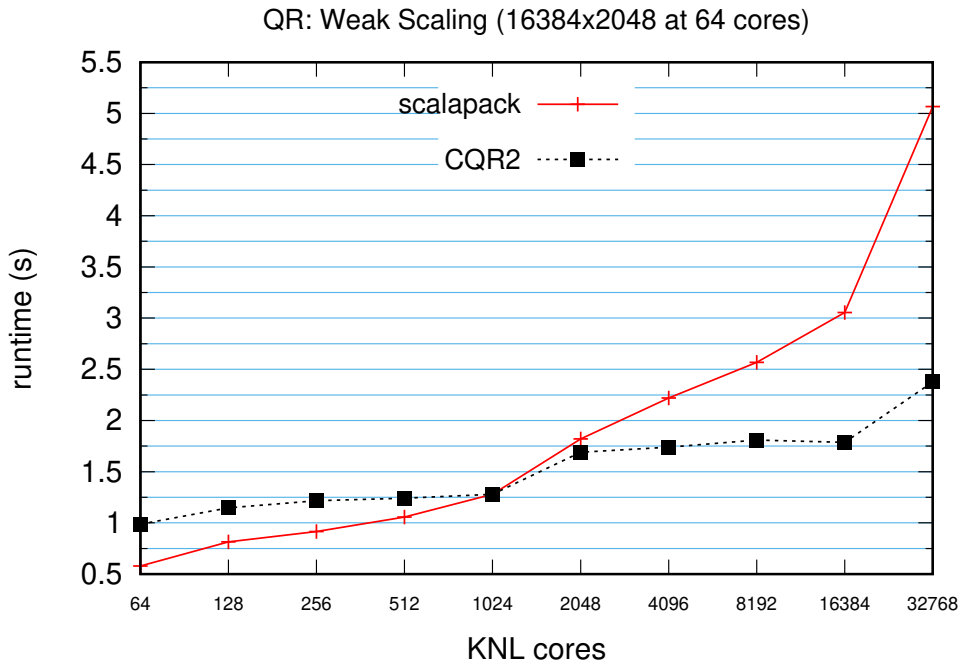
- decreases synchronization cost by $O(p^{2/3})$ on p processors with respect to known algorithms
- optimal communication for any number of right-hand sides
- MS thesis work by Tobias Wicky[1]

[1] T. Wicky, E.S., T. Hoefler, IPDPS 2017

Cholesky-QR2 for rectangular matrices

Cholesky-QR2[1] with 3D Cholesky gives a practical 3D QR algorithm[2]

- Compute $\boldsymbol{A} = \hat{\boldsymbol{Q}}\boldsymbol{R}$ using Cholesky $\boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{R}^T\boldsymbol{R}$
- Correct computed factorization by Cholesky-QR of $\hat{\boldsymbol{Q}}$
- Attains full accuracy so long as $\text{cond}(\boldsymbol{A}) < 1/\sqrt{\epsilon_{\text{mach}}}$



Edward Hutter

[1] T. Fukaya, Y. Nakatsukasa, Y. Yanagisawa, Y. Yamamoto, 2014

[2] E. Hutter, E.S., IPDPS 2019

Tridiagonalization

Reducing the symmetric matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ to a tridiagonal matrix

$$\boldsymbol{T} = \boldsymbol{Q}^T \boldsymbol{A} \boldsymbol{Q}$$

by an orthogonal similarity transformation is most costly in eigenvalue computation (SVD is similar).

- can be done by successive column QR factorizations

$$\boldsymbol{T} = \underbrace{\boldsymbol{Q}_n^T \cdots \boldsymbol{Q}_1^T}_{\boldsymbol{Q}^T} \boldsymbol{A} \underbrace{\boldsymbol{Q}_1 \cdots \boldsymbol{Q}_n}_{\boldsymbol{Q}}$$

- two-sided updates harder to manage than one-sided
- can use n/b QRs on panels of b columns to go to band-width $b+1$
- $b = 1$ gives direct tridiagonalization

Multi-stage tridiagonalization

Writing the orthogonal transformation in Householder form, we get

$$\underbrace{(\boldsymbol{I} - \boldsymbol{U}\boldsymbol{T}\boldsymbol{U}^T)^T}_{\boldsymbol{Q}^T} \boldsymbol{A} \underbrace{(\boldsymbol{I} - \boldsymbol{U}\boldsymbol{T}\boldsymbol{U}^T)}_{\boldsymbol{Q}} = \boldsymbol{A} - \boldsymbol{U}\boldsymbol{V}^T - \boldsymbol{V}\boldsymbol{U}^T$$

where columns of $\boldsymbol{U}$ contains Householder vectors and $\boldsymbol{V}$ is

$$\boldsymbol{V}^T = \boldsymbol{T}\boldsymbol{U}^T + \frac{1}{2}\boldsymbol{T}^T\boldsymbol{U}^T \underbrace{\boldsymbol{A}\boldsymbol{U}}_{\text{bottleneck}} \boldsymbol{T}\boldsymbol{U}^T$$

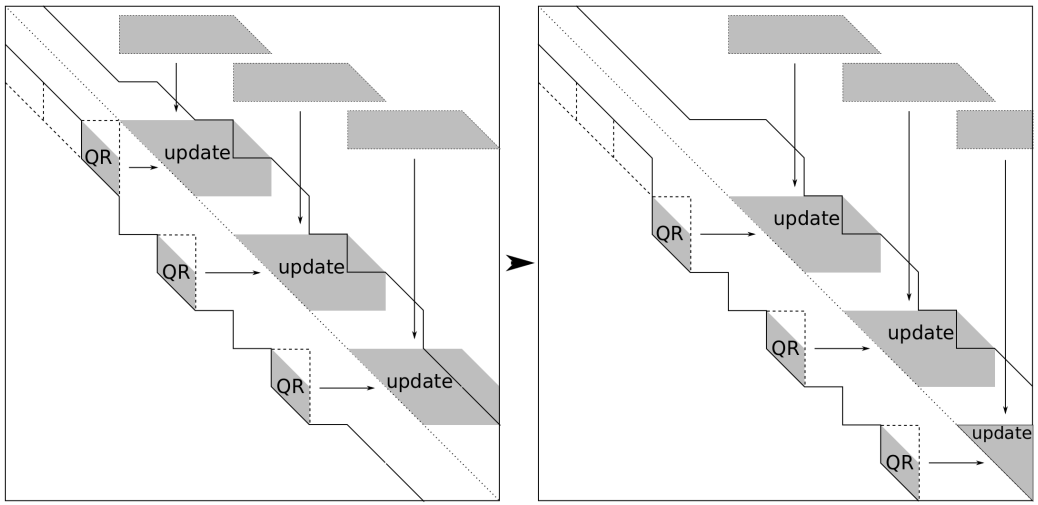
- if $b = 1$, $\boldsymbol{U}$ is a column-vector, and $\boldsymbol{A}\boldsymbol{U}$ is dominated by vertical communication cost (moving $\boldsymbol{A}$ between memory and cache)
- idea: reduce to banded matrix ($b \gg 1$) first[1]

[1] T. Auckenthaler, H. Bungartz, T. Huckle, L. Krämer, B. Lang, P. Willems 2011

Successive band reduction (SBR)

After reducing to a banded matrix, we need to transform the banded matrix to a tridiagonal one

- fewer nonzeros lead to lower computational cost, $F = O(n^2 b/p)$
- however, transformations introduce fill/bulges
- bulges must be chased down the band[1]



- communication- and synchronization-efficient 1D SBR algorithm known for small band-width[2]

[1] B. Lang 1993; C.H. Bischof, B. Lang, X. Sun 2000

[2] G. Ballard, J. Demmel, N. Knight 2012

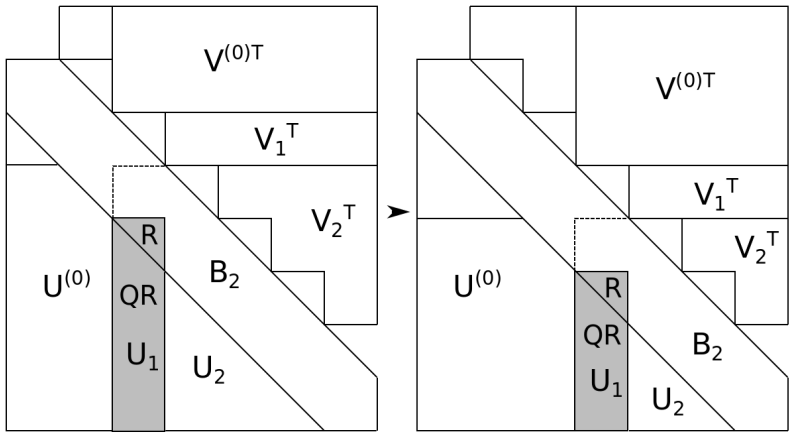
Communication-efficient eigenvalue computation

Previous work (start-of-the-art): two-stage tridiagonalization

- implemented in ELPA, can outperform ScaLAPACK[1]
- with $n = n/\sqrt{p}$, 1D SBR gives[2] $W = O(n^2/\sqrt{p})$, $S = O(\sqrt{p}\log^2(p))$

New results[3]: many-stage tridiagonalization

- $\Theta(\log(p))$ intermediate band-widths to achieve $W = O(n^2/p^{2/3})$
- communication-efficient rectangular QR with processor groups



- 3D SBR (each QR and matrix multiplication update parallelized)

[1] T. Auckenthaler, H. Bungartz, T. Huckle, L. Krämer, B. Lang, P. Willems 2011

[2] G. Ballard, J. Demmel, N. Knight 2012

[3] E.S., G. Ballard, J. Demmel, T. Hoefler 2017

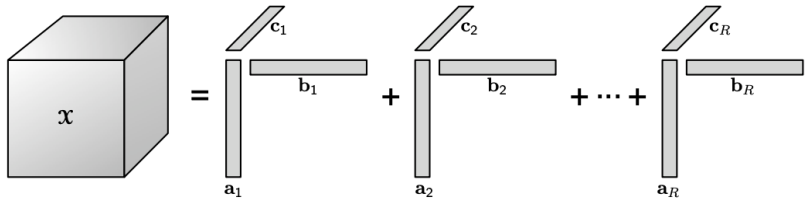
Symmetric eigensolver results summary

Algorithm	W	Q	S
ScaLAPACK	$n^2/\sqrt{p}$	n^3/p	$n\log(p)$
ELPA	$n^2/\sqrt{p}$	-	$n\log(p)$
two-stage + 1D-SBR	$n^2/\sqrt{p}$	$n^2\log(n)/\sqrt{p}$	$\sqrt{p}(\log^2(p)+\log(n))$
many-stage	$n^2/p^{2/3}$	$n^2\log(p)/p^{2/3}$	$p^{2/3}\log^2 p$

- costs are asymptotic (same computational cost F for eigenvalues)
- W – horizontal (interprocessor) communication
- Q – vertical (memory–cache) communication excluding $W + F/\sqrt{H}$ where H is cache size
- S – synchronization cost (number of supersteps)

Tensor Decompositions

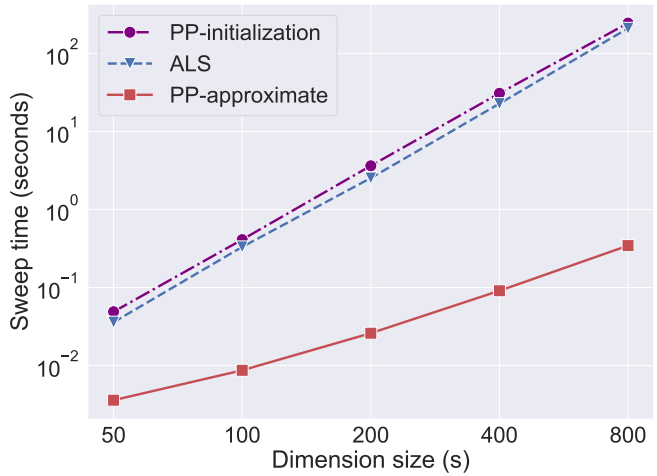
- Tensor of order N has N modes and dimensions $s \times \cdots \times s$
- Canonical polyadic (CP) tensor decomposition[1]

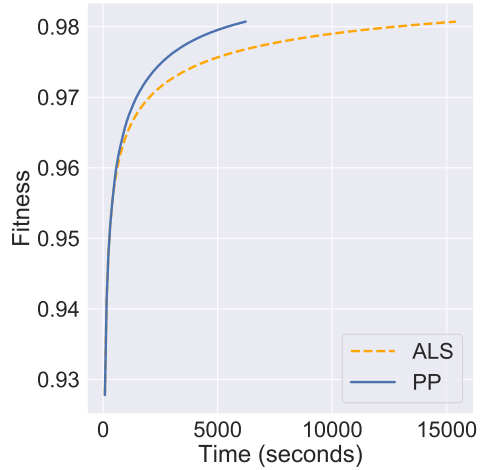


- Alternating least squares (ALS) is most widely used method
  - Monotonic linear convergence
- Gauss-Newton method is an emerging alternative
  - Non-monotonic, but can achieve superlinear convergence rate

[1] T.G. Kolda and B.W. Bader, SIAM Review 2009

Accelerating Alternating Least Squares





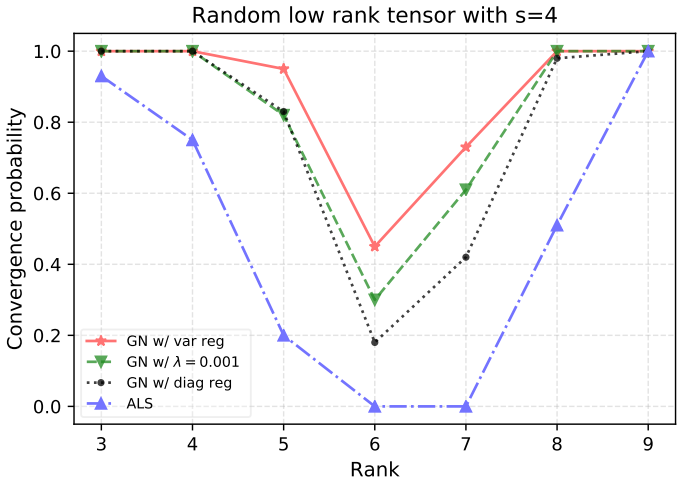
New algorithm: pairwise perturbation (PP)[1] approximates ALS

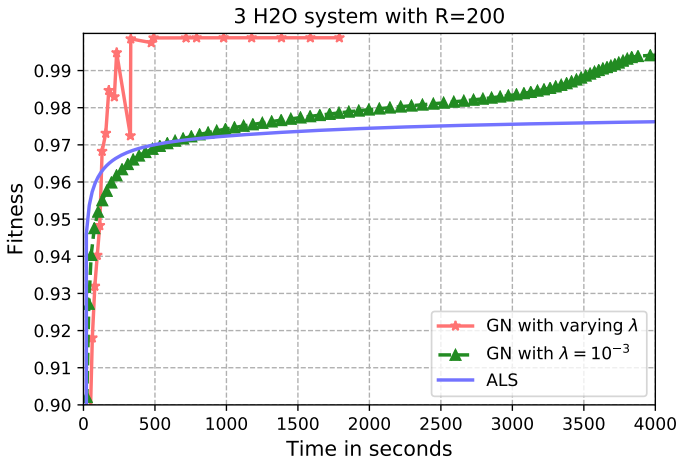
- accurate when ALS sweep over factor matrices stagnates
- rank $R < s^{N-1}$ CP decomposition:
  - ALS sweep cost $O(s^N R) \Rightarrow O(s^2 R)$, up to 600x speed-up
- rank $R < s$ Tucker decomposition:
  - ALS sweep cost $O(s^N R) \Rightarrow O(s^2 R^{N-1})$

Linjian Ma

[1] L. Ma, E.S. arXiv:1811.10573

Regularization and Parallelism for Gauss-Newton





New regularization scheme[1] for Gauss-Newton CP with implicit CG[2]

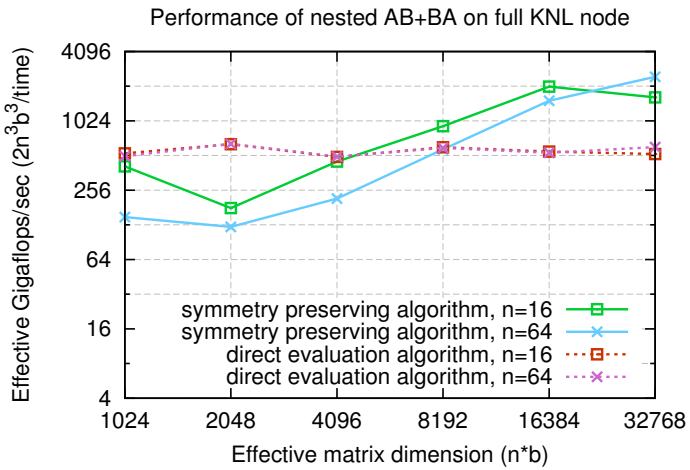
- Oscillates regularization parameter geometrically between lower and upper thresholds
- Achieves higher convergence likelihood
- More accurate than ALS in applications
- Faster than ALS sequentially and in parallel

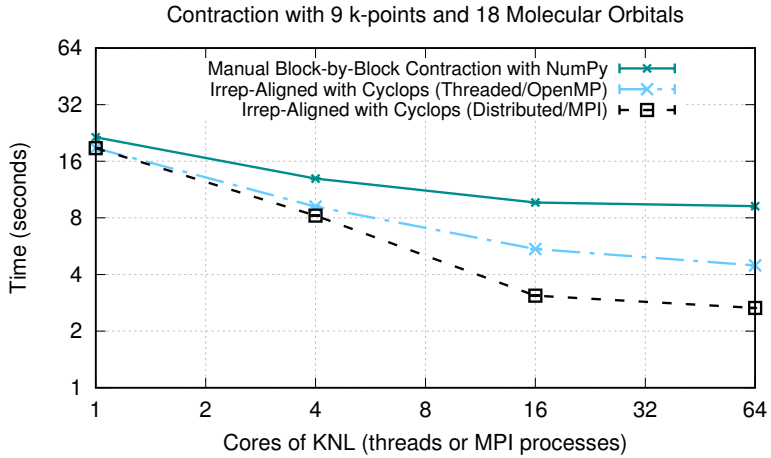
Navjot Singh

[1] Navjot Singh, Linjian Ma, Hongru Yang, and E.S. arXiv:1910.12331

[2] P. Tichavsky, A. H. Phan, and A. Cichocki., 2013

Symmetry in Tensor Contractions

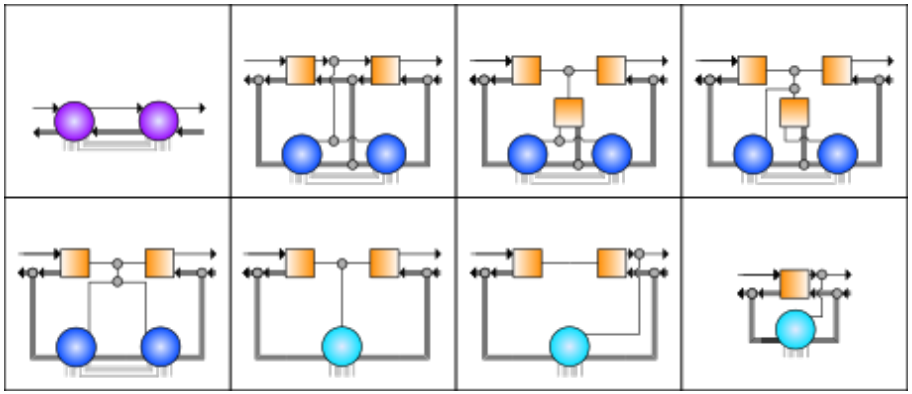




New algebraic transformations for contractions to exploit permutational symmetry[1] and group symmetry[2], prevalent in quantum chemistry/physics

$$\boldsymbol{C} = \boldsymbol{AB} + \boldsymbol{BA} \Rightarrow$$

$$c_{ij} = \sum_k [(a_{ij} + a_{ik} + a_{jk}) \cdot (b_{ij} + b_{ik} + b_{jk})] - \ldots$$



[1] E.S, J. Demmel, CMAM 2020

[2] collaboration with Yang Gao, Phillip Helms, and Garnet Chan at Caltech

Library for Massively-Parallel Tensor Computations

Cyclops Tensor Framework[1] sparse/dense generalized tensor algebra

- Cyclops is a C++ library that distributes each tensor over MPI
- Used in chemistry (PySCF, QChem)[2], quantum circuit simulation (IBM/LLNL)[3], and graph analysis (betweenness centrality)[4]
- Summations and contractions specified via Einstein notation

```
E["aixbjy"] += X["aixbjy"] - U["abu"]*V["iju"]*W["xyu"]
```

- Best distributed contraction algorithm selected at runtime via models
- Support for Python (`numpy.ndarray` backend), OpenMP, and GPU
- Simple interface to core ScaLAPACK matrix factorization routines

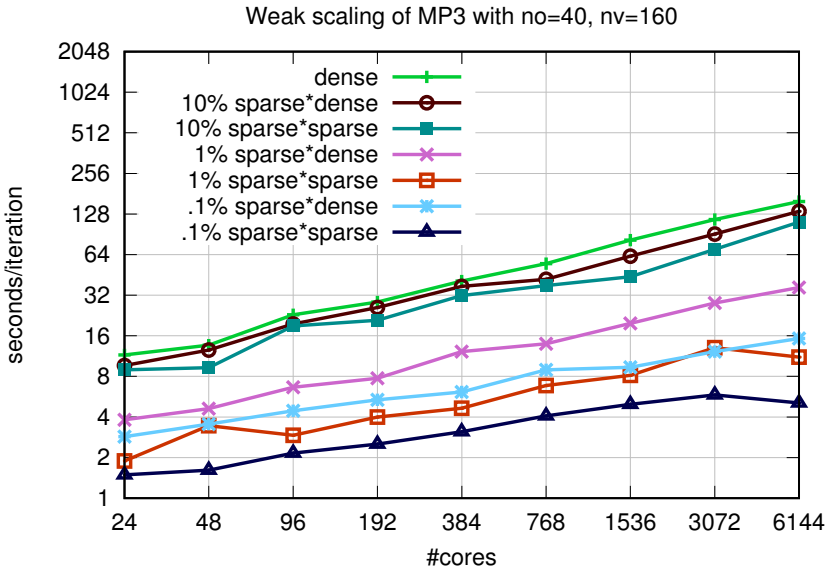
[1] https://github.com/cyclops-community/ctf

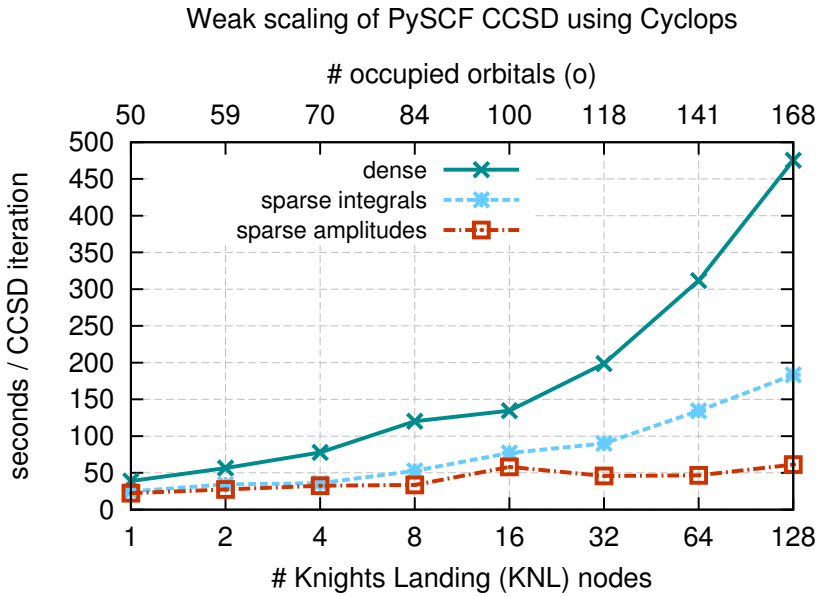
[2] E.S., D. Matthews, J. Hammond, J.F. Stanton, J. Demmel, JPDC 2014

[3] E. Pednault, J.A. Gunnels, G. Nannicini, L. Horesh, T. Magerlein, E. S., E. Draeger, E. Holland, and R. Wisnieff, 2017

[4] E.S., M. Besta, F. Vella, T. Hoefler, SC 2017

Sparsity in Tensor Contractions





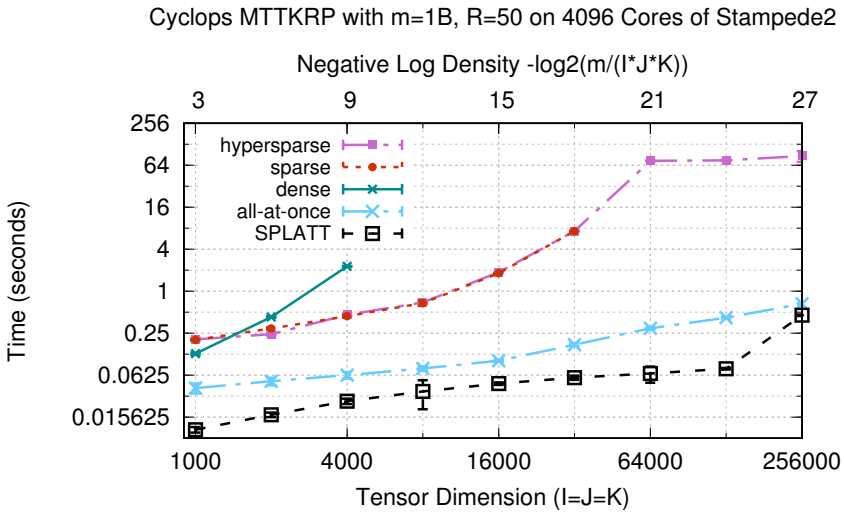
- Cyclops supports sparse representation of tensors[1]
- Choice of representation specified in tensor constructor
- CSR or DCSR[2] (2-index CSF[3]) representation used locally for contractions

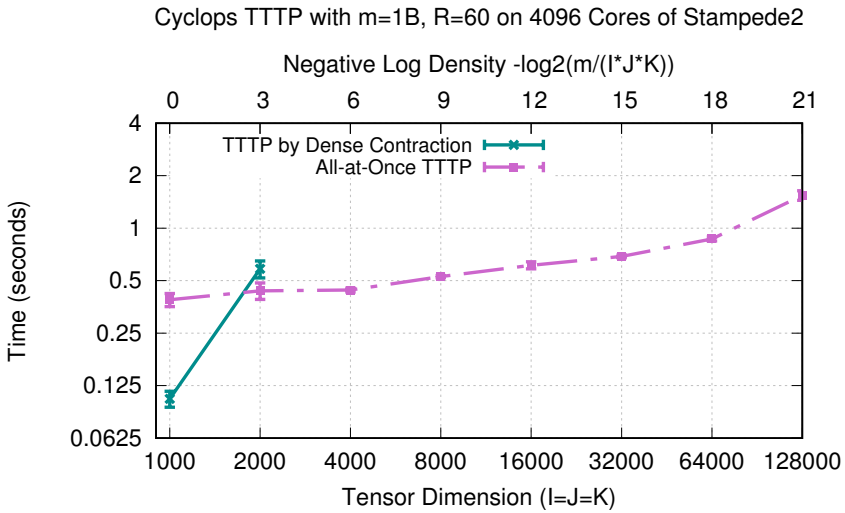
[1] E.S., T. Hoefler 2015

[2] A. Bulúc, J.R. Gilbert, 2008

[3] S. Smith, G. Karypis 2015

All-at-Once Multi-Tensor Contraction





With sparsity, all-at-once contraction[1] of multiple tensors can be faster[2].

- Sparse tensor CP decomposition is dominated by MTTKRP

$$u_{ir} = \sum_{j,k} t_{ijk} v_{jr} w_{kr}$$

- All-at-once sparse MTTKRP needs less communication than pairwise

- Tensor times tensor product (TTTP) enables sparse residual and CP tensor completion

$$r_{ijk} = \sum_{r} t_{ijk} u_{ir} v_{jr} w_{kr}$$

- Cost and memory footprint reduced asymptotically

[1] S. Smith, J. Park, G. Karypis, 2018

[2] Zecheng Zhang, Xiaoxiao Wu, Naijing Zhang, Siyuan Zhang, and E.S. arXiv:1910.02371

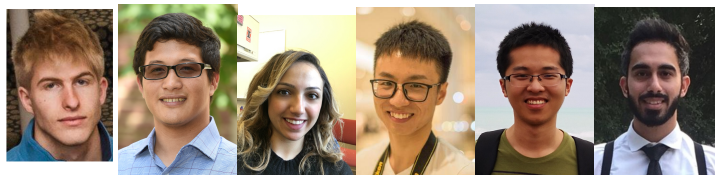
Conclusion

- Communication avoiding matrix factorizations
  - 3D algorithms move $O(p^{1/6})$ less data on p processors
  - Faster algorithms for TRSM, Cholesky, LU, QR, and symmetric eigensolve
- Optimization methods for tensor decomposition
  - Pairwise perturbation approximates ALS sweeps with asymptotically less cost
  - Gauss-Newton methods with new regularization scheme more effective than ALS
- Algorithms and software for sparse/symmetric tensors
  - Algebraic reorganizations to exploit permutational and group symmetry
  - Cyclops library for dense/sparse generalized distributed tensor algebra in C++/Python

Acknowledgements

- Laboratory for Parallel Numerical Algorithms (LPNA) at University of Illinois
- Key external collaborators: James Demmel, Torsten Hoefler, Garnet Chan
- NSF OAC CSSI funding (award #1931258)
- DOE Computational Science Graduate Fellowship
- Stampede2 resources at TACC via XSEDE

LPNA @CS@Illinois



DOE CSGF NSF

http://lpna.cs.illinois.edu