Scaling Uncertainty Quantification Studies to Millions of Jobs

Tamara Dahlgren, David Domyancic, Scott Brandon, Todd Gamblin, John Gyllenhaal, Rao Nimmakayala, and Richard Klein
Lawrence Livermore National Laboratory, Livermore, CA

SC15 Austin, TX | hpc transforms.

MOTIVATION

Department of Energy laboratories are increasingly interested in Uncertainty Quantification (UQ) studies to understand their simulation codes.

UQ Studies:
- evaluate input parameter impacts on outcomes
- assess likelihood of certain outcomes
- involve hundreds to millions of smaller simulation runs, each using a hundred plus processors
- can take months using current HPC resources

Key resource constraints:
- Resource management (or job) limits
- Front-end node (FEN) memory
- System limits

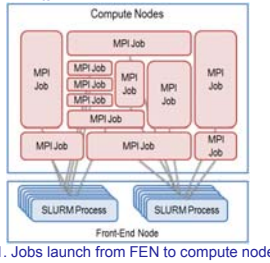

Figure 1. Jobs launch from FEN to compute nodes

Cielo's [LANL] resource manager ran the scheduling node out of memory at ~220 jobs.

Launching an ensemble on FEN quickly reaches system limits:
- Process count
  - srun uses 4 process ids (threads) per job
- Open sockets
- Memory
  - srun requires 15 MB per job
  - Python requires 50 MB per job

About 96 TB would be required to run 1.6 million simultaneous single-core jobs.

Increasing system limits and hardware resources solves only part of the problem. For example, raising the following on Sequoia increased the number of concurrent jobs by only an order of magnitude:
- descriptors and maximum process limits to 16384
- max number of socket connections to 8192
- memory on FEN from 32 GB to 256 GB

Still constrained to about 20k concurrent jobs

Figure 2. LLNL's Sequoia is one of the top 3 supercomputers in the GRAPH 500 and TOP500 lists.

LLNL UQ Pipeline (UQP)

Scientific workflow system, written primarily in Python, for UQ studies supporting:
- sampling high dimensional uncertainty spaces
- managing execution of simulation ensembles
- analyzing ensemble output
- constructing surrogate models
- performing sensitivity studies
- incorporating observational data
- performing statistical inferences
- estimating parameter values and probability distributions

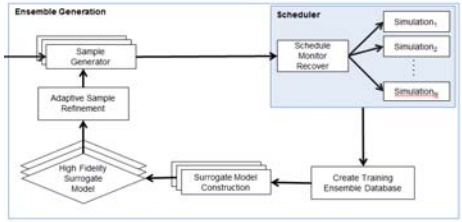

Figure 3. The UQP Scheduler manages ensembles

The UQP is actively used in production on machines at LLNL and LANL. On average, 27 UQP users have logged about 50.9 million CPU hours each year on LLNL production machines since 2013.

CRAM

- Packs a large number of MPI-based simulations into a single simulation run
- Compresses command line arguments, environments, and working directories into an input file
- May not require changes to the simulation code but it *must* link to the CRAM library, which
  - intercepts MPI calls
  - creates separate communicators in MPI_Init for each job
  - translates calls with MPI_COMM_WORLD to PMPI calls with local job's communicator

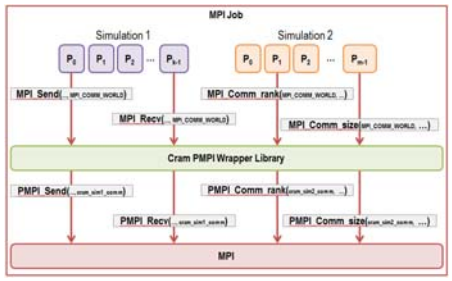

Figure 4. CRAM virtualizes MPI by intercepting and translating MPI calls

UQP with CRAM

UQP was extended to support:
- Splitting simulation runs into one group per CRAM input file
- Creating working directories in parallel
- Combining, in parallel, and writing the simulation results file
- Deleting temporary files and directories in parallel

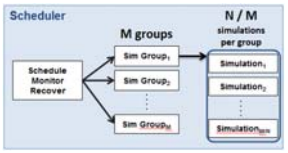

Figure 5. UQP packs simulation runs into M groups, one run per CRAM input file

```
Name:                cram.in
Number of Jobs:      8192
Total Procs:         8192
Cram version:        2
Max job record:      5125

Job command lines:
   0   1 procs   <exe> . 0.267279620002 0.518037995151
   1   1 procs   <exe> . 0.847661383995 0.139621280273
   2   1 procs   <exe> . 0.826606558447 0.409541056501
   3   1 procs   <exe> . 0.923455311230 0.108897478042
   4   1 procs   <exe> . 0.207357823719 0.481913129554
   5   1 procs   <exe> . 0.433447576895 0.702422378681
   6   1 procs   <exe> . 0.539133899953 0.369065382346
   7   1 procs   <exe> . 0.929428586592 0.507210539392
   8   1 procs   <exe> . 0.109160112935 0.500807383742
   9   1 procs   <exe> . 0.879604817416 0.663545105995
  ...
[8182 more]
```

Figure 6. Inspecting a CRAM input file

Massively Parallel UQ Ensembles

Running millions of single-task jobs
- Single-core application to produce Monte Carlo estimate of π based on ratio of areas
- X and Y are independent "UQ" variables
- UQP subdivided Sequoia into 192 512-node partitions
  - Each Sequoia node has 16 cores ➔ 1,572,864 cores
- Packed 8 K application runs into each job to test scaling limits of CRAM tool

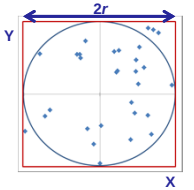

Figure 7. Monte Carlo example

$$\pi = \lim_{n\to\infty} \frac{4}{n} \sum_{\substack{i=1 \\ x_i, y_i \in [-r,r]}}^{n} \begin{cases} 1 & \text{if } \sqrt{x_i^2 + y_i^2} < r \\ 0 & \text{if } \sqrt{x_i^2 + y_i^2} \geq r \end{cases}$$

Achieved significant gains in throughput and concurrency
- Launched all 1.6 M application runs to fill all Sequoia cores
  - With CRAM: less than 40 minutes
  - Without CRAM: more than 4.5 days
- Achieved 400,000 concurrent simulations due to short execution time

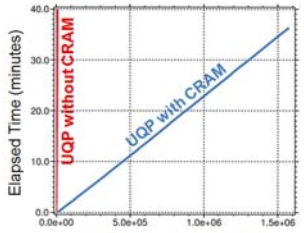

Figure 8. UQP launched 715 applications per second with CRAM versus four per second without it

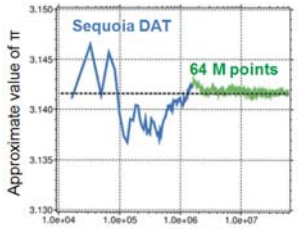

Figure 9. A more accurate estimate would have required 64 M points

Large-scale parameter studies
We are preparing to perform another run with a real, multi-physics simulation code
- Hydrodynamics simulations using Kull, developed at LLNL and SNL
- UQP will subdivide the machine into 4 or 16 partitions
- CRAM will pack 256 or 512 jobs into each application run

The goal of this demonstration will be to utilize CRAM to maximize the number of simulations we can run within our machine allocation.

SUMMARY

Extending UQP with CRAM allowed us to:
- use all of Sequoia – 1.6 million cores – with a single core application
- launch 1.6 million applications in under 40 minutes instead of 4.5 days
- achieve two orders of magnitude improvement in the time required to launch a single-core application

CONCLUSIONS

- UQP+CRAM can launch up to one UQ application per core
- HPC systems require adjustments to limits and increases in memory on front end (or scheduling) nodes in order to run large numbers of small jobs at scale

FUTURE WORK

Providing scientists with the ability to successfully complete ensemble simulations on modern HPC machines requires the ability to detect and recover from errors. CRAM does not currently provide that capability. So more work is needed to automate the assessment, recovery, and rescheduling of individual simulation runs spawned using the CRAM extension of the UQP.

ACKNOWLEDGEMENTS

We thank the following people for their contributions to this work:
- Don Lucas for his encouragement to create a poster as well as his suggestions and feedback on the content; and
- Livermore Computing for providing the resources and support for our Dedicated Access Time (DAT) run on Sequoia.

BIBLIOGRAPHY

[1] J. Tannahill, D. D. Lucas, D. Domyancic, S. Brandon, and R. Klein. Data Intensive Uncertainty Quantification: Applications to Climate Modeling. In *Proceedings of the 2011 companion on High Performance Computing Networking, Storage and Analysis Companion*. ACM, 2011.

[2] CRAM. https://github.com/scalability-llnl/cram.

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

LLNL-POST-676116