Scalable Tensor Computations with Cyclops and Faster Algorithms for Alternating Least Squares

Edgar Solomonik

Department of Computer Science
University of Illinois at Urbana-Champaign

Invited Workshop on Compiler Techniques for Sparse Tensor Algebra
Cambridge, MA

Jan 26, 2019
A library for parallel tensor computations

Cyclops Tensor Framework (CTF)$^1$, C++ (MPI/OpenMP) ⇒ Python

- distributed-memory symmetric/sparse/dense tensor objects
  
  ```
  Matrix<int> A(n, n, AS|SP, World(MPI_COMM_WORLD));
  Tensor<float> T(order, is_sparse, dims, syms, ring, world);
  T.read(...); T.write(...); T.slice(...); T.permute(...);
  ```

- parallel contraction/summation of tensors
  
  ```
  Z["abij"] += V["ijab"]; // C++
  Z.i("abij") << V.i("ijab") // Python
  W["mnij"] += 0.5*W["mnef"]*T["efij"]; // C++
  W.i("mnij") << 0.5*W.i("mnef")*T.i("efij") // Python
  einsum("mnef,efij->mnij",W,T) // numpy-style Python
  ```

- \(~2000\) commits since 2011, open source since 2013

---

$^1$E.S., D. Matthews, J.R. Hammond, J. Demmel, JPDC 2014
Electronic structure calculations with Cyclops

Coupled cluster engine in Aquarius (Devin Matthews)

Cyclops works with QChem, VASP, CC4S, Psi4, and PySCF

Is also being used for other applications, e.g. by IBM+LLNL collaboration to perform 49-qubit quantum circuit simulation\(^2\)

\(^2\)E. Pednault et al. arXiv:1710.05867
Sparse MP3 code

Strong and weak scaling of sparse MP3 code, with
(1) dense $V$ and $T$ (2) sparse $V$ and dense $T$ (3) sparse $V$ and $T$

Strong scaling of MP3 with no=40, nv=160

Weak scaling of MP3 with no=40, nv=160
Special operator application: betweenness centrality

Betweenness centrality code snippet, for $k$ of $n$ nodes

```c
void btw_central(Matrix<int> A, Matrix<path> P, int n, int k){
    Monoid<path> mon(...,
        [](path a, path b){
            if (a.w<b.w) return a;
            else if (b.w<a.w) return b;
            else return path(a.w, a.m+b.m);
        }, ...);
    Matrix<path> Q(n,k, mon); // shortest path matrix
    Q["ij"] = P["ij"];

    Function<int,path> append([](int w, path p){
        return path(w+p.w, p.m);
    });
    for (int i=0; i<n; i++)
        Q["ij"] = append(A["ik"],Q["kj"]);
    ...
}
```
Left plot compares different algorithms
- with CombBLAS
- with CA-MFBC (statically-mapped comm-efficient matrix distribution)

Right plot compares matrix representations (including push/pull)
- adjacency matrix sparse for all versions
- frontier sparse or dense rectangular matrix
- vertices adjacent to frontier (output) sparse or dense rectangular matrix
Tensor decomposition algorithms generally use a variant of gradient descent or alternating least squares (ALS)

ALS is effective for CP and Tucker as well as MPS/PEPS/DMRG

update each site/factor in network individually by quadratic optimization

---

Holtz, Rohwedder, and Schneider SISC 2012
Accelerating Alternating Least Squares

- Dimension trees amortize cost across quadratic subproblems
- Pairwise perturbation (PP) approximates ALS with less cost\(^4\), specifically for rank \( R \) decomposition for order \( N \) and \( s \times \cdots \times s \) tensor

<table>
<thead>
<tr>
<th></th>
<th>dimension tree ALS sweep</th>
<th>PP setup</th>
<th>PP approximate sweep</th>
</tr>
</thead>
<tbody>
<tr>
<td>CP</td>
<td>( 4s^N R )</td>
<td>( 6s^N R )</td>
<td>( 2Ns^2 R )</td>
</tr>
<tr>
<td>Tucker</td>
<td>( 4s^N R )</td>
<td>( 6s^N R )</td>
<td>( 2Ns^2 R^{N-1} )</td>
</tr>
</tbody>
</table>

- Cyclops-based implementation of PP shows improvements over regular dimension tree ALS for both synthetic and real-world tensors

\(^4\)Linjian Ma and E.S.  arXiv:1811.10573
Conclusion

Summary

- Cyclops is a distributed-memory sparse/dense tensor library
  - has seen adaptation in quantum chemistry and quantum circuit simulation
  - supports general semirings, efficient parallel graph algorithms
- Pairwise perturbation is a first-order-accurate approximation to ALS
  - its asymptotically faster in theory and 2-3X faster in practice

In-progress/future work

- Sparse tensor completion with Cyclops using ALS/CCD/SGD
- Perturbative ALS with low-rank updates

Acknowledgements

- Devin Matthews (UT Austin), Jeff Hammond (Intel Corp.), Maciej Besta, Flavio Vella, Torsten Hoefler (ETH Zurich), Zecheng Zhang (UIUC), Linjian Ma, James Demmel (UC Berkeley)
- Computational resources at NERSC, CSCS, ALCF, NCSA, and TACC