# New Methods and Software for Large-Scale Tensor Computations

Edgar Solomonik

L P N A @CS@Illinois

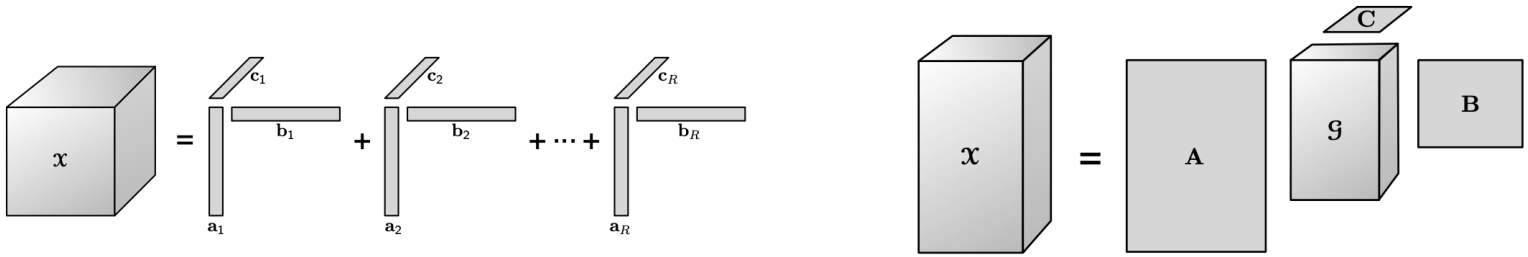
Department of Computer Science
University of Illinois at Urbana-Champaign

SIAM CSE

# Outline

# CP Tensor Decomposition Algorithms

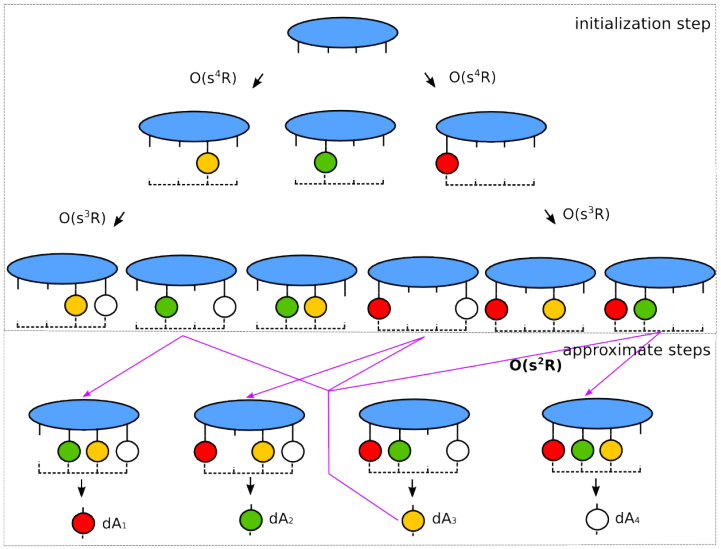
- Tensor of order $N$ has $N$ modes and dimensions $s \times \cdots \times s$
- CP and Tucker tensor decompositions[1]

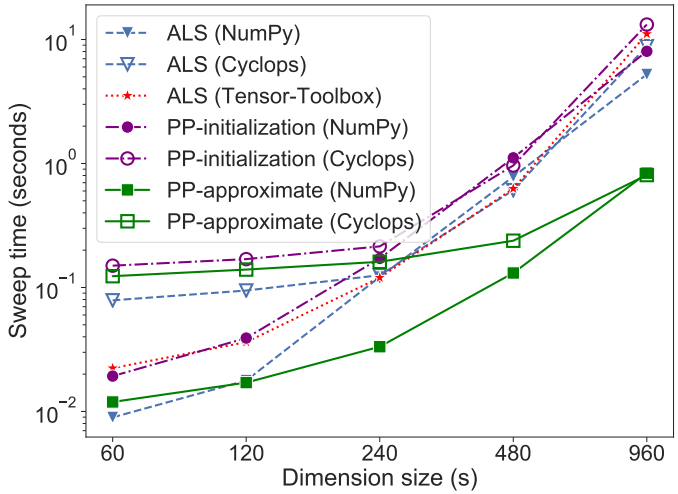



- Alternating least squares (ALS) is most widely used method
  - Optimize one factor matrix at a time, yielding quadratic optimization subproblems
  - Achieves monotonic linear convergence
- Gauss-Newton method is an emerging alternative
  - Optimizes all factor matrices at once by quadratic approximation of nonlinear objective function
  - Non-monotonic, but can achieve quadratic convergence

[1] Kolda and Bader, SIAM Review 2009

# Pairwise Perturbation Algorithm



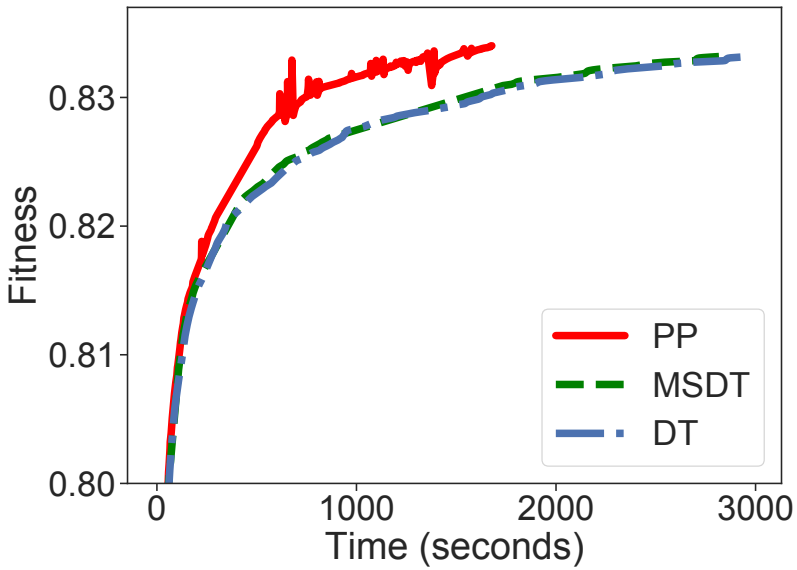

New algorithm: pairwise perturbation (PP)[1] approximates ALS

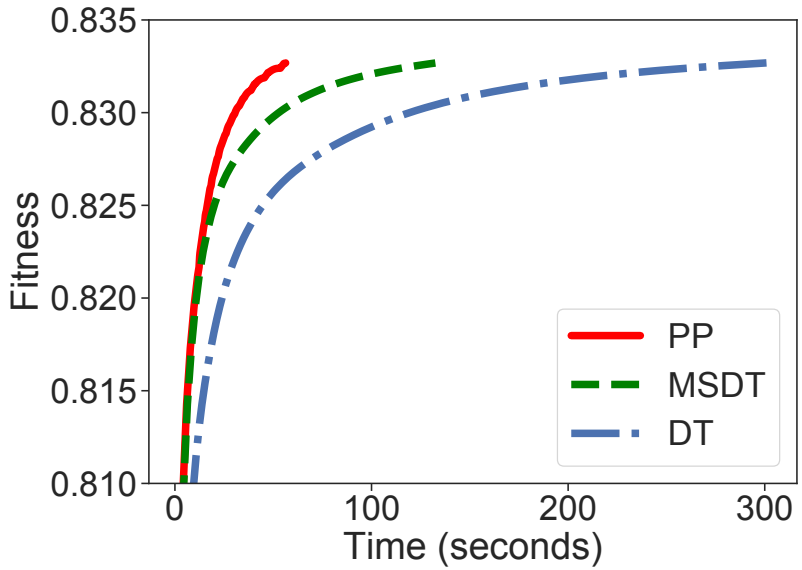
- based on perturbative expansion of ALS update to approximate MTTKRP
- approximation is accurate when ALS updates stagnate
- rank $R < s^{N-1}$ CP decomposition:
  - ALS sweep cost $O(s^N R) \Rightarrow O(s^2 R)$, up to 33x speed-up

Linjian Ma

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

# Parallel Pairwise Perturbation Algorithm



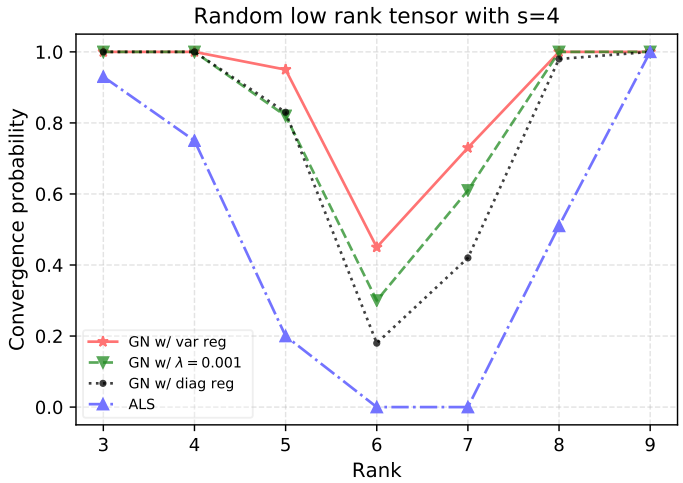

Effective parallelization by decomposing MTTKRP into local MTTKRPs [1]

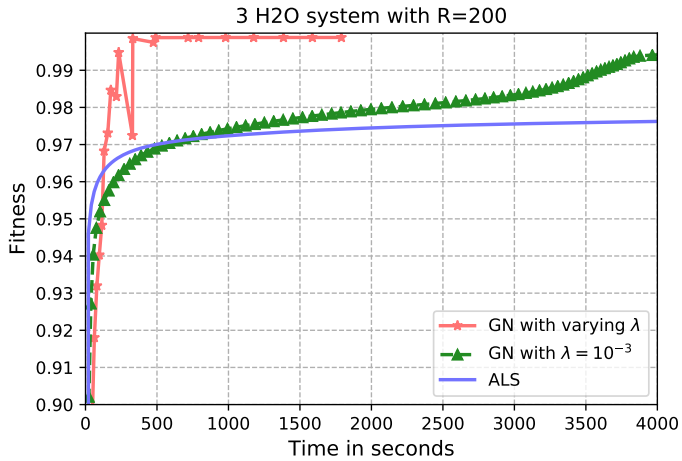
$$\boldsymbol{U} = \text{MTTKRP}(\boldsymbol{\mathcal{T}}, \boldsymbol{V}, \boldsymbol{W}) \Rightarrow \boldsymbol{U}_i = \sum_{j,k} \text{MTTKRP}(\boldsymbol{\mathcal{T}}_{ijk}, \boldsymbol{V}_j, \boldsymbol{W}_k)$$

- processor $(i, j, k)$ owns $\boldsymbol{\mathcal{T}}_{ijk}$, $\boldsymbol{V}_j$, and $\boldsymbol{W}_k$
- pairwise perturbation can be used to approximate local MTTKRPs
- multi-sweep dimension-tree (MSDT) amortizes terms across sweeps

[1] Linjian Ma, E.S. IPDPS 2021

# Regularization and Parallelism for Gauss-Newton




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

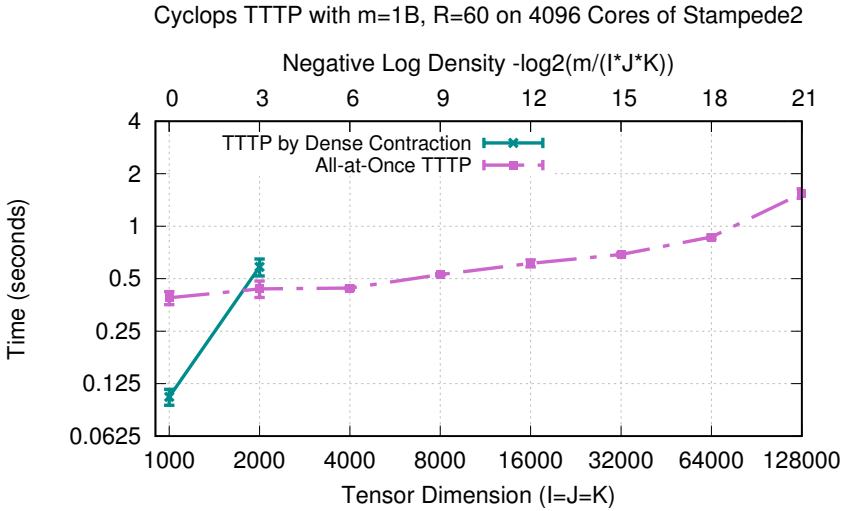
- Oscillates regularization parameter geometrically between lower and upper thresholds
- Achieves higher convergence likelihood
- More accurate than ALS in applications
- Faster than ALS sequentially and in parallel

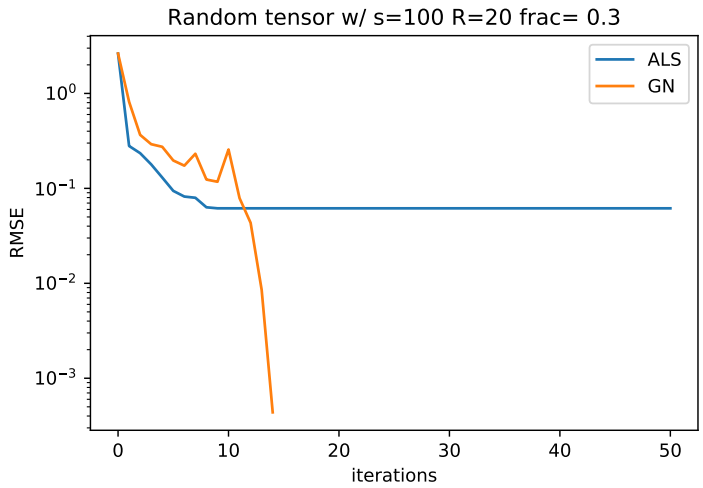
Navjot Singh

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

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

# Tensor Completion




- Tensor times tensor product (TTTP) enables CP tensor completion

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

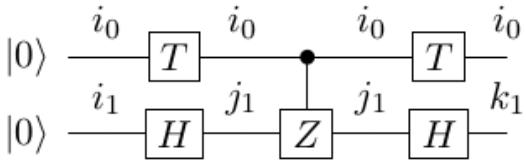
- For ALS, explicit parallel direct solves[1] are fastest
- Via the Cyclops Python interface, we have implemented parallel (over MPI) completion with SGD, CCD, ALS (with iterative and direct solves), and Gauss-Newton, with support for generalized loss[2]

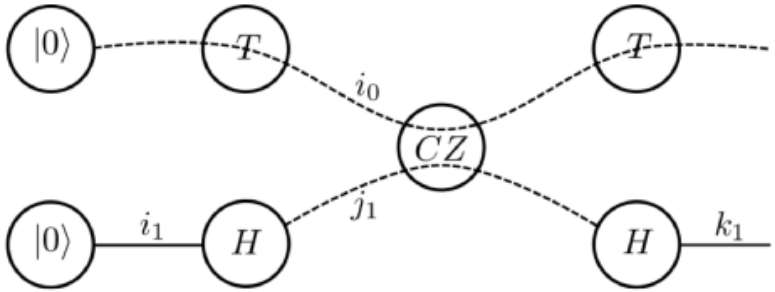
[1] Shaden Smith, Jongsoo Park, and George Karypis, 2016

[2] Navjot Singh, Zecheng Zhang, Xiaoxiao Wu, Naijing Zhang, Siyuan Zhang, and Edgar Solomonik arXiv:1910.02371

# Quantum Circuit Simulation with Tensor Networks

- A quantum circuit is a direct description of a tensor network[1]

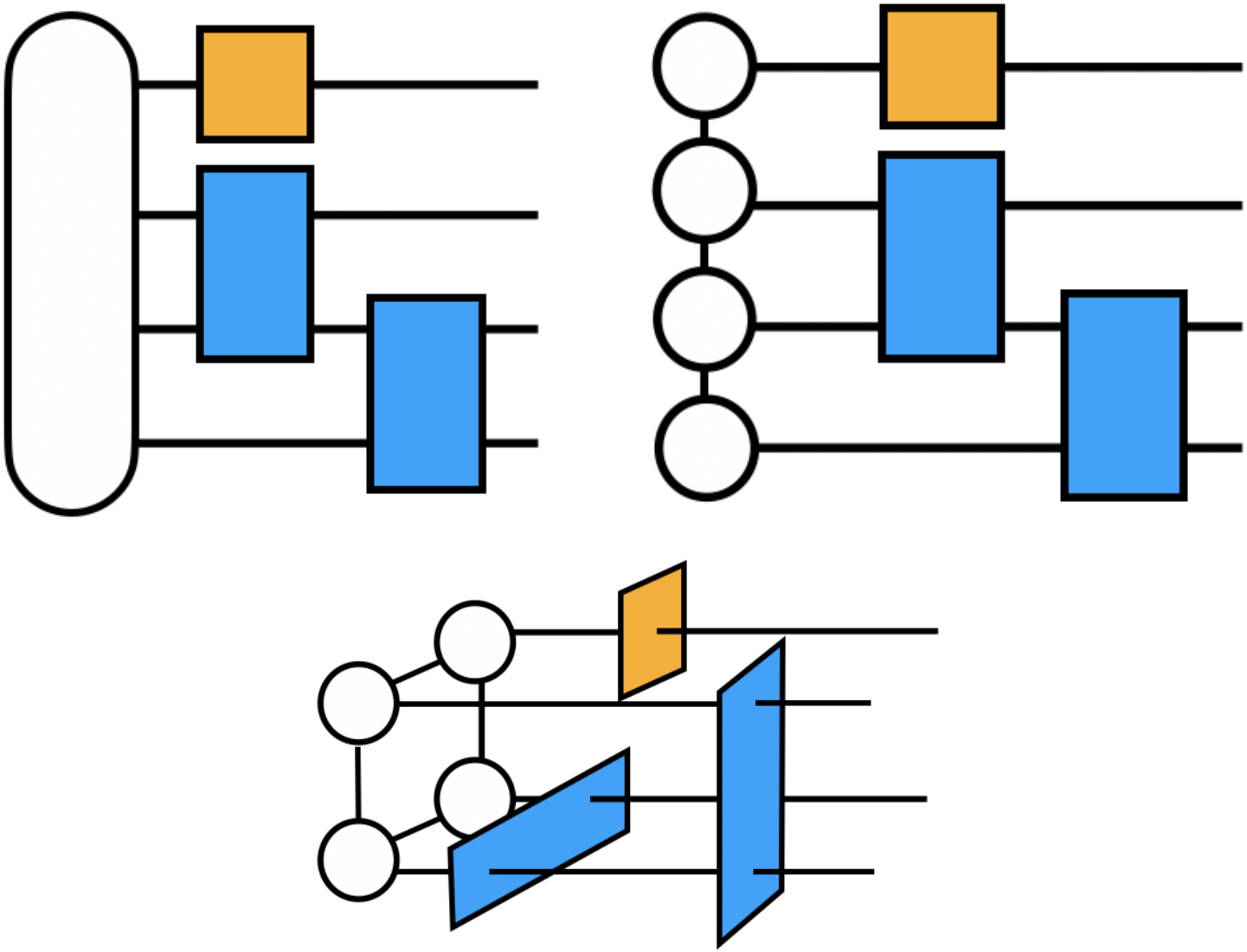



- Why use HPC to (approximately) simulate quantum circuits?
  - enable development/testing/tuning of larger quantum circuits
  - understand approximability of different quantum algorithms
  - quantify sensitivity of algorithms to noise/error
  - potentially enable new hybrid quantum-classical algorithms

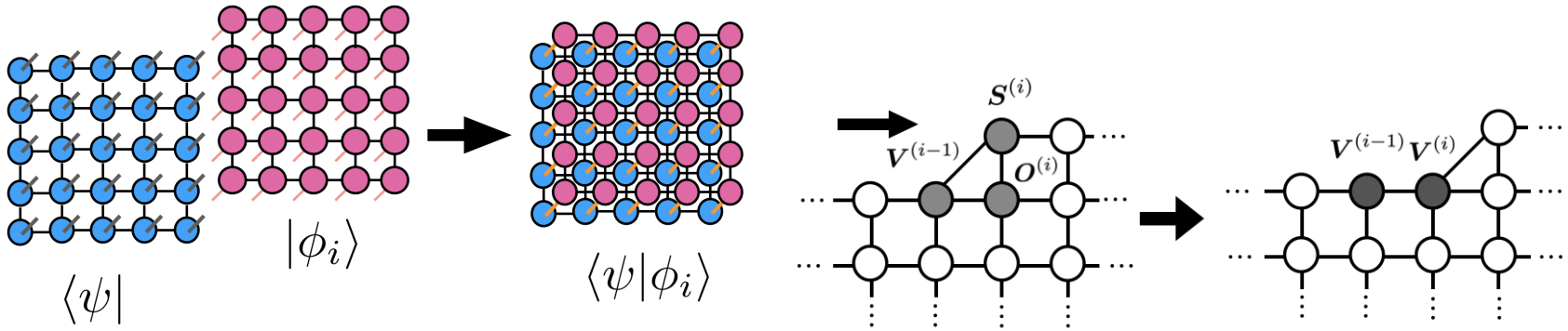
[1] Markov and Shi SIAM JC 2007

# Tensor Network State Simulation



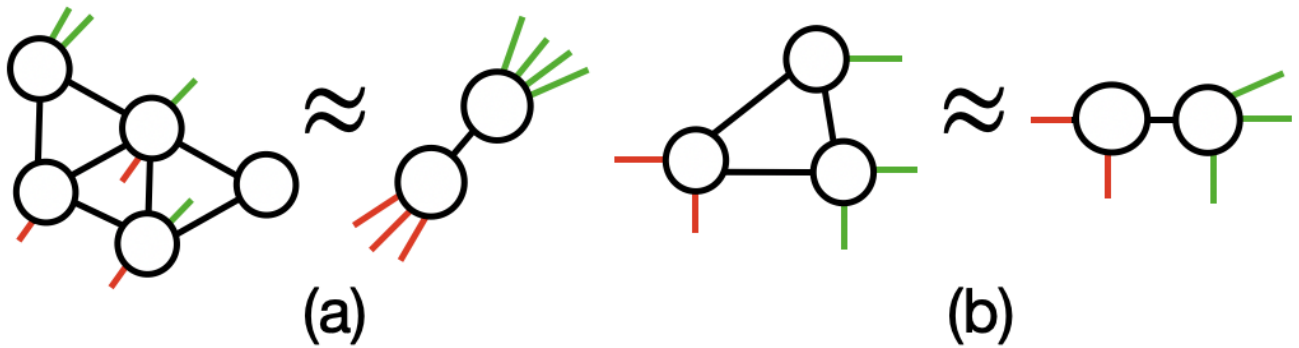
# PEPS Contraction

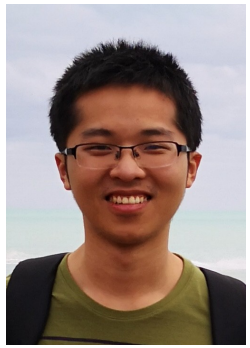
- Exact contraction of PEPS is #P-complete, so known methods have exponential cost in the number of sites
- PEPS contraction is needed to compute expectation values
- *Boundary contraction* is common for finite PEPS and can be simplified with `einsumsvd`



# Implicit Randomized `einsumsvd`

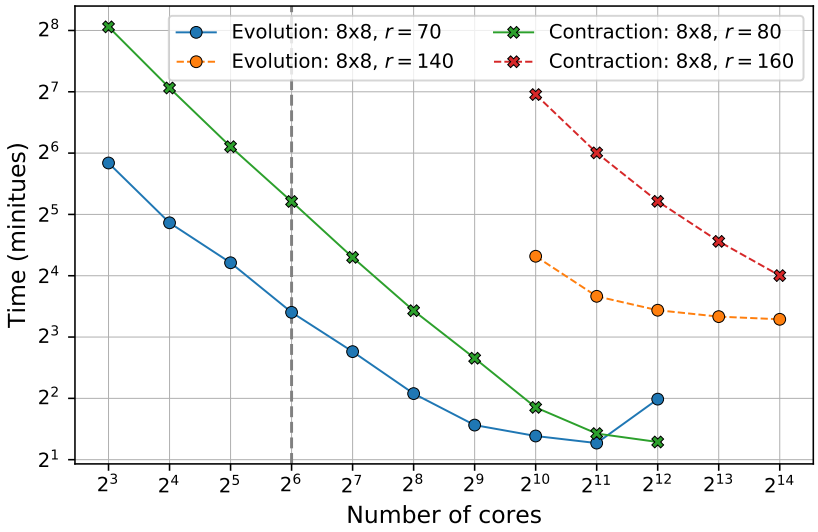
- The `einsumsvd` primitive provides an effective abstraction for tensor network simulation methods

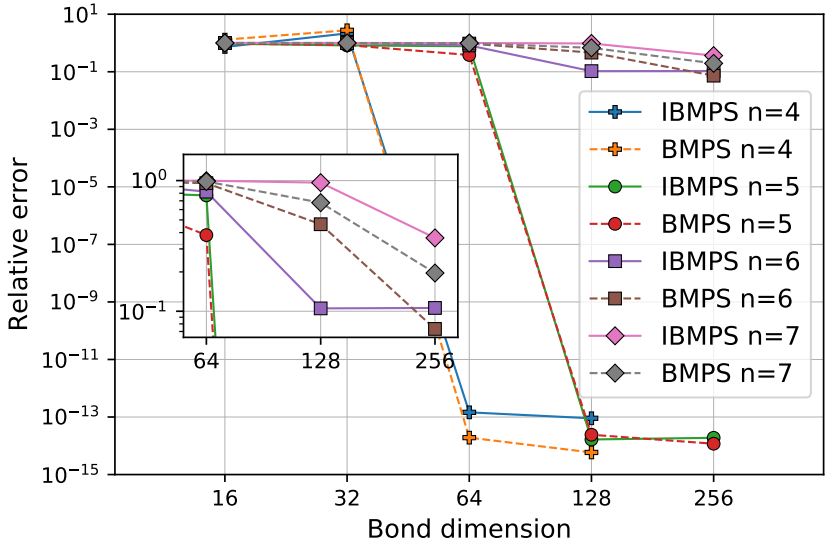


- Alternative algorithms:
  - contract then SVD
  - perform randomized SVD with implicit matrix–matrix products
  - perform QR factorization of operands and do einsumsvd on $R$ factors

# PEPS Benchmark Performance



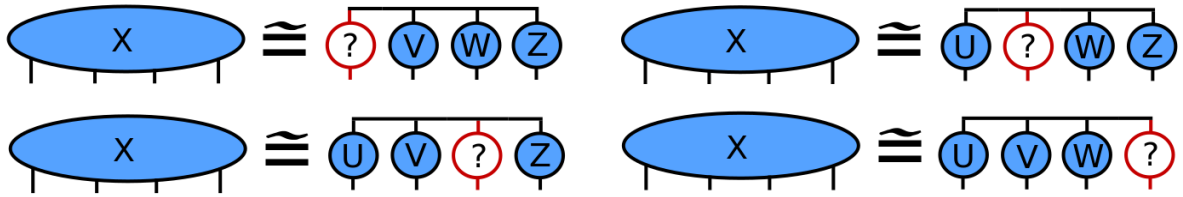

- We introduce a new library, Koala[1], for high-performance simulation of quantum circuits and time evolution with PEPS[2]
- Koala achieves good parallel scalability for approximate gate application (evolution) and contraction
- Approximation can be effective even for adversarially-designed circuits such as Google's random quantum circuit model (figure on right)

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

[2] Yuchen Pang, Tianyi Hao, Annika Dugad, Yiqing Zhou, and E.S. SC 2020

# Automatic Differentiation for Tensor Computations

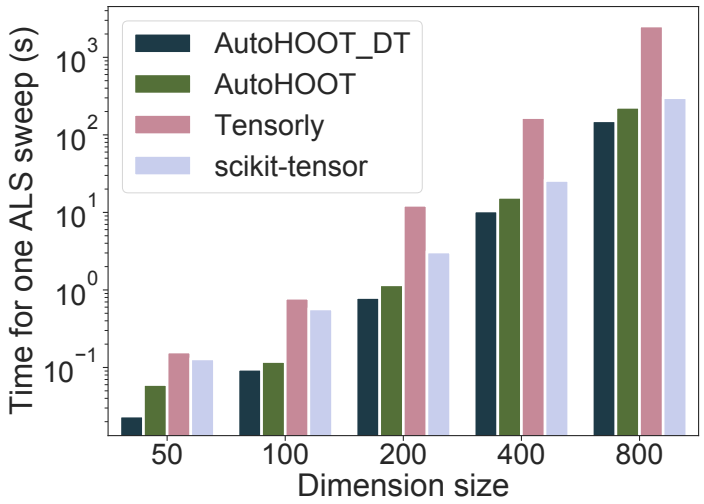
- Tensor network and tensor decomposition methods all typically based on applying Newton's method on a sequence of subsets of variables

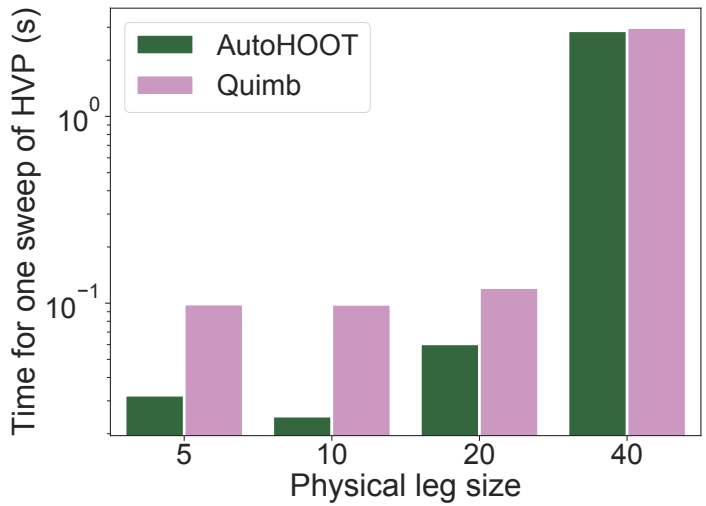



- Automatic differentiation (AD) in principle enables automatic generation of these methods
- Existing AD tools such as Jax (used by TensorFlow) are designed for deep learning and are ineffective for other tensor computations
  - these focus on first order optimization via Jacobian-vector products
  - unable to propagate tensor algebra identities such as $(\boldsymbol{A} \otimes \boldsymbol{B})^{-1} = \boldsymbol{A}^{-1} \otimes \boldsymbol{B}^{-1}$ to generate efficient code

# Automatic High-Order Optimization for Tensors

- AutoHOOT[1] provides a tensor-algebra centric AD engine
- Designed for `einsum` expressions and alternating minimization common in tensor decomposition and tensor network methods
- Python-level AD is coupled with optimization of contraction order and caching of intermediates
- Generates code for CPU/GPU/supercomputers using high-level back-end interface to tensor contractions



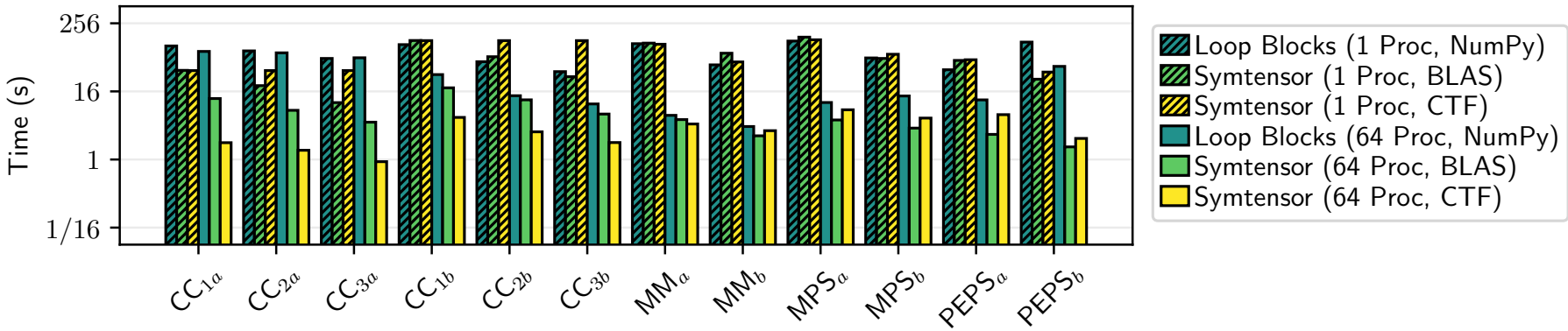

[1] Linjian Ma, Jiayu Ye, and E.S. PACT 2020

# Group Symmetry in Tensor Contractions

- Tensor with cyclic group symmetry can be represented as block-sparse

$$t_{ijk\ldots} = 0 \quad \text{if} \quad \lfloor i/G_1 \rfloor + \lfloor j/G_2 \rfloor + \lfloor k/G_3 \rfloor + \cdots \neq 0 \pmod{G}.$$

- Group symmetries of multiple types arise due to conservation laws when physical systems (quantum number symmetry, spin symmetry, rotational symmetry, translational symmetry)
- New contraction algorithm, *irreducible representation alignment* uses new dense reduced form tensor to handle group symmetry without looping over blocks or sparsity[1]

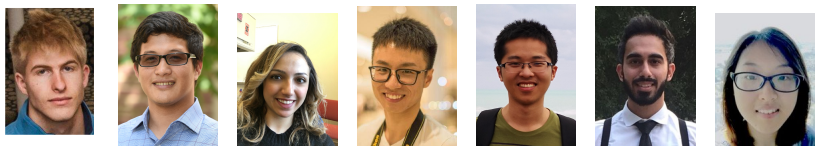



[1] Y. Gao, P. Helms, G. Chan, and E.S., arXiv:2007.08056

# Acknowledgements


- Our software is available via `github.com/cyclops-community` and `github.com/LinjianMa/AutoHOOT`
- See also our recent web-course "Tensor Computations" `relate.cs.illinois.edu/course/cs598evs-f20/`
- This work has been supported by NSF awards #1839204 (RAISE-TAQS), #1931258 (CSSI), #1942995 (CAREER)
- Stampede2 resources at TACC via XSEDE


L P N A @CS@Illinois



DOE CSGF NSF

`http://lpna.cs.illinois.edu`

# Backup slides

# Applications of CP Decomposition

- CP and Tucker are both used for data compression

# Applications of CP Decomposition

- CP and Tucker are both used for data compression
- In quantum chemistry, CP decomposition is used to obtain tensor hypercontraction (THC) format

$$t_{abij} = \underbrace{\sum_{s=1}^{P} d_{abs} d_{sij}}_{\text{Cholesky}}, \qquad d_{abs} = \underbrace{\sum_{r=1}^{R} u_{ar} u_{br} v_{sr}}_{\text{CP with repeating factor}}$$

# Applications of CP Decomposition

- CP and Tucker are both used for data compression
- In quantum chemistry, CP decomposition is used to obtain tensor hypercontraction (THC) format

$$t_{abij} = \underbrace{\sum_{s=1}^{P} d_{abs} d_{sij}}_{\text{Cholesky}}, \qquad d_{abs} = \underbrace{\sum_{r=1}^{R} u_{ar} u_{br} v_{sr}}_{\text{CP with repeating factor}}$$

  - THC asymptotically reduces cost of post-Hartree-Fock methods

# Applications of CP Decomposition

- CP and Tucker are both used for data compression
- In quantum chemistry, CP decomposition is used to obtain tensor hypercontraction (THC) format

$$t_{abij} = \underbrace{\sum_{s=1}^{P} d_{abs} d_{sij}}_{\text{Cholesky}}, \qquad d_{abs} = \underbrace{\sum_{r=1}^{R} u_{ar} u_{br} v_{sr}}_{\text{CP with repeating factor}}$$

  - THC asymptotically reduces cost of post-Hartree-Fock methods
- CP can be used to find fast bilinear algorithms, such as Strassen's matrix multiplication algorithm ($s = 4$, $R = 7$),

$$t_{ijklmn} = \delta_{lm}\delta_{ik}\delta_{nj} \quad \text{so} \quad c_{ij} = \sum_{klmn} t_{ijklmn} a_{kl} b_{mn} = \sum_{l} a_{il} b_{lj}$$

$$t_{ijklmn} = \sum_{r=1}^{R} u_{ijr} v_{klr} w_{mnr} \;\Rightarrow\; c_{ij} = \sum_{r=1}^{R} u_{ijr} \Big( \sum_{kl} v_{klr} a_{kl} \Big) \Big( \sum_{mn} w_{mnr} b_{mn} \Big)$$

# Randomized Methods for Sparse Tensor Decomposition

- When seeking a low-rank $R = O(1)$ decomposition for a sparse tensor, sketching schemes have been shown to be efficient
- In this regime, Tucker can be used to construct a CP decomposition
- Leverage score sampling on the rank-constrained least squares problem $\min_{\boldsymbol{X},\mathrm{rank}(\boldsymbol{X})\leq R} \|\boldsymbol{AX}-\boldsymbol{B}\|_F$ leads to a state-of-the-art cost-accuracy trade-off[1] in approximations to Tucker-ALS

| Algorithm for Tucker | LS solve cost | Sample size ($m$) |
|---|---|---|
| ALS | $O(\mathrm{nnz}(\boldsymbol{\mathcal{T}})R^{N-1})$ | / |
| ALS + TensorSketch[2] | $\tilde{O}(mR^N + msR)$ | $O(R^{2(N-1)} \cdot 3^{N-1}/(\epsilon^2\delta))$ |
| ALS + TTMTS[2] | $\tilde{O}(msR^{N-1})$ | $O(R^{2(N-1)} \cdot 3^{N-1}/(\epsilon^2\delta))$ |
| ALS + TensorSketch[1] | $\tilde{O}(mR^{2N-2} + sR^{N-1})$ | $O\big((R^{(N-1)} + 1/\epsilon^2) \cdot (3R)^{(N-1)}/\delta\big)$ |
| ALS + leverage scores[1] | $\tilde{O}(mR^{2N-2} + sR^{N-1})$ | $O(R^{(N-1)}/(\epsilon^2\delta))$ |

[1] Linjian Ma and E.S., in preparation

[2] O. Malik and S. Becker, 2018 (assuming unconstrained LSQ)