The Need for Resilience Research in Workflows of Big Compute and Big Data Scientific Applications

February 4, 2014

Franck Cappello, ANL&UIUC and Tom Peterka, ANL

Projections and reports about exascale failure modes conclude that we need to protect numerical simulations and data analytics from an increasing risk of hardware and software failures and silent data corruptions (SDC) [1, 4]. At this scale, hardware and software failures could be as frequent as ten or more per day. According to [9], the semiconductor industry will have increased difficulty presenting software with an efficient dependable hardware layer when feature size will become lower than 10 nm (11 nm is projected in 2015-2017 according to Intel and NVIDIA). For workflows of computation and data analytics at extreme scale, the challenge is to produce correct results in the presence of potentially unreliable hardware and software.

After many workshops and reports on exascale resilience [2, 3, 5, 6, 10], the need for resilience of parallel computations at extreme scale (big compute) is widely accepted. Roadmaps give priority to improving checkpoint restart, developing new programming models and runtimes for resilience, developing resilient algorithms and focusing on detection to limit SDC as much as possible. These roadmaps were primarily concerned with ensuring that large-scale simulations complete and produce correct results. In other contexts such as clouds and grids (big data), many research results concern resilience of workflows, including computation, data storage, and data analytics. However, we are not aware of such roadmaps or research efforts specifically concerning resilience of workflows of simulation and data analytics (big compute + big data) in the context of scientific applications on extreme-scale platforms such as the future exascale systems.

Big compute and big data scientific workflows on extreme-scale platforms have characteristics that distinguish them from other common types of parallel/distributed applications like single-component scientific simulations, coupled numerical simulations, and workflows running on grids and clouds. Essentially, both the characteristics of the workflows and the nature of the platforms are different.

Big compute and big data scientific workflows connect producer and consumer parallel components in complex data flow graphs. Extreme-scale platforms will run these workflows using high-performance communication (including in-memory communication) between computational and data analytics components. Figure 1 shows an example workflow coupling a cosmology simulation with a small subset of associated data analytics. The ovals represent analysis and visualization programs that convert data to other forms (shown in squares). For example, the simulation code, *HACC* [7] produces raw particles that can be viewed directly with *ParaView*, a production visualization tool. Or, particles can be converted to a mesh tessellation through the *tess* [8] parallel library, which can be visualized or further resampled onto a regular grid with the *dense* parallel tool. Other serial utilities can operate offline on the tessellation and density fields.

These workflows are much less regular than standalone scientific simulations: components may use different computing and communication models; communication occurs at multiple levels (within components and between components); orchestrators organize the execution of the workflow, and data flow may be scheduled to increase performance.

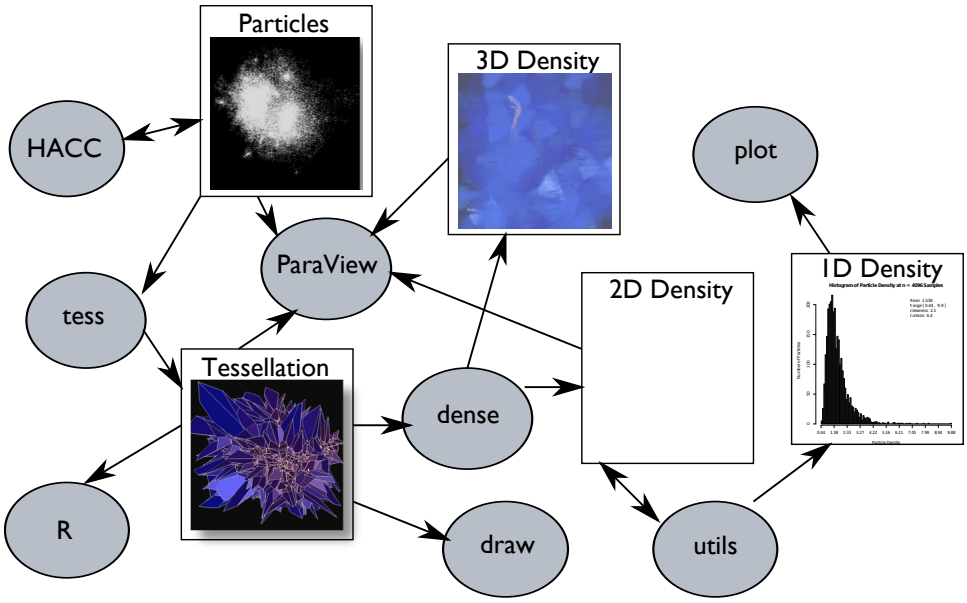

Figure 1: A workflow coupling a cosmology simulation with its associated data flow for the conversion of particle data to unstructured and regular grid analysis products. The ovals represent programs, some parallel and others serial, both tightly and loosely coupled. Raw and derived data products are denoted by squares.

Big compute and big data scientific workflows also differ from coupled simulations (for example multiphysics codes) because (i) workflows are fundamentally based on the producer consumer unidirectional communication model while coupled simulations involve bidirectional communications leading to more complex bidirectional dependencies (ii) communications between components take the form of data streams produced by the scientific simulation, and multiple stages transform the data in streams; in coupled simulations, communications occur typically in bursts followed by quieter periods (iii) components are of different nature: numerical simulations, data transformation, visualizations; (iv) ultimately at the end of a big compute and big data scientific workflows data is represented in compact ways that promote scientific understanding.

Other differences exist compared to workflows run on grid/cloud platforms. These platforms are typically loosely coupled, and data is stored to disk after each transformation, avoiding direct communication between components for performance and reliability reasons. Extreme-scale platforms are less heterogeneous; resources are supposed to be more reliable at a comparable scale; there are fewer security concerns (data are accessed within a single administration domain); there is less orchestration concern because resources are easier to reserve and allocate, and communication performance is orders of magnitude superior. In grid and cloud platforms, workflows execute over thousands of cores, while at extreme scale, we expect to execute workflows over millions of cores.

These differences along with the lack of research result concerning SDC effects on analysis products imply that there is a need to design new fault tolerance solutions for big compute and big data analytics workflows on extreme scale systems. Four main problems need to be addressed:

1) Understand the effects of SDC on the workflow results. We expect the probability of SDC to be higher at extreme-scale than it is in current platforms. While several results have been published concerning the impact of SDC in simulation applications, to our knowledge, there is no published research concerning SDC in workflows. Depending on the data product, the combination of resolution and location in the workflow may make some data products more sensitive to errors than others. For example, referring back to Figure 1, the raw particles and tessellation are high-resolution data, not visual or statistical summaries, and several hops separate them from sinks in the workflow. Does SDCs in these data affect more the end result than

SDC in data closer to the end of the workflow?. Do we observe like in simulation applications locations in the workflow where SDC have no effect (absorption effect) or in the contrary propagate and lead to detected failures (amplification effect)? Clearly, designing SDC mitigation techniques for such complex workflows requires first to understand their effects.
2) Establish clear response modes with respect to failure modes, including SDC. Depending on the failure type (including SDC) and on where it happens in the workflow, the user may have different expectations in terms of observed system behavior and result quality. For example, if the workflow produces an animation, it may be acceptable to drop several video frames because of a transient failure in a data transformation component; the large amount of compression and interframe coherence in movies may entirely hide the missing data.
3) Design workflow component coupling methods that respond to user needs: How to couple heterogeneous workflow components in an efficient way to maximize performance while at the same time providing failure containment that would avoid restarting the whole workflow for each transient failures within a workflow component. This problem is complex because on one hand performance can be gained by tighter coupling while at the same time resilience is improved by looser coupling.
4) Considering the three previous aspects, a difficult problem is to architect the right fault tolerance approach for each component, given a workflow, the fault model as well as the arsenal of available techniques for fault tolerance: checkpointing, replication, message logging, transactions, resilient algorithms, and their optimizations. This involves performing performance and energy measurements, developing performance/reliability/energy models, investigating optimization methods and designing software to help a user find the best combinations of techniques for his workflow.

References

[1] Franck Cappello, Al Geist, Bill Gropp, Laxmikant Kale, Bill Kramer, and Marc Snir. Toward exascale resilience. *Int. J. High Perform. Comput. Appl.*, 23(4):374–388, November 2009.

[2] John Daly. The Inter-Agency Workshop on HPC Resilience at Extreme Scale. Catonsville, MD, 2012.

[3] Nathan DeBardeleben, James Laros, John Daly, Stephen Scott, Christian Engelmann, and Bill Harrod. HighEnd Computing Resilience: Analysis of Issues Facing the HEC Community and PathForward for Research and Development, National HPC Workshop in Resilience. Arlington, VA, 2009.

[4] Jack Dongarra and Pete et al. Beckman. The international exascale software project roadmap. *Int. J. High Perform. Comput. Appl.*, 25(1):3–60, February 2011.

[5] Peter Kogge et al. Exascale computing study: Technology challenges in achieving exascale systems. 2008.

[6] Al Geist and Lucy Nowells. DOE Fault Management Workshop. Linthicum Heights, MD, 2012.

[7] Salman Habib, Vitaly Morozov, Hal Finkel, Adrian Pope, Katrin Heitmann, Kalyan Kumaran, Tom Peterka, Joe Insley, D. Daniel, Pat Fasel, Nick Frontiere, and Z. Lukic. The Universe at Extreme Scale: Multi-Petaflop Sky Simulation on the BG/Q. In *Proceedings of SC12*, Salt Lake City, UT, 2012.

[8] Tom Peterka, Juliana Kwan, Adrian Pope, Hal Finkel, Katrin Heitmann, Salman Habib, Jingyuan Wang, and George Zagaris. Meshing the Universe: Integrating Analysis in Cosmological Simulations. In *Proceedings of the SC12 Ultrascale Visualization Workshop*, Salt Lake City, UT, 2012.

[9] Chris Ramming, Mark Hill, Pat Lincoln, Steve Keckler, and Al McLaughlin. Resilient Computing Frameworks Workshop. In *Outbrief set of slides*, Marines Memorial Club and Hotel, San Francisco, 2013.

[10] Marc Snir and Bob Wisniewski. ICIS Workshop on Addressing Failures in Exascale Computing. Park City, Utah, 2012.