Climate Science Performance, Data and Productivity on Titan

Benjamin Mayer*, Patrick H. Worley*, Rafael Ferreira da Silva§, Abigail L. Gaddis*

* Oak Ridge National Laboratory, Oak Ridge, TN, USA

§ Information Sciences Institute, University of Southern California, Marina Del Rey, CA, USA

{mayerbw,worleyph,gaddisal}@ornl.gov, rafsilva@isi.edu

***Abstract*—Climate Science models are flagship codes for the largest of high performance computing (HPC) resources, both in visibility, with the newly launched Department of Energy (DOE) Accelerated Climate Model for Energy (ACME) effort, and in terms of significant fractions of system usage. The performance of the DOE ACME model is captured with application level timers and examined through a sizeable run archive. Performance and variability of compute, queue time and ancillary services are examined. As Climate Science advances in the use of HPC resources there has been an increase in the required human and data systems to achieve programs goals. A description of current workflow processes (hardware, software, human) and planned automation of the workflow, along with historical and projected data in motion and at rest data usage, are detailed. The combination of these two topics motivates a description of future systems requirements for DOE Climate Modeling efforts, focusing on the growth of data storage and network and disk bandwidth required to handle data at an acceptable rate.**

***Index Terms*—Climate Science, High Performance Computing, Performance Analysis, ACME.**

I. Introduction

Scientific workflows that run on high performance computing (HPC) systems can be very expensive in terms of both time and data [1]–[4]. In the Climate community, to complete a single instance or ensemble of high-resolution simulations can take on the order of six to nine months and generate hundreds of Terabytes (TB) of data. This time span is due to many different types of tasks and factors. Obviously compute time is a factor, but so is the time spent in the queue waiting to run, due to both competing jobs and machine down time. Having produced model output is only a partial goal; there are many other data management issues that take significant amounts of time, including generating analysis and archiving model output to long-term storage. These time spans will increase over the coming years due to the increase in spatial and temporal model resolution leading to better quality simulations. In this work, we examine this workflow process with the help of instrumentation, and look for areas where improvements can be made.

There are several scales at which development happens in climate models, from an individual equation or feature to how that feature interacts with the component model, and finally how all of the components interact together. The amount of computation and timeline for each of these start small on the development end, where modifications are happening on the minute to hour scale and seconds to minutes of computations to full scale simulations that take months and tens of millions of hours of computing. At every scale of development issues are identified. The faster one can do experiments at all scales, and especially at the large production scale, the faster the quality of the model can be improved.

A. Related Work

Performance evaluation and optimization of HPC applications is common practice, enabled by a variety of available tools and libraries (e.g., TAU [5], CrayPat [6], HPCToolkit [7], gptl [8], Vampir [9], etc). However, science productivity rarely comes from running a single application, but rather requires executing a series of tasks. These other tasks also consume resources and take time, which can be a significant factor [10] in the total time to solution. There are a number of science endeavors that exhibit these characteristics. Examples include those with a large data fusion component, e.g. astronomical data, those with a large data analysis component, e.g. data from large instruments such as accelerators, or those conducting large parameter sweeps, where monitoring and controlling the subtasks is itself a critical task.

One non-climate example that has taken an approach similar to what is described, and proposed, here, is Bellerophon. Bellerophon [11] is an n-tier software system built to support a production-level HPC application called CHIMERA, which simulates the temporal evolution of core-collapse supernovae. The CHIMERA team's workflow management needs are met with software tools that enable real-time data analysis and visualization, automate regression testing, dynamically generate code repository statistics, monitor the current status of several supercomputing resources, and integrate with other workflow tools utilized by the group. In addition, monitoring tools built into Bellerophon allow users to keep tabs on important quantities such as current physical time, wall time, and simulation progress.

The large computing centers also provide tools and methodologies for supporting workflows, and for center-wide management. For instance, the DOE National Energy Research Scientific Computing Center (NERSC) actively shows queue wait time [12] and provides history functionality. They are doing queue time data collection on a large scale. Workload archives [13]–[17] are widely used for research in distributed systems to validate assumptions, to model computational activity, and to evaluate methods in simulation or in experimental

conditions. These workloads mainly capture information about job executions, but lack detailed information for the applications, such as the separation between the initialization and run phases. Analyses of HPC workload characteristics including system usage, user population, and application characteristics have been conducted in [2], [18]. However, these analyses are focused on the overall workload, and not on the performance of a specific application.

B. Problem Statement

The climate modeling process involves a sequence of steps: 1) model configuration, 2) compilation, 3) model execution, 4) diagnostic analysis, 5) archive of model output and analysis to tape (e.g., HPSS), and 6) publication of model output to a content management system (e.g., Earth System Grid Federation (ESGF)). As part of this process there are some operations that use a shared file system and others that require data transfers (DTN) between file systems.

In projects like the DOE Ultra High Resolution Global Climate Simulation project [**?**] and the Accelerated Climate Modeling for Energy (ACME) project [19], where the goal is to exercise and analyze high resolution climate models, large amounts of simulation output are generated. The process of generating these data can take a significant amount of time, and not just the expected large amount of compute time. The workflow for producing a finished climate product involves many steps across several machines, and involves many people, with an associated cost in both people and pre- and post-processing time.

Given the nearly year long time frame for performing a 20 year high resolution simulation or running several hundred simulation years of moderate resolution simulation [20], there are likely places where optimization to reduce time to solution can happen. One of the goals of this work is to highlight where time is being spent in the workflow so that the process can be optimized.

Data volume in climate science is on the order of hundreds of TB of data produced yearly. For example, given this amount of data, the analysis that needs to be performed, transfer for distribution and archiving operations all take significant amounts of time. Automation of some tasks might help with this workload.

II. Examining Performance

A. Data Captured

In this work, we examined performance data for runs of the Community Earth System Model (CESM) and of the DOE ACME branch of the CESM over the period 2012–2015 on the Jaguar (Cray XT5) and Titan (Cray XK7) supercomputers hosted at Oak Ridge Leadership Computing Facility (OLCF). The CESM is equipped with workflow and model computer performance instrumentation, and this was further augmented for these studies. This augmentation is native to ACME, and we will refer to both ACME and this modified version of the CESM as ACME in the rest of this paper. We recorded various performance-related attributes (timings and configuration information), as well as timespans for some pre- and post-processing events. Captured configuration information includes model specific parameters, simulation period, and number and location of processing elements, MPI tasks, and OpenMP threads. Examples of recorded values are rates at which the components of the coupled model integrated and total throughput rate. Examples of recorded timesspans are when the model entered the queuing system, when a run started, etc. The information that was recorded changed over the 4 year period being examined, leading to a varying population size in the data presented here, dependent on the particular attribute we are examining.

ACME is a coupled model made up of several components. The components correspond to the physical domains to be modeled, i.e. atmosphere, ocean, land, sea ice, land ice, and rivers (river transport model). A *case* is a specific model configuration with input files, resolution, choice of actively simulating a physical domain or just reading in data, selection of physical processes and numerical methods to enable in each component, etc. The data called out in single cases here is for two resolutions: `t341` and `t85`, with `ne120` and `n30` cases also being run. `t341` is 0.35° resolution (512x1024 latitude/longitude points) and `t85` is 1° (128x256 latitude/longitude). Both of these use a spectral Eulerian numerical method for evolving the dynamical equations in the atmosphere, and both use 26 vertical levels. The `ne120` resolution is 0.25° and `ne30` is 1°. The `ne` numbers correspond to using a spectral element based numerical method for the atmosphere dynamics, and also both use 30 vertical levels. When a case is referred to as *high resolution* this currently refers to the `t341` and `ne120` resolutions, while *low resolutions* refers to the `t85` or `ne30` resolutions. The other part of the case name describes the model configuration. For example a leading `F` indicates that data read from a file are used for the ocean forcing communicated to the atmosphere, as opposed to `B` configurations where the ocean forcing is produced by an active ocean simulation. The `AMIP` vs `1850` runs are specific model configurations that differ in their initial input files and internal parameters. The desired simulation duration for high resolution runs is on the order of a few decades, while 100+ years is the target for the low resolution simulations.

Some of the data presented in this paper are from model runs done on the Jaguar system, which does not use Graphical Processing Units (GPU). The Titan system includes both a multicore processor and a GPU accelerator in each computational node, but the GPUs were not used in the ACME runs reported on here.[1]

B. Data Capture Infrastructure

The ACME model has an extensive system of control scripts that manage correctness checks and workflow tasks related to running the model. These scripts have been modified to include code that records the timestamp of particular events,

[1]The ACME science cases require a monotone limiter for tracer transport, which had not been ported to the GPU accelerators at the time of these experiments.

and archives this and other performance data in a project-specific central repository. Some of these data are captured only when the model run completes successfully, resulting in a selection bias in the presented performance data.

C. Analysis Techniques

Computer performance is typically measured at the climate simulation level using the inverse time metric Simulated Year per Day (SYPD), that is, how many simulation years can be computed for the particular configuration being measured if it ran for 24 hours straight. This metric is convenient for the climate scientists when specifying performance requirements as it relates directly to required productivity. This is also a sufficient metric if only application performance is of concern, as when tuning, or if the simulation is running on a dedicated system. In this paper, we also use a time to solution metric that takes into account all steps to a solution when run on shared resource systems.

Total time to solution is made up of several components (Fig. 1): Configure, Build, Queue (Fig. 1.a), Run (Fig. 1.b), Diagnostic Analysis, Transfer, and Archive. There are several other tasks that are required for provenance and output data archiving in a complete workflow, but the results of these tasks are typically not consumed directly and thus are not time critical for the initial results and will be ignored in our workflow optimization analysis. Another issue that can take significant time occurs when the model terminates abnormally. Currently, failed jobs are not automatically resubmitted to the queue, and progress is halted until a user resubmits the model (Fig. 1.c). Another situation when time can be spent waiting for a human to intervene is when a model finishes running and analysis and data transfers are required (Fig. 1.d).

The Configuration step is primarily limited by the principal investigators, who can take months to years to decide how to configure the model so as to best test a hypothesis. In contrast, the computer time required for configuration is on the order of several minutes. The build step takes on the order of tens of minutes, with 20–30 minutes being typical. This step needs to occur when there are changes to core system libraries, changes to the number of processors that are used, and for every new science configuration (*case*). Currently, builds happen infrequently except during debugging or performance optimization. Even though the configure and build steps are on the critical path and time blocking to a solution, given the infrequent nature of each step (typically once per case) and their relatively small cost, they will be ignored for this analysis.

D. Results

In this section, we examine model run time (Fig. 1.b), various aspects of queue time (Fig. 1.a), and the impact of human factors (Fig. 1.c). Data transfer time and archive time will be dealt with in the data section.

***Performance vs time (single cases)*:** Here we examine the computer performance of the model over time to see if there are changes or if it is fairly consistent. Seemingly small configuration changes can make large differences. When there is a case being run to answer a science question the model is typically run in a fixed configuration, including with a fixed number of MPI tasks and OpenMP threads. Comparing performance between configuration cases does not make sense unless it is the identical configuration either on the same system or when comparing performance between systems. What we will be looking at in this sub-section is several individual cases that have been run many times over a period of time, and comparing performance internally over time.

The initialization phase of ACME does a parallel I/O read of restart files and does a further distribution of that data, using a subset of the MPI tasks for I/O and then reordering the data from the output format to that used during computation before distribution to the rest of the tasks. The run phase of the program advances the simulation in time, serially in the time direction but exploiting parallelism spatially during the computation for each time step. Model history is also output periodically during the run phase, with data being gathered and reordered by the I/O tasks before being written out. Table I shows the statistical summary for both phases on three cases for a total of 247 runs.

	Initialization	Run
Mean (s)	86.31	5,276.98
Standard deviation (s)	87.63	628.34
Coefficient of variation	1.02	0.12
Count	118	118
Max (s)	962.22	6,817.39
Min (s)	55.10	76.17

(a) `t85 FAMIP`

	Initialization	Run
Mean (s)	547.82	17,899.15
Standard deviation (s)	206.55	869.54
Coefficient of variation	0.38	0.05
Count	42	42
Max (s)	1,735.69	19,990.46
Min (s)	407.01	16,822.68

(b) `t341 FAMIP`

	Initialization	Run
Mean (s)	585.14	16,264.20
Standard deviation (s)	178.44	1,006.05
Coefficient of variation	0.30	0.06
Count	87	87
Max (s)	1,450.79	19,686.51
Min (s)	455.04	15,054.04

(c) `t341 F1850`

TABLE I: Statistical summary of the ACME initialization and run phases.

The initialization time for the `t341` cases are similar with `t85` being smaller. This makes sense as there is less data to read in for the lower resolution `t85` case. By visual inspection (Fig. 2), there appears to be a good deal of variation in the `t341` cases, but further inspection with the coefficient of variation in the `t341` cases is 0.37 and 0.30 respectively, with t85 being 1. This indicates that there is less relative variability for the `t341` cases compared to `t85`.

Fig. 3 shows the distribution of run time for the three cases. There are few outliers in the run time dataset. The

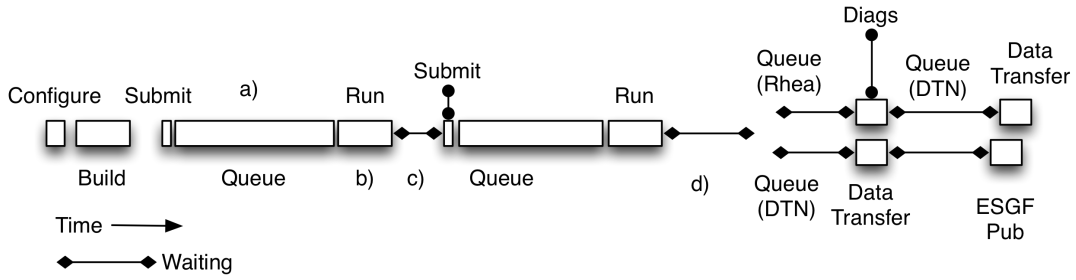

Fig. 1: Manual experiment and diagnostic workflow.

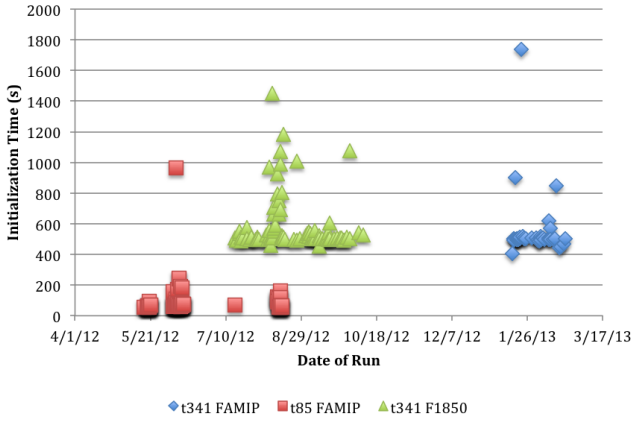

Fig. 2: Initialization time for the `t341` cases and the `t85` case (247 runs).

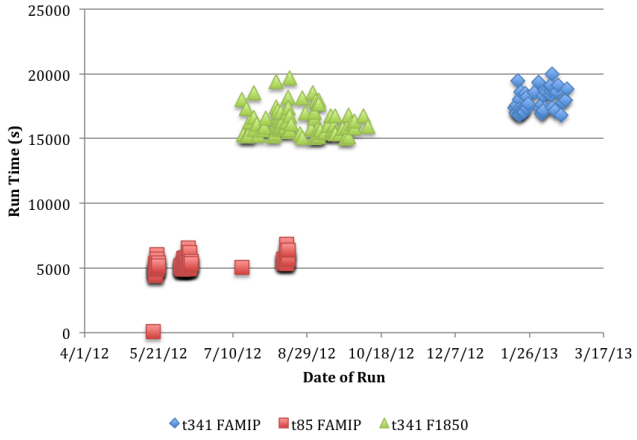

Fig. 3: Run time for the `t341` cases and the `t85` case (247 runs).

low run time outlier in the `t85` case is near the beginning of the run sequence and terminates almost immediately. Likely, this is early termination from a case misconfiguration. In comparing the variability in the performance between the run time and the initialization time, we note that the coefficient of variations are much lower (0.05, 0.11, 0.06 respectively) compared to the initialization times. The ratio between run time and initialization time is 30 for the `t341` cases, and 60 for the `t85` case showing that initialization time is only 1.5% to 3% of total run time. Even though the initialization time has a large amount of variability, the contribution to total time is small, thus the contribution to variability is also small.

***Queue time vs job size (bulk statistics)*:** Fig. 4 shows 485 runs from 2014 (1/11 to 11/20). From this diagram we can see that almost all requests are under 20k core hours and they are run within a day of being submitted. For the large requests there appears to be a uniform distribution with large range for how long the jobs take to be serviced, anywhere from seconds to a few weeks.

There is a question about what caused the larger runs to have such a range of queue wait times. Looking at the plot of queue wait time compared to the date of submission (Fig. 5), we can see several interesting things. There is a spike in wait times in March/April. The DOE INCITE (The Innovative and Novel Computational Impact on Theory and Experiment program) allocation period runs from January 1st to December 31st of

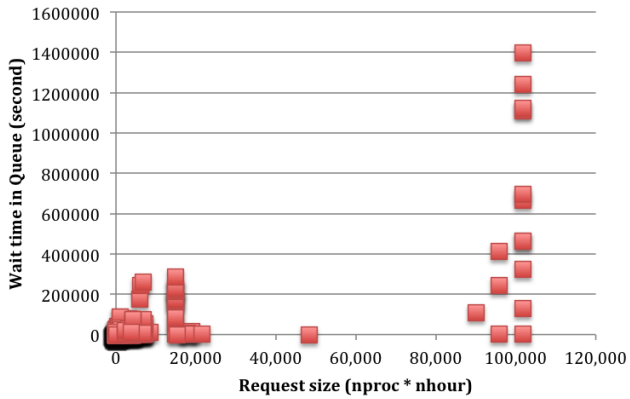

Fig. 4: Job queue time per job size (485 runs from 2014).

every year, with DOE ALCC (Advanced Scientific Computing Research Leadership Computing Challenge program) running July 1st to June 30th. Conventional wisdom suggests that projects are trying to use large portions their allocation near the end of those periods. However the spike we are seeing is not associated with either of those timeframes. What this queue wait time spike appears to be is projects doing large runs in preparation for major conferences such as SuperComputing (with full paper submission due April 11th, 2014), or Gordon Bell Prize attempts that were due May 1st.

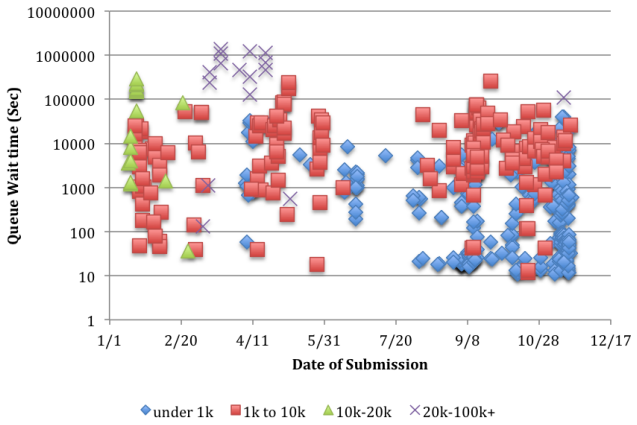

Fig. 5: Job queue time per submission date (485 runs from 2014).

***Queue request time vs actual run time*:** Queue time is on the order of a day while requested run time is on the order of 6 hours. The model run time is the component that is making progress, so we want to make sure that we are using as much of the requested time once it has been given to the process. Looking at the same `t341` and `t85` cases discussed above, the mean amount of requested time used for the cases was 74% for `t85`, 88% for `t341 FAMIP`, and 81% for `t341 F1850`.

For the `t85` case with a 2 hour request window using 74% of the request window means that there is 31 minutes left at the end of the run. Looking at the two standard deviations for the initialization and run time gives 24 minutes for 95% of the runs to finish. There is an additional buffer of 7 minutes past the two standard deviations model run time. Given the variability and the short request window a mean of 74% utilization is fairly good. A similar analysis with the t341 based models shows a similar result for `t341 FAMIP`, but an extra 34 minutes for `F1850`.

***Time spent waiting for people i.e. not running, not in queue*:** Another possible cause of additional time in completing a case is when a job fails and needs to be restarted manually, or when a job is stopped on purpose. We define the metric for the time spent waiting for people to respond as the timespan between when one job is expected to end and when the next job in a series is created. A positive value indicates that the next job is submitted after the expected run time of the projected job, which characterizes a delay in the execution. A negative value indicates that the next job in the sequence is submitted before the expected job deadline. The former is the expected outcome, since ACME resubmits itself to the queue once a run is succeeded, and on production infrastructure the walltime is often overestimated. Fig. 6 illustrates a situation where the metric would yield a negative value.

Using the same 485 instances of production runs as previously, we identified 7 occurrences of positive values (delays) from the above analysis, which represent about 1.4% of the total case runs. The delays range from hours to a few days in most cases, with two instances of multi-week delays. These

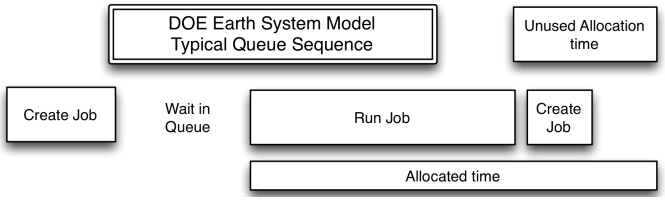

Fig. 6: Typical sequence of events in queue.

delays have been explained as occasions when the model run did not complete successfully, but, when resubmitted, runs to completion and continues running. Possibly this is caused by a hardware fault, but no information was archived that allows us to confirm this at this point in time. (At the time of a failure, the job id can be used to query system logs and identify possible causes for failures.) Even if these were hardware failures, they occurred only intermittently and had a low impact on throughput. The small hours to day delay likely is from when the person minding the run is not alerted that the run has failed and it is restarted only when they notice that it is no longer running. The longer delays are from a case where model output analysis showed that the simulation was starting to drift in an undesirable direction. The simulations were then halted while the source of the issue could be identified and a solution formulated.

III. Diagnostic Analysis

While the model is running, analyses are computed to track the progress of the simulation. These can be, for example, single values such as RESTOM, which measures the rate at which energy is entering or leaving the model, summary statistics and plots of cloud layers, and/or extent of ice sheets, all to ensure that the simulation is not "nonphysical". It is important that these analyses are carried out in a timely manner so corrections can be made, or the simulation can de discarded.

There are standard analysis packages used for each of the components. The available data is exclusively for the Atmosphere diagnostics, of which there are two software packages. One is the Atmosphere Model Working Group (AMWG) diagnostics package, which is based on NCAR Command Language (NCL) analysis tools [21]. The other package replicates the behavior of AMWG with additional calculations being performed. This second package is based on the python CDAT library, which is part of the UV-CDAT package [22].

	Time (hr)
NCL AMWG	6.45
UV-CDAT Total	58.74
UV-CDAT climatologies	55.79
UV-CDAT diagnostics	2.95

TABLE II: Analysis operations timespan.

These operations can run in parallel, but are critical for feedback about how the model is progressing.

IV. Transfer Time

As part of the end-to-end workflow model, output data needs to be archived as well as being published to the ESGF system.

The allocation of long term storage and infrastructure for publication is often located at a different site from the compute allocation. Hence, data transfer across different networks is required. We have two sets of transfers: 1) *wide area*–between two national laboratories; and 2) *local area*–internal to a single laboratory but between security zones and different disk systems. All transfers under one gigabyte have been removed from the data set as they have little impact on total time and are typically log files of negligible scientific use other than for reporting such as this paper. Table III shows the statistical summary of the two data transfer sets.

	Rate (Mbit/s)	Size of Transfer (GB)
Mean	591.0	430.7
Standard deviation	621.8	872.2
Coefficient of variation	1.05	2.02
Count	34	34
Max	2,2210.1	5,110
Min	1.58	1.0

(a) Transfer from one national laboratory to another (wide area)

	Rate (Mbit/s)	Size of Transfer (GB)
Mean	642.3	2166.2
Standard deviation	405.7	2259.7
Coefficient of variation	0.63	1.04
Count	38	38
Max	1330.8	5930
Min	19.3	7.2

(b) Transfer internally across security zones (local area)

TABLE III: Statistical summary of data transfers.

Typical setup of these transfers is a model case needing to be copied. A transfer session per model component or transfer per file type (multiple filetypes per component) are started. These transfers run in parallel.

At this point we don't have enough data to provide statistical information, but enough to make an estimate. A high resolution simulation with coupled ocean model can produce 130TB of data. If all of this information is transferred at one time, instead of as it is generated, and if there is only one transfer process working, this example would take 19.6 days. If there are four parallel transfers the transfer time drops to 5 days. Transferring the data as it is generated does not add to the total amount of time to run an experiment as the transfers can happen in parallel to the case running.

A. Current Dataset Size

The amount of data generated by climate related projects over the course of their year long duration can be quite sizeable. From three climate projects one of the authors participated in, final data volume saved to tape archive was of 161 TB, 53 TB, and 252 TB.

V. Performance Summary

To this point we have detailed several contributions to total experiment run time. The summary of that information is presented here:

- *Queue time:* progress through the queue happens in under a day in 94.6% of the cases in our record. There is some variability around busy times of computer usage;
- *Model performance* is fairly consistent. Approximately 20% of the variability comes from the model initialization phase with the remainder in the rest of the model which includes numerical algorithms, communication, and writing of restart and history files;
- *Diagnostic Analysis:* the process consumes from a quarter day to two days;
- *Transfers* in parallel after simulation is complete take $\sim$5 days. Transfers as the model is running happen in parallel with the simulation.

VI. Projections

In this section, we examine parameters and planned changes that impact overall workflow performance. In particular, we observe how changes to the model, along with the platforms that run the experiments, can influence the experiment performance.

A. Relationship between data size and compute time

The amount of computing time awarded to a project has an impact on the ability of the project to generate data. In particular, the above mentioned projects received: 33.5 million hours (Mh) for `t341 FAMIP`; 116 Mh for `t85 FAMIP`; and 50 Mh for `t341 F1850`. These were allocation hours on Jaguar. The data intensity in TB per million allocation hour for each project was: 4.8, 0.5, and 5.0, respectively. The two projects with high data intensity for the amount of allocation were focused on high resolution production runs and looking at data output at a high frequency with respect to simulation time.

B. Changes in simulation impacting compute intensity

The ACME climate model is changing atmospheric models from `CAM4` to `CAM5`. `CAM5` is more computationally expensive than `CAM4` due to enhanced physics, increased aerosols, and improved cloud properties, which significantly increases the amount of time simulated (i.e., more compute intensive). In practice, `CAM5` is 2–4x more computationally expensive for the same amount of data, with the range of performance depending on the exact configuration [23]. Also impacting the computation data balance, the DOE community standard configuration is moving to higher spatial resolution (1° to 0.25°). In the atmosphere, the theoretical computational impact of the resolution change represents an increase of 16x (4 times the resolution in the horizontal dimensions). The number of vertical levels is independent to the horizontal resolution, and it is only modified for experiment concerns. In practice, when comparing the performance of `CAM5 ne30` to `ne120` configurations, there is a 27x difference in performance. Combining this with the 16x increased size of the output data fields, the amount of required computation is once again scaling faster than the data output, in this case by nearly 1.7x given the single sample size for this estimate.

Additional physical details can be revealed by increasing the temporal resolution of the output. Increasing the output frequency allows more physics to be simulated from the governing equations as opposed to being represented by physical parameterizations. This leads to more realistic interactions between the different simulated scales in the simulation. Examples include precipitation, wind speed, tropospheric moisture convection, Atlantic storm tracks, and eddy fluxes. For instance, increased output frequency (e.g., daily or hourly) may be required in simulations where the number of hurricanes is examined. These properties lead to the ability to generate better simulations, and do more detailed analysis, which leads to better founded paper or a greater volume of papers. How far the balance is pushed towards gathering more data is largely controlled by available storage resources, with the later being an ongoing discussion.

C. Machine Changes

Future generations of supercomputers at the Leadership Computing Facilities are aiming to be 5-10x more capable than current generation machines, with the follow on generation another 5-10x again. [24] The percentage allocation that Climate Science has on these large systems has been fairly consistent historically.

From the Summit announcement we have seen machine computational performance scale up with I/O rates staying constant, albeit with a change in technology. If the delivered performance to rated performance stays constant with these changes in technology we will continue to see widening gaps between the computation and I/O. However, looking back at the model changes we see that the model is changing in a similar way, with a larger increase in computation than in I/O.

D. Projections of compute and data given projected machine speeds

In the previous section the transition from CAM4 to CAM5 gave a 2-4x increase in computational demands, while the resolution change needed 27x the amount of computation with a 16x increase in data volume. To analyze higher resolution model behavior the I/O output volume needs to increase, but is limited by available storage space.

VII. Automation

Much of the reporting (simulation status) and ancillary tasks (publication of data, archival storage) for these experiments can happen in parallel with the simulation, but currently consumes large amounts personal time while being time sensitive. To alleviate the time demands on staff, improve consistency and reduce need for high speed sub-systems automation has been introduced into the experimental process.

A. First Attempt

OLCF operates as a moderate security data processing facility. This imposes restrictions such as using two factor authentication to login, submit jobs, or start data transfers from external locations. Also, there are no reliable facilities for running daemon processing internally to OLCF, motivating early attempts to automate the ACME workflow within OLCF to use a novel one-sided workflow (see Fig. 7). This method consisted of commands being issued outside of OLCF from the Compute and Data Environment for Science (CADES) infrastructure with one time passwords and the processes running in OLCF doing automated reporting of status. This arrangement has been abandoned due to the inability to directly gather information about process status, this being critical when exceptions occur. Other considerations include the inability to start transfers that moved simulation output from OLCF to CADES Open Research in an automated fashion, and the dependence on fragile scripting to carry out complex tasks across several machines.

B. Current Path

Pegasus is a workflow management system that is targeted at large scale workflows on large resource. It uses a file based task and flow configuration system with a web based UI for status and reporting. The Pegasus system is highly portable, performance oriented, scaleable to large numbers of tasks and resources utilized, automatically captures provenance, provides reliability in workflow execution and has error recovery.

We have encoded the core ACME workflow (Configuration, Build, Run and periodic Analysis) into a Pegasus Workflow as a demonstration of capability. This simulation has run successfully at NERSC. At OLCF and CADES Open Research workflow nodes have been installed into the infrastructure on an experimental basis with the intent of transitioning to production ready systems. The ACME workflow is currently being ported to these new Pegasus workflow instances.

As demonstrations of capability and portability for the ACME project are made, additional capabilities will be added to the workflow such as the data transfers, ESGF publication and archiving to HPSS.

C. Automated Workflow Impacts

Up to now we have been doing mass processing of data that has required days to weeks to transfer data between systems and store it to HPSS. With automation, treating this as a single event where a large volume of data is moved through the system and injected to the end points can end. Instead as progress is being made each small piece of data can be handled as it is being generated.

Taken very approximately we see that the model runs once per day, with a requested run time of 6 hours, actually running for 4.5 to 5 hours of the 6 hour allocation. Half an hour of that time is spend doing I/O at a rate of approximately 50MB/s. The network transfer rates are about 80MB/s, thus needing 20 minutes per day to stream the data to the destination machine. The OLCF HPSS system ingests data at around 200MB/s, again only needing minutes once the proper tapes are loaded. Moving to a processing flow where each step is incrementally updated results in lower demand on hardware and human resources.

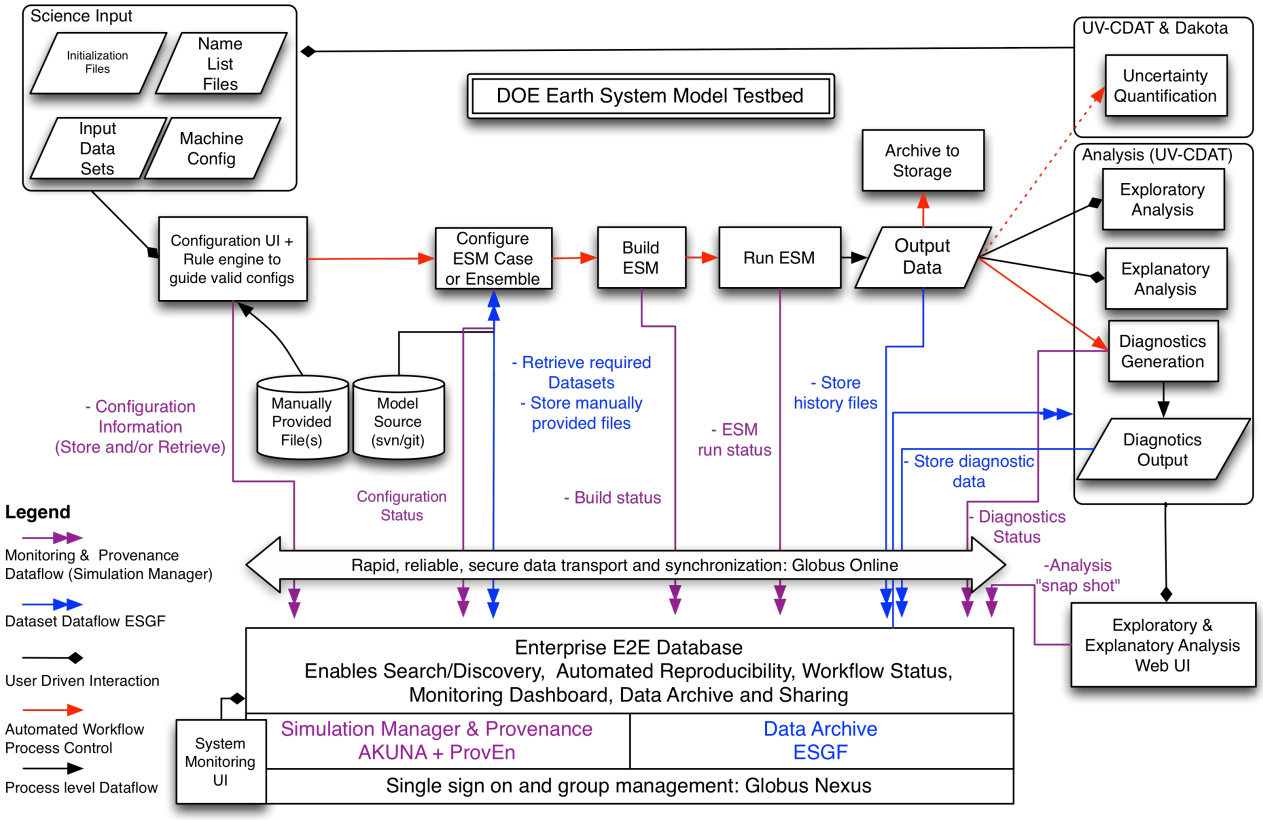

Fig. 7: One sided workflow

VIII. Conclusion and Future Work

The ACME model is changing to become more compute and data intense with the ratio of the two leaning towards becoming more compute intense. Machine performance is moving in a similar way. As there is little information at this point on future machine performance it is impossible to quantify how the changes in machine balance compare to model balance, but they are moving in similar directions.

Increase the number of places where performance information is captured, such as the transfer and publication times. Increase the number of fields captured Gather data and build an analytic model for increased queue request time vs longer queue wait time

Reduce run time of applications: Optimize run time where appropriate, Automate where appropriate

Acknowledgements

This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world- wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

This work was supported by DOE under contract #DE-SC0012636, "Panorama—Predictive Modeling and Diagnostic Monitoring of Extreme Science Workflows" and by the Office of Biological and Environmental Research as part of the project "Accelerated Climate Modeling for Energy".

We thank Ashley Barker, Christopher Fuson, Judy Hill, Gideon Juve, Eric Lingerfelt, Vickie Lynch, Ross Miller, Matt Norman, Suzanne Parete-Koon, Duane Rosenberg and Brian Smith for their valuable contributions.

References

[1] G. Mehta, E. Deelman, J. A. Knowles, T. Chen, Y. Wang, J. Vöckler, S. Buyske, and T. Matise, "Enabling data and compute intensive workflows in bioinformatics," in *Euro-Par 2011: Parallel Processing Workshops*. Springer, 2012, pp. 23–32.

[2] W. Tang, N. Desai, D. Buettner, and Z. Lan, "Job scheduling with adjusted runtime estimates on production supercomputers," *Journal of Parallel and Distributed Computing*, vol. 73, no. 7, pp. 926–938, 2013.

[3] R. Ferreira da Silva, G. Juve, E. Deelman, T. Glatard, F. Desprez, D. Thain, B. Tovar, and M. Livny, "Toward fine-grained online task characteristics estimation in scientific workflows," in *8th Workshop on Workflows in Support of Large-Scale Science*, 2013, pp. 58–67.

[4] R. Ferreira da Silva, G. Juve, M. Rynge, E. Deelman, and M. Livny, "Online task resource consumption prediction for scientific workflows," *Parallel Processing Letters*, vol. accepted, 2015.

[5] S. S. Shende and A. D. Malony, "The TAU parallel performance system," *International Journal of High Performance Computing Applications*, vol. 20, no. 2, pp. 287–311, 2006.

[6] S. Kaufmann and B. Homer, "Craypat-cray x1 performance analysis tool," *Cray User Group (May 2003)*, 2003.

[7] L. Adhianto, S. Banerjee, M. Fagan, M. Krentel, G. Marin, J. Mellor-Crummey, and N. R. Tallent, "Hpctoolkit: Tools for performance analysis of optimized parallel programs," *Concurrency and Computation: Practice and Experience*, vol. 22, no. 6, pp. 685–701, 2010.

[8] J. Rosinski, "General purpose timing library (gptl): A tool for characterizing performance of parallel and serial applications," *Cray User Group (May 2009)*.

[9] W. E. Nagel, A. Arnold, M. Weber, H.-C. Hoppe, and K. Solchenbach, "Vampir: Visualization and analysis of mpi resources," 1996.

[10] W. Chen and E. Deelman, "Workflow overhead analysis and optimizations," in *Proceedings of the 6th workshop on Workflows in support of large-scale science*. ACM, 2011, pp. 11–20.

[11] E. J. Lingerfelt, O. Messer, S. S. Desai, C. A. Holt, and E. J. Lentz, "Near real-time data analysis of core-collapse supernova simulations with bellerophon," *Procedia Computer Science*, vol. 29, pp. 1504–1514, 2014.

[12] "NERSC," https://www.nersc.gov/users/queues/queue-wait-times.

[13] "Parallel workloads archive," http://www.cs.huji.ac.il/labs/parallel/workload.

[14] A. Iosup, H. Li, M. Jan, S. Anoep, C. Dumitrescu, L. Wolters, and D. H. J. Epema, "The grid workloads archive," *Future Gener. Comput. Syst.*, vol. 24, no. 7, pp. 672–686, 2008.

[15] C. Germain-Renaud, A. Cady, P. Gauron, M. Jouvin, C. Loomis, J. Martyniak, J. Nauroy, G. Philippon, and M. Sebag, "The grid observatory," *IEEE International Symposium on Cluster Computing and the Grid*, pp. 114–123, 2011.

[16] R. Ferreira da Silva and T. Glatard, "A science-gateway workload archive to study pilot jobs, user activity, bag of tasks, task sub-steps, and workflow executions," in *Euro-Par 2012: Parallel Processing Workshops*, ser. Lecture Notes in Computer Science, I. Caragiannis, M. Alexander, R. Badia, M. Cannataro, A. Costan, M. Danelutto, F. Desprez *et al.*, Eds., 2013, vol. 7640, pp. 79–88.

[17] R. Ferreira da Silva, W. Chen, G. Juve, K. Vahi, and E. Deelman, "Community resources for enabling and evaluating research on scientific workflows," in *10th IEEE International Conference on e-Science*, ser. eScience'14, 2014, pp. 177–184.

[18] D. L. Hart, "Measuring teragrid: workload characterization for a high-performance computing federation," *International Journal of High Performance Computing Applications*, vol. 25, no. 4, pp. 451–465, 2011.

[19] "ACME: Accelerated climate modeling for energy," http://climatemodeling.science.energy.gov/projects/accelerated-climate-modeling-energy.

[20] K. Evans, D. Bader, G. Bisht, J. Caron, J. Hack, S. Mahajan, M. Maltrud, J. McClean, R. Neale, M. Taylor, J. Truesdale, M. Vertenstein, and P. Worley, "Mean state and global characteristics of the t341 spectral community atmosphere model," *CESM Atmosphere Working Group*, 2012.

[21] NCAR, "Atmosphere working group diagnostics package," https://www2.cesm.ucar.edu/working-groups/amwg/amwg-diagnostics-package.

[22] L. Cinquini, D. Crichton, C. Mattmann, J. Harney, G. Shipman, F. Wang, R. Ananthakrishnan, N. Miller, S. Denvil, M. Morgan *et al.*, "The earth system grid federation: An open infrastructure for access to distributed geospatial data," *Future Generation Computer Systems*, vol. 36, pp. 400–417, 2014.

[23] S. Tilmes, "Cam4/cam5 comparison," in *Chemistry-Climate Working Group Meeting*. CESM Annual Workshop, 2012.

[24] B. Bland, "Present and future leadership computers at olcf," *OLCF User Group Conference Call December 3, 2014*.