Load-Adaptive Continuous Coupled-Simulation Ensembles with DataStorm and Chameleon

Hans Walter Behrens, Mao-Lin Li, Ashish Gadkari, Yash Garg, Xilun Chen, Sicong Liu, K. Selçuk Candan
Arizona State University
Tempe, Arizona 85281–3673

***Abstract*—Recent developments in the literature have illustrated the practicality and advantages of coupled simulation ensembles, which empower sophisticated causality explorations between interrelated but technically-distinct models. However, in addressing the challenges of allocating a limited computational budget in a near-infinite potential simulation space, new technical challenges arise in harmonizing desired theoretical allocations with the concrete reality of server instantiations within a compute cluster. In this work, we describe DataStorm, a system for defining, deploying, and executing continuous coupled-simulation ensembles for disaster response, built on Chameleon. We also present a software interface layer for declarative control over cluster state, from meta-level instantiation information to user-level application control. We also provide a qualitative assessment of the Chameleon testbed, and the advantages and disadvantages encountered in the course of this investigation, and suggestions for future improvements.**

***Index Terms*—systems simulation, grid computing, big data applications, scientific computing**

I. Introduction

Designing cyber-infrastructures to support decision making in many critical applications is difficult due to the inherent sparsity of available data. This is caused by the difficulty of sensing of complex systems in their entireties and the fact that making decisions requires information about not only what has happened in the past and what is happening in the present, but also about likely (or unlikely) future events, which are inherently unknown and cannot be sensed. Thus, cyber-infrastructures that support understanding of complex systems with only sparse observational data may not be effective. Data driven models and computer simulations that take into account available domain knowledge, in addition to past data and observations, represent a promising approach in understanding and predicting complex dynamic processes (such as the evolution of disasters) and effectively planning responses through a diverse set of intervention measures. Creating such simulation ensembles, however, is difficult: complexity of systems implies that we need to generate and maintain a large number of simulations, each with different parameter settings, corresponding to different, and plausible, scenarios. Critically, many complex systems require multiple models that reflect different components of the system and simulations across these complementary models need to be integrated into a single coupled simulation.

Funding for this research was provided by the National Science Foundation under the proposal "DataStorm: A Data Enabled System for End-to-End Disaster Planning and Response." (NSF Award No. 1610282).

Simulation models often require extensive domain knowledge and expertise to produce realistic results, and the number and variety of such models matches the profusion of specialized domains to be predicted. However, despite interrelationships between models, such as that between flooding (hydrological) and hurricanes (meteorological), linkage between models was a tedious, manual process.

To address this challenge, we are developing DataStorm, a system for creating simulation ensembles through model linkage [1]. Through the creation of a workflow-based continuous simulation system, DataStorm provides an extensible framework for linking spatio-temporal simulation models.

Simulation ensembles of interrelated but domain-distinct models have also been previously studied in the literature. [2] examined a scenario linking epidemic transmission models with a variety of different human mobility simulators in order to produce more-accurate predictions of epidemic spread.

For a heterogeneous simulation, where each constituent model may take tens or hundreds of individual variables, the possible parameter space for the combined system is extremely large. Given simulation time constraints and a limited computational budget, the sampled simulation space will therefore by extremely sparse. [3] previously examined the impact of sparse simulation space sampling, by leveraging the decomposition process to better use available (sparse) results.

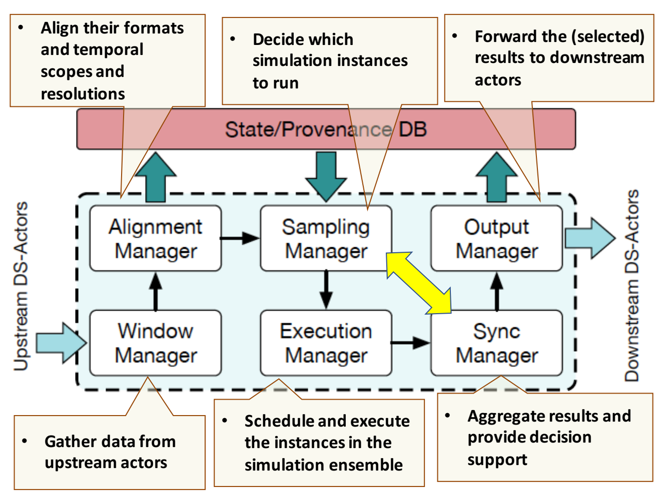

Fig. 1. A brief outline of the model orchestration process.

II. Research Challenges: Creation of Continuous Coupled-Simulation Ensembles on a Cluster

Generating potential scenarios whose simulation would improve the accuracy of the overall ensemble is a difficult task. Moreover, once selected, these simulation scenarios must be executed efficiently on a computing cluster, whose instances must adapt in order to execute specialized simulation models.

Additionally, due the continuous and elastic nature of the simulation load, model and instance requirements will rapidly evolve over time. Sampling budgets may change over time for a specific simulator, across multiple simulators, or even according to utility/fit functions derived from currently-available results. Aligning cluster resources to support these changing requirements is an ongoing challenge.

Consider a simulator that receives updated real-world data as an input to a previously-executed model; this model will likely absorb most of the cluster resources while the data is incorporated. Similarly, early in a simulation, users may generate several plausible outcomes immediately, and refine their respective likelihoods over time in a balanced way.

As part of our work, we aim to answer the following research questions:

1) Is real-time cluster adaptation for evolving heterogeneous simulation ensembles practical?
2) Does this adaptation reduce overall cluster execution time?
3) Can adaptation overhead be reduced to improve efficiency?

III. Implementation Details

A. Deployment Configuration

Technical requirements for the core orchestration system are minimal; three fixed instances fill the roles of cluster control, data storage, and visualization access respectively. Although intermediate and provenance-related data storage can be considerable, these challenges are extraneous to the challenge of simulation deployment and rebalancing.

As each simulator requires different underlying software systems, dependencies, versioning, and hardware capabilities, configuring and deploying these systems pose a challenge of configuration management.

B. Chameleon

To facilitate the automatic creation and deletion of simulation instances, integration with the OpenStack API of the Chameleon system is necessary. This is accessible through the OpenStack Python client, and is subsequently wrapped by additional middleware layers to simplify interaction.

The computational capabilities of the testbed are sufficiently varied to satisfy the requirements of all models that we evaluated. However, since the OpenStack portion of the Chameleon testbed does not support instances with GPU integration, models which require OpenCL or CUDA computation would not be suitable for integration with this system. The prescheduled nature of the GPU instances does not match well with the requirements of real-time instance balancing.

C. Software Artifacts

The initial implementation of DataStorm has been published [1], and its source code is available [4].

We also intend to publish, in conjunction with our next public release, cluster control middleware to provide a feature-rich declarative-style layer for OpenStack-based deployments.

IV. Chameleon Analysis

A. Advantages and Disadvantages

Overall, the Chameleon testbed provides a flexible infrastructure suitable for the implementation and experimentation phases of our work. During this project, we experienced a number of positive and negative aspects of the testbed.

OpenStack's API was successfully leveraged in the creation of a simplified cluster-control interface, which was crucial for integration into the workflow/orchestration system.

Individual contributors' access to data was secured through Chameleon's robust key management systems. Cluster communications were transferred on the cluster intranet, a crucial feature offered by the testbed.

The time required to spin up servers was greater than expected, and required the use of pre-configuration and snapshot recreation. Snapshot restoration times significantly reduced the waiting times observed during image-based instance creation, by roughly 1 order of magnitude.

Reliability issues with the network backplane caused frequent roadblocks when remote accessibility of the cluster was lost. Technical support staff affiliated with the testbed were quick to resolve the issues when they arose. Outage frequency necessitated improvements ot the fault tolerance of the system, a beneficial side effect for real-world deployments.

B. Suggestions and Improvements

Increases in reliability and fault tolerance by the underlying system architecture would be a welcome improvement for the Chameleon testbed. This is the easiest and most obvious course of action that the maintainers can undertake.

However, it could prove beneficial to provide a lower level of access between the user level currently available and the administrative level used by the maintainers. This access would expose a number of new tools for users to address common failure modes without opening a ticket. To ensure that such access does not undermine system reliability further, a certification process could be instituted for interested users.

V. Acknowledgement

Results presented in this work were obtained using the Chameleon testbed supported by the NSF, under the Chameleon projects CH-817461 and CH-820361.

References

[1] H. W. Behrens *et al.*, "Datastorm-fe: a data-and decision-flow and coordination engine for coupled simulation ensembles," *VLDB*, 2018.

[2] S. R. Poccia *et al.*, "Simdms: Data management and analysis to support decision making through large simulation ensembles," in *EDBT*, 2017.

[3] X. Li *et al.*, "M2td: multi-task tensor decomposition for sparse ensemble simulations," in *ICDE*, 2018.

[4] Y. Garg, "Datastorm release." https://github.com/EmitLab/DataStorm-Release, 2018.