IMPROVING CHECKPOINT RESTART WITH LOSSY COMPRESSION

Jon Calhoun, University of Illinois at Urbana-Champaign
2014-2015 Graduate Fellow

RESEARCH SUMMARY

Checkpoint restart is a fundamental component of HPC applications and is used to recover the application from system failure and to extend execution beyond a single time allocation. Checkpointing to the parallel file system is prohibitively expensive, but is often unavoidable due to its non-volatility. Multi-level checkpointing schemes have been developed to utilize the memory hierarchy to improve checkpointing times, therefore reducing expensive file system operations. The next generation of HPC systems such as ANL's Aurora and ORNL's Summit are expected to increase the amount of total system memory by 5-9x over current 10 and 27 petaflop machines. However, the file system bandwidth will remain around 1 TB/s. Failure to consistently scale file system bandwidth can have negative effects on HPC applications. Applications that are scalable on current systems may not be scalable on future systems as file system operations will become a larger fraction of runtime. As machine size continues to scale, the system's mean time between failure (MTBF) is expected to decrease, resulting in more checkpoints being taken [1] resulting in a further reduction in time spent on computation. Reducing data movement is key for efficient usage of HPC systems.

Data compression techniques can be used to reduce dataset size and come in two forms: lossless and lossy. Lossless compression schemes provide small compression factors for floating-point data sets [2]. Lossy compression schemes [3, 4], can provide significantly higher compression factors at the expense of a small but controllable error added to the data. For lossy compression to be used in HPC applications, understanding this compression error is paramount. Selection of this compression error is naively a trial and error based approach and requires detailed knowledge of what is simulated to judge acceptance. Expressing the compression error in mathematical terms that would be familiar to application scientists will help adoption by providing a generic methodology to interpret the compression error.

Many HPC applications approximate the solution of partial differential equations (PDE) or ordinary differential equations (ODE) by using a numerical method. This numerical method approximates the solution to a given accuracy dependent on the spatial discretization of the problem. The accuracy of the simulation as expressed by the numerical methods and spatial discretizations can be leveraged in the selection of the lossy compression error tolerance. For example, if we know our simulation is accurate to 1e−4, then a numerical solution to this problem, u^h, is indistinguishable from a slightly perturbed solution, $\tilde{u}^h = u^h + \epsilon$, if $\epsilon < 1e{-}4$. This observation creates an upper bound on the compression error tolerance such that restart from a lossy checkpoint is indistinguishable from that of a lossless checkpoint. If we seek higher compression factors by setting $\epsilon < 1e{-}4$ (adding in error), we can assign compression error tolerances to the accuracy of related numerical methods, e.g. varying order Runge-Kutta methods. This selection methodology is outlined in Fig. 1.

We test our compression error selection methodology using plasma physics code PlasComCM. PlasComCM solves the Navier-Stokes equations using Runge-Kutta 4. Our problem simulates flow past a fixed cylinder and is accurate to 1.8e−5. We use SZ version 0.5.14 [3] with compression error tolerance ϵ=1e−5 and restart from a lossy compressed checkpoint every 5000 time-steps. We plot the inf-norm of the error

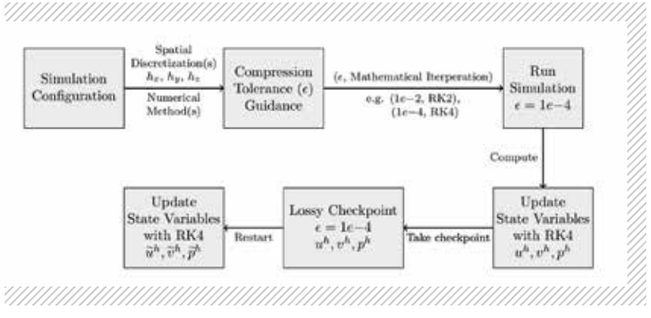

FIGURE 1: Overview of the method to select lossy compression error tolerance. Compression error tolerance can be related to accuracy of numerical methods by knowing the simulation's spatial discretization.

in each state variable against time (Fig. 2). Error in the state variables never exceeds the accuracy of the simulation's mathematics. There is an accumulation of error with each additional checkpoint, but it decreases due to non-periodic boundaries allowing the perturbed flow to exit the domain. Propagation and removal of error in the momentum variable, between the time-step after the first lossy restart until the time step just before the second restart is observed (Fig. 3). Initially, error propagates until around time 310, after which the error begins to exit the domain. Exploitation of boundary conditions and physical properties yields knowledge for selecting an optimal compression error. For problems that attenuate error, it may be possible to compress to a tolerance above the accuracy without affecting the outcome of the simulation.

Lossy compression is an exciting new research direction that shows great promise in reducing the volume of data transmitted by HPC applications. Compared with standard lossless compression, lossy compression can achieve higher compression factors, but at the expense of adding error into the simulation. The results discussed above, and our ongoing research, suggest this error can be interpreted mathematically. Thinking of lossy compression error in this way shifts its application from a trial and error process to the well-understood domain of understanding numerical error.

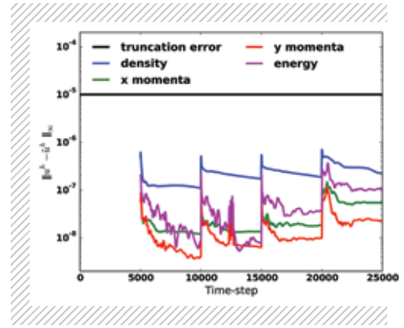

FIGURE 2: Inf-norm between numerical solution u^h and compressed numerical solution $\tilde{u}^h$. There is no error before time-step 5000, as the simulation has yet to be restarted. At each restart, we see a large spike in the error, but error remains less than the simulation's accuracy (truncation error). This error is partially removed by our selection of boundary conditions.

WHY BLUE WATERS

Blue Waters represents a balanced HPC system where fast compute nodes are complimented by I/O bandwidth of more than 1 TB/s. This **advantage**, combined with the heterogeneous nature of the XK nodes, makes Blue Waters an **ideal** system for testing next generation optimizations for checkpoint restart.

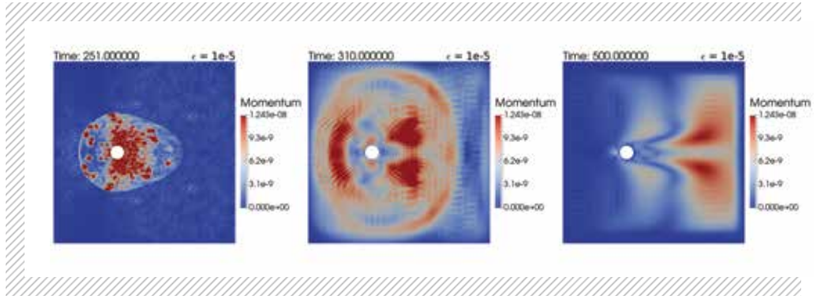

FIGURE 3: Location of error in the domain after first restart from a lossy compressed checkpoint, time-steps 5001-9999. After restart, error is confined near the cylindrical object (left plot). As time evolves, error is propagated though the domain (center plot), and eventually exits the domain due to our selection of boundary conditions (right plot).

John Calhoun is in the fourth year of his Computer Science Ph.D. studies at the University of Illinois at Urbana-Champaign. He expects to graduate in August 2017 and aspires to teach at a university or work at a national laboratory.

"Working on the Blue Waters system has given me valuable experience on a flagship HPC system. Without access to such a system, it would be difficult to test the new resilience ideas I am investigating. The research that I am conducting on Blue Waters serves as a stepping-stone to what I will accomplish post-graduation."