GA

ML

DETECTION OF SILENT DATA CORRUPTIONS USING MACHINE LEARNING

Allocation: Exploratory/40 Knh
PI: Marc Snir[1]
Collaborator: Franck Cappello[2]

[1]University of Illinois at Urbana–Champaign
[2]Argonne National Laboratory

EXECUTIVE SUMMARY

Future supercomputers are expected to encounter more frequent bit-flips as the number of devices increases and their size shrinks; furthermore, one could reduce power consumption by tolerating more frequent errors. Some errors may escape the notice of the error detection mechanisms provided in hardware. To handle those, it may be necessary to detect errors in software. One promising approach is to periodically test the state of a long-running computation and identify "anomalous" states indicating that a bit-flip occurred.

The research team is using a convolutional neural network (CNN) as a detector. The CNN is trained with multiple examples of correct and incorrect computation states, next used as a classifier. Using this method, the team achieved a high detection rate, with an acceptable overhead. The method is practical and could be used on future systems should their rate of undetected bit-flips require software detection.

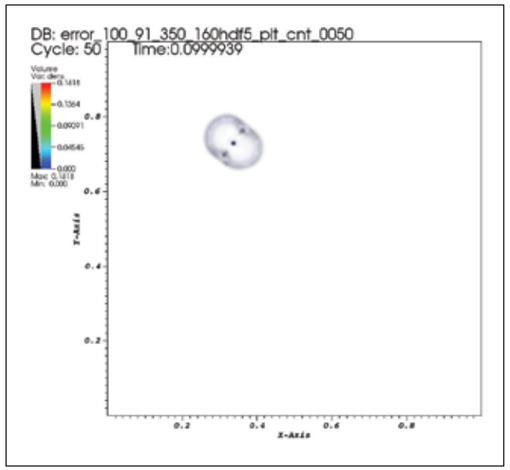

Figure 1: Error propagation after 50 iterations of the Sod application.

RESEARCH CHALLENGE

Future supercomputers are expected to encounter more frequent hardware errors, owing to the increase in the number of devices and the decrease in their size. Furthermore, one can decrease power consumption of supercomputers by tolerating more frequent errors. Some of these errors may not be detected and could corrupt the computation output. The research group's goal is to detect silent data corruptions by periodically running tests on the state of an ongoing computation.

METHODS & CODES

A single bit-flip in one variable of a large simulation can result in an error that propagates through the entire data set over time. Figs. 1 and 2 illustrate error propagation for two different iterative simulations. The research team's hypothesis is that machine learning can be used to detect such error patterns. The problem is that these error patterns are superimposed onto the correct computation state.

The team is treating the correct simulation as "noise" and is looking for the signal of a propagating error added to the noise. This is done by training a convolutional neural network (CNN) with multiple examples of correct and faulty simulations and then using the network as a detector. The training needs to be repeated once for each code, but the training procedure does not depend on the code.

RESULTS & IMPACT

Using this technique, the researchers have achieved a much higher detection rate than was previously achieved [1]. Furthermore, unlike previous methods, errors can be detected multiple iterations after they occurred so that the error detector need not be run at each iteration: The recall rate is around 90% and the overhead for running the detector could be as low as 1%. Follow-up work, to be published, aims at better understanding which errors do lead to a noticeable corruption of the final result, as only those need to be detected; and to extend these methods to work on 3D data sets. This research indicates that it will be possible to use future supercomputers efficiently, even if undetected bit-flips become common.

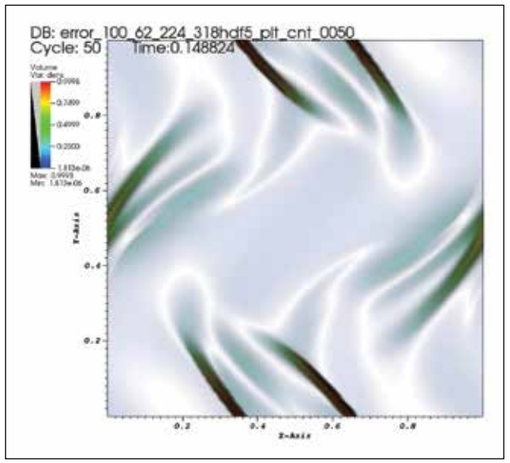

Figure 2: Error propagation after 50 iterations of the Orzag–Tang application.

WHY BLUE WATERS

GPUs are routinely used to accelerate the training of deep neural networks; such training is time consuming. A system with a large number of GPUs, such as Blue Waters, enables the research team to explore various alternatives at a much faster clip.