HPC DEVELOPMENT OF DEEP LEARNING MODELS IN SCIENTIFIC COMPUTING AND FINANCE

Allocation: Illinois/173.65 Knh
PI: Justin Sirignano[1]
Collaborators: Jonathan B. Freund[1], Jonathan F. MacArt[1]

[1]University of Illinois at Urbana–Champaign

EXECUTIVE SUMMARY

This allocation was used to support several deep learning projects across a variety of applications. The three applications were: (1) scientific computing, (2) modeling high-frequency financial data, and (3) image recognition. The first application develops and evaluates deep learning closure models for large eddy simulation (LES) models of turbulent flows. The deep learning LES models are trained on direct numerical simulation (DNS) data from the Naiver–Stokes equation. The DNS data sets are generated using Blue Waters and the deep learning LES models are trained using Blue Waters' GPU nodes. In the second application, deep learning models are trained and evaluated on massive high-frequency data sets. In the third application, which only used a small amount of the overall allocation, the asymptotics of deep learning models were analyzed on image data sets such as the Modified National Institute of Standards and Technology and Canadian Institute for Advanced Research 10 data sets.

RESEARCH CHALLENGE

Deep learning has revolutionized fields such as image, text, and speech recognition. Owing to this success, there is growing interest in applying deep learning to other fields in science, engineering, medicine, and finance. Blue Waters was used to develop and test deep learning methods and models for important applications in scientific computing and quantitative finance. The research team developed deep learning models for turbulence, which has been a longstanding challenge in computational science and engineering. The preliminary results demonstrate the ability of the machine learning-based models to predict turbulent flow energy spectrums more accurately than traditional turbulence models. In another project, the team developed a deep learning model for high-frequency financial data. Financial institutions have a strong interest in developing and using machine learning models for financial data and decisions.

METHODS & CODES

In the scientific computing project, the team developed a deep learning closure model for LES and generated DNS data sets. These DNS data sets were filtered and downsampled to yield appropriate data sets for training the deep learning LES model, which was parallelized across 24 GPU nodes. Synchronous gradient descent was used to train the deep learning model.

As part of this project, the research team also developed a new DNS/LES code, PyFlow, for training and simulation of the deep learning LES model. This code is GPU-accelerated and has potential to address modeling challenges over a wide range of flows. Similarly, the high-frequency financial data project used synchronous gradient descent (distributed across multiple compute nodes) to train deep learning models.

RESULTS & IMPACT

In the scientific computing project, the deep learning LES model is able to more accurately predict the filtered DNS data than traditional LES models. For example, in out-of-sample tests, the deep learning LES model more accurately reproduces the energy decay and energy spectrum as compared to traditional LES models such as Smagorinsky and Dynamic Smagorinsky. The use of deep learning and machine learning methods in scientific modeling has the potential to advance simulation and design in engineering.

WHY BLUE WATERS

Deep learning uses multilayer neural networks (deep neural networks) to build statistical models of data. This training of the deep learning model can be computationally intensive owing to both the large number of parameters and the large amounts of data. GPUs can be used to accelerate training of deep learning models. The researchers leveraged Blue Waters' large amount of GPU resources to develop deep learning models for applications in engineering and finance. Blue Waters XE nodes were also extensively used to generate DNS data sets, which was extremely computationally expensive and required parallelization over thousands of cores. The Blue Waters technical staff provided invaluable help throughout the project, including solving a number of technical issues related to deep learning computational frameworks such as PyTorch.

PUBLICATIONS & DATA SETS

J. B. Freund, J. F. MacArt, and J. Sirignano, "DPM: A deep learning PDE augmentation method with application to large-eddy simulation," submitted, 2019, arXiv: 1911.09145.

J. Sirignano and K. Spiliopoulos, "Mean field analysis of deep neural networks," submitted, 2019, arXiv: 1903.04440.

OPTIMIZATION OF A FIELD DATA PARALLEL OUTPUT LIBRARY

Allocation: Industry/3 Knh
PI: David E. Taflin[1]

[1]Tecplot Inc.

EXECUTIVE SUMMARY

TecIO–MPI is Tecplot, Inc.'s parallel data output library. It is designed to output to Tecplot's native file format the field data results of numerical simulations, such as those produced by computational fluid dynamics (CFD). Its initial implementation suffered from unacceptable parallel performance in some cases. In this work, the profiling software tools available on Blue Waters were used to identify and remedy the sources of this unacceptable performance. The researchers then ran test cases, scaling up to 912 nodes, to confirm acceptable performance and scaling. The results confirm order-of-magnitude reductions in output time and indicate that output times scale linearly with output file size.

RESEARCH CHALLENGE

Parallel simulation codes break large problems into smaller pieces and solve these smaller pieces simultaneously by using many CPU cores. At the end of the solution (and perhaps periodically during the solution), each core's portion of the overall solution must be written to disk, perhaps all to a single file. This parallel file output must be fast enough to not hinder the overall solution progress significantly. Thus, any code that performs this output must be optimized for parallel performance. But parallel optimization requires tools that can not only examine the execution time of blocks of code but can also identify wait states caused by communication delays among the multiple processes that run together to produce the solution or output. Further, to optimize a massively parallel code requires testing that code on hardware representative of the hardware on which it is intended to run. Blue Waters provided the answer to these challenges.

METHODS & CODES

The research team ran the TecIO–MPI library with a test harness that loaded data previously serialized to disk and then called into TecIO–MPI to output the data. The test harness was a stand-in for the sort of simulation code with which TecIO-MPI is designed to run. This allowed repeated tests of the software without the overhead of an actual CFD-type solver. The researchers performed parallel profiling with Cray's Performance and Analysis Tools, and data were written to a single file on Blue Waters' LUSTRE file system. Hotspots that were identified were recoded to improve performance, and the software was retested to ensure the fixes were effective.

RESULTS & IMPACT

The performance of TecIO–MPI was improved to the point that CFD-solver authors should find it acceptable—roughly the cost of a single solver iteration—and measured wall-clock output times appear to scale linearly with output file size. The file format the library outputs enables Tecplot to visualize on typical desktop workstations—via contour flooding, streamline generation, slicing, etc.—much larger files than can typically be handled on such machines by loading from disk only the data required to produce the artifacts. The memory required in this postprocessing scales with the number of solution cells raised to the two-thirds power, such that its advantage grows exponentially with larger file sizes.

WHY BLUE WATERS

Blue Waters provided access to both hardware and software required for this project to be successful. Its ability to scale problems to many CPU cores, and its use of the popular LUSTRE file system, ensured that the software was being tested in a representative environment, such that performance improvements seen in this work would also be seen by the intended users of this software. Blue Waters' Cray performance-measuring tools were indispensable in identifying performance bottlenecks and confirming their removal. And the technical support staff were very helpful with identifying the appropriate configuration of the compilers and profiling software required for this work.