NSF Large Scale Networking (LSN) workshop on Huge Data: A Computing, Networking and Distributed Systems Perspective

White Paper

Huge Data Analytics with Transparent In-Network Memory Computing

Ling Liu, James Bae, Wenqi Cao*, Semih Sahin*, Yanzhao Wu, Qi Zhang*

School of Computer Science
Georgia Institute of Technology
Atlanta, GA 30332, USA

Conventional distributed systems manage a cluster of computing nodes through cluster- wide coordination with respect to communication, computation and storage, represented by Hadoop Clusters and Spark Clusters. Huge data can be partitioned and distributed by partitions to different nodes in a cluster. Computation can be done in either local mode or distributed mode. In local mode, computation needs to handle both location computing and data movements to and from other nodes in the cluster. In distributed mode, the local computation needs to be synchronized through inter-node communications across the cluster. For huge data movements across a compute cluster, the inter-node communication for distribution synchronization can be prohibitively expensive.

In the age of big data powered Artificial Intelligence (AI) and Machine Learning (ML), Data has become the No. 1 in exponential growth, faster than big data hardware and software combined. Such huge data growth rate has further increased the demand for HPC and deep learning platforms to handle huge data at faster speed and analyze them in real time. Unleashing the potential of huge data for advancing science and engineering presents a pressing challenge to the computer systems and network systems research community. One approach to enabling and advancing huge data sciences and engineering research is to investigate architecture and algorithm for software-defined transparent in-network memory computing.

Unlike conventional distributed computing and cluster computing, where the distributed synchronization and communication control is tightly coupled with the data partitioning and data distribution, we advocate a clean separation of in-network huge data transportation (data plane) from in-network huge data computation control (control plan), coined as a software defined transparent computing paradigm. By separating computation control from data partitioning, data distribution and data movement in network, it allows each compute node in a cluster to adapt to unpredictable temporal variations of its working memory consumption, and be able to transparently utilize cluster-wide free memory across in-network executors, which are either residing on the same host or on remote nodes in the cluster. Such on-demand and transparent memory expansion and elastic in-memory computing paradigm provide stable peak time performance (w.r.t. latency and throughput) for huge data analytics and seamless response to the unpredictable temporal variations of memory demands. We below present two example developments towards the software defined approach to transparent in-network memory computing.

Host-coordinated Transparent In-Network Memory Ballooning. It is widely recognized that huge data systems and applications hosted in a virtualized compute cluster enjoy the peak-time performance when their working set fully fit in the working memory of each node in the cluster, demonstrated by Memcached, Redis, VoltDB and so forth. As soon as certain percentage of the working set no longer fits into the available memory of some executors, be it container, virtual machine or JVM, the huge data systems and applications will experience drastic performance degradation due to excessive paging or out of memory errors, even when there are sufficient free memory available in the other executors of the same cluster. By enabling transparent in-network memory ballooning, the conventional cluster computing systems and applications can enjoy faster huge data access at either local memory (DRAM) speed through cross-executor memory sharing or remote memory speed through cluster-wide remote memory sharing before resorting to the external slow storage I/O media. Concretely, the transparent in-network memory ballooning extends the virtual memory management into a hierarchical memory disaggregation architecture with local memory, cluster-wide remote memory, followed by external slower storage I/O. Such cluster wide in-network memory pool can be utilized for transparent in-network caching, transparent in-network paging, and transparent in-network file storage and access. Our preliminary research results with transparent in-network paging have shown that with the transparent in-network memory ballooning, we can enable huge data key-value systems and huge data machine learning workloads to enjoy stable peak time performance even when only partial working set (75% or 50% or 25%) can fit into the available working memory for some compute nodes in a Hadoop or Spark cluster. A number of technical challenges should be addressed for developing a high performance transparent in-network memory computing paradigm, such as how to provide efficient and reliable huge data placement in cluster wide network memory, how to transparently and seamlessly control the remote access with desired security and privacy guarantee, and how to balance the in-network memory workloads across nodes in a cluster.

Cloud-coordinated and Transparent in-network federated learning. Traditionally, training ML models requires all training data to reside in a central location and to be partitioned and distributed over a cluster of compute nodes through a centralized master coordinator, with Spark and Hadoop MapReduce as the two representative platforms. For huge data, the conventional platform suffers from prohibitively expensive communication and synchronization cost. At the same time, privacy concerns and legislations, such as General Data Protection Regulation (GDPR) and Health Insurance Portability and Accountability Act (HIPAA), have further hindered the collection of data collections from sensors on massive mobile devices (clients).

Federated learning has emerged as an attractive framework for transparent in-network memory computing, and is fueled by a number of big data companies, represented by Google, Facebook, Amazon, Apple. In a typical federated deep learning system, each data owner (participant or client) maintains its own data locally and follows a federated communication protocol where only updates of the model training parameters are shared with the trusted parameter server (federated server for parameter aggregation). Participants are also the workers and are responsible for training the same model on different mini-batches of the huge data (compute intensive tasks). In each training iteration, each participant sends its local parameter updates to the parameter server, typically hosted in the Cloud, which aggregates and maintains a set of shared parameters. In the next iteration round, the parameter server shares the aggregated parameters with each of the participants in the federated learning system and each participant updates its local parameters in the subsequent iteration. This distributed training process repeats in iteration rounds until the federated training reaches the pre-defined convergence condition. In a federated learning scenario, participants are heterogeneous compute nodes that communicate with the parameter server in either a client-server style or decentralized peer to peer style. We refer to this in-network collaborative model training paradigm a Cloud coordinated and transparent in-network federated learning.

Although in federated learning systems, the participants have high transparency and high autonomy for their local training data and the mobile devices only need to share the parameter updates from their local training to the federated server in the Cloud, several open technical challenges need to be addressed systematically and methodically, such as when synchronous federated learning is more beneficial than asynchronous federated learning? For synchronous federated learning, how to manage straggler workers, select drop out workers, and optimize communication overheads of sharing large number of high precision parameters, can we guarantee the security and privacy of federated learning with full transparency, while maintaining the desired training performance in terms of both training accuracy and overall training time (cost).

At Georgia Tech, we have several active projects that address the above challenges from different dimensions, aiming to promote a software defined approach to transparent in-network memory computing for high performance huge data powered machine learning and artificial intelligence.

Acknowledgement: Wenqi Cao (FaceBook, USA), Semih Sahin (Google, USA) and Qi Zhang (IBM TJ Watson) contributed to this research during their PhD dissertation research at Georgia Institute of Technology.