High-Performance Computing for Big Data Analytics: My Reflection of a Decade Adventure

Hang Liu
Rutgers, The State University of New Jersey
hang.liu@rutgers.edu

Data is the new crude oil of today! However, just like crude oil must be refined into gasoline to deliver values to the combustion engine, only the "intelligence" that is extracted from the raw data will fuel business wins. Starting with data, lots of it; if you are like most entities, you are probably drowning in the big raw data that brings little value to you. Clearly, the computing systems that manage and analyze the raw big data play a pivotal role.

Big data analytics needs to be fast! Otherwise, the extracted insights will quickly expire and bring diminished impacts. Using our recent machine learning for architecture simulation paper at SC '22 [1] as an example, architecture simulation is the key tool on design space exploration for processor designs. Unfortunately, traditional cycle accurate simulators, such as, gem5 [2] would need months to simulate a typical benchmark suit, such as SPEC [3] for one processor configuration. That said, exploring a few hundred of configurations would lead to decades of simulation time. Clearly, the speed of this traditional simulator prohibits architect to thoroughly explore design space with existing architecture simulators as chip companies often need to roll out a new product annually. Please refer to our article [1] for more details about architecture simulation details. Therefore, big data analytics needs to be fast, a.k.a., HPC is the right way for big data analytics since HPC is a field that attempts to make anything fast.

Overview remarks. My feeling is that HPC is not a field like the traditional ones, such as, Networking, Architecture, or Machine Learning. Specifically, *traditional fields often restrict themselves to several particular applications.* For example, architecture field is really concentrated in research work that focuses on architecture. In contrast, *HPC focuses on making things fast.* Therefore, the application can be anything that is important but slow. Just like the name of the flagship conference in HPC - Supercomputing, a.k.a., International Conference for High Performance Computing, Networking, Storage and Analysis. HPC focuses on making things high-performance (i.e., fast): the subject could be computing, networking, storage, or analysis. Further, when it goes to the methodology, HPC is again not restricted to any particular methods, i.e., the solution can be hardware or software, heuristic or theoretical. One key metric that HPC researchers value is that whether your proposed solution is implemented. In this regard, HPC could be a field that is closely related to system research.

I will humbly use my research experience to explain how to navigate through different research fields and identify the crucial applications for HPC research, as well as leveraging various software and hardware solutions to claim the high-performance target.

HPC should address "critical" applications that suffers from relatively long turnaround time. I will use my research experience to explain how we identify candidate applications for my project. I started my research by working on accelerating the solving procedure of the Navier Stokes Equations for the Computational Fluid Dynamics (CFD) problems [4, 5]. When the research proceeds, the core of this problem is accelerating distributed

Sparse Matrix Vector Multiplication (SpMV) for solvers in Krylov space. Subsequently, I noticed that sparse matrix is also inherently a graph. In addition, social network analysis is extremely popular during that period of time, I propose to move my research focus to graph traversal, which my advisor also agreed. Subsequently, I published many research papers about graph algorithms, such as, graph traversal [6, 7, 8, 9, 10], triangle counting [11, 12, 13, 14], connected components [15], graph mining [16], graph systems [17, 18]. With the emergence of machine learning and graph learning, we shifted our focus to graph learning [19, 20, 21] and machine learning [22, 23, 24, 25] after I start my professorship career. Along the road, we have also explored applications such as Locality Sensitive Hashing (LSH) [26], LU factorization [27], architecture simulation [28]. The reason we shifted our focused applications was rooted from the fact that those applications are important for various disciplines, and were regarded slow.

It is important to mention that the "speed" could be a problem even it seems already to be fast. Using our recent E.T. paper at SC '21 [24] as an example, popular enterprise Natural Language Processing (NLP) solutions, e.g., PyTorch, consume μs or longer to compute one layer for transformers. At the first glance, 680 μs could seem to be fast. However, a production-level engine would need the computation time to go below 100 μs for real-time translation or live chat to succeed. Therefore, it is not the absolute speed that determines whether a further acceleration is required. Instead, we should focus on what is the desired speed. If the current speed falls below the desired speed, we need to accelerate the computation process. In that regard, this is a proper application for HPC researchers.

HPC for big data should seek for innovations from every aspect of the computing system(s). While HPC researchers would like to observe orders of magnitude speedup over the existing solutions, cutting the turnaround time by such a large margin is a mounting challenge. Therefore, *high-performance computing needs to explore every possible aspect of a computing system, including hardware and software, to claim that goal.* I will, again, use my experience to illustrate my approaches. On the one hand, HPC should leverage both emerging hardware [6, 7, 12, 18, 8, 19, 22, 29, 1], such as Graphics Processing Units (GPUs), and customizing new hardware platforms [9, 30, 31, 32, 33, 28, 25], e.g., Field-Programmable Gate Arrays (FPGAs) to curb the overhead of data movement in conventional computing platforms such as Central Processing Units (CPUs). On the other hand, HPC folks should also delve deeply into the software stack (including algorithm design and system implementations) [5, 17, 10, 15, 34, 23, 35, 36] to maximally extract the benefits from GPUs and FPGAs. Not limited there, the novel algorithms and system designs can also benefit traditional CPU platforms in case end users cannot access emerging hardware, such as GPUs/FPGAs.

However, it is also important to notice how much effort should be expected when we shift our focused applications and platforms for future projects in HPC. I will use our recent experience about FPGA and GPU programming as an example. We were supposed to perform an equirectangular projection computation [37] for a real-time streaming data that is 60 frames per second at the resolution of 8K. We thought FPGA would be the perfect platform for such a task. However, it turns out that we spend one Ph.D. student year only arriving at the speed of around 1 second per frame. Therefore, we switched the gear to GPU. And thanks to the ease of the programming interface, we can achieve beyond 60 frames per second throughput using roughly one week of efforts. Of note, both efforts were performed by the same PhD student.

In summary, HPC is one of the most young and vibrating fields. It attracts many researchers from their traditional fields, such as my collaborators from Machine Learning, Architecture, Networking and Security. In pursuit of the next critical application, researchers are always poised to learn new applications and platforms. Therefore, it is a field that you will never feel boring. I feel blessed to have a chance to work in HPC.

References

[1] Santosh Pandey, Lingda Li, Thomas Flynn, Adolfy Hoisie, and Hang Liu. Scalable Deep Learning-Based Microarchitecture Simulation on GPUs. In *SC-International Conference for High Performance Computing, Networking, Storage and Analysis*, 2022.

[2] Jason Lowe-Power, Abdul Mutaal Ahmad, Ayaz Akram, Mohammad Alian, Rico Amslinger, Matteo Andreozzi, Adrià Armejach, Nils Asmussen, Brad Beckmann, Srikant Bharadwaj, et al. The gem5 simulator: Version 20.0+. *arXiv preprint arXiv:2007.03152*, 2020.

[3] John L Henning. Spec cpu2006 benchmark descriptions. *ACM SIGARCH Computer Architecture News*, 34(4):1–17, 2006.

[4] Hang Liu, Jung-Hee Seo, Rajat Mittal, and H Howie Huang. Gpu-accelerated scalable solver for banded linear systems. In *2013 IEEE International Conference on Cluster Computing (CLUSTER)*, pages 1–8. IEEE, 2013.

[5] Rajat Mittal, Jung Hee Seo, Vijay Vedula, Young J Choi, Hang Liu, H Howie Huang, Saurabh Jain, Laurent Younes, Theodore Abraham, and Richard T George. Computational Modeling of Cardiac Hemodynamics: Current Status and Future Outlook. *Journal of Computational Physics*, 2016.

[6] Hang Liu and H Howie Huang. Enterprise: Breadth-First Graph Traversal on GPUs. In *SC-International Conference for High Performance Computing, Networking, Storage and Analysis*, 2015.

[7] Hang Liu et al. iBFS: Concurrent Breadth-First Graph Traversal on GPUs. In *Proceedings of the 2016 International Conference on Management of Data*, 2016.

[8] Anil Gaihre, Zhenlin Wu, Fan Yao, and Hang Liu. XBFS: eXploring Runtime Optimizations for Breadth-First Search on GPUs. In *Proceedings of the 28th International Symposium on High-Performance Parallel and Distributed Computing*, pages 121–131, 2019.

[9] Eric Finnerty, Zachary Sherer, Hang Liu, and Yan Luo. Dr. BFS: Data Centric Breadth-First Search on FPGAs. In *2019 56th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2019.

[10] Anil Gaihre, Yan Luo, and Hang Liu. Do Bitcoin Users Really Care About Anonymity? An Analysis of the Bitcoin Transaction Graph. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 1198–1207. IEEE, 2018.

[11] Hang Liu, Yang Hu, and H Howie Huang. High-Performance Triangle Counting on GPUs. In *2018 IEEE High Performance extreme Computing Conference (HPEC)*, pages 1–5. IEEE, 2018. Awarded Champion.

[12] Yang Hu, Hang Liu, and H Howie Huang. TriCore: Scalable Triangle Counting on GPUs. In *SC-International Conference for High Performance Computing, Networking, Storage and Analysis*, 2018.

[13] Santosh Pandey, Xiaoye Sherry Li, Aydin Buluc, Jiejun Xu, and Hang Liu. H-INDEX: Hash-Indexing for Parallel Triangle Counting on GPUs. In *2019 IEEE high performance extreme computing conference (HPEC)*, pages 1–7. IEEE, 2019 Awarded Champion.

[14] Santosh Pandey, Zhibin Wang, Sheng Zhong, Chen Tian, Bolong Zheng, Xiaoye Li, Lingda Li, Adolfy Hoisie, Caiwen Ding, Dong Li, et al. Trust: Triangle counting reloaded on gpus. *IEEE Transactions on Parallel and Distributed Systems*, 32(11):2646–2660, 2021.

[15] Yuede Ji, Hang Liu, et al. iSpan: Parallel Identification of Strongly Connected Components with Spanning Trees. In *SC-International Conference for High Performance Computing, Networking, Storage and Analysis*, 2018.

[16] Bibek Bhattarai, Hang Liu, et al. CECI: Compact Embedding Cluster Index for Scalable Subgraph Matching. In *Proceedings of the 2016 International Conference on Management of Data*, 2019.

[17] Hang Liu and H Howie Huang. Graphene: Fine-Grained IO Management for Graph Computing. In *Proceedings of the 15th Usenix Conference on File and Storage Technologies*, 2017.

[18] Hang Liu and H Howie Huang. SIMD-X: Programming and Processing of Graph Algorithms on GPUs. *arXiv preprint arXiv:1812.04070*, 2018.

[19] Santosh Pandey, Lingda Li, Adolfy Hoisie, Xiaoye S Li, and Hang Liu. C-SAW: A Framework for Graph Sampling and Random Walk on GPUs. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15. IEEE, 2020.

[20] Chengying Huan, Shuaiwen Leon Song, Yongchao Liu, Heng Zhang, Hang Liu, Charles He, Kang Chen, Jinlei Jiang, and Yongwei Wu. T-gcn: A sampling based streaming graph neural network system with hybrid architecture. In *Proceedings of the International Conference on Parallel Architectures and Compilation Techniques*, pages 69–82, 2022.

[21] Chengying Huan, Shuaiwen Leon Song, Santosh Pandey, Hang Liu, Yongchao Liu, Baptiste Lepers, Changhua He, Kang Chen, Jinlei Jiang, and Yongwei Wu. Tea: A general-purpose temporal graph random walk engine. 2023.

[22] Anil Gaihre, Da Zheng, Scott Weitze, Lingda Li, Shuaiwen Leon Song, Caiwen Ding, Xiaoye S Li, and Hang Liu. Dr. Top-k: Delegate-Centric Top-k on GPUs. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–14, 2021.

[23] Linnan Wang, Wei Wu, Junyu Zhang, Hang Liu, George Bosilca, Maurice Herlihy, and Rodrigo Fonseca. FFT-based Gradient Sparsification for the Distributed Training of Deep Neural Networks. In *Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing*, pages 113–124, 2020.

[24] Shiyang Chen, Shaoyi Huang, Santosh Pandey, Bingbing Li, Guang R Gao, Long Zheng, Caiwen Ding, and Hang Liu. E.T.: Re-thinking Self-Attention for Transformer Models on GPUs. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–18, 2021.

[25] Hongwu Peng, Shaoyi Huang, Shiyang Chen, Bingbing Li, Tong Geng, Ang Li, Weiwen Jiang, Wujie Wen, Jinbo Bi, Hang Liu, et al. A Length Adaptive Algorithm-Hardware Co-design of Transformer on FPGA Through Sparse Attention and Dynamic Pipelining. In *Proceedings of the 59th ACM/IEEE Design Automation Conference*, 2022.

[26] Bolong Zheng, Zhao Xi, Lianggui Weng, Nguyen Quoc Viet Hung, Hang Liu, and Christian S Jensen. PM-LSH: A Fast and Accurate LSH Framework for High-Dimensional Approximate NN Search. *Proceedings of the VLDB Endowment*, 13(5):643–655, 2020.

[27] Anil Gaihre, Xiaoye Sherry Li, and Hang Liu. GSOFA: Scalable Sparse Symbolic LU Factorization on GPUs. *IEEE Transactions on Parallel and Distributed Systems*, 2021.

[28] Lingda Li, Santosh Pandey, Thomas Flynn, Hang Liu, Noel Wheeler, and Adolfy Hoisie. SimNet: Accurate and High-Performance Computer Architecture Simulation Using Deep Learning. 6(2), jun 2022.

[29] Zhen Xie, Wenqian Dong, Jiawen Liu, Hang Liu, and Dong Li. Tahoe: tree structure-aware High Performance Inference Engine for Decision Tree Ensemble on GPU. In *Proceedings of the Sixteenth European Conference on Computer Systems*, pages 426–440, 2021.

[30] Bingbing Li, Zhenglun Kong, Tianyun Zhang, Ji Li, Zhengang Li, Hang Liu, and Caiwen Ding. Efficient Transformer-based Large Scale Language Representations using Hardware-Friendly Block Structured Pruning. *arXiv preprint arXiv:2009.08065*, 2020.

[31] Bingbing Li, Santosh Pandey, Haowen Fang, Yanjun Lyv, Ji Li, Jieyang Chen, Mimi Xie, Lipeng Wan, Hang Liu, and Caiwen Ding. Ftrans: Energy-Efficient Acceleration of Transformers using FPGA. In *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, pages 175–180, 2020.

[32] Hongwu Peng, Shaoyi Huang, Tong Geng, Ang Li, Weiwen Jiang, Hang Liu, Shusen Wang, and Caiwen Ding. Accelerating Transformer-based Deep Learning Models on FPGAs using Column Balanced Block Pruning. In *2021 22nd International Symposium on Quality Electronic Design (ISQED)*, pages 142–148. IEEE, 2021.

[33] Geng Yuan, Payman Behnam, Zhengang Li, Ali Shafiee, Sheng Lin, Xiaolong Ma, Hang Liu, Xuehai Qian, Mahdi Nazm Bojnordi, Yanzhi Wang, et al. Forms: Fine-Grained Polarized reRAM-based in-situ Computation for Mixed-Signal DNN Accelerator. In *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, pages 265–278. IEEE, 2021.

[34] Bibek Bhattarai, Hang Liu, and H Howie Huang. CECI: Compact Embedding Cluster Index for Scalable Subgraph Matching. In *Proceedings of the 2019 International Conference on Management of Data*, pages 1447–1462, 2019.

[35] Shaoyi Huang, Dongkuan Xu, Ian Yen, Yijue Wang, Sung-En Chang, Bingbing Li, Shiyang Chen, Mimi Xie, Sanguthevar Rajasekaran, Hang Liu, et al. Sparse Progressive Distillation: Resolving Overfitting under Pretrain-and-Finetune Paradigm. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 190–200, 2022.

[36] Heng Zhang, Lingda Li, Hang Liu, Donglin Zhuang, Rui Liu, Chengying Huan, Shuang Song, Dingwen Tao, Yongchao Liu, Charles He, Yanjun Wu, and Shuaiwen Leon Song. Bring Orders into Uncertainty: Enabling Efficient Uncertain Graph Processing via Novel Path Sampling on Multi-Accelerator Systems. In *Proceedings of the 36th ACM International Conference on Supercomputing*, ICS '22, New York, NY, USA, 2022. Association for Computing Machinery.

[37] Bappaditya Ray, Joel Jung, and Mohamed-Chaker Larabi. A low-complexity video encoder for equirectangular projected 360 video content. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1723–1727. IEEE, 2018.