frontiers | Frontiers in High Performance Computing

TYPE Original Research
PUBLISHED 01 May 2024
DOI 10.3389/fhpcp.2024.1360720

OPEN ACCESS

EDITED BY
Anshu Dubey,
Argonne National Laboratory (DOE),
United States

REVIEWED BY
Xingfu Wu,
Argonne National Laboratory (DOE),
United States
Jay Lofstead,
Sandia National Laboratories (DOE),
United States

*CORRESPONDENCE
Scott Callaghan
scottcal@usc.edu

RECEIVED 23 December 2023
ACCEPTED 19 February 2024
PUBLISHED 01 May 2024

CITATION
Callaghan S, Maechling PJ, Silva F, Su M-H, Milner KR, Graves RW, Olsen KB, Cui Y, Vahi K, Kottke A, Goulet CA, Deelman E, Jordan TH and Ben-Zion Y (2024) Using open-science workflow tools to produce SCEC CyberShake physics-based probabilistic seismic hazard models.
Front. High Perform. Comput. 2:1360720.
doi: 10.3389/fhpcp.2024.1360720

Using open-science workflow tools to produce SCEC CyberShake physics-based probabilistic seismic hazard models

Scott Callaghan[1]*, Philip J. Maechling[1], Fabio Silva[1], Mei-Hui Su[1], Kevin R. Milner[1], Robert W. Graves[2], Kim B. Olsen[3], Yifeng Cui[4], Karan Vahi[5], Albert Kottke[6], Christine A. Goulet[2], Ewa Deelman[5], Thomas H. Jordan[7] and Yehuda Ben-Zion[1]

[1]Statewide California Earthquake Center, University of Southern California, Los Angeles, CA, United States, [2]United States Geological Survey, Earthquake Science Center, Pasadena, CA, United States, [3]Department of Geological Sciences, San Diego State University, San Diego, CA, United States, [4]San Diego Supercomputer Center, University of California, San Diego, San Diego, CA, United States, [5]Information Sciences Institute, University of Southern California, Marina del Rey, CA, United States, [6]Pacific Gas and Electric Company, San Francisco, CA, United States, [7]Department of Earth Sciences, University of Southern California, Los Angeles, CA, United States

The Statewide (formerly Southern) California Earthquake Center (SCEC) conducts multidisciplinary earthquake system science research that aims to develop predictive models of earthquake processes, and to produce accurate seismic hazard information that can improve societal preparedness and resiliency to earthquake hazards. As part of this program, SCEC has developed the CyberShake platform, which calculates physics-based probabilistic seismic hazard analysis (PSHA) models for regions with high-quality seismic velocity and fault models. The CyberShake platform implements a sophisticated computational workflow that includes over 15 individual codes written by 6 developers. These codes are heterogeneous, ranging from short-running high-throughput serial CPU codes to large, long-running, parallel GPU codes. Additionally, CyberShake simulation campaigns are computationally extensive, typically producing tens of terabytes of meaningful scientific data and metadata over several months of around-the-clock execution on leadership-class supercomputers. To meet the needs of the CyberShake platform, we have developed an extreme-scale workflow stack, including the Pegasus Workflow Management System, HTCondor, Globus, and custom tools. We present this workflow software stack and identify how the CyberShake platform and supporting tools enable us to meet a variety of challenges that come with large-scale simulations, such as automated remote job submission, data management, and verification and validation. This platform enabled us to perform our most recent simulation campaign, CyberShake Study 22.12, from December 2022 to April 2023. During this time, our workflow tools executed approximately 32,000 jobs, and used up to 73% of the *Summit* system at Oak Ridge Leadership Computing Facility. Our workflow tools managed about 2.5 PB of total temporary and output data, and automatically staged 19 million output files totaling 74 TB back to archival storage on the University of Southern California's Center for Advanced Research Computing systems, including file-based relational data and large binary files to efficiently

store millions of simulated seismograms. CyberShake extreme-scale workflows have generated simulation-based probabilistic seismic hazard models that are being used by seismological, engineering, and governmental communities.

KEYWORDS

scientific workflows, probabilistic seismic hazard analysis, high performance computing, seismic simulations, distributed computing, computational modeling

1 Introduction

Large-scale simulation has become a key technique in scientific research. Referred to as the "third pillar of science" (Reed et al., 2005), computational modeling enables investigation of phenomena inaccessible to traditional experimentation. Simulation-based science is heavily utilized by disparate domains, such as astrophysics, climate science, biomedical engineering, and seismology, to evaluate hypotheses and predict future behavior.

Growing computational capability—approximately 7 orders of magnitude in the past 30 years (Top500, 2023)—has expanded the scope of scientific computation, but with this expansion comes challenges. Larger, more complex simulations require organization beyond the capacity of a single developer with shell scripts. To assist with automated job management, data management, and other challenges of extreme-scale simulations, many computational researchers now utilize scientific workflow tools.

The Statewide (formerly Southern) California Earthquake Center (SCEC) has utilized computational modeling to investigate earthquake system science for decades. One way to categorize physics-based earthquake ground motion simulations is into (1) validation simulations; (2) scenario simulations; and (3) probabilistic seismic hazard analysis (PSHA) simulations. Validation simulations, such as the 2010 El Mayor-Cucapah earthquake (Graves and Aagaard, 2011), the 2014 La Habra earthquake (Taborda et al., 2018; Hu et al., 2022), and the 2019 Ridgecrest mainshock (Yeh and Olsen, 2023), model historic earthquakes and compare simulation results against observed ground motions. Scenario earthquake simulations, such as TeraShake (Olsen et al., 2006), ShakeOut (Bielak et al., 2010), M8 (Cui et al., 2010), the EQSIM project (McCallen et al., 2021), and nonlinear simulations (Roten et al., 2023), model possible future earthquakes to quantify the hazard associated with a particular earthquake. Physics-based PSHA simulations such as CyberShake (Graves et al., 2011; Jordan et al., 2018) model all scientifically plausible earthquakes for a region of interest. As a result, physics-based PSHA hazard calculations require large ensembles of scenario earthquake simulations and typically require significantly more computing resources than the other types of simulations.

In this paper, we describe how SCEC uses open-science workflow tools and leadership class high performance computer resources to calculate CyberShake Study 22.12, a physics-based PSHA model for Southern California. We identify the challenges confronted in performing CyberShake simulation campaigns: (1) hardware heterogeneity; (2) software heterogeneity; (3) data type heterogeneity; (4) complexity and scalability; (5) reproducibility; (6) verification and validation; (7) resource provider policies; and (8) personnel and resource constraints. We believe that many of the computing challenges we face performing CyberShake studies are domain-independent issues, common to many existing and future large-scale computational research efforts, including the following. CyberShake includes a heterogeneous collection of both short-running serial CPU and long-running parallel GPU codes written in multiple programming languages. The CyberShake workflow consists of multi-stage calculations that require an ordered sequence of execution, with multiple data dependencies between the computational stages. CyberShake calculations are distributed among multiple resource providers based on resource availability and the required computing resource type. Automation is required to perform large calculations on supercomputers with limited numbers of personnel, and the multi-stage calculations may need to be restarted, and rerun, without restarting from the beginning. The provenance of CyberShake results must be preserved to support transparency and reproducibility.

In this paper, we describe how the CyberShake computational framework uses the Pegasus Workflow Management System (Pegasus WMS) (Deelman et al., 2019) and HTCondor (Thain et al., 2005) workflow tools to address these challenges and coordinate the extreme-scale computational and data management requirements for the CyberShake 22.12 study, using computing and storage resources from the Oak Ridge Leadership Computing Facility (OLCF), from the Center for Advanced Leadership Computing (CARC) at the University of Southern California (USC), and from SCEC.

2 Materials and methods

2.1 Background and methodology of CyberShake

Probabilistic Seismic Hazard Analysis (PSHA) (Cornell, 1968) has been widely used for over 50 years by governments and industry in applications with significant societal impact such as deciding safety criteria for nuclear power plants, constructing national hazard models, developing building code requirements, and determining earthquake insurance rates. PSHA is the basis for defining the seismic loads on civil structures in terms of ground shaking and is used in seismic design activities. PSHA provides the probability of exceeding a certain level of shaking over a given period of time by performing an integration over two

main types of input: (1) a list of possible earthquakes given by an earthquake rupture forecast (ERF) and (2) a ground motion model (GMM) to determine the shaking from each earthquake. The main products of PSHA are hazard curves that relate the probability of exceedance (or rate) to the ground motion intensity measure of interest, e.g., acceleration at a certain period or peak velocity. ERFs describe the earthquake sources, including faults, and detail their location, geometry, and the distribution of earthquakes of various magnitudes that the faults can generate, along with the expected long-term rate, or probability, of each possible earthquake (e.g., Field et al., 2009). These parameters are derived from seismological principles that integrate results from diverse disciplines including geodesy, paleoseismology and earthquake geology. ERFs may also account for yet unknown earthquake sources by the inclusion of background seismicity models. GMMs are used to calculate ground motions at a site for earthquakes from the ERF. The current standard practice in PSHA uses empirically-based ground motion models developed using global seismicity catalogs and observed ground motions (e.g., Bozorgnia et al., 2014).

SCEC has developed the CyberShake method as the first three-dimensional physics-based computational PSHA platform (Graves et al., 2011; Jordan et al., 2018). Instead of using empirically-based GMMs, CyberShake calculates ground motions from finite fault ruptures and wave propagation simulation. Physics-based ground motions are produced up to a predefined frequency, typically 1 Hz, but can be augmented with stochastic-based methods to produce broadband (low- and high-frequency) results up to 25 Hz (Graves and Pitarka, 2015). CyberShake can sample from an existing ERF to generate rupture models for a large number of earthquake sources using a kinematic rupture generator, which generates a model of rupture slip without simulating the underlying physics (Pitarka et al., 2021), or it can use an earthquake simulator to generate fault ruptures through evolving seismicity and stress on a California fault system (Milner et al., 2021). A CyberShake study is defined by selecting sites of interest and specifying a specific set of input configuration parameters, then calculating PSHA seismic hazard products for each site. A typical CyberShake study produces seismic hazard curves for hundreds of sites by performing physics-based earthquake wave propagation calculations to generate ground motions for hundreds of thousands of unique earthquakes. Since CyberShake uses wave propagation simulations, it generates seismograms in addition to the intensity measures, hazard curves, and hazard maps produced by standard empirically-based PSHA methods. This makes the resulting CyberShake hazard model useful for a broad variety of applications interested in simulated seismograms for large-magnitude earthquakes.

A key modeling ingredient in CyberShake is the 3D velocity model, which is a volumetric representation of Earth material properties that impacts earthquake wave propagation. Different consensus community velocity models (CVMs) are used depending on the region of interest or the goals of the project (Small et al., 2017). The CVMs used in Study 22.12 were tested during project planning and shown to improve validation results compared to a 1D model. Figure 1 illustrates the difference between using a 3D and 1D velocity model on hazard. Wave propagation software is then used to propagate seismic waves and generate Strain Green Tensors (SGTs), which represent the relationship between fault slip at one location and ground motion at another. Each individual seismogram in a CyberShake study is simulated by convolving the SGTs with the slip time history for that earthquake, a process referred to as reciprocity (Zhao et al., 2006). The CyberShake simulation results are two-component orthogonal horizontal seismograms at each site for each earthquake source. Derived parameters of interest to the community, including peak acceleration, peak velocity, and shaking duration, are calculated from these ground motion timeseries. These results are then combined with the long-term earthquake rates on the faults, as defined by the ERF, to generate seismic hazard curves at selected sites. Additional CyberShake output data products include regional hazard maps, disaggregation results showing the largest contributors to hazard, and databases containing engineering design metrics. The relationship between various CyberShake output data products is illustrated in Figure 2.

An advantage of the CyberShake method is that it produces full time series (seismograms), as opposed to empirically-based GMMs that only provide peak ground motion intensity measures. This is critical given the paucity of observed strong ground motion data and the increased reliance of engineering designs on non-linear response history analyses, where seismograms are direct inputs to computer models of engineering structures (Gerstenberger et al., 2020). Simulated seismograms can be used by a wide range of researchers in science and engineering to perform novel research. Examples of research enabled by CyberShake include the impact of site location in relation to fault ruptures and sedimentary basin structures (Baker and Chen, 2022; Nweke et al., 2022), the proper characterization and modeling of rupture directivity effects (Milner et al., 2021), near-surface site response (Villani and Abrahamson, 2015), and the combined impacts of these effects on buildings and other structures (Teng and Baker, 2019; Bijelic et al., 2020). By incorporating 3D wave propagation effects, CyberShake results capture 3D effects not included in empirical ground motion models including rupture directivity and its interaction with 3D geologic structures, particularly basins. This is especially critical in estimating the vulnerability of infrastructure distributed over large areas, such as roads and pipelines. We expect that CyberShake physics-based PSHA models will play an increasing role in improving hazard estimation and enhancing societal earthquake preparedness as this method is applied to more seismically vulnerable regions.

2.2 CyberShake computing and data requirements

The large-scale computing and data management requirements of CyberShake hazard models are imposed by the scientific goals and the specific scientific software elements which are used.

In this paper, we use the following terms to describe various aspects of the CyberShake workflows. A "simulation" is run by executing software on a computer to model a physical process like earthquake waves propagating through the solid earth. Within CyberShake workflows, simulations consist of a series of computational "tasks", where a task is an individual execution

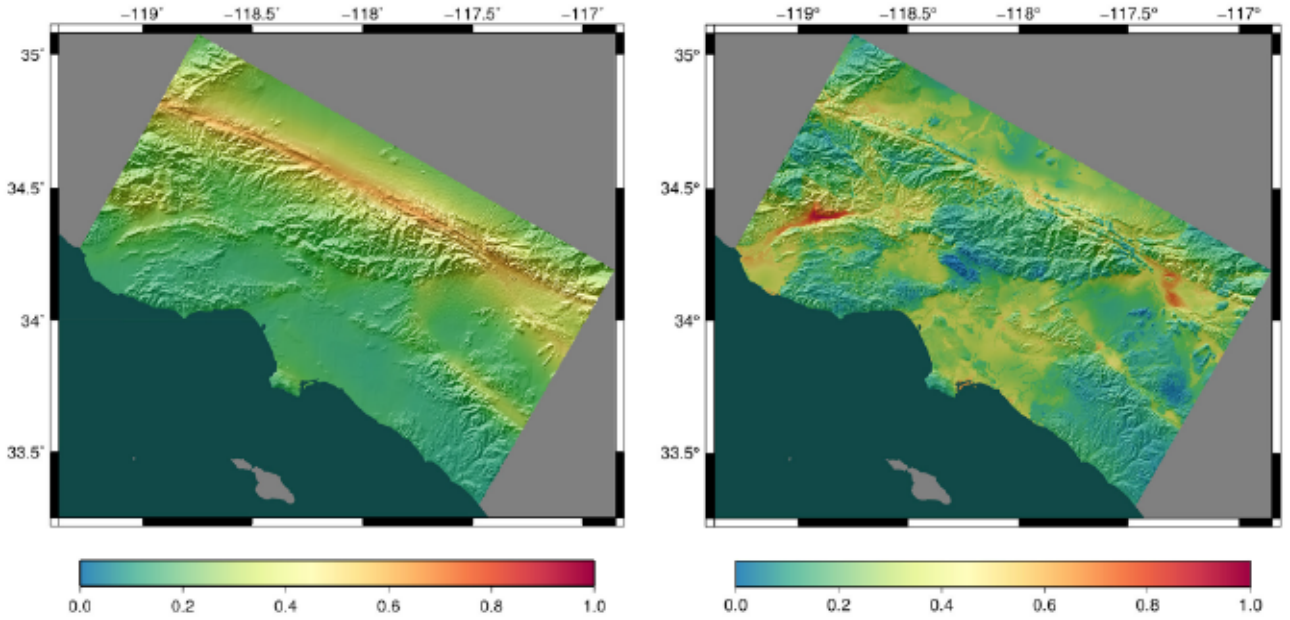

FIGURE 1
Probabilistic seismic hazard maps for southern California calculated using a 1D velocity model (**Left**) and a 3D velocity model (**Right**), indicating the ground motion level for which the model gives a 2% chance of exceedance in 50 years. Units are g (acceleration due to gravity). Warm colors indicate areas of higher hazard. The 3D PSHA hazard model shown on the right captures basin amplification effects near the western (Ventura Basin) and eastern (San Bernardino Basin) edges of the region. PSHA models that make use of a 3D velocity model can provide detailed seismic hazard information that includes the impact of established geological features in a region.

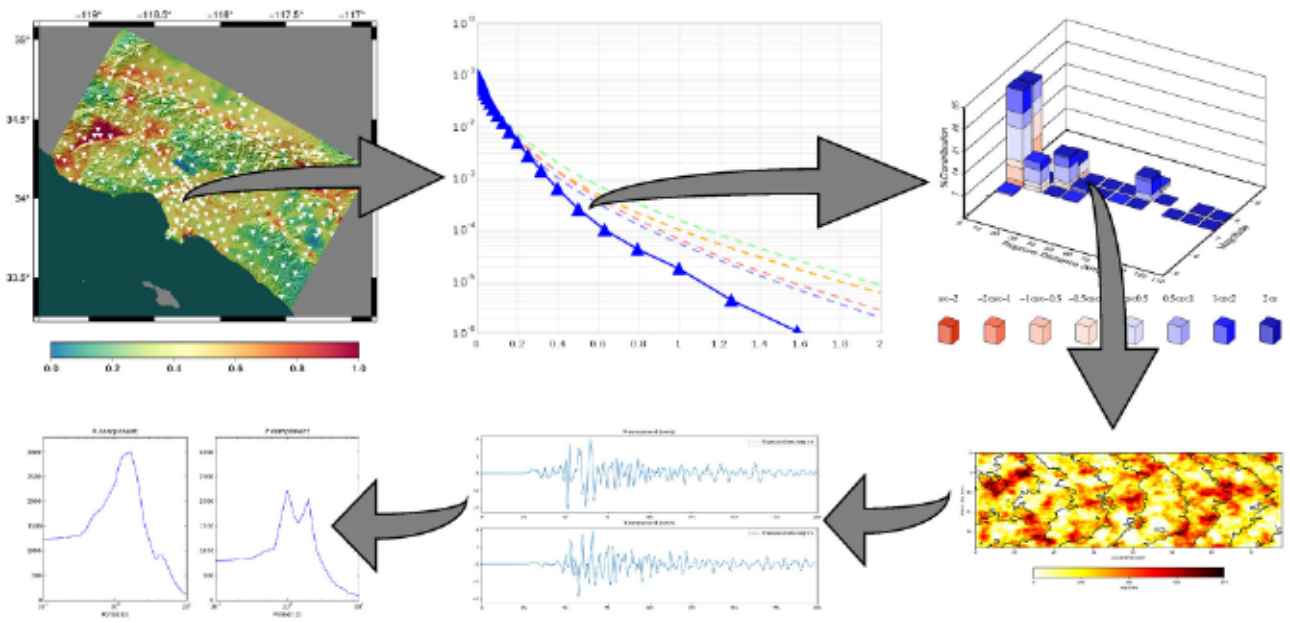

FIGURE 2
The different layers of data products produced by a CyberShake (Graves et al., 2011; Jordan et al., 2018) study. Clockwise from upper left: a regional hazard map; a site-specific hazard curve; hazard disaggregation at a point on the curve; slip details for an individual rupture; the seismogram simulated for that rupture; intensity measures derived from the seismogram.

of a code. We define a "job" as a computer program that is queued, then run, by a scheduler. Tasks and jobs are similar, but multiple tasks can be combined into a single job, then the job is submitted to a scheduler. A "workflow" is a sequence of jobs with a defined execution order and data dependencies that produce specific data products. We typically describe workflows using

directed acyclic graphs (DAGs). We define a "CyberShake Study" as a collection of one or more probabilistic seismic hazard models for a specific geographic region. During a CyberShake Study, most input parameters, such as the earthquake rupture forecast, are held constant, and results are calculated using a consistent methodology for multiple sites. A CyberShake Study may require use of multiple workflows. Makespan as applied here refers to the elapsed time between the submission of the first job and the termination of the last job of a study.

The CyberShake processing components in CyberShake have been developed and updated during the development of the CyberShake project. We designed the CyberShake system to implement a standard PSHA hazard calculation that retains well-established standard practices, while replacing the use of ground motion prediction equations with the use of 3D wave propagation simulations to calculate ground motion amplitudes at sites of interest. Since initially assembling all the processing codes required to perform a full PSHA site-specific hazard calculation, we have updated and improved our workflows by integrating the best available science codes, and selectively optimized the most computationally expensive processing stages for use on available leadership class open-science supercomputers.

For CyberShake Study 22.12, CyberShake consists of 16 different software codes (SCEC Cybershake-Core Software Repository, 2022), but we usually think of the CyberShake workflow as having four stages:

- The SGT stage (8 unique codes), which determines the simulation volume, creates a velocity mesh (a Cartesian mesh populated with material properties at each node), and runs two 3D physics-based wave propagation simulations to generate SGTs. The key codes are an MPI CPU code for generating a velocity mesh, Unified Community Velocity Model (Small et al., 2017), and an MPI GPU fourth-order staggered grid finite difference wave propagation code, AWP-ODC-SGT (Cui et al., 2013). These two codes are responsible for 97% of the computational requirements of this stage.
- The post-processing stage (two unique codes), which uses the SGTs generated in (1) to synthesize low-frequency deterministic seismograms for each earthquake in the earthquake rupture forecast. Intensity measures and durations are also derived from these seismograms. The key code is an MPI CPU manager-worker code for producing seismograms and intensity measures, responsible for over 99% of the computational requirements of this stage (DirectSynth Github Code Repository, 2024).
- The broadband stage (three unique codes), which generates high-frequency stochastic seismograms for each earthquake in the earthquake rupture forecast and combines them with the seismograms in (2) to produce broadband seismograms, intensity measures, and durations. This stage is optional. The key codes are serial CPU codes from the SCEC Broadband Platform (Maechling et al., 2015) for synthesizing high-frequency seismograms and combining high- and low-frequency results.
- The data products stage (three unique codes), which populates a relational database with a subset of the most commonly used intensity measures produced in (2) and (3), checks that all expected output files are present, and produces derived aggregate data products such as hazard curves and maps. These data products are constructed using an open-source seismic hazard analysis toolkit, OpenSHA (Field et al., 2003). This stage is run separately on the outputs from (2) and (3). The codes are all CPU serial codes.

Stages 1, 2, and 3 require high performance computing resources, and are typically executed on remote clusters. Not all CyberShake simulations include stage 3. Stage 4 can be executed anywhere, but is fastest when run on a local system, close to the archived output data files and the relational database. Graphical depictions of the steps involved in each stage are illustrated in Figure 3. The computational and data requirements of each workflow stage are given in Table 1. Note that the number of jobs and amount of data varies greatly depending on the workflow stage.

To perform a CyberShake study, also referred to as a simulation campaign or production run, we perform seismic hazard calculations for several hundred sites. Given the scale of the available computational resources, CyberShake studies have a makespan, or elapsed time between the submission of the first job and the termination of the last job, of 1–3 months when using shared, open-science, leadership-class supercomputers, depending on the number of sites included in the study and the number of earthquakes in the ERF. This long makespan imposes additional requirements on an execution solution since it must be able to run jobs in an automated fashion over a long period of time. We also expect that during the course of a simulation campaign, we will encounter job and filesystem errors, so a CyberShake execution solution also requires checkpoint and restart capability.

2.3 Need for extreme-scale scientific workflows

To achieve the scientific goal of performing large-scale PSHA, CyberShake requires the use of extreme-scale scientific workflows.

The basic CyberShake computational unit of work is an end-to-end workflow, running all four stages, for a single site of interest. This workflow performs all the calculations required to produce PSHA results for that location. When performing studies for several hundred sites in a geographic region, CyberShake must execute several hundred independent site-specific workflows in parallel.

The CyberShake project moves through different phases as we prepare for a new Study. First, we develop scientific goals for the Study, such as a new geographic region, new physics in the wave propagation codes, a new velocity model, or a different earthquake rupture forecast. Then, based on these goals, we identify changes that need to be made to the CyberShake calculations. These may include the addition of new codes, updates to previous codes, optimizations to be able to complete the calculations using the available resources, or migration to a new system. We then integrate these changes and perform verification tests, comparing CyberShake results on a new system with those generated on an established system, and validation tests, including a comparison

TABLE 1 Approximate computational and data requirements of each stage of the CyberShake Study 22.12 workflow for a single site of interest.

Workflow stage	Job count	CPU-hours	GPU-hours	Temporary data	Output data
1 (SGT)	12	355	425	3.2 TB	1.5 TB
2 (Post-processing)	3	36,750	0	1.5 TB	38 GB
3 (Broadband)	77,000	33,850	0	37 GB	190 GB
4 (Data products)	7	1	0	228 GB	2 MB
Total	77,022	70,936	425	5.0 TB	1.7 TB

CPU and GPU hour requirements are for jobs running on *Summit* at the Oak Ridge Leadership Computing Facility. Note that stages 1 and 2 mostly consist of large parallel jobs, while stage 3 consists of high throughput serial jobs.

between simulated and observed seismograms for a selected earthquake, to establish confidence in our results. Finally, we perform a production run, typically lasting multiple months, to calculate one or more seismic hazard models for the selected region. Each seismic hazard model contains a variety of seismic hazard products including rupture models, synthetic seismograms, hazard curves, and hazard maps for the study region. Once the production run is complete, the cycle starts again as we evaluate the results, identify potential computational or geophysical model improvements, and define new scientific goals. We assign a CyberShake Study number based on the year and month that the production run started for the study. For example, CyberShake Study 22.12 was started in December, 2022.

The software elements which make up the CyberShake workflow are periodically updated through the addition of new physics, additional parameters, or more complex representations which improve the ability of the codes to model ground truth. Therefore, we must be able to update or add new individual software elements as the science behind PSHA progresses, while still utilizing the same workflow approach. Computational stages we have improved over the years include the wave propagation codes, the rupture generator software, and the codes that calculate amplitudes from seismograms. Most CyberShake studies require either the addition of a new code or an update to an already existing one, representing scientific enhancements since the previous study. To enable us to meet the computational, data, and scientific needs of the project, we use a complex stack of workflow tools to execute our extreme-scale workflows.

Since the first CyberShake study in 2008 (Jordan et al., 2018), CyberShake workflows have run on 13 different clusters, typically large-scale open science systems funded by the National Science Foundation (NSF) or the Department of Energy (DOE) (Comparison of CyberShake Studies, 2024). We have performed 12 studies over those 15 years. This paper will focus on the workflows and cyberinfrastructure created and used to support the most recent CyberShake production study, Study 22.12.

2.4 Design of CyberShake workflows

While developing the CyberShake architecture, we analyzed a wide range of workflow systems and technologies including shell and Python scripts, business process workflows, graphical workflow construction tools, and open-source scientific workflow systems. During this architectural phase of CyberShake development, we identified five essential features that our CyberShake research calculation required from a workflow system. (1) A format for expressing program execution order and data dependencies that does not tie execution to a particular computing resource. (2) A job scheduler that automates the orderly execution of the programs. (3) File and metadata management to track files in the system and transfer them when necessary. (4) Support for distributed processing across multiple computing resources. (5) A solution which does not require modifications to scientific codes. We then identified an open-source scientific software stack based on Pegasus WMS (Deelman et al., 2015, 2021) and HTCondor (Thain et al., 2005) that provides these capabilities. A number of alternative workflow tools with these capabilities exist today, such as Makeflow (Albrecht et al., 2012), Nextflow (Di Tommaso et al., 2017), Parsl (Babuji et al., 2019), and Apache Airflow (2024). Pegasus and HTCondor have continued to provide excellent support for CyberShake workflows since our initial selection of these tools in 2007. In the following sections, we describe how we combine these five basic capabilities into the CyberShake extreme scale workflows.

Our CyberShake workflows make an important distinction between an abstract workflow and an executable workflow. An abstract workflow defines the jobs, files, and dependencies between jobs, but uses placeholders and variable names for the programs and files involved. It depicts the flow of the execution without regard to system specifics and could theoretically be executed on any system. We use directed acyclic graphs (DAGs) to represent our abstract workflow. A program called a workflow planner, part of Pegasus WMS (*pegasus-plan*), converts the abstract workflow into an executable workflow before the workflow is submitted to a job scheduler. The executable workflow defines a series of execution-ready job descriptions, with system-specific paths to executables and files and scheduler-specific parameters such as runtime, number of nodes, and submission queue. In addition to replacing the placeholders in the abstract workflow with actual executables and files in the executable workflow, the workflow planner adds additional jobs, such as working directory creation jobs, data product registration jobs, and data transfer jobs, that are implied, but unspecified, in the abstract workflow. A simple example is shown in Figure 4. For example, if parts of the workflow are executed on different systems, files generated from early jobs on the first system might need to be transferred ("staged") to the second system to be consumed by later jobs. Data staging jobs are added by Pegasus and are performed using Globus Online (Foster, 2011; Allen et al., 2012) during execution. The planner also wraps executables to gather runtime provenance information, such as execution time, execution environment, and task success

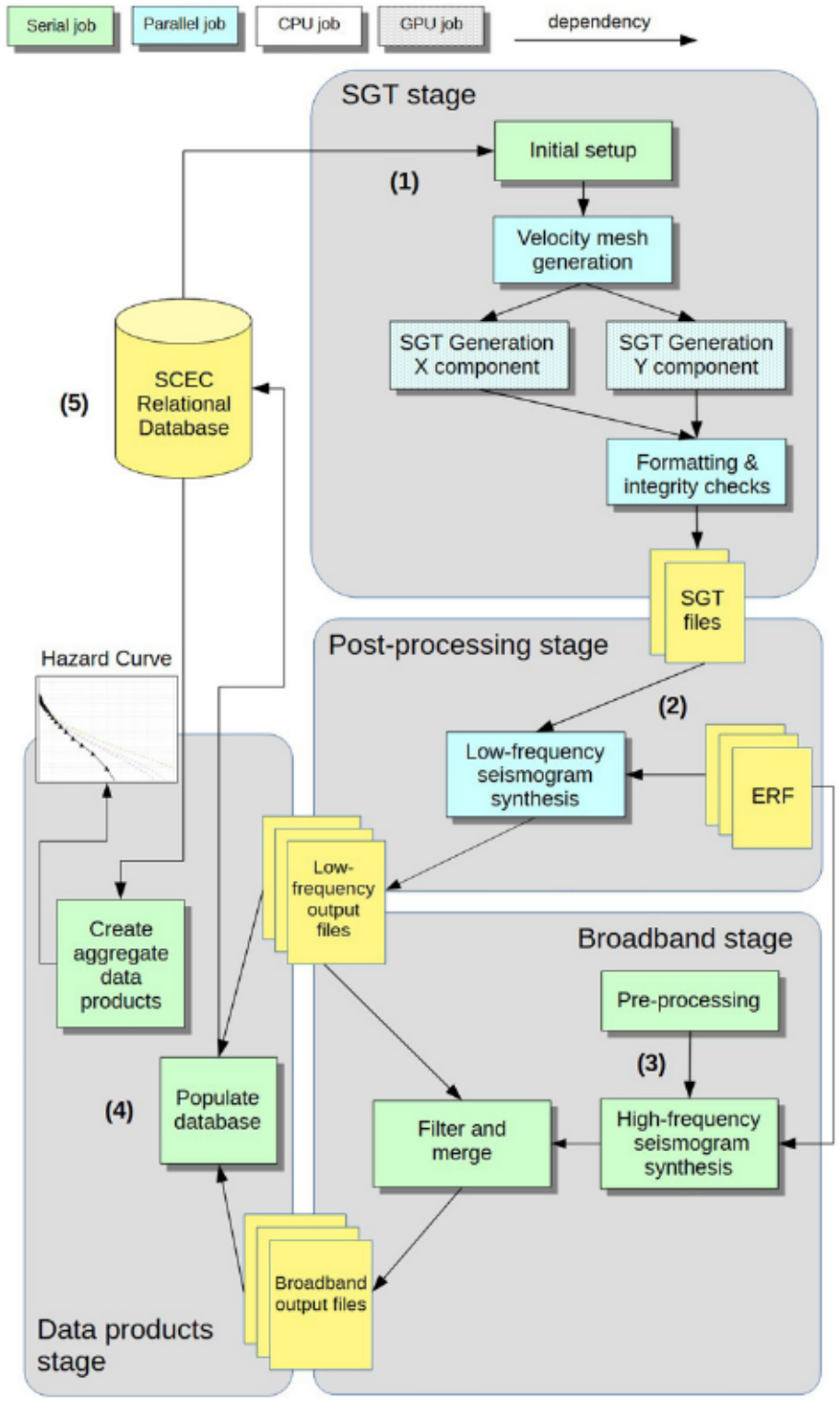

FIGURE 3
Flow chart showing the CyberShake workflow for a single site with serial processing programs (green boxes), parallel executable programs (blue boxes), parallel GPU programs (blue checked boxes), and data stores (yellow boxes). The workflow stages include (1) the SGT stage, which constructs a velocity model and performs wave propagation simulations; (2) the post-processing stage, which uses the SGTs to synthesize low-frequency seismograms; (3) the broadband stage, which synthesizes high-frequency stochastic seismograms; and (4) the data product stage which produces aggregate data products. The SCEC relational database contains data products once a run completes (5).

or failure. This approach also enforces separation between task implementation details and the abstract workflow. Thus, it is simple to update or replace scientific codes while continuing to use the same abstract workflow.

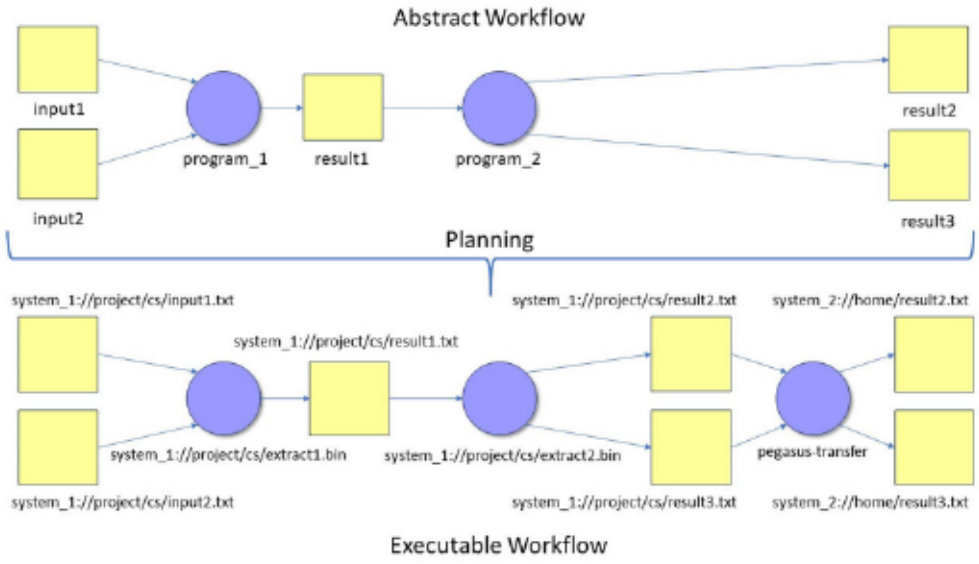

FIGURE 4
A comparison between an abstract workflow **(Top)** and an executable workflow **(Bottom)** shows that the abstract workflow defines the programs and data dependencies in the workflow and that an executable workflow defines the specific computers, files, and executables to be run. Files (yellow boxes) are inputs and results, and programs (blue circles) consume and produce results. Note that the planning process adds a Pegasus-transfer data staging job (Deelman et al., 2019) that uses Globus data transfer (Foster, 2011; Allen et al., 2012) to move output files to a different system where they are needed to complete the workflow.

The Pegasus WMS workflow planning software manages the transformation from abstract to executable workflow by utilizing three data stores called the Replica Catalog, the Transformation Catalog, and the Site Catalog (Pegasus WMS Documentation, 2023). The Replica Catalog (RC) contains a database of input files involved in the workflow and after execution is populated with information about the output files. The relational model associates logical file names for each file with one or more physical file names, which represent actual files on a file system. When the workflow is planned, the RC is used to identify the location of any files which are external inputs to the workflow and stage them in, if needed. The planner may also add jobs to register in the RC output files generated by the workflow, by creating new RC entries containing a mapping of logical filename to physical pathname(s). The Transformation Catalog (TC) contains a list of executables or "transformations" in the workflow. For CyberShake, these correspond to our scientific application programs. The transformation catalog contains entries for each logical transformation name, and associates with it one or more actual executables, along with system-specific runtime information. For example, perhaps one system has twice as many cores per node as a second system, so the job should request twice as many nodes when running on the second system. By allowing us to specify a logical program and then providing options to use alternative executables, the TC supports flexibility in selecting target systems for execution.

The Site Catalog (SC) provides system-specific information which is shared by all jobs running on that system. This includes paths to scratch storage, the type of batch scheduler, and paths to Pegasus and HTCondor installs. When we plan the workflow, Pegasus WMS produces an executable workflow, ready for the job scheduler, that directs each processing stage to an appropriate computer resource with all needed parameters. This includes the addition of data transfer jobs and RC data registration jobs, so that as the workflow executes and generates files, the physical files that are produced are registered into the RC.

Pegasus includes the DAGMan component of the HTCondor software as a job manager to manage the execution of executable workflows (Deelman et al., 2019). Many of the jobs in our CyberShake workflows execute on shared, open-science supercomputer systems, such as *Summit* at the Oak Ridge Leadership Computing Facility (OLCF), and these shared computer resources typically manage jobs using a batch scheduler, such as Slurm (Jette and Wickberg, 2023) or LSF (IBM Spectrum LSF Documentation, 2024). DAGMan works as a meta-job-scheduler. Pegasus submits the executable workflows to DAGMan/HTCondor, which runs on a local, USC-based dedicated workflow submission host. DAGMan manages the executable workflow and identifies which jobs are ready to be run. Then DAGMan, through the HTCondor *schedd*, submits the jobs to the batch scheduler on the target remote system. When the job completes, HTCondor receives a notification, and based on the structure of the workflow, new jobs may be ready to run. If a job fails, it is automatically retried a user-configurable number of times. If it fails repeatedly, DAGMan will execute any other jobs in the workflow not dependent on the failed job, write errors to a log file, and create a workflow checkpoint file (called a "rescue" file by DAGMan). Once the developer has fixed the underlying problem based on information from the error logs, the workflow can be resumed from the failed job using the rescue file. One can also resubmit the original workflow to Pegasus to replan the workflow on different resources. Pegasus will only replan

the portions of the workflow that remain to be executed. Since CyberShake studies may run for multiple months, the availability of required computing resources may change during that time. If the available resources change, we can update our Site Catalog, replan the abstract workflow, and re-distribute our workflows to the most appropriate resources during a Study.

There are many complexities involved in these HTCondor meta-scheduling capabilities. A few examples may serve to make this point. Communications between our local HTCondor workflow submission host, and the remote systems running our workflow jobs, present security issues for both systems. Automated remote job submission for systems that require two-factor authentication can be an issue. Each external system has its own policies for the number of jobs a user can submit to its queue, so HTCondor must comply with the appropriate policy for each resource provider. The utilization and throughput that our workflow jobs can obtain on an external supercomputer depends strongly on how the number, size, and runtime of the workflow jobs mesh with the scheduling priorities of the external system. For leadership class systems such as OLCF *Summit*, priorities are often given to jobs that use a substantial portion of the system. Serial and short running jobs and jobs that produce many small files are discouraged. Solutions for these issues, supported by Pegasus WMS, include bundling of serial and short running jobs into large, multi-node long-running jobs (Rynge et al., 2012). HTCondor also provides a technology called glideins that enables users to utilize pilot jobs to run many-task workloads efficiently on supercomputers. We provide technical details on how we addressed these issues for CyberShake 22.12 in Section 3.2 below.

The concept of planning has proved to be particularly important and valuable for our CyberShake workflows because we depend on open-science shared computing resources allocated via a proposal process. We don't have access to dedicated computer systems that can support our CyberShake workflows. As a result, our access to specific computer resources varies by year. Once we receive an allocation, the availability of any specific computer resource is still dependent on both scheduled and unscheduled maintenance and workload. Heterogeneity in computing hardware and job requirements leads us to direct each of our computing stages to the most appropriate resource provider, and in many cases the workflow is distributed between several resource providers. The ability to update our workflow planner data stores, including the RC, the TC, and the SC, then replan a workflow to redirect to a different system without changing the abstract workflow has been particularly valuable. We have used this capability to dynamically allocate CyberShake workflows to resources based on their real-time workload, system queue times, and to avoid resources during periods of planned maintenance (Callaghan et al., 2019).

The CyberShake workflow architecture also establishes an interface between the scientific software elements that are run as workflow jobs and OpenSHA, our aggregate data product generation tool (Field et al., 2003). We designed a relational database schema that stores necessary PSHA input parameters from the ERF generated by OpenSHA. We also defined a schema that stores the data products produced by the CyberShake workflows, including ground motion intensity measures and probabilistic seismic hazard curves, that can be read by OpenSHA. When CyberShake workflows execute, they read their input parameters from the relational database, execute multiple computing stages, and write a subset, consisting of the most commonly used results, into a relational database. OpenSHA and end users query the relational database for these data products. This architectural approach separates the interactive stages of the PSHA calculations, including configuration of the inputs, and analyzing and visualizing the results, from the extended high-performance processing stages which are asynchronous and automated and therefore would be slowed down by human interactions. End users may need to process seismograms to access results not written to the relational database.

We have chosen to preserve the stages described in Section 2.2 in our workflow representation. Although these stages could be combined, keeping them loosely coupled has provided additional flexibility. For example, Stage 3 can be easily omitted, Stage 2 can be run multiple times with different earthquake input parameters for comparisons, or different stages can be distributed to different HPC resources.

The SCEC-developed software tools to support CyberShake workflows are available through the CyberShake tools code repository (SCEC Cybershake-Tools, 2022).

3 Results

To accomplish our scientific goals of advancing PSHA in Southern California, in early 2022 we began planning for the execution of a new CyberShake set of broadband simulations, Study 22.12, for 335 sites in the greater Los Angeles area with an updated set of events and an improved 3D velocity model. Below we describe the computational system used to perform this study, summarize the technical scope of the simulations, and briefly describe the scientific advancements enabled by this new model.

3.1 Description of CyberShake study 22.12 computational system

To perform a CyberShake study, we implement the workflow system described above, using a dedicated SCEC workflow submission host (WSH) located at SCEC headquarters at USC to orchestrate the workflows and a dedicated relational database server, both for metadata and the storage of output data products.

Before the study can begin, we perform initial setup. This includes performing verification and validation of the software stack, selection of the geographic sites to be used in the study, and insertion into the relational database of needed input data (for example, the latitude and longitude of the sites, and the earthquakes in the ERF with their metadata). This may also include the installation or updating of software packages. On the SCEC WSH, we install both Pegasus (Deelman et al., 2015, 2021) and HTCondor (Thain et al., 2005) and any required dependencies. On the remote system, Pegasus is required to support transferring, wrapping executables, and running jobs in a high-throughput manner. HTCondor is also required to support our methods of remote job submission, detailed below. Both Pegasus and HTCondor can be installed in user space on the remote system.

Once the scientific goals and configuration for a study have been determined, prior to launching a large-scale CyberShake production run, we conduct group meetings that we call "readiness reviews" to review the scientific and technical details of the planned study. We conduct our scientific readiness review with our project's scientific and engineering collaborators. These reviews ensure that the input simulation configuration, and the planned data products that will be produced by the study, properly address the scientific goals of the study. We also conduct technical readiness reviews with software, computer science, and cyberinfrastructure developers on the project, and we include representatives from the computer system resource providers. These reviews discuss the study's target computing resources, identify the software versions to be used, review the computing and storage estimates from the study, and confirm that appropriate computing and storage resources are available for the planned study.

Next, we perform a stress test, in which we run many workflows simultaneously to test the software stack under production-level load. Once any identified issues are resolved, we begin the main production run.

An individual CyberShake workflow, which produces PSHA results for a single geographic site, is initiated by invoking the Pegasus Java API on the WSH to create a programmatic description of the workflow. At this time, scientific parameters are selected (the site of interest, the specific input models to use, the maximum seismic frequency, etc.), and an entry for the run is created in the relational database, so the run's status can be monitored. Once the abstract workflow is created, it is written to the filesystem on the WSH. This abstract workflow is then converted into an executable workflow for execution on one or more specific systems, using *pegasus-plan*, which is then passed off to HTCondor DAGMan (Deelman et al., 2019) to manage the runtime execution.

For a CyberShake study, individual workflows define the processing required to produce a PSHA hazard model for a specific site; Study 22.12 required 335 sites and therefore 335 workflows. We obtain higher throughput by running multiple workflows simultaneously. We find that 30–40 works well; this is high enough to keep a variety of jobs in the remote scheduler and low enough so that workflows are still able to make progress and aren't blocking each other. We set HTCondor configuration parameters to enforce a per-user limit on the number of jobs the HTCondor scheduler will send to the remote system, so that we don't exceed the jobs per user limit on the system. To keep multiple workflows running simultaneously without human interaction, we developed an additional layer of automation. We have constructed a Python workflow submission script which parses a Pending Site File with a list of sites to run, and then creates, plans, and submits the workflow for the sites in this file. This script supports execution of workflows on distributed systems and includes a limit as to how many workflows can be simultaneously executing on each system. Because the Pegasus Java planning processes can require extensive memory resources, we also limit the number of new workflows submitted per script execution cycle to 3. We then set up a cron job on the WSH to run this script every 10 min and submit new workflows if there are still sites to run and the remote resource has available workflow slots. Thus, to start up a study, we construct the Pending Site File with a list of all the sites, along with scientific parameters such as the velocity model to use and the maximum simulated frequency. The workflow submission script slowly digests the Pending Site File over the course of the study. The Pending Site File also supports workflow restart, so that if a workflow experiences an error and is fixed, it can be added back to the Pending Site File and restarted automatically. The workflow submission system used in CyberShake is illustrated in Figure 5.

As workflows are executing, Pegasus via HTCondor DAGMan keeps a queue of workflow jobs and submits them for execution on their target resource, either *Summit* or the WSH for Study 22.12. Our workflows support both push-based and pull-based job submission approaches. Jobs submitted to *Summit* use the push-based rvGAHP approach by default (Callaghan et al., 2017), since for most jobs we prefer a low-overhead push-based solution. rvGAHP is supported by a long-running daemon on an OLCF data transfer node, which initiates the connections to the WSH so that the two-factor authentication token is not needed for job submission. However, the SGT wave propagation code is executed in large bundles using pull-based HTCondor glideins to enable job bundling. To support glideins, we created a Python script to check the WSH DAGMan queue from the remote system and submit a glidein job when a minimum threshold of queued SGT jobs is met. We use a cron job on an OLCF data transfer node to run this Python script and submit glidein jobs when needed. For more details about our remote job submission approaches, see Section 3.2.7.

3.2 CyberShake workflow challenges and solutions

In the process of designing the CyberShake workflow system and utilizing it for large-scale scientific investigation, we have encountered and solved a variety of challenges that may be applicable to other extreme-scale workflow applications.

3.2.1 Hardware Heterogeneity

High performance computing is a rapidly changing field, and applications which make use of it must be able to adapt. Over the 15-year lifetime of CyberShake, we have seen clusters advance from several hundred CPUs to hundreds of thousands of cores to predominantly GPUs and accelerators. A challenge for CyberShake has been to keep up with these changes in hardware.

Scientific workflow tools have been key in enabling CyberShake to migrate to the latest cutting-edge HPC systems. The Pegasus WMS abstract/executable workflow paradigm described in Section 2.4 enables reuse of the abstract workflow, since the overall processing workflow remains the same regardless of the execution system. Entries are added to the Transformation Catalog for the executables and the Site Catalog for the filesystem and scheduler, and then the abstract workflow can be planned into an executable workflow on the new system. This separation of the high-level workflow description from the system-specific details simplifies migration, enabling us to take advantage of the best available hardware. Pegasus also supports distributed execution, so that different parts of the workflow can be executed on different systems. This enables us to run the GPU codes on systems with high GPU

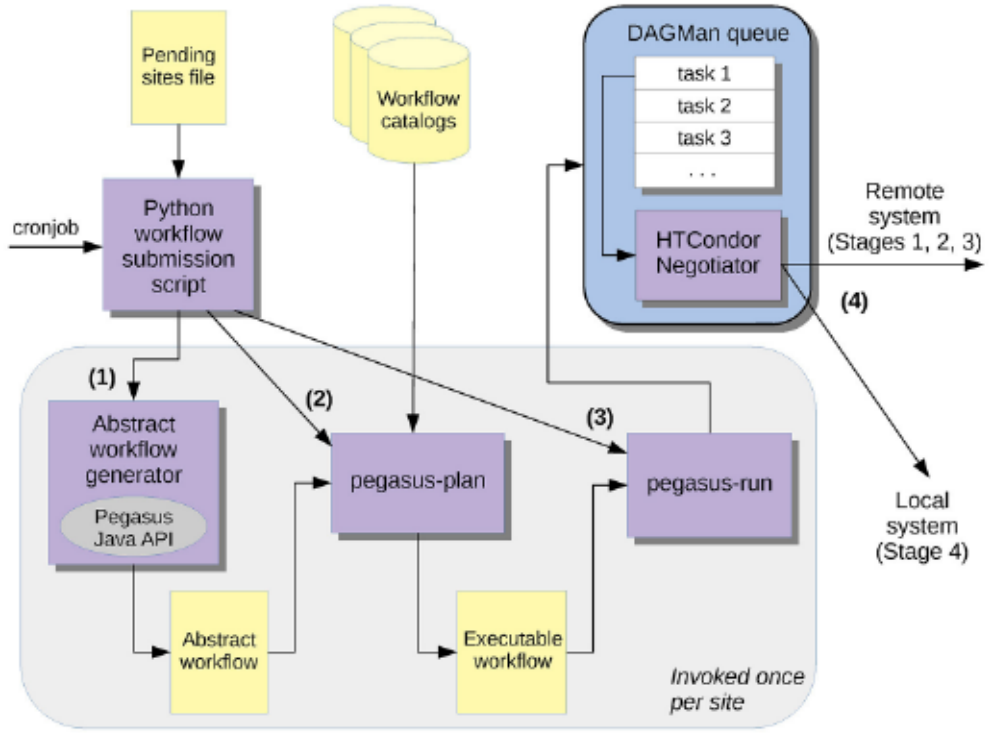

FIGURE 5
Schematic of the workflow submission system utilized by CyberShake. Purple boxes are software executables; yellow boxes are data stores. The workflow submission script, invoked by a cronjob, processes the pending sites file to construct a list of sites to run. A site is removed from the pending site list when a workflow has completed for that site. A study is complete when the pending site list is empty for that study. For each site, the script calls (1) the workflow generator, which constructs an abstract workflow; (2) pegasus-plan, which uses the TC, RC, and SC to convert the abstract workflow into a concrete workflow; (3) pegasus-run, which sends the concrete workflow to DAGMan (Deelman et al., 2019) for execution on local or remote resources. (4) Tasks are then run on either a remote system for stages 1–3 or locally for Stage 4.

counts, the CPU codes on predominantly CPU systems, and data product generation codes locally, close to the relational database and the scientific output files. The coordination of different computing resources is presented in Figure 6.

We have also found that testing is vital when running on heterogeneous hardware. When CyberShake is migrated to new hardware, each code must be tested to confirm that it is producing correct results. This is discussed in more detail in Section 3.2.6.

required to return a zero exitcode upon successful execution, and a non-zero exitcode if an error is encountered, for the Pegasus wrappers to correctly identify jobs which have succeeded and failed. The CyberShake workflow software stack also supports heterogeneous task execution through a variety of execution paradigms, including push-based, pull-based, bundled jobs through glideins, and *pegasus-mpi-cluster* jobs, discussed further in Section 3.2.7.

3.2.2 Software heterogeneity

Another challenge for CyberShake is the heterogeneity of the scientific software elements. CyberShake consists of over 15 codes, written by six different developers, in C, Fortran, Python, and Java. These codes use several processing methods including serial, OpenMP-parallel, tightly coupled MPI-parallel, and manager/worker MPI-parallel. Our GPU wave propagation code, AWP-ODC-SGT (Cui et al., 2013), has both CUDA and HIP implementations for running on NVIDIA and AMD GPUs, respectively. The shortest codes take a few seconds to run, and the longest can take over 8 h.

Pegasus supports the execution of a heterogeneous code base like CyberShake. Since Pegasus wraps scientific codes without changing their contents, scientific software development can remain separate from technical workflows. Scientific codes are

3.2.3 Data type heterogeneity

CyberShake workflows have large data requirements driven by diverse types of data and metadata. This includes workflow catalog data, temporary data produced and consumed by workflow jobs, and final output data products conserved and archived for usage by seismic researchers.

Workflow catalog data—the Replica Catalog, Transformation Catalog, and the Site Catalog—used for converting the abstract workflow description into an executable workflow are kept in files and SQLite databases on the workflow submission host.

CyberShake workflows produce large volumes of temporary data. Most jobs in the workflow communicate through the filesystem, so temporary data produced by one job is consumed by a later job. The file-based interfaces between the codes are used in a CyberShake workflow because they typically require

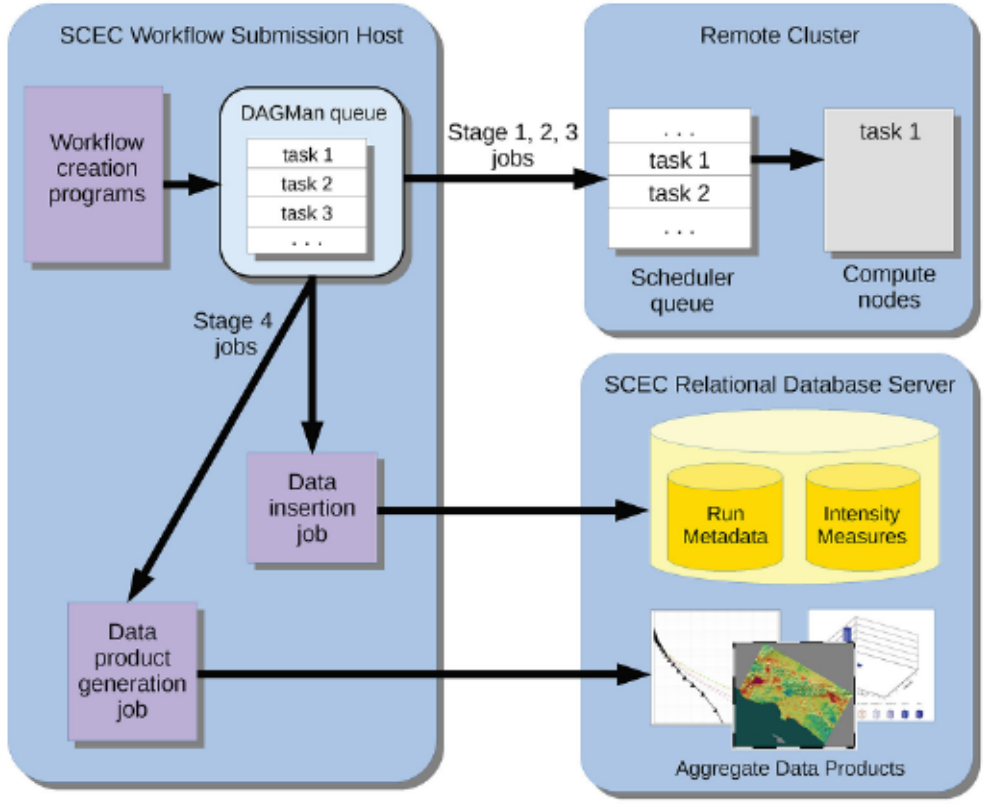

FIGURE 6
Schematic showing the computational coordination between the workflow submission host, the remote system, and the relational database.

minimal modifications to the original scientific codes as they are integrated into the workflow processing. Study 22.12 produced about 2.5 PB of temporary data, most of which was the velocity meshes and SGT files. Pegasus automatically adds data transfer jobs at planning time to stage temporary data when necessary, a feature especially important when CyberShake workflows are running on distributed systems. This way, we can run the Stage 1 workflows on a GPU-heavy system to produce the SGTs efficiently, and then automatically stage them to a CPU-heavy system for Stages 2 and 3.

Users of CyberShake data are interested in both aggregate hazard products, such as hazard curves and regional maps, and also in earthquake-specific data products such as individual seismograms and intensity measures. Thus, we archive these output data products for researchers to access. CyberShake produces four types of output files: seismograms, geometric mean intensity measures, rotational intensity measures, and durations. Initially CyberShake produced over two million small (300 B—100 KB) files per run, so we modified our codes so that output data for earthquakes that share a magnitude and fault is aggregated into the same file. This reduced the total number of files to ~28,000 per site, or 19 million files for Study 22.12. To facilitate storage of these files, the workflow tools automatically stage them back to long-term SCEC disk storage at USC CARC using Globus for high-speed transfer performance. We use a directory hierarchy to keep the number of files per directory low and to avoid overwhelming the filesystem. Since the intensity measure data is particularly useful to our community, we select a subset (about 25%) of the intensity measure data and store it in our relational database to streamline access. This also enables fast generation of hazard curves, disaggregations, hazard maps, and other aggregate data products in Stage 4. The different types of data and their storage locations are presented in Figure 7.

3.2.4 Complexity and scalability

The trend in scientific modeling is toward larger, more complex models. Most projects start with a simple model, and as sources of mismatch with observations are identified, additional model complexity is added to more closely reproduce reality. As computing resources expand, larger, more complex simulations become computationally feasible. Thus, a challenge for scientific workflows is to scale with the growing needs of modelers.

In CyberShake, we have seen this trend as well. Over the lifetime of the CyberShake project, the number of sites, maximum frequency, number of earthquakes, velocity model complexity, and rupture complexity have all increased to produce more accurate and useful PSHA estimates (Comparison of CyberShake Studies, 2024). Individual codes have gone through optimization cycles as well. Serial pre- and post-processing codes which initially took a few seconds to run became bottlenecks as the data and compute scales increased, so we converted them to parallel versions. Our main Stage 2 code, which synthesizes low-frequency seismograms,

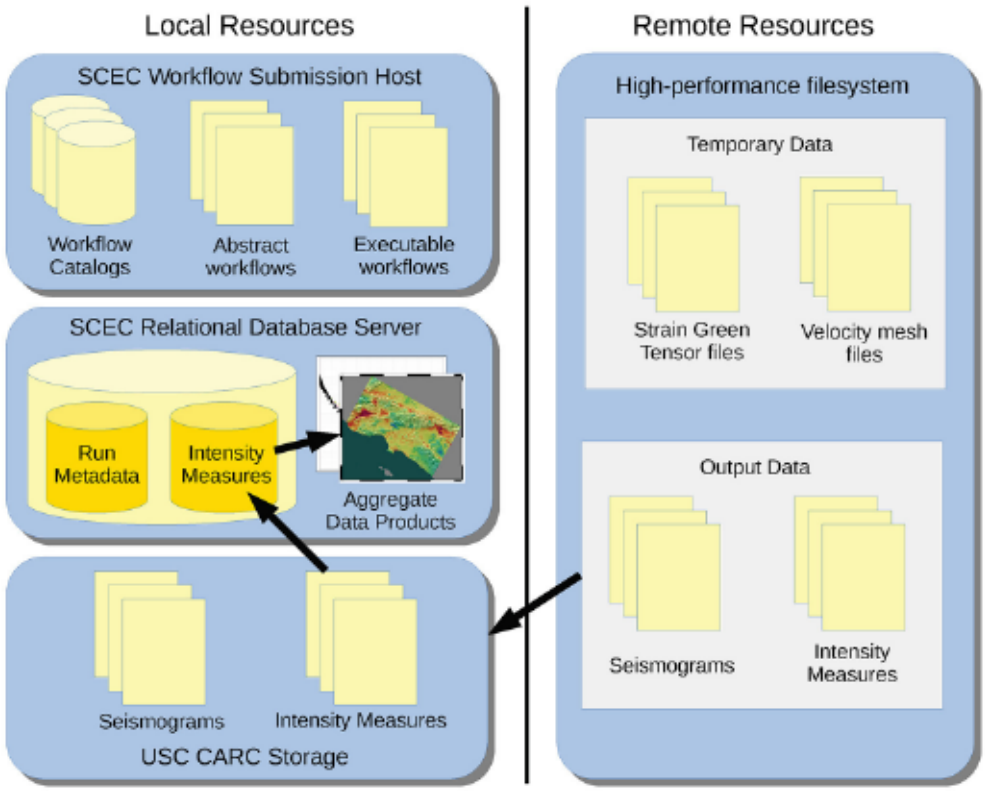

FIGURE 7
Schematic of the different types of data utilized in CyberShake and their storage locations. The majority (97%) of the data requirements come from the temporary files stored on the remote high performance filesystem.

was initially a series of serial jobs which wrote intermediate files and then used them to produce one seismogram at a time, but as CyberShake computational demands grew to match the scientific goals, we converted this code to a manager-worker MPI code for scalability. Both our main Stage 2 code and our Stage 3 codes were modified so that instead of reading earthquake rupture forecast files from disk, ruptures were generated in memory to reduce filesystem load.

To support scientific and technical optimizations, we rely on the scientific workflow stack to easily update codes with new physics or new implementations and add additional codes when necessary. For example, one of the steps in CyberShake is the creation of a Cartesian velocity mesh, where each node in the mesh contains earth material properties such as the speed of seismic waves obtained from a CVM. In early CyberShake studies, the mesh was created by querying a single velocity model. However, as we expanded the geographic scope of CyberShake, no single velocity model covered the entire simulation region, requiring the use of multiple velocity models. However, this can lead to sharp, artificial interfaces between models. To mitigate this, we created a code to smooth the transitions between models to minimize reflection and refraction artifacts, and added it to the abstract workflow as a new job. The simplicity of adding new jobs to the workflow has been vital in enabling CyberShake to improve. Other modifications have included updates to the SGT code to support GPUs, checks on output files to identify file system errors, and the addition of Stage 3 to enable high-frequency stochastic calculations. Problems that have occurred during previous large workflow runs have motivated the development of testing scripts that we integrate into the workflows at crucial stages in the processing. These scripts detect errors such as empty files, correct file sizes, or files with NaNs, and confirm the integrity of the calculation before proceeding. These kinds of issues reflect what we have generally found: although performance optimizations are helpful for reducing computational cost, our biggest bottlenecks come from the overall complexity of CyberShake. The flexibility of scientific workflows enables CyberShake to continue to produce the best available scientific results.

3.2.5 Reproducibility

As the scale and importance of computational research continues to increase, there has been an increased focus on reproducibility in scientific simulations. In the CyberShake context, we use reproducibility to refer to the ability to use the CyberShake codebase to rerun a simulation, using the same inputs, and produce exactly the same outputs as in the previous run. Reproducibility is vital for scientific results to be trusted and for other researchers to evaluate the work. CyberShake provides reproducibility through its use of open-source scientific codes, official USGS earthquake rupture models, and published seismic velocity models. This capability is also used to perform verification

testing when CyberShake is migrated to a new system, as described in Section 3.2.6.

To support reproducibility, we tag both the scientific and supporting code in Github before beginning a production run. This has also proved useful in determining if bugs have affected past results. Pegasus wraps all executables in *pegasus-kickstart*, a lightweight wrapper which captures the command-line parameters, the job runtime, and the execution environment (Papadimitriou and Deelman, 2021). Additional scientific parameters, such as the velocity model, the rupture generator, and the maximum frequency are tracked in the relational database, for transparency as to what set of parameters have yielded a particular set of output data products. Also, since CyberShake workflows are described in an abstract form, other scientists can potentially run these workflows in their environments, using Pegasus to map the workflows to their own resources.

3.2.6 Verification and validation

Based on the intended use of CyberShake results in broad impact seismic hazard applications, verification and validation are essential elements of every CyberShake study. In this paper, we refer to verification as the process of confirming code correctness, and validation as the process of confirming that the model expressed by the code reflects reality (Boehm, 1981).

Whenever we migrate CyberShake to new platforms, we go through a process of verification to ensure that the codes are producing comparable results on the new platform. Typically, we follow the following procedure: (1) Run each code by hand on a small problem, using batch scheduler submission scripts. Compare the results of each code to a reference solution. (2) Using Pegasus, run the small problem end-to-end and compare the final outputs to reference solutions. (3) Using Pegasus, run a production-sized problem end-to-end and compare the final outputs to reference solutions. This approach enables us to identify problems quickly and confirm that both the individual scientific codes and the workflow tools are running correctly on a new system. The Pegasus workflow tool streamlines this process, as the separation of the abstract workflow from system-specific details means that the same abstract workflow can be used on a new system by just adding catalog entries. Pegasus also captures command-line parameters, runtime metadata, and environment details at runtime through executable wrappers, simplifying the process of tracking down differences between two systems. Due to accumulated single-precision floating-point errors and rounding operations in the codes, we do not expect to exactly reproduce reference solutions. Typically, we see differences of O(0.1%) when comparing results generated on different systems, largely driven by differences in earthquake rupture descriptions due to integer rounding.

An ongoing effort for CyberShake—and, indeed, all scientific modeling codes—is to perform validation, to show that the scientific codes are able to reproduce a natural phenomenon closely enough to be a useful model. Validation results provide vital evidence to end users that simulation results can be relied upon. In CyberShake, we perform validation by simulating the ground motions produced by historic earthquakes in our study region and comparing the simulated ground motions against the observed ground motions. We begin the validation process by creating a new earthquake rupture forecast which contains moderate magnitude (M_w 5.5–7.5) historic earthquakes. We then run workflows using this ERF to simulate the historic earthquakes in CyberShake, and compare the simulated results to observed seismograms. In general, we have found that the 3D physics-based approach in CyberShake generates synthetic seismograms that more closely match observed seismograms from historic earthquakes in Southern California on earthquake engineering goodness-of-fit measures than empirically-defined GMM methods (Callaghan et al., 2022). Validation is an ongoing process which must be repeated as new earthquakes occur, new physics are integrated into the codes, and improved input datasets are made available.

3.2.7 Resource provider policies

Computing clusters are large, complex systems, and each system has its own policies regarding security, authentication, network access, and scheduling. To reduce CyberShake time to solution on diverse systems and optimize throughput, resource provider policies must be considered when designing a scientific workflow solution.

3.2.7.1 Scheduler policies

One challenge is that CyberShake consists of both short-running (<1 min) serial loosely-coupled high-throughput tasks and long (>6 h) tightly-coupled parallel tasks. However, most cluster schedulers have particular job profiles that they favor, based on the goals of the cluster. For example, OLCF *Summit* favors large jobs, giving a 5-day priority boost to jobs which use at least 20% of the system, and a 15-day boost to jobs that use at least 50% (Summit User Guide, 2023). To shape CyberShake workflow job size to best fit a target system, we have developed two approaches. The first is to use HTCondor glideins to bundle together multiple workflow jobs into a single cluster scheduler job, referred to as a pilot or glidein job. For example, on *Summit* a job requires at least 922 nodes to trigger the 5-day priority boost. Since our largest SGT tasks only required 67 nodes, we gathered the SGT tasks into groups of 14 (since $67 \times 14 = 938$ nodes, more than 20% of the system). This was done by letting SGT tasks wait in the DAGMan queue until at least 14 were ready to run. We then submitted a job to *Summit* for 938 nodes. When this job started, it advertised 14 slots to the HTCondor collector process, and DAGMan could then assign 14 SGT jobs to the available resources simultaneously. Bundling can be done in time as well; for instance, if a task typically takes an hour to run, a glidein can be submitted for 3 h, and three tasks can be run sequentially. This also can improve throughput, since often scheduler queue times are shorter for 1 job on H hours × N nodes than for H jobs on 1 h × N nodes. To automate the process of requesting glideins, we used a long-running daemon running on the remote cluster, as described in Section 3.1.

To improve throughput, we bundled short serial tasks together and ran them as a single job using *pegasus-mpi-cluster* (PMC) (Rynge et al., 2012), a tool included in Pegasus WMS and initially developed for use with CyberShake. PMC is a wrapper that executes an executable workflow as a single MPI job. A manager process digests the DAG file describing the workflow and assigns workflow tasks to workers, honoring dependencies. When tasks

complete, the workers ask the manager for new tasks until the workflow finishes. This enables CyberShake to execute tens of thousands of short-running tasks with high throughput, since the task scheduling is handled by PMC over MPI rather than by the batch scheduler, which is typically not configured to handle large numbers of short-running jobs. Both HTCondor glideins and PMC enable the CyberShake workflow system to create jobs with runtime parameters optimized for the target systems, reducing overall makespan.

3.2.7.2 Two-factor authentication

In the past 5 years, most clusters have moved to mandatory two-factor authentication (2FA) when using SSH to access login and compute nodes. One factor is the user's password; the other factor is a one-time token with a brief (<1 min) lifetime, typically delivered through a dedicated fob, an app, or a push notification. Requiring 2FA for cluster logins is popular because it offers additional security. However, 2FA complicates automated job submission since it does not permit traditional automated access approaches such as SSH keys. To enable automated job submission—critical for long-running simulation campaigns like CyberShake—on machines with 2FA, we have developed two approaches. The first is rvGAHP, a push-based approach which is an implementation of the HTCondor GAHP protocol (Callaghan et al., 2017). This approach was initially developed by the Pegasus team for use with CyberShake. A long-running daemon is started up on the remote system and initiates an SSH connection from the remote resource back to the workflow submission host. Once this connection is established, Pegasus/HTCondor can use it to submit ready-to-run jobs to the cluster scheduler. 2FA is not needed because the connection was initiated from the cluster, behind the 2FA. The rvGAHP approach is illustrated in Figure 8. The second approach is to use HTCondor glideins, a pull-based approach, that runs a process on the supercomputer to monitor the CyberShake HTCondor task queue. The glidein daemon schedules pilot processes on supercomputer nodes when there are workflow tasks waiting. Again, since the connection is initiated from the glidein on the cluster side, 2FA is not required. This arrangement requires collaboration with the resource providers on both sides of the process and may require whitelisting specific IP addresses to allow the required network connections between specific known trusted research computing systems. The HTCondor glidein approach is illustrated in Figure 9.

3.2.8 Personnel and resource constraints

To make continued progress performing our CyberShake research we must deal with several characteristics of this large-scale calculation. The CyberShake research process is divided into three phases: preparation, computation, and analysis. We have completed about 1 CyberShake study per year, on average, over the last 15 years. While time in each phase varies, typical durations are 8 months for preparation, 2 months for production run, and 2 months for analysis. The preparation phase must coincide with the availability of the research scientists. The computation phase must coincide with the availability of computing resources and the computing staff. The analysis phase involves reporting to stakeholders and potential users.

As with many groups, over time, we are faced with accomplishing more work with fewer human resources. While the scale of the CyberShake calculations has increased nearly six orders of magnitude, the staff available to perform the calculations has been reduced. Workflow tools, with their automation capability and error recovery, are the essential technical elements that have allowed us to increase the amount of computing work we can perform with a smaller number of personnel.

The extended, multi-month makespan for our CyberShake production exceeds what individual developers can manually track and monitor. Our production runs are submitting jobs around the clock for weeks, so automated monitoring tools are essential. In most cases, detecting error conditions and alerting an operator is sufficient; the monitoring tools do not need to fix the problems. In addition to monitoring the DAGMan queue, we have developed a login-protected web interface to obtain the status of individual runs in the relational database. By accessing the website, developers can monitor run status ("running," "completed," "error") as well as see what percentage of a study is complete and what the estimated end date is. The website provides study-level status information and provides optional email notifications to alert the workflow operators when, for example, a proxy certificate is about to expire, or a workflow completes. A typical error resolution process begins with operators noticing an issue through the web interface. A developer can then investigate the HTCondor error log for that workflow, resolve the error, and put that workflow back into the Pending Sites File for the automated system to restart.

We approach our CyberShake production runs as collaborations with resource providers, such as OLCF and USC CARC, and cyberinfrastructure providers, such as Pegasus WMS and HTCondor. All the groups involved want the available software and computer systems to produce useful research results. To build a shared interest in a positive result, and increase the chances of help when needed, we keep the resource providers apprised of our computational plans, and give them a chance to provide input and modifications. We recognize that the open-science computing resources we use are shared resources, and we respect the administrative policy decisions of the system operators. Throughout the study we continue to communicate; sometimes we request policy changes to improve throughput, and sometimes they request throughput adjustments to reduce impact on other users. We describe our results at conferences and in science articles and we acknowledge resource providers and collaborators in our presentations and publications.

3.3 CyberShake study 22.12 technical results

Previous CyberShake studies have been run both on multiple remote clusters and on a single remote cluster (Jordan et al., 2018). For this study, we selected to use a single resource, OLCF *Summit*, as our target remote execution system. *Summit* was chosen for a number of reasons. It is operated by OLCF as an open-science system, and significant computing time is awarded to academic research groups through the ALCC and INCITE allocation programs. The overall capability of the system, including

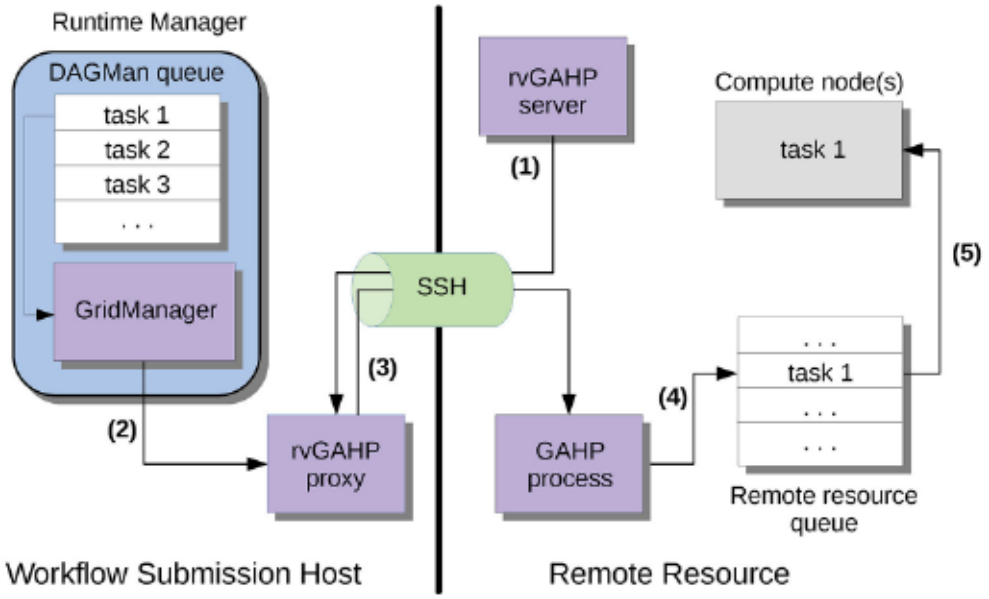

FIGURE 8
The approach used by CyberShake for push-based job submission using rvGAHP (Callaghan et al., 2017). The blue region denotes the workflow task list, and the purple squares show the separate computing processes involved: (1) The rvGAHP server on the remote system creates an SSH connection to the workflow submission host and starts up the rvGAHP proxy. (2) When a task is ready to run, the local GridManager submits it through the rvGAHP proxy. (3) The rvGAHP proxy sends the job over the SSH connection to a GAHP process on the remote resource. (4) The GAHP process sends the tasks to the remote queue. (5) The task starts up and runs on a compute node. More details are available in Callaghan et al.2(017).

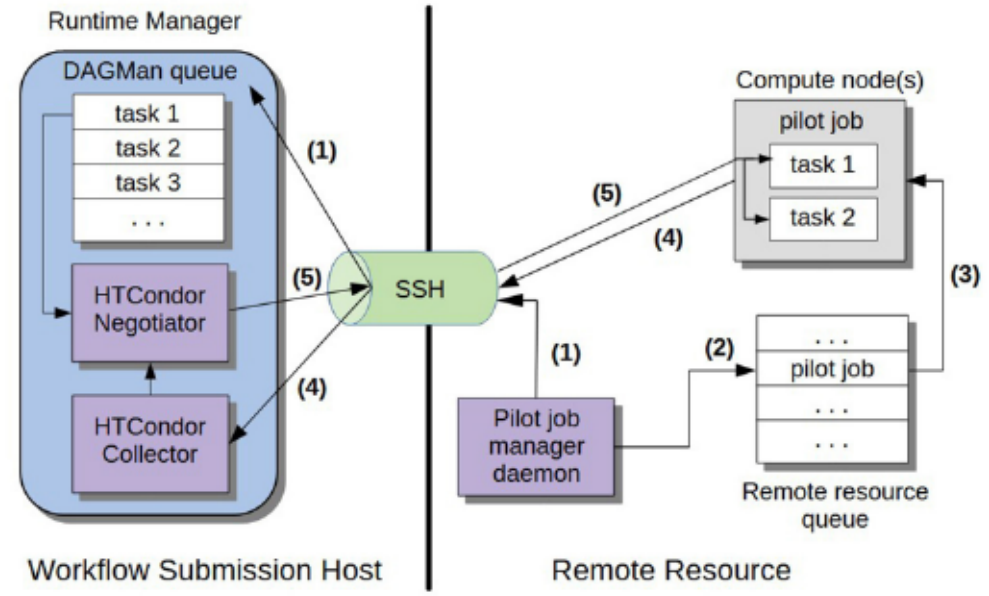

FIGURE 9
The approach used by CyberShake for pull-based job submission with HTCondor glideins. The blue region denotes the workflow task list, and the purple squares show the separate computing processes involved. (1) The pilot job manager daemon on the remote system monitors the DAGMan queue over SSH. (2) When a user-specified number of jobs are waiting, the pilot job manager daemon submits a pilot job to the remote queue. (3) A pilot job that can run tasks on the nodes start on compute nodes. (4) Each pilot job calls back via SSH to the HTCondor collector and advertises its available resources. (5) The HTCondor negotiator matches tasks with the advertised slots.

a large number (4,000+) of hybrid CPU and GPU compute nodes and a large and fast filesystem, matched the resource requirements for our planned CyberShake study. Its large number of GPUs is a good match for our GPU wave propagation code, and the large number of compute nodes enabled many of our site-specific workflows to run simultaneously, reducing makespan.

Additionally, though the CyberShake workflow system supports execution across an arbitrary number of systems, performing stages 1, 2, and 3 on the same system reduces the amount of data transferred over the network.

The Study 22.12 stress test ran from December 17, 2022 to January 7, 2023, and ran all four stages of the workflows for 20 (6%) of the sites. After all the stress test workflows completed, we examined the results and identified and fixed several issues, before we began the rest of the Study. We identified selected parameters that needed to be propagated between workflow jobs. We found problems that prevented some glidein jobs from starting promptly, and a problem where bundling was set up incorrectly for certain tasks. We found that we needed to correct runtime and memory requirements for some of our codes. Once these issues were addressed, we began our CyberShake Study 22.12 production run. We have consistently found that running a stress test before beginning the full study enables us to find and fix problems more easily and improves our overall time to solution.

The production run for CyberShake Study 22.12 ran from January 17, 2023 to April 4, 2023. We chose to run the four workflow stages as described in Section 2.2 sequentially. This was to simplify management of the study and utilize compute time on *Summit* strategically, in case the computational cost of the study meant that a second system would be needed. It was our highest priority to finish Stage 1 (SGT) workflows due to their GPU jobs; *Summit* has over 18,000 GPUs. Our second priority was to finish all Stage 2 workflows, to avoid having to migrate the SGT intermediate data files to a separate filesystem. We finished all Stage 1 (SGT) workflows on January 26, the Stage 2+4 (post-processing and data products) workflows on February 22, Stage 3 (broadband) on March 6, and the remaining Stage 4 (data products) workflows on April 4. Thankfully, we were able to complete the entire study in the available allocation on *Summit*.

We have found that the extreme scale of the CyberShake workflows often leads to unforeseen problems during production runs. In Study 22.12, we ran out of disk space on our relational database, so the last month was spent clearing additional space and running the remaining Stage 4 workflows. Periodically the Globus proxy would expire; this was fixed by easily renewing it. Occasionally the long-running daemons to support glideins and rvGAHP jobs would terminate due to a node restart; these were simple to restart. Additionally, we encountered a number of issues on *Summit* in which files were not copied or written to the filesystem correctly and parts of some of the files were missing, even though the copy command or write call had a successful return code. We configured Globus to include checksum integrity checks for all transfers, and we added a post-transfer file checking stage that signals an error if it detects any empty files. We mitigated the data transfer issues by manually modifying workflows to resume at either the data generation or data copy job. We then automatically restarted the workflows via the Pending Site File. Problems like this illustrate the importance of checkpointing and restart capabilities in CyberShake workflows.

Overall, Study 22.12 required 772,000 node-hours on *Summit*, equivalent to using the entire system [#5 on the Top500 list (Top500, 2023)] for a week. The makespan was 1,829 h, yielding an average of 442 nodes (9%) usage on *Summit* during the entire study. Our workflow system executed jobs on a peak of 3,382 nodes (73% of *Summit*) without using a reservation. We were able to obtain this high level of system utilization by bundling our SGT jobs and submitting jobs to *Summit* with node counts large enough to obtain queue priority. We managed about 2.5 PB of data, of which 74 TB was output data products. The output data was comprised of about 19 million files, which were staged back to USC CARC for long-term storage. This included 420 million two-component seismograms, and 83 billion intensity measures.

3.4 CyberShake study 22.12 scientific results

This study advanced the state-of-the-art in physics-based PSHA for Southern California. Scientific advancements in this model included (1) integration of the low-frequency deterministic results generated in Stage 2 with the high-frequency stochastic results generated in Stage 3 to produce a broadband CyberShake hazard model up to 50 Hz; (2) increased sampling of hypocentral, slip, and rupture velocity variability; (3) updates to the rupture generator used to produce the suite of earthquakes from the earthquake rupture forecast; and (4) improvements to the 3D velocity model to reduce the near-surface velocities outside of the sedimentary basins (Callaghan et al., 2023). While previous CyberShake studies have been used in broad impact public seismic hazard products, including the 2023 National Seismic Hazard Model (NSHM) (Field et al., 2023), the results from CyberShake Study 22.12 are provisional and under scientific review. More details about the data from the study are given in the Data Availability Statement. Multiple seismological, civil engineering, and risk management research groups have retrieved selected data products from the CyberShake Study 22.12 model to further their research (e.g., Lee et al., 2023), using the results to investigate the impact of rupture directivity on ground motion models, seismic hazard and risk assessments of distributed water and transportation infrastructure, tall building response to strong ground motions, and basin effects on ground motions. We anticipate that as the CyberShake Study 22.12 results are fully validated, they will be used for a variety of broad impact seismic hazard and risk assessment purposes like previous CyberShake studies.

Figure 10 presents a sample hazard map produced from Study 22.12.

The CyberShake workflow framework executed 31,897 jobs, of which 19,184 were remote, in 988 workflows. Individual workflow makespans varied from 2.5 h to 30 days. We used the task bundling techniques described in Section 3.2.7 to bundle 27 million short-running serial tasks in the broadband stage into 349 longer jobs. On average, each remote job waited for 5.5 h before executing, reflecting queue times on *Summit*.

4 Discussion

We have highlighted a series of technical challenges encountered by the CyberShake collaboration, and solutions engineered with the help of extreme-scale scientific workflow tools. We anticipate that as the CyberShake project continues, we will encounter new challenges driven by scientific demands. For

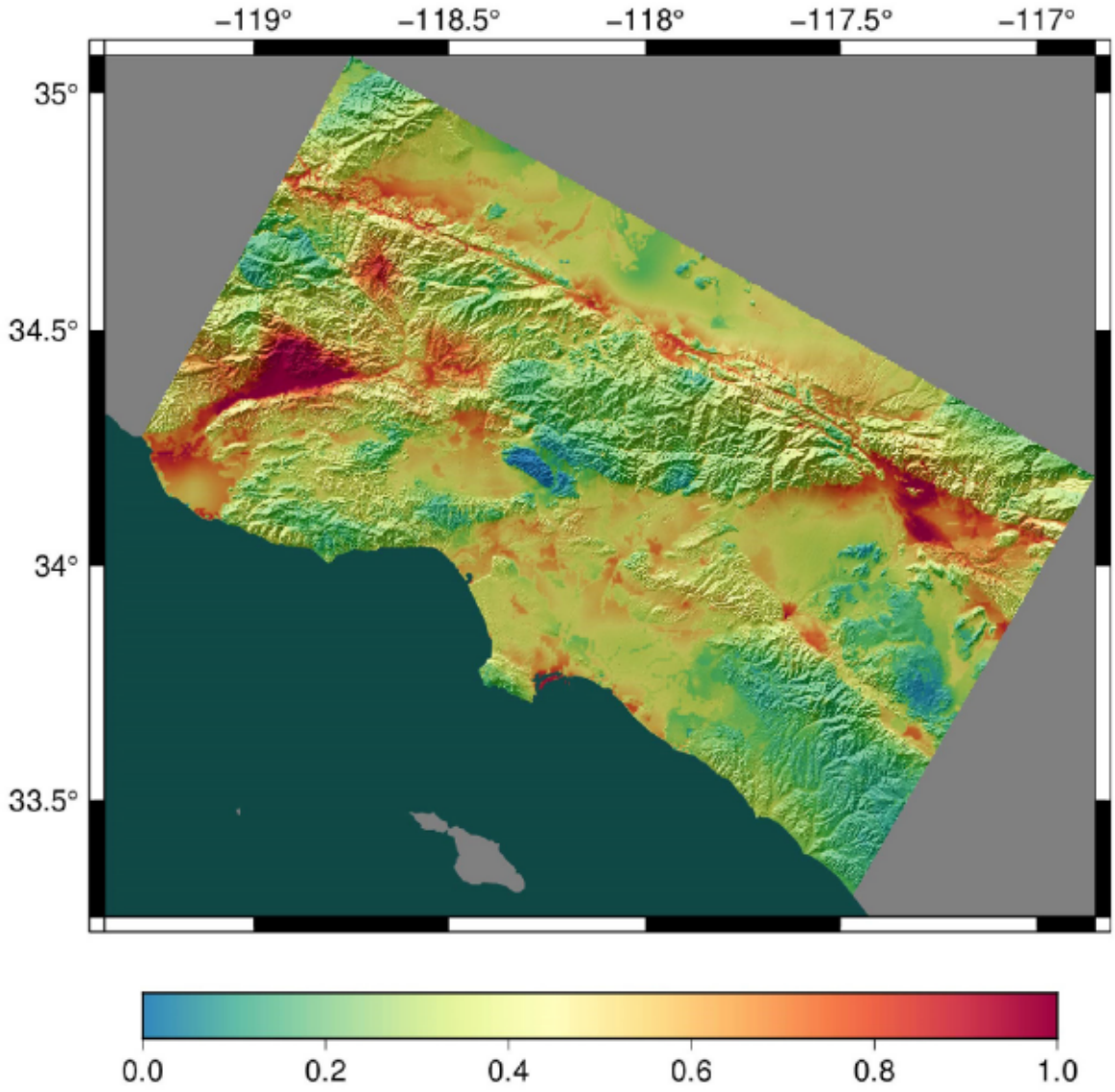

FIGURE 10
Hazard map for southern California generated with Study 22.12, showing the ground motion expected with a 2% chance of exceedance in 50 years. Units are g (acceleration due to gravity). Warm colors indicate higher hazard.

example, if we increase the frequency of the wave propagation simulations, we may need to more tightly couple our velocity mesh and wave propagation codes to reduce filesystem load. We will continue to evolve the workflow solution presented here to accomplish our science goals.

Our CyberShake studies have provided impactful scientific and technical developments over 15 years. CyberShake results have informed the USGS National Seismic Hazard Model Project (Moschetti et al., 2017); the Building Seismic Safety Council and the American Society of Civil Engineers for use in building designs (Crouse et al., 2018); and the City of Los Angeles for adoption in tall building design regulations (Crouse and Jordan, 2016). CyberShake simulations have been also used to train the Earthquake Early Warning system developed for California (Böse et al., 2014).

The computing requirements CyberShake are already significant and are likely to increase as future CyberShake PSHA hazard models are developed with higher resolution earth models and more realistic physics. Our workflow tools help to automate complex calculations, and extensive automation is essential for dealing with extreme scale computing without extreme scale personnel. The workflow solutions described here have helped SCEC advance leading-edge seismic hazard analysis from terascale to exascale—six orders of magnitude—and through more than four generations of supercomputers, while accommodating increasing computational and data capabilities and requiring fewer people to perform the simulations. There is a good match between the computational needs of CyberShake, the capabilities of Pegasus and HTCondor workflow tools, and the availability of open-science supercomputing resources through the NSF and DOE. Our workflow tools also support computational reproducibility by providing definition, tracking, and logging of complex computing essential for scientific transparency and reproducibility.

CyberShake shows how extreme-scale scientific workflows can utilize open-science supercomputers to produce transformative, simulation-based, seismic hazard models.

Data availability statement

The datasets presented in this study can be found in online repositories. The names of the repository/repositories and accession number(s) can be found at: Globus Collection: SCEC CyberShake Study 22.12 BB (https://app.globus.org/file-manager?origin_id=863d9e0a-d8f9-4c82-a04c-14fdc0767478). Globus Collection: SCEC CyberShake Study 22.12 LF (https://app.globus.org/file-manager?origin_id=024d5492-9104-46aa-8cd0-fe867c6fbad9). CyberShake Study 22.12 results are managed using USC CARC storage resources. They are distributed to SCEC community members using web-based tools and custom data access software. Details for accessing Study 22.12 data are available at https://github.com/SCECcode/cs-data-tools/.

Author contributions

SC: Data curation, Formal analysis, Investigation, Software, Validation, Visualization, Writing—original draft, Writing—review & editing. PM: Project administration, Resources, Supervision, Writing—original draft, Writing—review & editing. FS: Software, Validation, Writing—review & editing. M-HS: Software, Writing—review & editing, Validation. KM: Formal analysis, Software, Visualization, Writing—review & editing. RG: Methodology, Software, Validation, Writing—review & editing. KO: Methodology, Software, Validation, Writing—review & editing. YC: Software, Writing—review & editing. KV: Software, Writing—review & editing. AK: Writing—review & editing, Funding acquisition, Methodology, Validation. CG: Conceptualization, Funding acquisition, Resources, Supervision, Validation, Writing—review & editing. ED: Writing—review & editing, Methodology, Software. TJ: Conceptualization, Methodology, Writing—review & editing. YB-Z: Conceptualization, Funding acquisition, Supervision, Writing—review & editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This research was supported by the Southern California Earthquake Center with funding from National Science Foundation Cooperative Agreements EAR-1600087 and EAR-2225216 and United States Geological Survey Cooperative Agreements G17AC00047 and G22AC00070 (SCEC Contribution #13382). Additional support was provided by the Pacific Gas and Electric Company. This research used the Pegasus Workflow Management System funded by the National Science Foundation under the OAC SI2-SSI program, grant #1664162. This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which was supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725. This computational work was performed using the DOE ALCC allocation, Improved Seismic Hazard Modeling Using Physics-based Simulations.

Acknowledgments

The authors acknowledge the Center for Advanced Research Computing (CARC) at the University of Southern California for providing computing resources that have contributed to the research results reported in this publication (https://carc.usc.edu). We thank Matthew Norman, Bill Jendrzejek, and Steven Lin for their technical assistance. We thank George Papadimitriou, Mats Rynge, and Rajiv Mayani for assistance with Pegasus. We thank Brad Aagaard for his thorough and helpful feedback. Any use of trade, firm, or product names is for descriptive purposes only and does not imply endorsement by the U.S. Government.

Conflict of interest

AK was employed by Pacific Gas and Electric. This study received funding from Pacific Gas and Electric. The funder had the following involvement with the study: (1) input on derived data products, and (2) development of validation methodology.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Albrecht, M., Donnelly, P., Bui, P., and Thain, D. (2012). "Makeflow: a portable abstraction for data intensive computing on clusters, clouds, and grids," in *Proceedings of the 1st ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies (SWEET '12)* (Scottsdale, AZ).

Allen, B., Bresnahan, J., Childers, L., Foster, I., Kandaswamy, G., Kettimuthu, R., et al. (2012). Software as a service for data scientists. *Commun. ACM* 55, 81–88. doi: 10.1145/2076450.2076468

Apache Airflow (2024). Available online at: https://airflow.apache.org/ (accessed March 11, 2024).

Babuji, Y., Woodard, A., Li, Z., Katz, D. S., Clifford, B., Kumar, R., et al. (2019). "Parsl: pervasive parallel programming in Python," in *Proceedings of the 28th International Symposium on High-Performance Parallel and Distributed Computing (HPDC '19)* (Phoenix, AZ).

Baker, J. W., and Chen, Y. (2022). "Spatial correlation analysis of CyberShake simulations, considering multiple ruptures," in *Proceedings of the 12th National Conference on Earthquake Engineering* (Earthquake Engineering Research Institute). Available online at: https://eeri.org/what-we-offer/digital-library/?lid=12784 (accessed April 9, 2024).

Bielak, J., Graves, R. W., Olsen, K. B., Taborda, R., Ramírez-Guzmán, L., Day, S. D., et al. (2010). The ShakeOut earthquake scenario: verification of three simulation sets. *Geophys. J. Int.* 180, 375–404. doi: 10.1111/j.1365-246X.2009.04417.x

Bijelic, N., Lin, T., and Deierlein, G. G. (2020). Efficient intensity measures and machine learning algorithms for collapse prediction of tall buildings informed by SCEC CyberShake ground motion simulations. *Earthq. Spectra.* 36, 1188–1207. doi: 10.1177/8755293020919414

Boehm, B. W. (1981). *Software Engineering Economics*. Englewood Cliffs, NJ: Prentice-Hall.

Böse, M., Graves, R. W., Gill, D., Callaghan, S., and Maechling, P. J. (2014). CyberShake-derived ground-motion prediction models for the Los Angeles region with application to earthquake early warning. *Geophys. J. Int.* 198, 1438–1457. doi: 10.1093/gji/ggu198

Bozorgnia, Y., Abrahamson, N. A., Al Atik, L., Ancheta, T. D., Atkinson, G. M., Baker, J. W., et al. (2014). NGA-West2 research project. *Earthq. Spectra.* 30, 973–987. doi: 10.1193/072113EQS209M

Callaghan, S., Juve, G., Vahi, K., Maechling, P. J., Jordan, T. H., and Deelman, E. (2017). "rvGAHP—push-based job submission using reverse SSH connections," in *Proceedings of the 12th Workshop on Workflows in Support of Large-Scale Science (WORKS '17)* (Denver, CO).

Callaghan, S., Maechling, P. J., Goulet, C. A., Milner, K. R., Su, M., Vahi, K., et al. (2019). Computational tools to support large-scale CyberShake PSHA simulations. *Seismol. Res. Lett.* 90, 876–877.

Callaghan, S., Maechling, P. J., Silva, F., Goulet, C. A., Milner, K. R., Shaw, B. E., et al. (2022). *Updates to the CyberShake PSHA platform. 2022 SCEC Annual Meeting, poster #240*. Available online at: https://www.scec.org/meetings/2022/am/poster/240 (accessed April 9, 2024).

Callaghan, S. A., Maechling, P. J., Silva, F., Milner, K. R., Su, M., Goulet, C. A., et al. (2023). Updated broadband CyberShake PSHA model for Southern California. *Seismol. Res. Lett.* 94: 1202.

Comparison of CyberShake Studies (2024). Available at https://strike.scec.org/scecpedia/Comparison_of_CyberShake_Studies (accessed April 9, 2024).

Cornell, C. A. (1968). Engineering seismic risk *analysis. Bullet. Seismol. Soc. Am.* 58, 1583–1606. doi: 10.1785/BSSA0580051583

Crouse, C. B., and Jordan, T. H. (2016). "Development of new ground-motion maps for Los Angeles based on 3-D numerical simulations and NGA West2 equations," in *Proceedings of the SMIP16 Seminar on Utilization of Strong Motion Data* (Irvine: University of California, Irvine). Available online at: https://www.conservation.ca.gov/cgs/documents/program-smi/seminar/SMIP16_P1_Paper_by_Crouse.pdf (accessed October 6, 2016).

Crouse, C. B., Jordan, T. H., Milner, K. R., Goulet, C. A., Callaghan, S., and Graves, R. W. (2018). "Site-specific MCER response spectra for Los Angeles Region based on 3-D numerical simulations and the NGA West2 equations," in *Proceedings of the 11th National Conference in Earthquake Engineering* (Earthquake Engineering Research Institute), 518. Available online at: https://11ncee.org/images/program/papers/11NCEE-000518.pdf (accessed April 9, 2024).

Cui, Y., Olsen, K. B., Jordan, T. H., Lee, K., Zhou, J., Small, P., et al. (2010). "Scalable earthquake simulation on petascale supercomputers," in *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage, and Analysis (SC'10)* (New Orleans, LA).

Cui, Y., Poyraz, E., Olsen, K. B., Zhou, J., Withers, K., Callaghan, S., et al. (2013). "Physics-based seismic hazard analysis on petascale supercomputers," in *Proceedings of the 2013 ACM/IEEE International Conference on High Performance Computing, Networking, Storage, and Analysis (SC'13)* (Denver, CO). doi: 10.1145/2503210.2503300

Deelman, E., Ferreira da Silva, R., Vahi, K., Rynge, M., Mayani, R., Tanaka, R., et al. (2021). The Pegasus workflow management system: translational computer science in practice. *J. Comput. Sci.* 52:101200. doi: 10.1016/j.jocs.2020.101200

Deelman, E., Vahi, K., Juve, G., Rynge, M., Callaghan, S., Maechling, P. J., et al. (2015). Pegasus, a workflow management system for science automation. *Future Gen. Comp. Sy.* 46, 17–35. doi: 10.1016/j.future.2014.10.008

Deelman, E., Vahi, K., Rynge, M., Mayani, R., da Silva, R. F., Papadimitriou, G., et al. (2019). The evolution of the pegasus workflow management software. *Comput. Sci. Eng.* 21, 22–36. doi: 10.1109/MCSE.2019.2919690

Di Tommaso, P., Chatzou, M., Floden, E. W., Barja, P. P., Palumbo, E., and Notredame, C. (2017). Nextflow enables reproducible computational workflows. *Nat. Biotechnol.* 35, 316–319. doi: 10.1038/nbt.3820

DirectSynth Github Code Repository (2024). Available online at: https://github.com/SCECcode/cybershake-core/tree/main/DirectSynth (accessed April 9, 2024).

Field, E. H., Dawson, T. E., Felzer, K. R., Frankel, A. D., Gupta, V., Jordan, T. H., et al. (2009). Uniform California earthquake rupture forecast, version 2 (UCERF2). *B. Seismol. Soc. Am.* 99, 2053–2107. doi: 10.1785/0120080049

Field, E. H., Jordan, T. H., and Cornell, C. A. (2003). OpenSHA; a developing community-modeling environment for seismic hazard analysis. *Seismol. Res. Lett.* 74, 406–419. doi: 10.1785/gssrl.74.4.406

Field, E. H., Milner, K. R., Hatem, A. E., Powers, P. M., Pollitz, F. F., Llenos, A. L., et al. (2023). The USGS 2023 conterminous U.S. *time-independent earthquake rupture forecast. B. Seismol. Soc. Am.* 114, 523–571. doi: 10.1785/0120230120

Foster, I. (2011). Globus online: accelerating and democratizing science though cloud-based services. *IEEE Internet Comput.* 15, 70–73. doi: 10.1109/MIC.2011.64

Gerstenberger, M. C., Marzocchi, W., Allen, T., Pagani, M., Adams, J., Danciu, L., et al. (2020). Probabilistic seismic hazard analysis at regional and national scales: State of the art and future challenges. *Rev. Geophys.* 58:e2019RG000653. doi: 10.1029/2019RG000653

Graves, R., and Aagaard, B. (2011). Testing long-period ground-motion simulations of scenario earthquakes using the Mw 7.2 El Mayor-Cucapah mainshock: evaluation of finite-fault rupture characterization and 3D seismic velocity models. *B. Seismol. Soc. Am.* 101, 895–907. doi: 10.1785/0120100233

Graves, R., Jordan, T. H., Callaghan, S., Deelman, E., Field, E., Juve, G., et al. (2011). CyberShake: a physics-based seismic hazard model for Southern California. *Pure Appl. Geophys.* 168, 367–381. doi: 10.1007/s00024-010-0161-6

Graves, R., and Pitarka, A. (2015). Refinements to the Graves and Pitarka (2010) broadband ground-motion simulation method. *Seismol. Res. Lett.* 86, 75–80. doi: 10.1785/0220140101

Hu, Z., Olsen, K. B., and Day, S. M. (2022). 0–5 Hz deterministic 3-D ground motion simulations for the 2014 La Habra, California, Earthquake. *Geophys. J. Int.* 2162–2182. doi: 10.1093/gji/ggac174

IBM Spectrum LSF Documentation (2024). Available online at: https://www.ibm.com/docs/en/spectrum-lsf/10.1.0 (accessed April 9, 2024).

Jette, M. A., and Wickberg, T. (2023). "Architecture of the slurm workload manager," in *Job Scheduling Strategies for Parallel Processing. JSSPP 2023. Lecture Notes in Computer Science, Vol 14283*, eds D. Klusáček, J. Corbalán, and G. P. Rodrigo (Cham: Springer).

Jordan, T. H., Callaghan, S., Graves, R. W., Wang, F., Milner, K. R., Goulet, C. A., et al. (2018). "CyberShake models of seismic hazards in southern and central California," in *Proceedings of the 11th National Conference on Earthquake Engineering* (Earthquake Engineering Research Institute). Available online at: https://11ncee.org/images/program/papers/11NCEE-001458.pdf (accessed April 9, 2024).

Lee, Y., Goulet, C. A., Hu, Z., and Callaghan, S. (2023). "Evaluation of the impacts on risk assessments for distributed infrastructure systems from ground motion median, variability, and spatial correlation in CyberShake simulations," in *Southern California Earthquake Center Annual Meeting, Poster #192*. Available online at: https://www.scec.org/meetings/2023/am/poster/192 (accessed April 9, 2024).

Maechling, P. J., Silva, F., Callaghan, S., and Jordan, T. H. (2015). SCEC broadband platform: system architecture and software implementation. *Seismol. Res. Lett.* 86, 27–38. doi: 10.1785/0220140125

McCallen, D., Petersson, A., Rodgers, A., Pitarka, A., Miah, M., Petrone, F., et al. (2021). EQSIM–A multidisciplinary framework for fault-to-structure earthquake simulation on exascale computers part I: computational models and workflow. *Earthq. Spectra* 37, 707–735. doi: 10.1177/8755293020970982

Milner, K. R., Shaw, B. E., Goulet, C. A., Richards-Dinger, K. B., Callaghan, S., Jordan, T. H., et al. (2021). Toward physics-based nonergodic PSHA; a prototype fully deterministic seismic hazard model for Southern California. *B. Seismol. Soc. Am.* 111, 898–915. doi: 10.1785/0120200216

Moschetti, M. P., Luco, N., Baltay, A. S., Boyd, O. S., Frankel, A. D., Graves, R., et al. (2017). "Incorporating long-period (T>1 s) ground motions from 3D simulations into the U.S. National Seismic Hazard Model," in *Proceedings of the 16th World Conference on Earthquake Engineering* (International Association for Earthquake Engineering). Available online at: http://wcee.nicee.org/wcee/article/16WCEE/WCEE2017-4423.pdf (accessed April 9, 2024).

Nweke, C. C., Stewart, J. P., Graves, R. W., Goulet, C. A., and Brandenberg, S. J. (2022). Validating predicted site response in sedimentary basins from 3D ground motion simulations. *Earthq. Spectra.* 38, 2135–2161. doi: 10.1177/87552930211073159

Olsen, K. B., Day, S. M., Minster, J. B., Cui, Y., Chourasia, A., Faerman, M., et al. (2006). Strong shaking in Los Angeles expected from southern San Andreas event. *Geophys. Res. Lett.* 33:25472. doi: 10.1029/2005GL025472

Papadimitriou, G., and Deelman, E. (2021). "A lightweight GPU monitoring extension for Pegasus Kickstart," in *Proceedings of the 16th Workshop on Workflows in Support of Large-Scale Science (WORKS '22)* (Dallas, TX).

Pegasus WMS Documentation (2023). *Documentation*. Available online at: https://pegasus.isi.edu/documentation/ (accessed December 19, 2023).

Pitarka, A., Graves, R., Irikura, K., Miyakoshi, K., Wu, C., Kawase, H., et al. (2021). Refinements to the Graves-Pitarka kinematic rupture generator, including a dynamically consistent slip rate function, applied to the 2019 Mw 7.1 Ridgecrest earthquake. *B. Seismol. Soc. Am.* 112, 287–306. doi: 10.1785/0120210138

Reed, D. A., Bajcsy, R., Fernandez, M. A., Griffiths, J.-M., Mott, R. D., Dongarra, J., et al. (2005). *Computational Science: Ensuring America's Competitiveness*. Available online at: https://apps.dtic.mil/sti/citations/ADA462840 (accessed April 9, 2024).

Roten, D., Yeh, T., Olsen, K. B., Day, S. M., and Cui, Y. (2023). Implementation of Iwan-Type nonlinear rheology in a 3D high-order staggered-grid finite-difference method. *B. Seismol. Soc. Am.* 113, 2275–2291. doi: 10.1785/0120230011

Rynge, M., Callaghan, S., Deelman, E., Juve, G., Mehta, G., Vahi, K., et al. (2012). "Enabling large-scale scientific workflows on petascale resources using MPI master/worker," in *Proceedings of the 1st Conference of the Extreme Science and Engineering Discovery Environment (XSEDE '12)* (Chicago, IL), 1–8.

SCEC Cybershake-Core Software Repository (2022). *SCEC Cybershake-Core Software Repository, study_22_12_v2 tag*. Available online at: https://github.com/SCECcode/cybershake-core/tree/study_22_12_v2 (accessed April 9, 2024).

SCEC Cybershake-Tools (2022). *SCEC Cybershake-Tools, Study 22_12_v2 Tag*. Available online at: https://github.com/SCECcode/cybershake-tools/tree/study_22_12_v2 (accessed April 9, 2024).

Small, P., Gill, D., Maechling, P. J., Taborda, R., Callaghan, S., Jordan, T. H., et al. (2017). The SCEC unified community velocity model software framework. *Seismol. Res. Lett.* 88, 1539–1552. doi: 10.1785/0220170082

Summit User Guide (2023). *Scheduling Policy*. Available online at: https://docs.olcf.ornl.gov/systems/summit_user_guide.html#scheduling-policy (accessed December 19, 2023).

Taborda, R., Olsen, K. B., Graves, R. W., Silva, F., Khoshnevis, N., Savran, W. H., et al. (2018). "High-frequency simulations: verification and validation of the M5.1 La Habra, CA, Earthquake," in *Proceedings of the 11th National Conference on Earthquake Engineering* (Earthquake Engineering Research Institute). Available online at: https://11ncee.org/images/program/papers/11NCEE-001869.pdf (accessed April 9, 2024).

Teng, G., and Baker, J. (2019). Evaluation of SCEC CyberShake ground motions for engineering practice. *Earthq. Spectra*. 35, 1311–1328. doi: 10.1193/100918EQS230M

Thain, D., Tannenbaum, T., and Livny, M. (2005). Distributed computing in practice: the Condor experience. *Concurr. Comp.-Pract. E.* 17, 323–356. doi: 10.1002/cpe.938

Top500 (2023). *Frontier Remains No. 1 in the TOP500 but Aurora With Intel's Sapphire Rapids Chips Enters With a half-Scale System at No. 2*. Available online at: https://www.top500.org/ (accessed December 19, 2023).

Villani, M., and Abrahamson, N. A. (2015). Repeatable site and path effects on the ground-motion sigma based on empirical data from Southern California and simulated waveforms from the CyberShake platform. *B. Seismol. Soc. Am.* 105, 2681–2695. doi: 10.1785/0120140359

Yeh, T., and Olsen, K. B. (2023). Fault damage zone effects on ground motions during the 2019 Mw 7.1 Ridgecrest, California, earthquake. *B. Seismol. Soc. Am.* 113, 1724–1738. doi: 10.1785/0120220249

Zhao, L., Chen, P., and Jordan, T. H. (2006). Strain Green's tensors, reciprocity, and their applications to seismic source and structure studies. *B. Seismol. Soc. Am.* 96, 1753–1763. doi: 10.1785/0120050253