Computation Institute

ExTENCI: Using multiple cyberinfrastructures

Daniel S. Katz

Senior Fellow, Computation Institute (University of Chicago & Argonne National Laboratory)

Open Science Grid

TeraGrid™

MAPPER Seasonal School (1 Feb 2012)

Argonne NATIONAL LABORATORY

www.ci.anl.gov
www.ci.uchicago.edu

ExTENCI

- Website: https://sites.google.com/site/extenci/
- Joint Open Science Grid (OSG) and TeraGrid project
  - Funded by the National Science Foundation Office of Cyberinfrastructure (NSF OCI)
- PIs: Paul Avery (University of Florida), Ralph Roskies (Pittsburgh Supercomputing Center), and Daniel S. Katz (University of Chicago).
- Goal: Develop and provide production quality enhancements to the National Cyberinfrastructure that will enable specific science applications to more easily use both OSG and TeraGrid or broaden access to a capability to both TeraGrid and OSG users
- Metric: We will have reached the goal if a science project makes production use of the new capability and/or there is a measured performance improvement or additional throughput of science application

- Project rule: No Demos!
  - If it's not a step towards production usage with the users involved, don't do it

ExTENCI Subprojects

- Technologies – Application/User – Leads (Sites)
  - Distributed File System (Lustre-WAN) – CMS/ATLAS HEP – Ralph Roskies & Paul Avery (PSC, U Florida)
  - Virtual Machines – STAR & CMS – Carol Song, Sebastian Goasguen, Todd Tannenbaum (Purdue, Clemson, Wisconsin)
  - Workflow & Client Tools – SCEC & Protein Folding – Daniel S. Katz, Mike Wilde (U Chicago)
  - Job Submission Paradigms – Cactus Application – Shantenu Jha (Rutgers)

SCEC CyberShake

- Part of SCEC (PI: Tom Jordan, USC)
- Using large scale simulation data, estimate probabilistic seismic hazard (PSHA) curves for sites in southern California (probability that ground motion will exceed some threshold over a given time period)
- Used by hospitals, power plants, schools, etc. as part of their risk assessment
- Based on Rupture Variations (RupVar) set
  - ~14,000 Potential ruptures, with likelihood derived from earthquake rupture forecast (ERF)
- Use cases:
  1. Build map of area (run 1k-10k locations) based on latest RupVarset
  2. Run one location, based on latest RupVar set

  (~270 locations run to-date by SCEC, all on USC or TG)

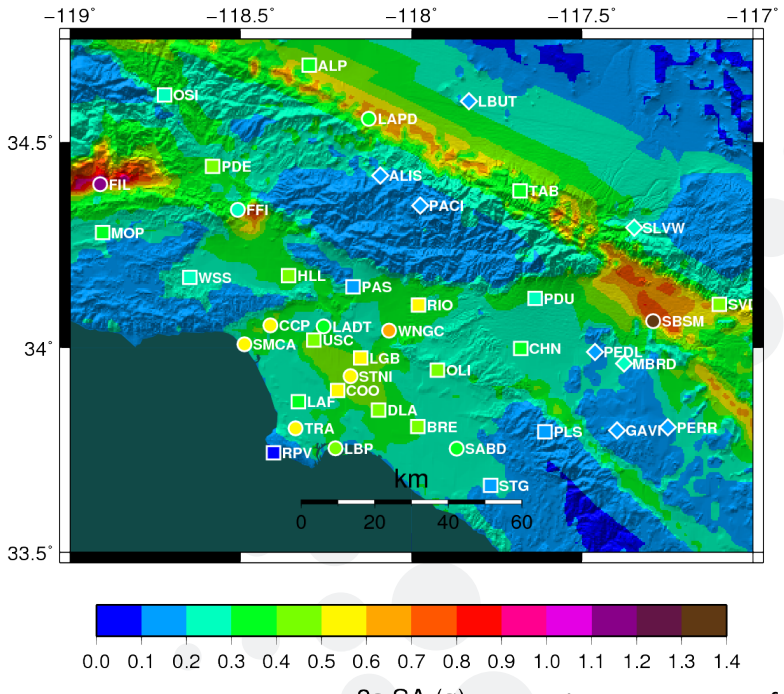


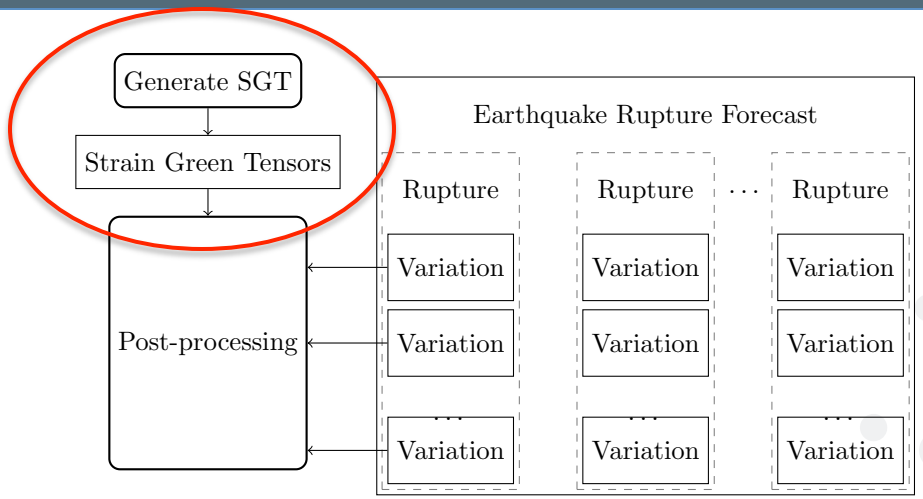
Image courtesy of Philip Maechling

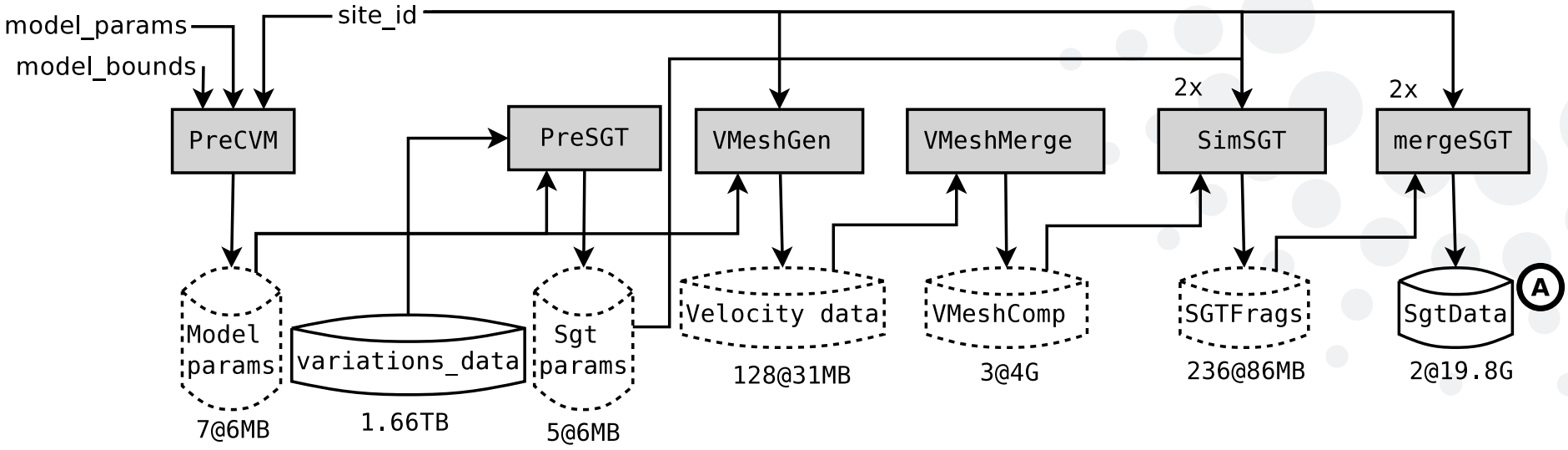
Managing these requires effective grid workflow tools for job submission, data management and error recovery. Currently running production on TG using Pegasus (ISI) and DAGman (Wisconsin).

SCEC CyberShake

- For each location
  - SGT: a pair of MPI runs (~400 cores, ~10 hours)
  - PP: ~840,000 parallel short jobs (420,000 synthetic seismograms, 420,000 extractions of peak ground motion)
- RupVar dataset
  - ~14,000 Potential ruptures, with likelihood derived from earthquake rupture forecast (ERF)
    - ~7000 will affect any one site
  - 2-1000 variations (45 average) per rupture, all equally likely (~600k rupture variations total)
  - Data size: ~2 TB total
    - 146 MB per rupture (average)
    - ~3 MB per rupture variation (average)

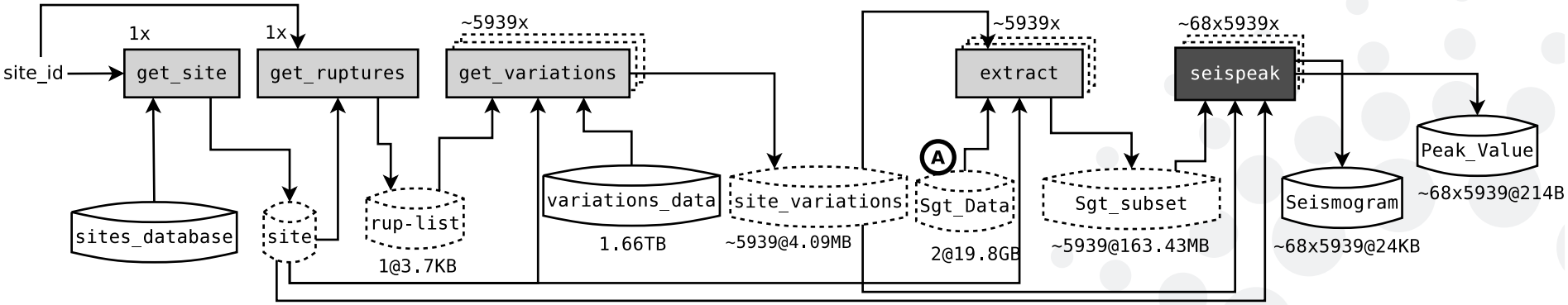
SGT Data Flow

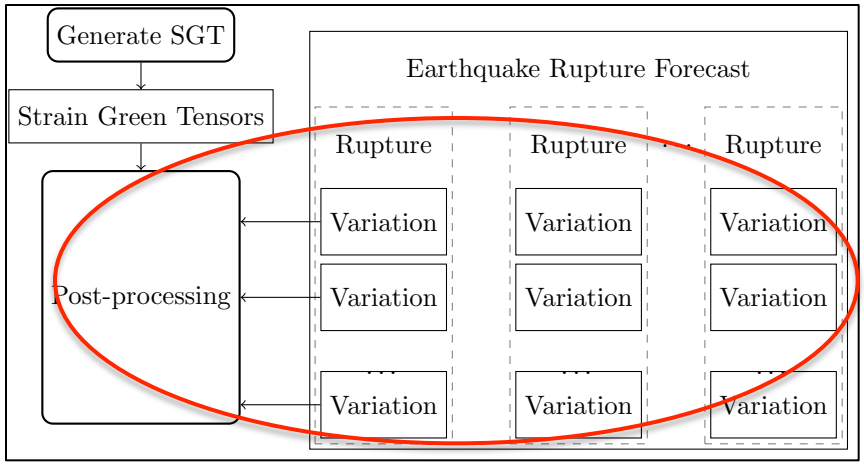


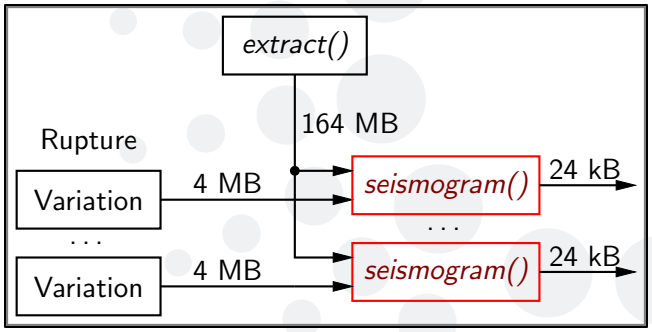


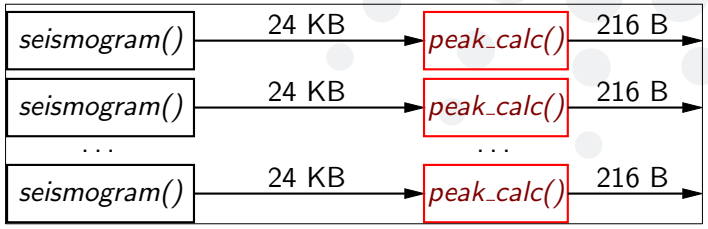
ExTENCI – d.katz@ieee.org

Postprocessing Data Flow









TeraGrid baseline

- Pre-stage RupVar data to TG sites X, Y, Z
- For each location:
  - Run SGT on TG site X, Y, Z (Pegasus workflow)
  - Run PP on TG site X, Y, Z (Pegasus workflow)

- Goal of ExTENCI is to use multiple cyberinfrastructures where it makes sense
- Does it make sense here?
  - It seems to, but let's see how to do it

Swift

- Idea: use Swift instead of Pegasus/DAGman
  - To get more control – Pegasus/DAGman is a black box – We can change Swift as needed
- Use Coasters multi-level scheduling (Swift's pilot job mechanism)
  - We can opportunistically acquire about 2000 CPUs on OSG in a few hours
- Swift handles backend operation
  - Can move files to/from and launch jobs on TG, OSG, etc.

Swift - http://www.ci.uchicago.edu/swift

Straightforward Swift implementation

```
Site site = "LGU";
Sgt sgt<"From MPI code computed in
TeraGrid">;

Rupture rups[] = get_rupture(sgt);
```

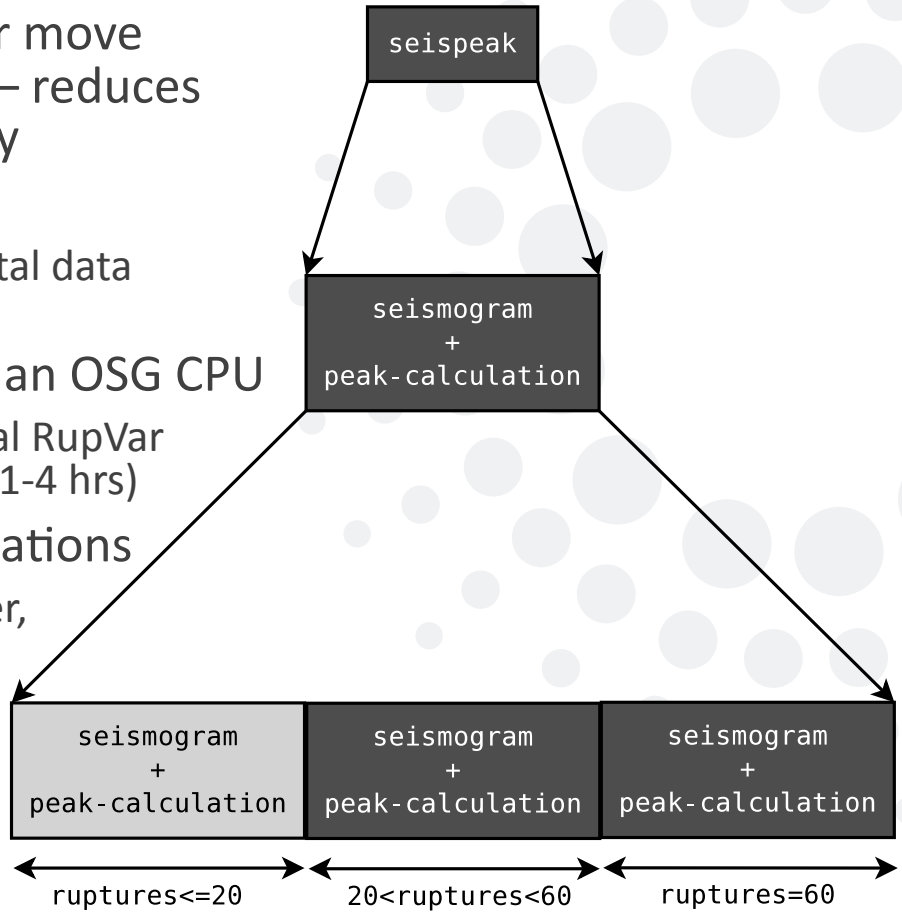
```
Foreach rup in rups {
  Sgt sub;
  sub = extract(sgt, rup);

  Variations vars = get_variations(site, rup);

  Seismogram seis[];
  PeakValue peak[];

  foreach var,i in vars {
    seis[i] = seismogram(sub, var);
      peak[i] = peak_calc(seis[i], var);
  } // end foreach over vars
} // end foreach over rups
```

Problems with straightforward code

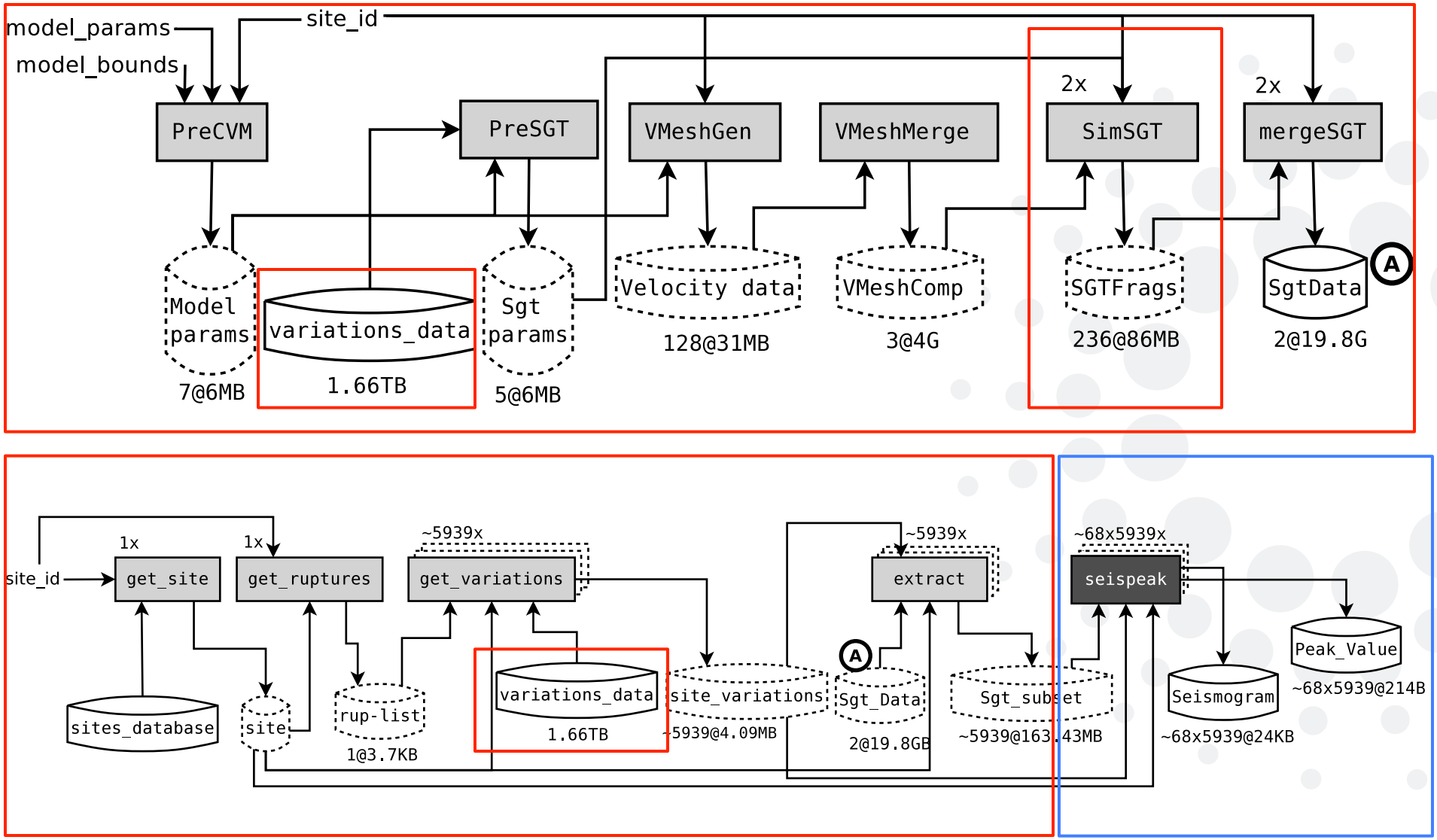
- Can't keep data on OSG sites
  - Need full 2 TB on each site to use that site, because Swift doesn't currently have a good data-affinity-aware scheduling mechanism, and because OSG resources are opportunistic
- So, need to move data to where resources are available
  - Basically – move 40 GB for each rupture, extract 160 MB for that rupture, move 3 MB for each rupture variation, make calculations associated with the variations
- But, data movement times overwhelmed compute times (when trying to do this at 2000x parallel)
- So, be more clever in scheduling
  - With the idea that this could be automated in Swift later

Current solution

- Combine everything into a single Swift script
- Do SGT on TeraGrid
- Split PP across TG and OSG, based on # of rupture variations per rupture
  - Do all SGT extractions on TG (rather move 160 MB subSGTs than 40 GB SGTs) – reduces overall data movement dramatically
  - <20: do work on TG
    - No data transfer needed, reduced total data transfer by ~50%
  - >20, <60: send a bundle of work to an OSG CPU
    - Limited subSGT data transfer, minimal RupVar transfers, reasonable work/transfer (1-4 hrs)
  - >60: divide into bundles of ~60 variations
    - Still have limited subSGT data transfer, minimal RupVar transfers, reasonable work/transfer (~4 hrs)
    - Some redundant subSGT data transfer (~20%)



SGT Data Flow



ExTENCI – d.katz@ieee.org

ExTENCI Status

- Technologies – Application/User – Leads (Sites)
  - Distributed File System (Lustre-WAN) – CMS/ATLAS HEP – Ralph Roskies & Paul Avery (PSC, U Florida)
  - Virtual Machines – STAR & CMS – Carol Song, Sebastian Goasguen, Todd Tannenbaum (Purdue, Clemson, Wisconsin)
  - Workflow & Client Tools – SCEC & Protein Folding – Daniel S. Katz, Mike Wilde (U Chicago)
  - Job Submission Paradigms – Cactus Application – Shantenu Jha (Rutgers)
- All making good progress
- SCEC work – even though we have multiple infrastructures, it is easier (policy and tools) and better (network) to use the supercomputing one mostly