# Design and Implementation of 'Many Parallel Task' Hybrid Subsurface Model

Khushbu Agarwal, Jared M. Chase, Karen L. Schuchardt, Timothy D. Scheibe, Bruce J. Palmer, Todd O. Elsethagen

Khushbu.Agarwal@pnnl.gov

Pacific Northwest
NATIONAL LABORATORY

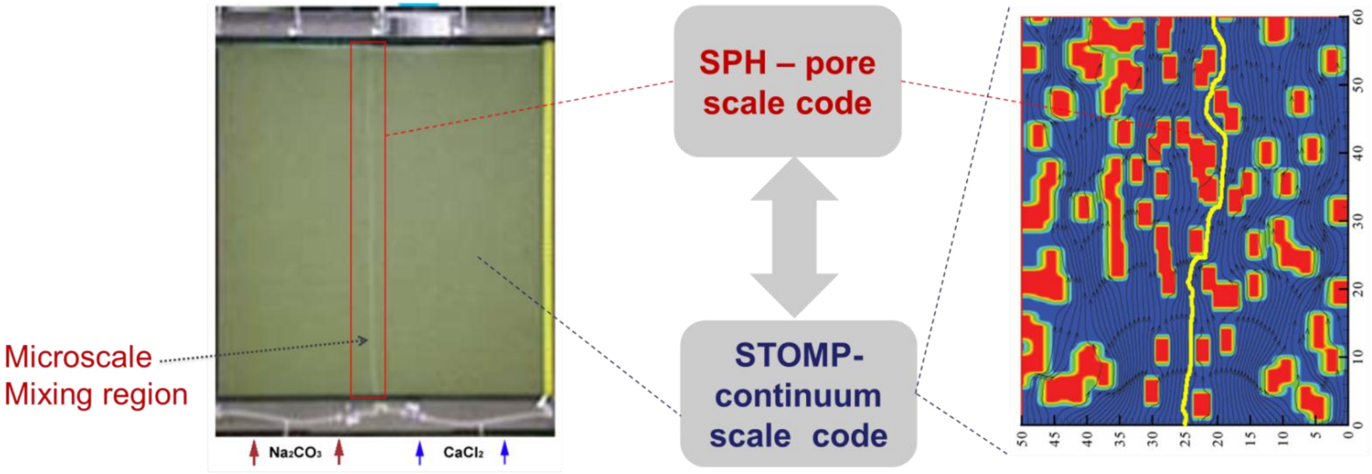
Proudly Operated by Battelle Since 1965

# Motivation: Why couple subsurface models?

- Continuum scale methods
  - Represent subsurface phenomenon in average sense, e.g. flow and transport
- Pore Scale methods
  - Provide detail on particle level interactions.
  - Adequate for modeling fractures, precipitation etc.
  - Cannot scale to realistic domains which contain billions of particles
- A coupled model provides balance b/w computation time and model accuracy.
  - Use macroscale methods to identify regions of high reactivity
  - Use microscale simulations to study “reactive” grid cells.

# Target Problem

- Mixing controlled kinetic precipitation reaction:
    - Two solutes (A,B) are injected in a porous medium domain.
    - Study formation of precipitate C
- Homogeneous medium: Solutes are expected to mix along centerline.
    - Fixed or variable number of equal size pore scale simulations
- For a non-homogeneous case, mixing zone is unknown.
    - Variable number of unequal size pore-scale runs



**Case 1 (Uniform):**
**Fixed/variable number of**
**Equal size pore runs**

**Case 2 (Adaptive):**
**variable number of**
**unequal size pore runs**

# Hybrid Subsurface Model

- The coupled model consists of
  - A ‘macroscale’ simulator, STOMP (Subsurface transport over multiple phases)
  - A pore-scale simulator, SPH (Smooth Particle Hydro-dynamics) (SPH):
  - Modules for mathematical exchange of information across scales (serial python codes)
    - Pore Generator(PG) : processes STOMP results, identifies the location, number and initial conditions for each of SPH simulations.
    - Grid Parameter Generator (GPG): process all SPH results and prepares STOMP input for next iteration.
- Visualization and provenance capabilities are integrated for enabling run time analysis.

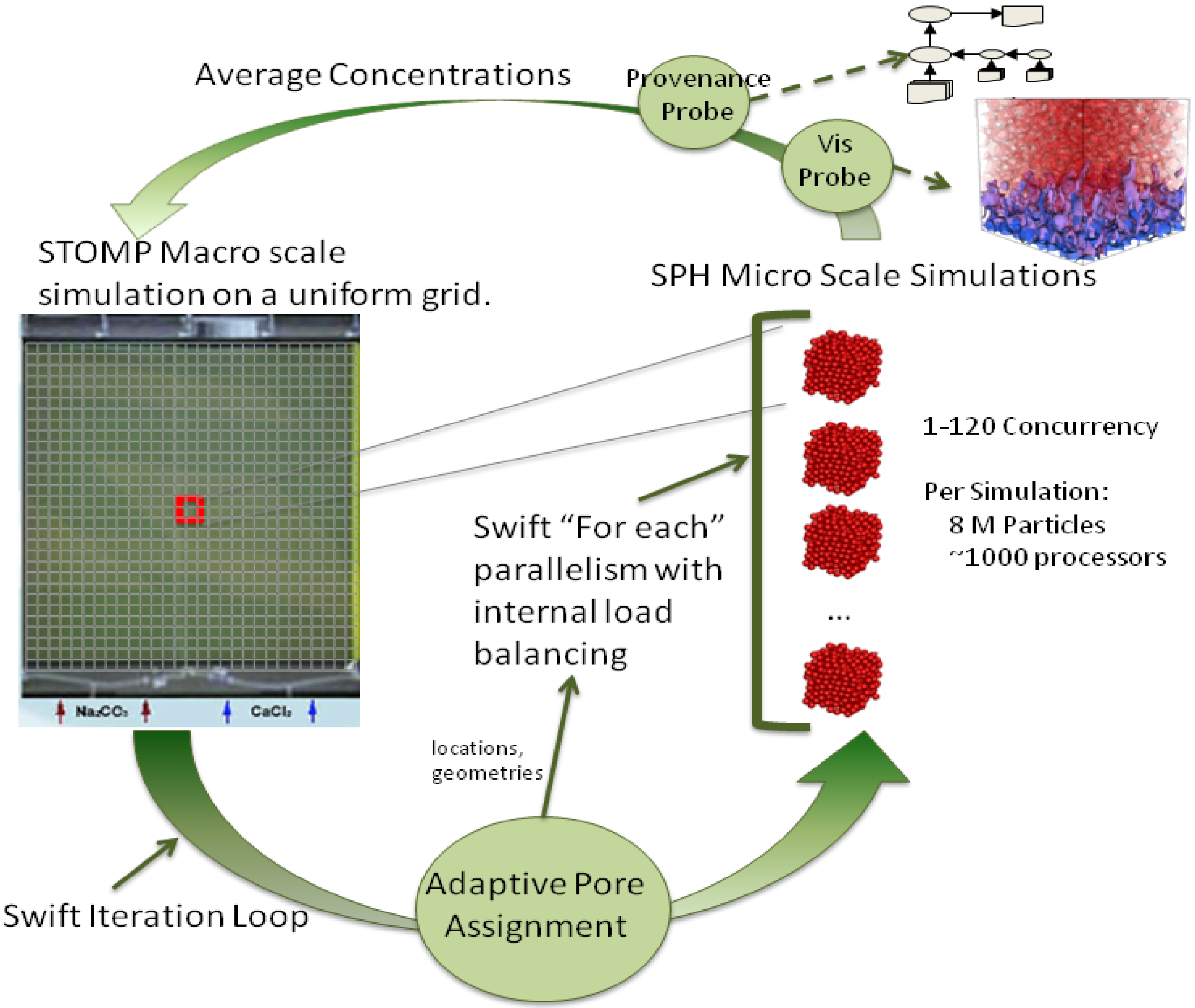
# Using Swift

- We develop the coupled subsurface model using Swift workflow system
- Swift provides scripting language for distributed parallel computing
  - An inherently parallel execution model
  - Simple interface definitions, to apply programs to file collections
  - Internal Error handling
  - Simple C-like syntax
  - Portable

# Hybrid Subsurface Model Workflow

- Single STOMP run per iteration.
  - Serial version of STOMP is used for initial problem domain.
- Multiple iterations of the coupled model executed in the workflow, using Swift’s ‘iterate’ construct
  - User provides #iterations at command line
- Each iteration, multiple independent pore-scale (SPH) simulations  executed simultaneously.
  - Each pore-scale run is itself parallel.
  - Many Parallel Task Computing (MPTC) Paradigm
  - The number and size of pore- scale domains vary across iterations

# Hybrid Model Workflow



Average Concentrations

Provenance Probe

Vis Probe

STOMP Macro scale simulation on a uniform grid.

SPH Micro Scale Simulations

1-120 Concurrency

Per Simulation:
8 M Particles
~1000 processors

Swift "For each" parallelism with internal load balancing

...

locations, geometries

Swift Iteration Loop

Adaptive Pore Assignment

# Implementing Swift Workflow

- An ‘app’ (interface) is defined for each of the system executables.
  - Identifies input and output parameters
  - Each ‘app’ is registered with Swift’s transformation catalogue which maps these executable to their system paths.
- The Swift’s ‘sites.xml’ is configured to run jobs  on the localhost with a single processor.
- The job submission script invokes a single instance of Swift workflow on the login node.
  - Avoids complexity of launching Swift on compute nodes.

# Workflow Charactersctics : Adaptive and Dynamic Scheduling

- Single job submission to the job queue, with internal task management for SPH simulations.
- Dynamic Process Allocation
  - Each iteration, processes allocated equally between SPH tasks
  - Load balancing issues if task are unequal in size
  - Weighed scheduling to be implemented
- Tasks are queued and executed in multiple batches if insufficient resources.
  - Worst case each task executes as a serial job, in an iterative manner.

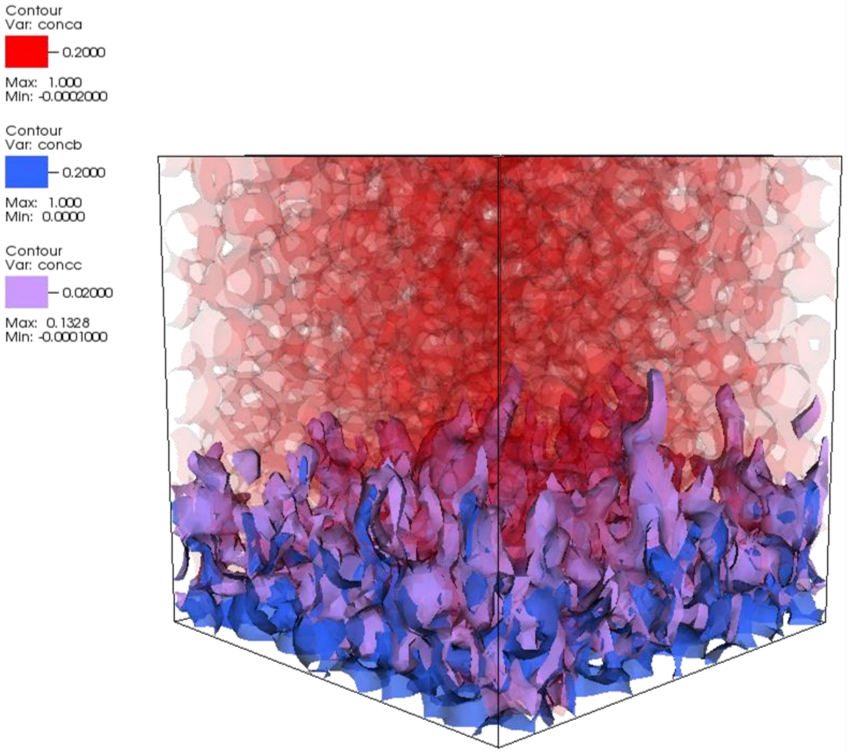
# Portability and Data Management

- Portable
  - The underlying resource manager is identified at run time through configuration file to create the launch command
  - PBS and SLURM support.

- Data Management
  - Each iteration is executed in a separate directory using Swift’s mapping capabilities
  - Any given iteration, all SPH run are identified based on their unique id in the STOMP mesh.

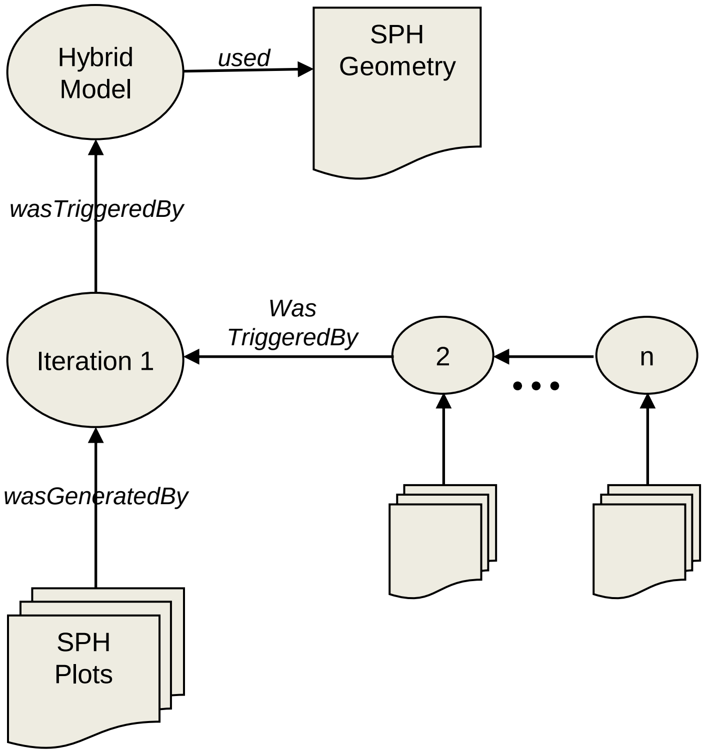
# Workflow : Visualization

- Automated Image Extraction
  - Use VisIt's python API
  - Performed for each workflow iteration.
- Simulation Plots
  - SPH (*Pore Scale*)
    - A (*red*) and B (*blue*) are the reactants.
    - C (*purple*) is the precipitate.
  - STOMP (*Continuous Scale*)



# Provenance capture

- Specialized Swift applications dedicated to provenance capture.
- Collected at every STOMP iteration
- Supports real-time and post analysis efforts.
- Standards based provenance model – Open Provenance Model (OPM).



# Hybrid Subsurface Model : Other Challenges

- Workflow
  - The first iteration of the workflow is different from other iterations.
    - No restart files
    - Create separate interfaces for first iterations or provide dummy files.
  - File dependencies need to be explicit in interface definitions, may get lost otherwise.
  - filesys_mapper fails
    - using ‘tar’ and untar all SPH runs every iteration may be slow
  - Swift allows single write to any parameter.
  - IO may impact performance

# Other Challenges (contd.)

- Application
  - Defining particle properties as they move across SPH domains
  - Identifying appropriate geometry, concentration of particles, mesh properties, unit conversions/
  - Implementing Pore and Grid Parameter Generator.
  - Changes to STOMP and SPH

# Experimental Configuration

- Initial run on NERSC's Franklin and Hopper system.
- Serial components take order of seconds
- An 3.4M particle SPH simulation(1K processes) completes in 2 min 36 seconds.
  - 8M particles for target problem domain,
  - using 256 processes, the SPH simulation is expected to take 15-20 minutes.
  - Each iteration executes 1-120 of such SPH simulations
- Complete workflow may run b/w 100-1000 of iteration.
- Computing costs will be significantly high for 3D non uniform problems. I/O costs may be significant.
- Workflow tested with stubs

# Summary

- Developed Hybrid Subsurface Model using Swift Workflow system.
- Manages multiple parallel tasks (MPTC) per iteration, as size and number of tasks changes across each iteration.
- Portable workflow adapts to the underlying resource manager.
- Data Management using Swift’s mapping capabilities.
- Integrated visualization and provenance for run time data analysis and validation.

# Future Work

- Weighed Task Scheduling
- Runtime analysis for I/O costs and solutions to minimize them.
- Integration with user interface
- Task Manager improvements:
    - Task management when #Tasks << # Processes
- Addressing challenges associated with modeling non-homogeneous domain.

# Acknowledgement

- Thanks to the Swift team at ANL for their support during the course of this effort.
- This work was done under Office of Science, SCIDAC project.

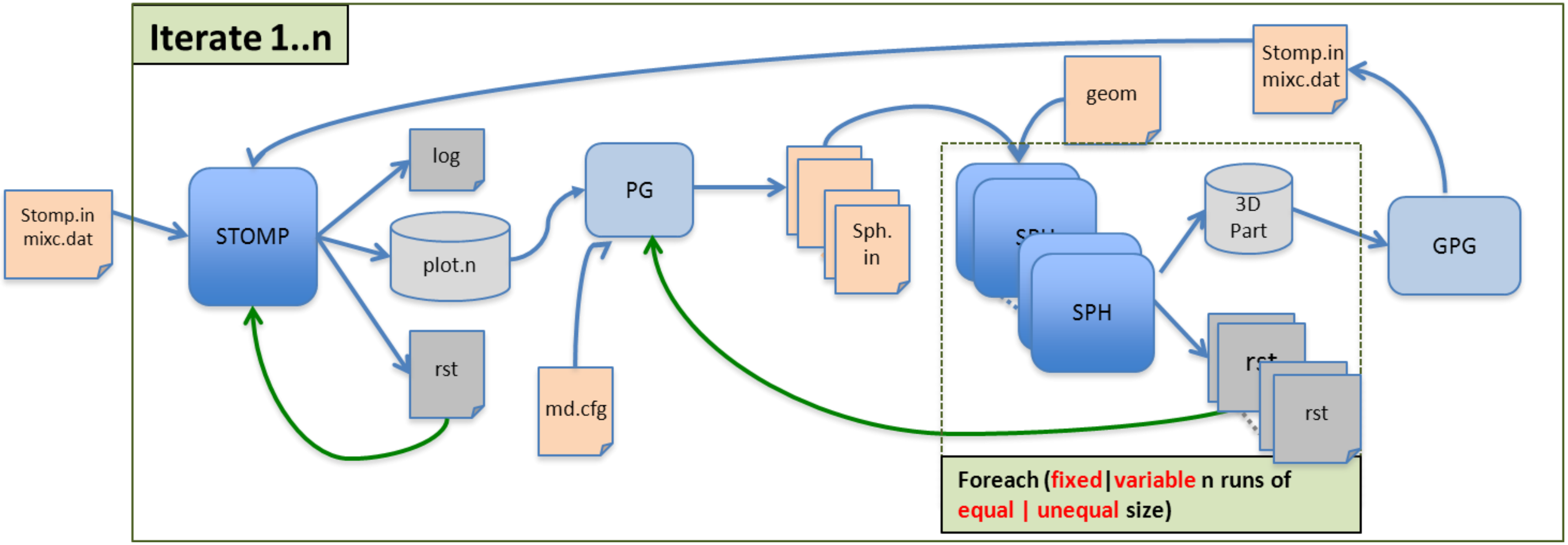
# Questions

# More

# Initial results : IO benchmarking for SPH

| Franklin - 256 Processors (Aggregate Bandwidth for IO (MB/s) | | | |
|---|---|---|---|
| | POSIX | POSIX + LUSTRE | MPI + LUSTRE |
| OSTS=10 | 539.5 | 504.8 | 491.2 |
| OSTS=20 | 524.2 | 542.6 | 511.8 |
| OSTS=40 | 510.4 | 404.4 | 219.9 |

# Hybrid Subsurface Model Workflow



- Swift collective data management
- Tell Swift to not do staging