Trillion-Tensor: Trillion-Scale CP Tensor Decomposition

Zeliang Zhang,Xiao-Yang Liu,Pan Zhou

Huazhong Uniersity of Science and Technology

Department of Electrical Engineering, Columbia University, USA

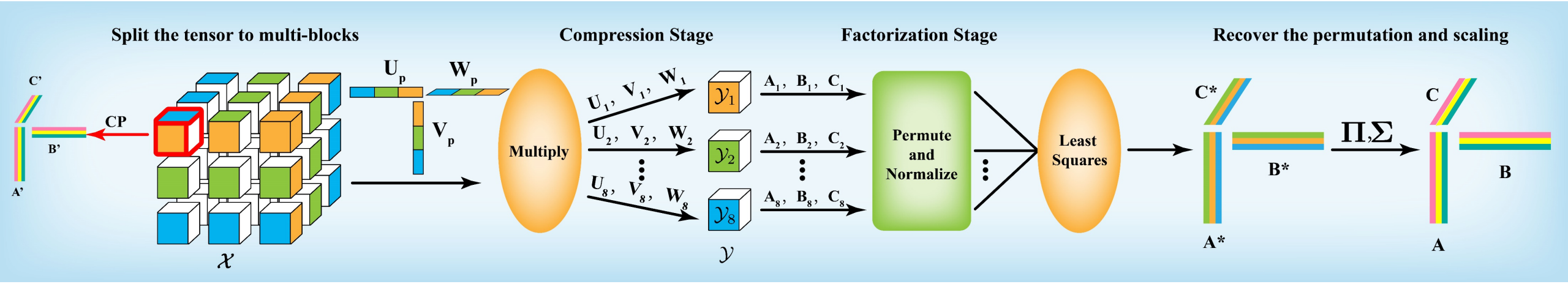
Motivations

- Tensor decomposition is the basis of many machine learning applications, including graphic analysis,image classification, data mining, etc. There are two major tensor decomposition models, CP tensor decomposition and Tucker tensor decomposition.
- In recent years, the size of tensors becomes increasingly large, approaching millions to trillions of nonzero elements. Due to the limitation of the main memory, both CP and Tucker tensor decomposition algorithms are impractical for large-scale tensors.
- Many proposed methods may handle tensors with each mode up to millions. However, the input tensors are required to be extremely sparse, where the total number of nonzero elements is up to millions.
- Exploiting the high-performance large scale tensor decomposition scheme.

Challenges

- Designing a compression method to process an “out-of-memory” tensor;
- Exploring an appropriate trick to recover the permutations and scaling of the full mode factorization matrices;
- implementing efficient parallel schemes both on CPUs and GPUs, respectively.

Trillion-Tensor Decomposition



Overview of the proposed trillion-tensor decomposition

- The trillion-tensor algorithm mainly consists of three stages, namely compression stage, factorization stage, and a scaling and permutation stage, as given in the above Figure.
- First, a tensor $X \in \mathrm{R}^{300\times300\times300}$, consisting of 27 tensor blocks $B_n \in \mathrm{R}^{100\times100\times100}$, n=1,2, ...,27, is first compressed into 8 replicas $\mathrm{Y}_p \in \mathrm{R}^{50\times50\times50}$, p= 1,2, ...,8.
- Second, each replica is factored to mode matrices independently.
- Thirdly, with a proper permutation and normalization, solving a least squares problem for each mode will result in the full mode matrices (A∗,B∗,C∗). Using the factorization of the first block B_n, we get the permutation matrix Π and the scaling matrix Σ. Then, we use Π and Σ to obtain the full mode matrices (A,B,C)

Optimization

Bottleneck:

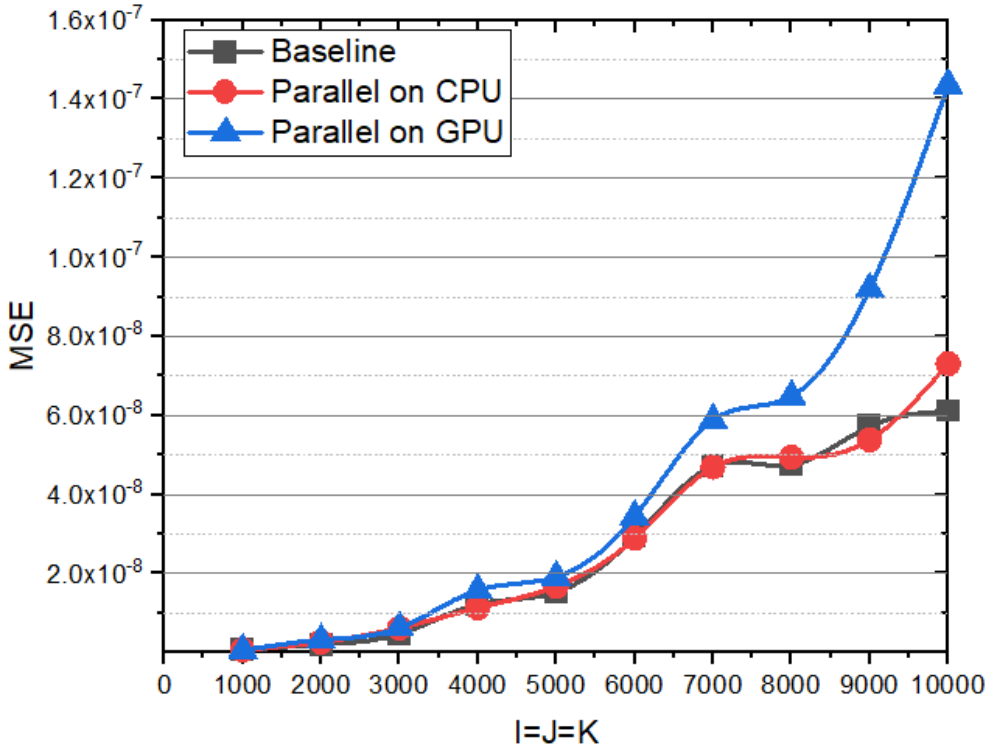
The computational complexity of the compression stageis O(P IJK×min(L, M, N)) versus O(IJF) of the factorization stage. Huge time consumption of the compression stage is the bottleneck of this algorithm with the size of tensor increasing.

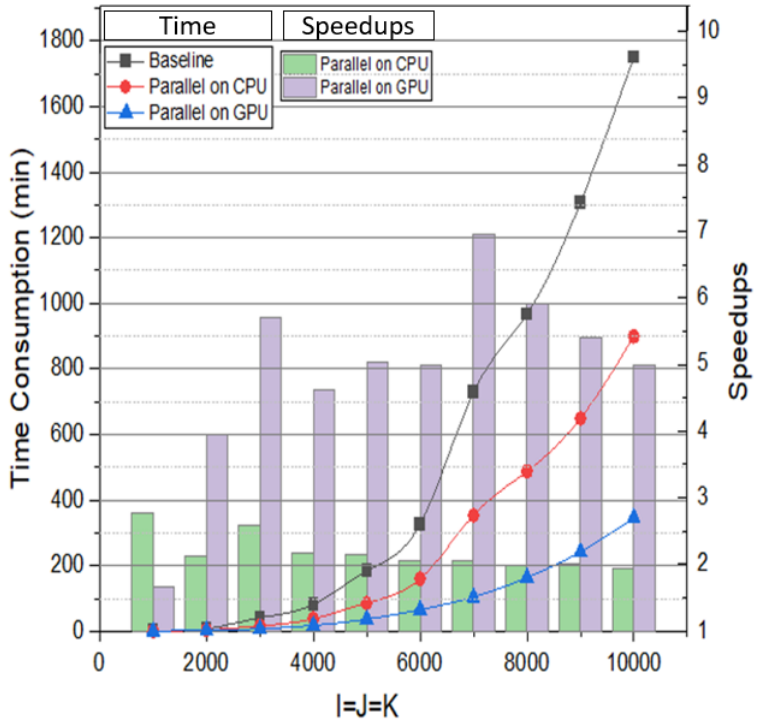
Methods:

- Parallel implementation on CPUs: This scheme advocates exploiting the property of multi-core processors on CPUs.

- Parallel implementation on GPUs: This scheme advocates exploiting GPUs' high performance in matrixmultiplication.

Performance Evaluations





In our experiments, we test tensors ranging from million-scale to trillion-scale and obtain a relatively low mean squared error. Evaluation results show that trillion-tensor supports up tp large scale tensors up to 1w * 1w *1w, and a speedup up to 6.95×, compared with the baseline on CPU.

THANK YOU!

Zeliang Zhang
2021.1