ElastiSim

Ozden, Beringer, Mazaheri, et al.

James Osborne, Khue Le

The Why

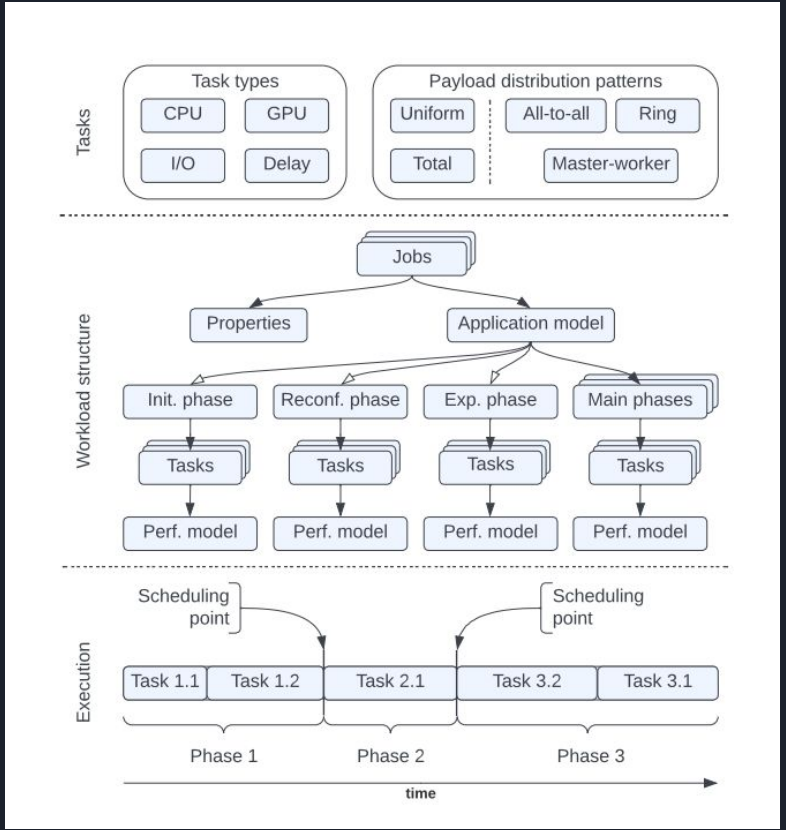
- Malleable jobs are jobs where the scheduler can reconfigure the resources allotted to the job and the job cna still adapt and run
  - This property should allow them to improve on system performance
    - This would happen because of increased resource allocation to speed up a job, or less resources so that the machine can run more jobs
- A number of HPC simulators have been made, whether from scratch or otherwise, but none of them yet are able to model adaptive workloads

The What

- ElastiSim is…
  - A batch-system simulator supporting the combined scheduling of rigid and malleable jobs.
  - A workload format to facilitate the description of malleable workloads using performance models.
  - A simulation approach for custom schedulers enabling the evaluation of topology-, I/O-, and progress-aware scheduling algorithms.
  - A SimGrid extension to simulate large-scale malleable GPU workloads.
  - A detailed malleability study of DL training workloads.

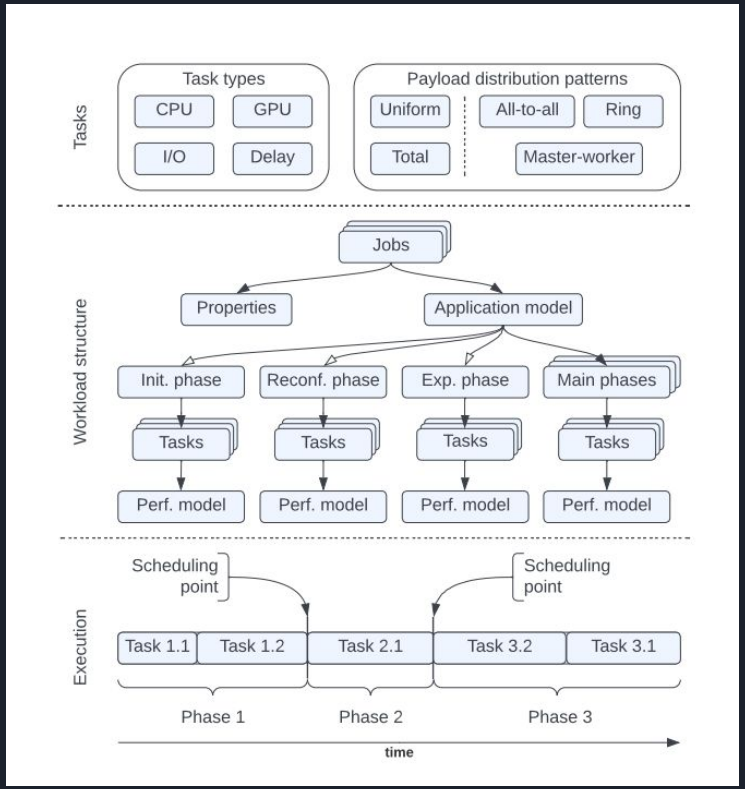
Workload Modeling - Structure

- Each job has a application model which represents the running application fo the job
- Each application is split into phases to represent different parts of the applications workload over time
- Each phase is split into task which are things like computations or network operations.
    - They act as a model of different simulated activities



Workload Modeling - Structure

- Scheduling events are placed between phases so that the scheduler can adjust resource usage of any given job
- This creates computational overhead, shown in the form of a Reconfiguration phase, this runs on all resources reconfigured
- If more resources are allotted, this also results in a Expansion phase that runs only on newly allotted resources
- The actions that occur on the initial configuration make up the Initialization phase

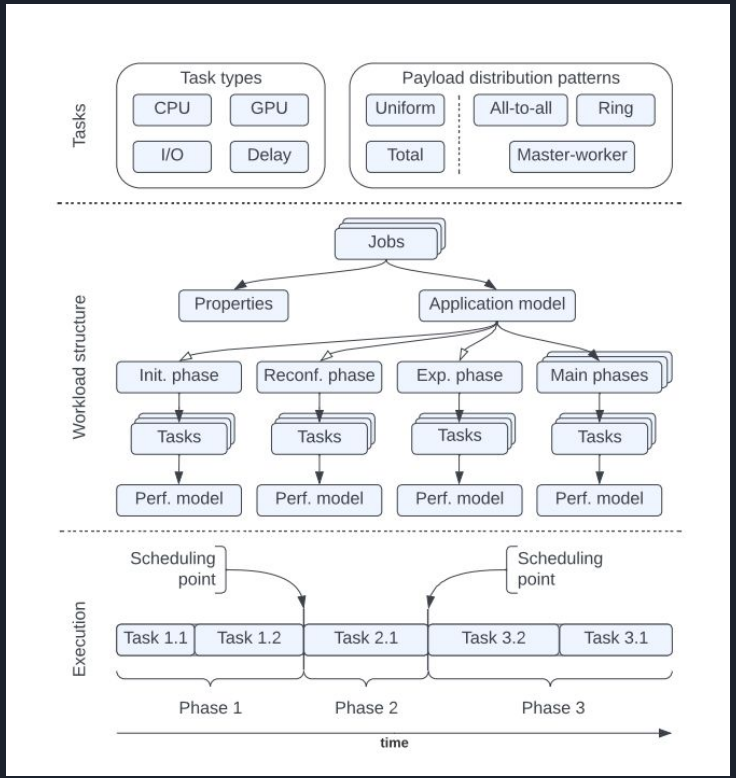


Workload Modeling - Tasks and Payloads

- There are 3 types of tasks
  - Compute, I/O, and generic delays
- Any given task describes the amount of work done in its payload
  - FLOPs (floating point operations), data in bytes read or written to a specific storage system, and time used up respectively
- Sequence tasks are tasks which can contain any number of other tasks including sequences
  - Made for added capacity for modeling regure events like checkpointing
- For define the distribution of payloads among resources, they introduced payload distribution patterns
  - Categorized into 2 types, regular and communication patterns
    - Regular patterns define payload distribution for compute, I/O, and delay tasks
    - Communication patterns define payload distribution in communication tasks

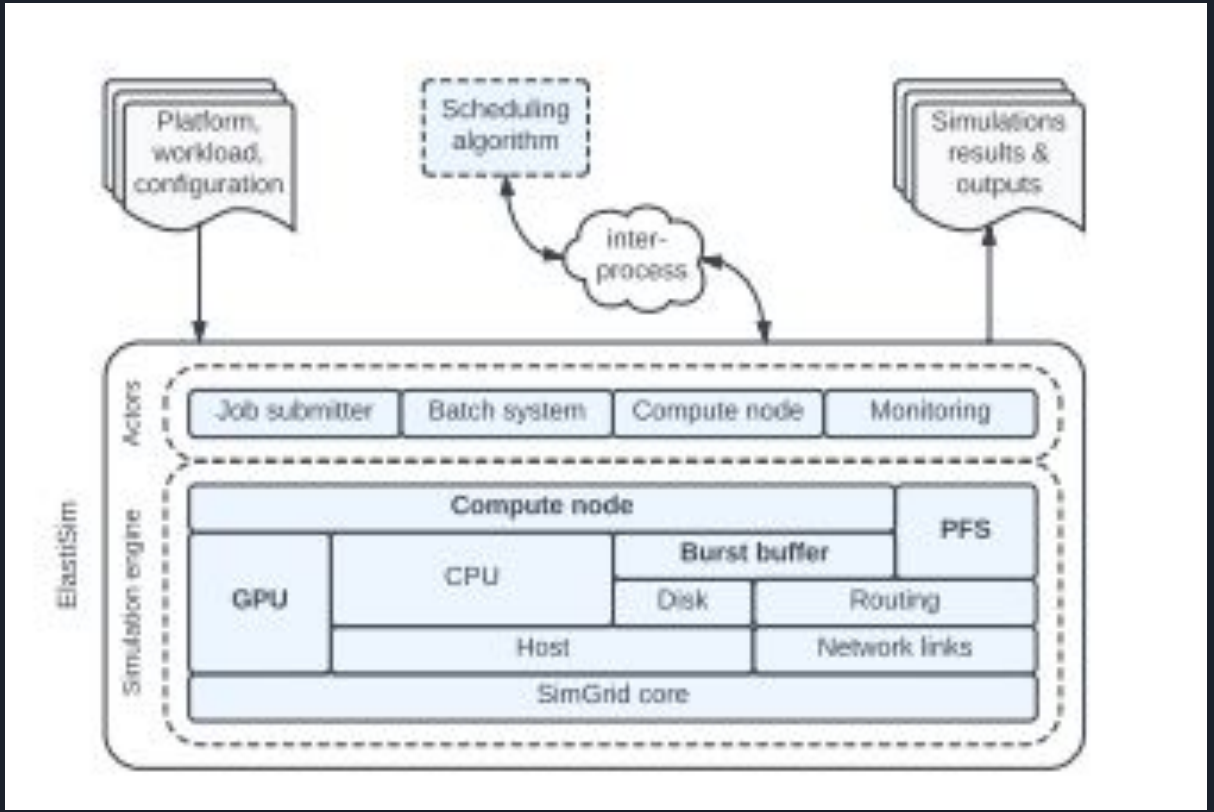
Workload Modeling - Structure

- To account for malleable jobs, dynamic means for adjustment to payloads are needed, and these are Performance models
- Essentially just a math model to quantify resource usage
  - The performance is updated every time tasks (re)configure



Architecture

- Simulation engine: extend SimGrid to support GPUs, introduce parallel file system and burst buffer semantics
- System actors
- Compute nodes are part of the simulation engine and system actors as they are considered resources but also continuously interact with the batch system.
- User provide external scheduling algorithm

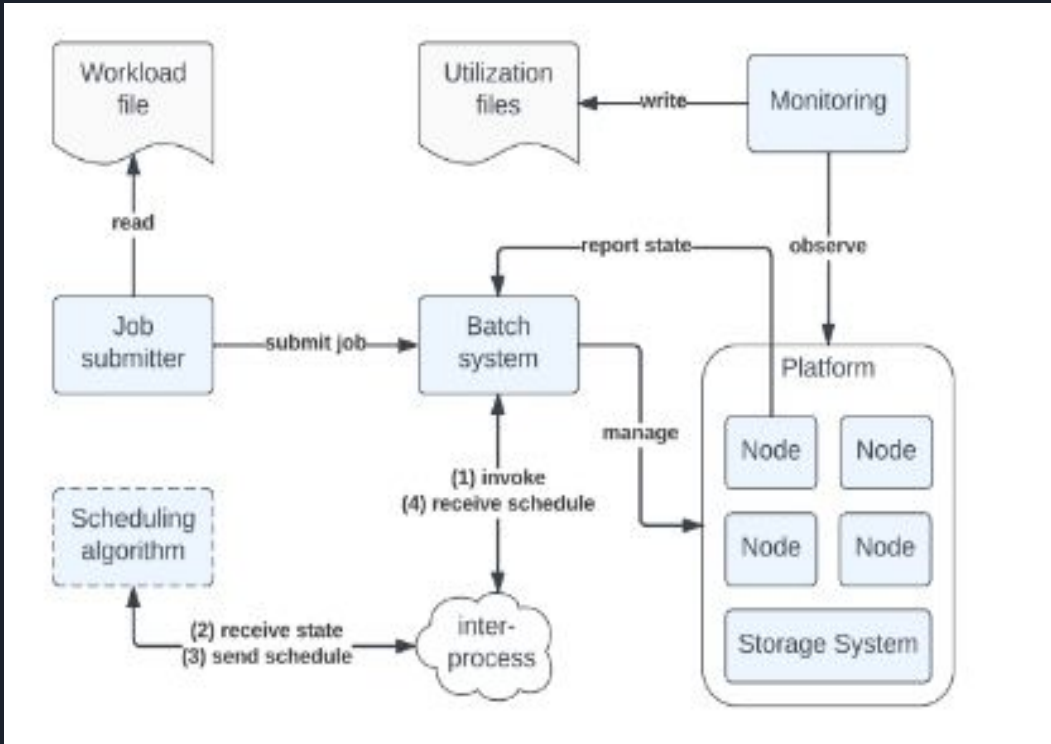


Storage and GPU model

- 2 types of storage system:
  - Parallel file system
  - Node-local burst buffers
- Provide GPU as separate computational resources during task execution because SimGrid does not support GPUs.
- GPU model: Users specify the number of GPUs per node, their computational performance, and the bandwidth of each connecting GPU link. Then distribute GPU computations among the requested GPUs located on the assigned compute node.

System actors

- Job submitter
- Batch system
- Compute nodes
- Monitoring/Observing actor



Scheduling algorithm

- Batch system invokes the algorithm periodically to allow reconfiguration of malleable jobs during runtime.
- Specify minimum scheduling intervals to prevent high number of jobs submission within a short period
- Each invocation contains: job queue, state of compute node, and the utilization of I/O system
- Scheduling algorithm is responsible for assigning compute node to jobs
- Initially, batch system applies scheduling decision immediately and allocates the corresponding nodes.
- Reconfigurations are stored separately and applied when the job reach next scheduling point.

Validate ElastiSim

Compare runtime between simulated and real DL training

Setup:

- Testbed: trained various convolutional neural networks (CNNs) on a GPU cluster with different configurations ranging from one to eight compute nodes
- Workload testing: Randomly chose 400 jobs from traces of the Microsoft DL cluster to simulate various DL workloads.

DL Application model

- Replicating the distributed training of convolutional neural networks (CNNs)
- Simulate various CNNs and dataset without modifying the underlying application model.
- Each compute node initially read the training and validation dataset from the PFS
- If the scheduler assigns additional nodes during runtime, each newly allocated node must perform the same tasks as a reconfiguration penalty before taking part in the simulated DL training
- Simulate a training step for each batch on a GPU as a sequence of tasks, repeated for the number of batches divided by the number of allocated GPUs.

- DL training step:
  - Forward/backward pass
  - Gradient synchronization
  - Gradient update
- Runtime analysis: similar runtime to real runtime of various training.

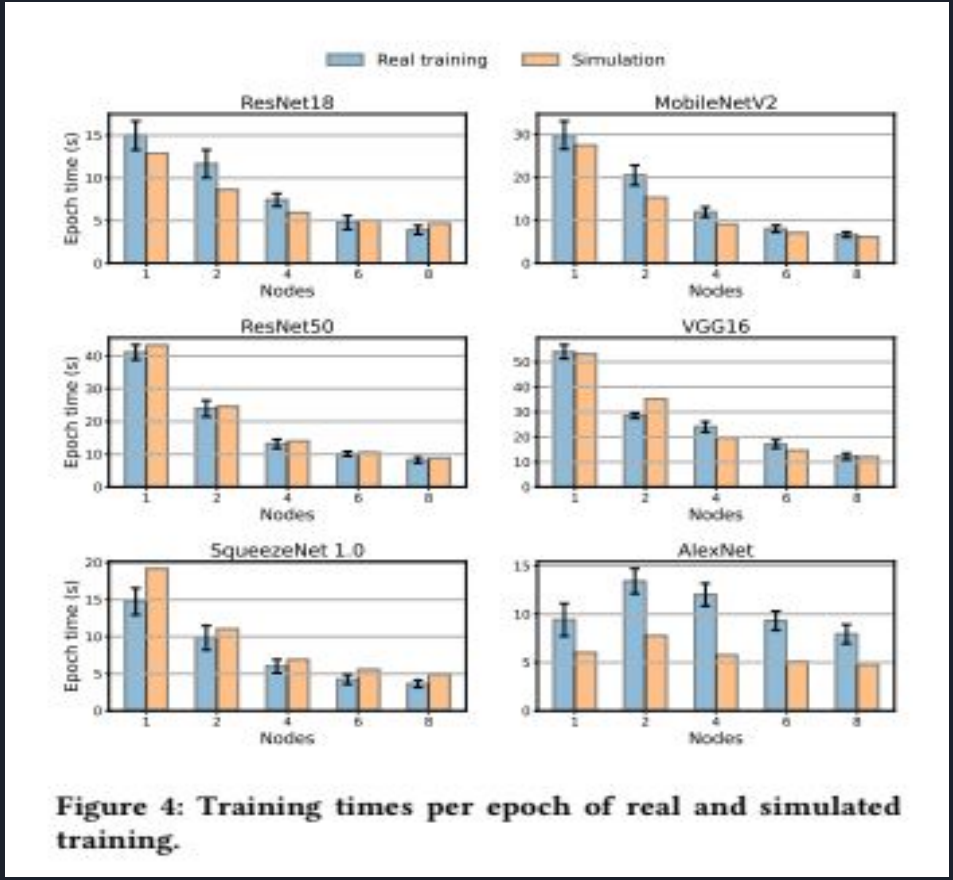


Figure 4: Training times per epoch of real and simulated training.

Thanks for Coming

Questions?