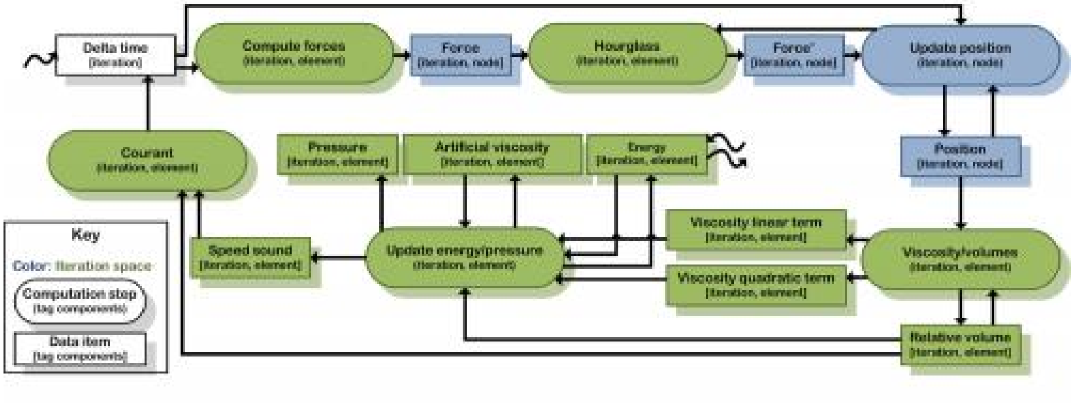
PHYS ORG

Scientists apply new graph programming method for evolving exascale applications

April 18 2014



This formal CnC graph was developed from an initial sketch of LULESH mapped on a whiteboard and embodies good software design practice.

(Phys.org) —Hiding the complexities that underpin exascale system operations from application developers is a critical challenge facing teams designing next-generation supercomputers. One way that computer scientists in the Data Intensive Scientific Computing group at Pacific Northwest National Laboratory are attacking the problem is by developing formal design processes based on Concurrent Collections (CnC), a programming model that combines task and data parallelism. Using the processes, scientists have transformed the Livermore Unstructured Lagrangian Explicit Shock Hydrodynamics (LULESH) proxy application code that models hydrodynamics (the motion of materials relative to each other when subjected to forces) into a complete CnC specification. The derived CnC specification can be implemented and executed using a paradigm that takes advantage of the massive parallelism and power-conserving features of future exascale systems.

Application performance on future exascale systems will be immediately impacted by massive parallelism, where many calculations are conducted simultaneously and solved concurrently, and restricted by energy consumption, heat generation, and data movement. While exascale systems are expected to have the computing power to affect broad areas of science and engineering research, applications developers will need to create code that takes advantage of the added complexity. By developing formal processes that capture data and control dependencies and separate computations from implementation issues, the complexities of exascale systems can be hidden, dramatically decreasing development cost and increasing opportunities for automatic performance optimizations.

Rather than plugging away at machines generating code via trial and error, initiating a CnC specification begins by manually depicting dataflow between software components and formalizing opportunities for analysis and optimization of parallelism, energy efficiency, data movement, and faults. For example, developing the CnC model for LULESH started with a whiteboard sketch at an application workshop. Domain experts with functional knowledge provided the application logic for the original assessment. After converting the sketch into a formal graph—yet before writing any code—the PNNL scientists were able to perform static analysis, apply optimization techniques, and detect bugs, reducing some costs commonly associated with development and testing processes.

"The formalization of scientific applications as graphs is extremely important and enlightening," said Dr. John Feo, director of the Center for Adaptive Supercomputer Software and Data Intensive Scientific

Computing group lead at PNNL. "In addition to providing a natural and obvious pathway for application development, we identified communications and optimization issues that could be addressed with added clarity before the computation steps were even implemented."

Ultimately, LULESH code was pared into chunks that corresponded to the formal CnC procedures. Then, the LULESH code was wrapped in CnC steps before executing the application to evaluate its correctness.

The CnC application method now is being applied to a second software code, MiniGMG, another compact geometric multigrid benchmark for optimization, architecture, and algorithmic research. PNNL's Data Intensive Scientific Computing group also is engaged in using LULESH to develop and test other tuning models.

Provided by Pacific Northwest National Laboratory

Citation: Scientists apply new graph programming method for evolving exascale applications (2014, April 18) retrieved 20 March 2024 from https://phys.org/news/2014-04-scientists-graph-method-evolving-exascale.html