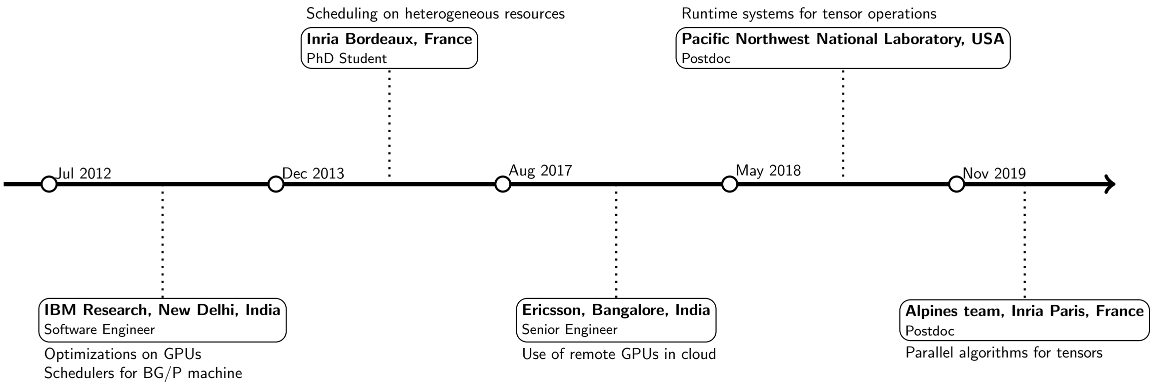
Parallel Strategies for Quantum Computations

Suraj Kumar

Inria Paris

January 17, 2022

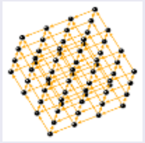
Resume in timeline


Scheduling on heterogeneous resources
Inria Bordeaux, France
PhD Student
Runtime systems for tensor operations
Pacific Northwest National Laboratory, USA
Postdoc
Jul 2012
Dec 2013
Aug 2017
May 2018
Nov 2019
IBM Research, New Delhi, India
Software Engineer
Optimizations on GPUs
Schedulers for BG/P machine
Ericsson, Bangalore, India
Senior Engineer
Use of remote GPUs in cloud
Alpines team, Inria Paris, France
Postdoc
Parallel algorithms for tensors

Past Experience

Parallelization in Polyhedral Model

- Linked-list operations
- Improved spatial locality
- Parallelization using OpenMP



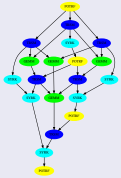
Seismic Imaging on GPUs

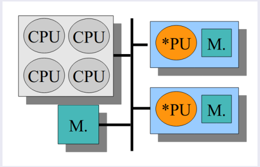
$$\begin{aligned}H_1 &= \sin^2\theta\cos^2\phi\,\frac{\partial^2}{\partial x^2} + \sin^2\theta\sin^2\phi\,\frac{\partial^2}{\partial y^2}\\ &\quad + \cos^2\theta\,\frac{\partial^2}{\partial z^2} + \sin^2\theta\sin 2\phi\,\frac{\partial^2}{\partial x\partial y}\\ &\quad + \sin 2\theta\sin\phi\,\frac{\partial^2}{\partial y\partial z} + \sin 2\theta\cos\phi\,\frac{\partial^2}{\partial x\partial z}\\ H_2 &= \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2} - H_1\end{aligned}$$

Schedulers for Blue Gene Supercomputers

- GASNET API
- Unbalanced Tree Search benchmark
- Comparison to Charm++

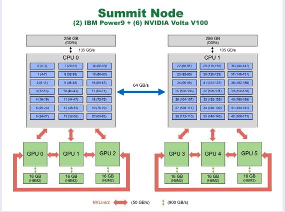
Scheduling on Heterogeneous Platforms





Molecular Simulations on Supercomputers

- NWChemEx Project
- TAMM library
- Hartree Fock and CCSD applications



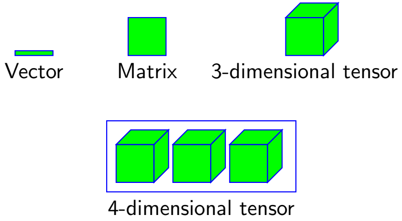
Present Collaborations

- Parallel algorithms for high dimensional tensors – with Laura Grigori (Inria Paris) and Olivier Beaumont (Inria Bordeaux)
- Communication optimal algorithms for Tensor computations – with Laura Grigori, Grey Ballard (Wake Forest University, USA), Kathryn Rouse (Inmar Intelligence, USA) and Hussam Al Daas (STFC Rutherford Appleton Laboratory, UK)
- Theoretical models to perform molecular dynamics simulations with a few number of parameters – with Laura Grigori, Yvon Maday (Sorbonne University), Eric Cances (Ecole des Ponts ParisTech) and Jean-Philip Piquemal (Sorbonne University)
- Efficient implementation of density matrix renormalization group (DMRG) algorithm – with Laura Grigori and Julien Toulouse (Sorbonne University)

Outline

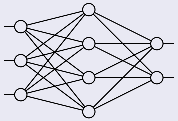
Tensors are used in Several Domains

- **Neuroscience**: Neuron $\times$ Time $\times$ Trial
- **Transportation**: Pickup $\times$ Dropoff $\times$ Time
- **Media**: User x Movie x Time
- **Ecommerce**: User x Product x Time
- **Cyber-Traffic**: IP x IP x Port x Time
- **Social-Network**: Person x Person x Time x Interaction-Type



High Dimensional Tensors

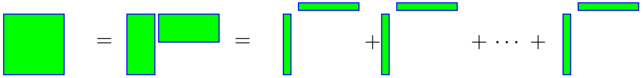
- **Neural Network**:



- **Molecular Simulation**: To represent wave functions
- **Quantum Computing**: Tensor network based models for computations

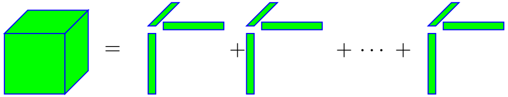
Tensor Computations

- Memory and computation requirements are exponential in the number of dimensions
  - A simulation involving just 100 spatial orbitals manipulates a huge tensor with 4^{100} elements
- People work with low dimensional structure (decomposition) of the tensors
  - A tensor is represented with smaller objects
  - Improves memory and computation requirements
- Most tensor decompositions rely on Singular Value Decomposition (SVD) of matrices
  - SVD represents a matrix as the sum of rank one matrices, $A = U\Sigma V^T = \sum_i \Sigma(i;i) U_i V_i^T$

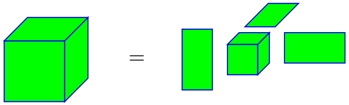


Popular Tensor Decompositions (Higher Order Generalization of SVD)

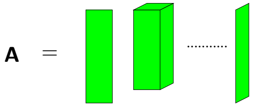
- Canonical decomposition (Also known as Canonical Polyadic or CANDECOMP/PARAFAC)



- Tucker decomposition

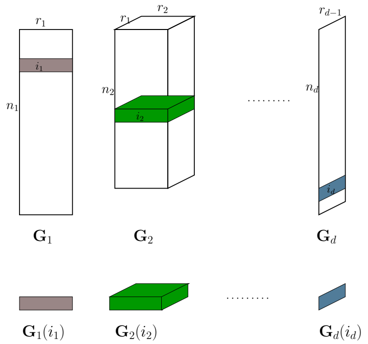


- Tensor Train decomposition (equivalently known as Matrix Product States)



Tensor Train Representation: Product of Matrices View

- A d-dimensional tensor is represented with 2 matrices and d-2 3-dimensional tensors.



$$\mathbf{A}(i_1, i_2, \cdots, i_d) = \mathbf{G}_1(i_1)\mathbf{G}_2(i_2)\cdots\mathbf{G}_d(i_d)$$

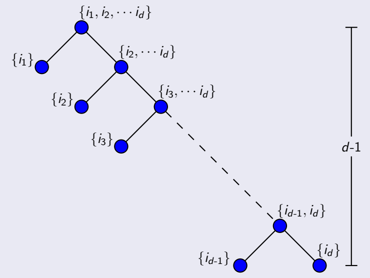
An entry of $\mathbf{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ is computed by multiplying corresponding matrix (or row/column) of each core.

Previous Activities

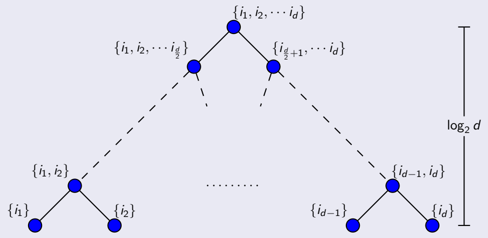
Tensor Train algorithms and Separation of dimensions

- A sequential algorithm to compute Tensor Train decomposition exists [Oseledets, 2011]

Sequential algorithm



For better parallelization

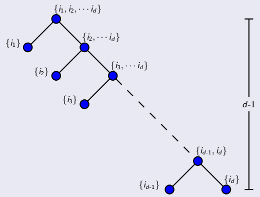


$\{i_1, i_2, \cdots i_d\}$
$\{i_1, i_2, \cdots i_{\frac{d}{2}}\}$
$\{i_{\frac{d}{2}+1}, \cdots i_d\}$
$\log_2 d$
$\{i_1, i_2\}$
$\{i_{d-1}, i_d\}$
$\{i_1\}$
$\{i_2\}$
$\{i_{d-1}\}$
$\{i_d\}$

- Can obtain better parallelism by expressing the operation in a balanced binary tree shape
  - Proposed a parallel algorithm based on this idea

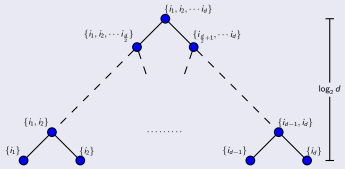
Tensor Train approximation algorithms

Sequential algorithm [Oseledets, 2011]



- Unfolding matrix: matricized representation of the tensor
- Perform truncated SVD of unfolding matrix A, $A = U\Sigma V^T + E_A$
- Work with ΣV^T on the right subtree

Our parallel algorithm



$\{i_1, i_2, \cdots i_d\}$
$\{i_1, i_2, \cdots i_{\frac{d}{2}}\}$
$\{i_{\frac{d}{2}+1}, \cdots i_d\}$
$\log_2 d$
$\{i_1, i_2\}$
$\{i_1\}$
$\{i_2\}$
$\{i_{d-1}, i_d\}$
$\{i_{d-1}\}$
$\{i_d\}$

- Perform truncated SVD of unfolding matrix A, $A = U\Sigma V^T + E_A$
- Find diagonal matrices X, Y, and S, such that $\Sigma = XSY$
- Call left (resp. right) subtree with UX (resp. YV^T)

Approach 1: $X = I$, $Y = \Sigma$, $S = I$
Approach 2: $X = Y = \Sigma^{\frac{1}{2}}$, $S = I$
Approach 3: $X = Y = \Sigma$, $S = \Sigma^{-1}$

Comparison of our approaches

- A 12-dimensional tensor with 4^{12} elements (generated with a popular low rank function)
- prescribed accuracy $= 10^{-6}$
- Compr: compression ratio, NE: number of elements, AA: approximation accuracy

Metric	Sequential Algo	Parallel Algo		
		Approach 1	Approach 2	Approach 3
Compr	99.993	99.817	99.799	99.993
NE	1212	30632	33772	1212
AA	2.271e-07	3.629e-08	2.820e-08	2.265e-07

SVD is expensive

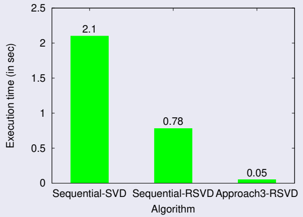
- Good alternatives to SVD: QR factorization with column pivoting (QRCP), randomized SVD (RSVD)

Approach	Rank	Compr	NE	Sequential-AA	Approach3-AA
SVD	5	99.994	992	6.079e-06	6.079e-06
QRCP+SVD				1.016e-05	1.384e-05
RSVD				6.079e-06	6.079e-06
SVD	6	99.992	1376	1.323e-07	1.340e-07
QRCP+SVD				3.555e-07	5.737e-07
RSVD				1.322e-07	1.322e-07

Performance comparison

Single core performance

- Number of computations for both RSVD algorithms = $\mathcal{O}(n^d)$
- Approach3-SVD is very slow
- Approach3-RSVD is much faster



RSVD algorithm of Approach3-RSVD

- Input matrix is A and the desired rank is r
- Multiply with a random sketch matrix (depends on r), Y = A *RS
- Perform QR factorization, [Q, $\sim$] = QR(Y)
- Compute SVD decomposition, [U S V] = SVD(Q^T * A)
- Update U, U = Q*U

Parallel performance counts on P processors

Communication cost analysis along the critical path

- To perform A*RS, #data transfers $= \mathcal{O}(\frac{n^{\frac{d}{2}}}{\sqrt{P}} \log P)$
- To perform $Q^T * A$, #data transfers $= \mathcal{O}(\frac{n^{\frac{d}{2}}}{\sqrt{P}} \log P)$
- To perform reshape operation, #data transfers $= \mathcal{O}(\frac{n^{\frac{d}{2}}}{\sqrt{P}} \log P)$
- At each step, #messages $= \mathcal{O}(\log P)$

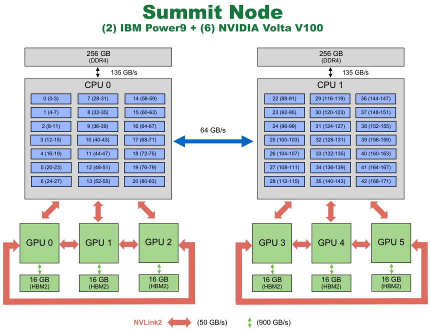
Algorithm	# Computations	Communications[1]	# Messages
Approach3-RSVD	$\mathcal{O}(\frac{n^d}{P})$	$\mathcal{O}(\frac{n^{\frac{d}{2}}}{\sqrt{P}} \log P)$	$\mathcal{O}(\log d \log P)$
Sequential-RSVD	$\mathcal{O}(\frac{n^d}{P})$	$\mathcal{O}(\frac{n^{d-1}}{P}(1+\frac{\log P}{d}))$	$\mathcal{O}(d \log P)$

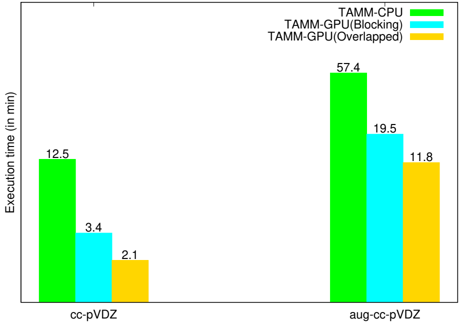
[1]Assuming n is large.

Previous Activities

Minimizing impact of communications on Summit supercomputer

- Maximizing the overlap of communications and computations
- Implemented proposed approaches in Tensor Algebra for Manybody Methods (TAMM) library
- Molecular chemistry application (CCSD), Ubiqtin molecule, cc-pVDZ (737 basis functions, 220 nodes), aug-cc-pVDZ (1243 basis functions, 256 nodes)





Joint work with S. Krishnamoorthy and M. Zalewski during my postdoc at PNNL, USA

Figure source: `https://www.olcf.ornl.gov`

Project: Parallel Strategies for Quantum Computations

1. Teaching plan
2. Validation of quantum algorithms on classical computers
   - Scalability of the algorithms
3. Design of new parallel algorithms for quantum computations
   - Tensor network (equivalent to quantum circuit) based algorithms

Teaching plan for the next academic year

- Introduction to Tensors
  - tensor notations and various tensor decompositions, tensor networks, use of tensors in data analysis and quantum molecular simulations, use of existing tensor libraries for some selected problems, analysis of the various tensor decompositions, computation and storage complexities of tensor computations, challenges with tensor computations, create new algorithms to mitigate some of these challenges

- Algorithms and Data Structures
  - stack, queue and heap data structures, time and space complexities of algorithms, popular ways of search and sort items, algorithm design strategies – divide and conquer, greedy methods, dynamic programming, algorithms for various popular problems based on these strategies, parallelization of algorithms, few hands-on sessions for some selected algorithms, P, NP and NP-complete classes

Other courses in the coming years

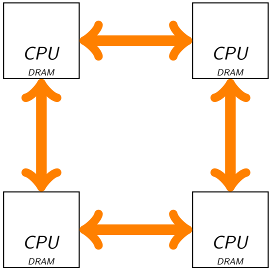
- Specialized courses
  - Parallel and communication avoiding algorithms
  - Architecture aware algorithms
- Basic courses
  - Compiler design
  - Computer architecture
  - Program analysis and verification

Research Plan

1. Teaching plan
2. Validation of quantum algorithms on classical computers
    - Scalability of the algorithms
3. Design of new parallel algorithms for quantum computations
    - Tensor network (equivalent to quantum circuit) based algorithms

Communication and its importance in classical computing

- Running time of an algorithm depends on
  - Computations
    - Number of operations * time-per-operation
  - Data movement
    - Volume of communication / Network-bandwidth
    - Number of messages * Network-latency



- Gaps growing exponentially with time (Source: Getting up to speed: The future of supercomputing)

	time-per-operation	Network-bandwidth	Network-latency
Annual improvements	59 %	26 %	15 %

- Avoid communication to save time (and energy)

Scalable algorithms on classical computers

- Most quantum circuits can be expressed as tensor operations
- Analyze the existing algorithms and communication performed by them
- Propose new scalable communication optimal algorithms
- Implement the proposed algorithms
  - Load balancing
  - Memory awareness
  - Efficient scheduling of computations

Communication lower bounds for quantum computations

How people did it on classical computers?

- People obtain results for matrix multiplication operations
- Same lower bounds apply to almost all direct linear linear algebra operations using reduction [Ballard et. al., 09] , for instance, bound for LU factorization

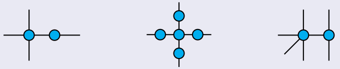
$$\begin{pmatrix} I & & -B \\ A & I & \\ & & I \end{pmatrix} = \begin{pmatrix} I & & \\ A & I & \\ & & I \end{pmatrix} \begin{pmatrix} I & & -B \\ & I & AB \\ & & I \end{pmatrix}$$

- Extend this approach for quantum computations on classical computers
- Analyze this approach to understand how it can be extended for the quantum computers

Approach to compute lower bounds for tensor computations

Notation: Tensors are denoted by solid shapes and number of lines denote the dimensions of the tensors. Connecting two lines implies summation (or contraction) over the connected dimensions.

- Obtain bounds for basic tensor operations

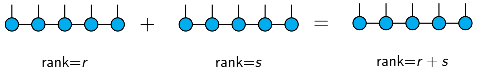


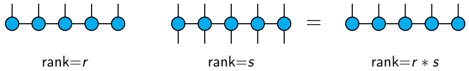
Research Project

1. Teaching plan
2. Validation of quantum algorithms on classical computers
    - Scalability of the algorithms
3. Design of new parallel algorithms for quantum computations
    - Tensor network (equivalent to quantum circuit) based algorithms

Challenges with low-rank representation of high dimensional tensors

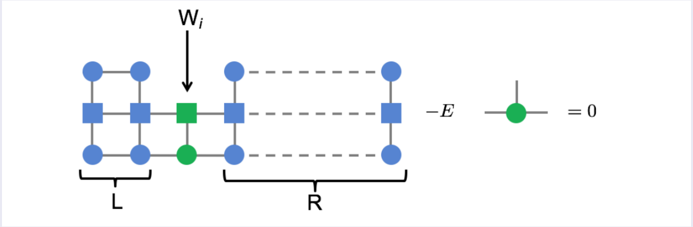
- Tensor Train is a popular representation to work with high dimensional tensors
- Adding tensors and aplying an operator in this representation





- Requires a truncation process which iterates over cores one by one
- This representation is not much suited to work in parallel

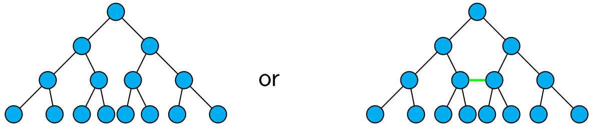
Density Matrix Renormalization Group (DMRG) algorithm



(Figure source: Markus Reiher)

Parallel algorithms to work with new tensor representations

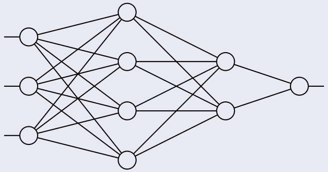
- Look at new representations in tree format – suitable for parallelization



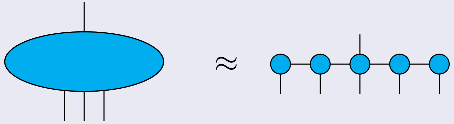
- Data will be stored at the leaf nodes
- Interal nodes will help to manipulate tensors in parallel
- Some tensor representations exhibit tree structure
  - Mainly designed to reduce storage or model long range interactions
  - Not suitable to work with them in parallel
- Design new (or modify existing) algorithms to work with the proposed representations

Parallelization of tensorized deep neural network models

Conventional deep neural network



Tensorized neural networks



- Parallel methods to train tensorized neural networks

Bringing additional skills in the team

- High dimensional dense tensor computations
- Communication lower bounds for tensor computations
- Parallel algorithms for large computing systems
- Scheduling strategies to make better utilization of resources

Thank You!