

Scalable Parallel Analysis of Power Grid Models Using Swift

*Ketan Maheshwari**, Victor M Zavala, Justin Wozniak,
Mark Hereld, Michael Wilde

MCS Division, Argonne National Laboratory

*correspondence: ketan@mcs.anl.gov

Overview

- Motivation
- Swift Parallel scripting
- Power Grid Applications
 - Large-Scale Scenario Analysis
 - Inference Analysis for Stochastic Programs
- Summary



Motivation

- Grand scheme of things: Think parallel
 - Conceptualize: write sequential program doing one thing at a time
 - Implementation: Using conventional languages and libraries
 - Easily and readily scale to large machines: Parallelize
- Need for Programming Environments that Enable Easy Parallel Execution of Complex Applications in Heterogeneous Architectures
- Write, test in small -- Deploy in large
 - Application level programming
 - suites multiple infrastructures
 - Rapid pickup from laptop to supercomputers (and everything in between)



Tame Heterogeneous Computational Infrastructure

Remote Access Interface (ssh, tcp)

Middleware and Schedulers

Condor

SLURM

PBS

Eucalyptus

Compilers and libraries

GNU

intel suite

PGI

MPI

Domain Tools

Matlab

AMPL

IPOPT

CBC-solver



Parallel Systems



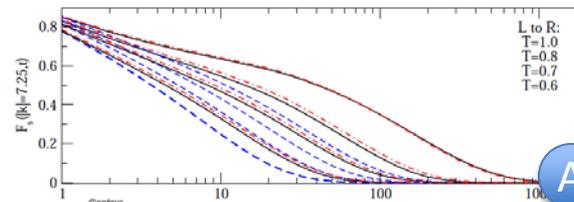
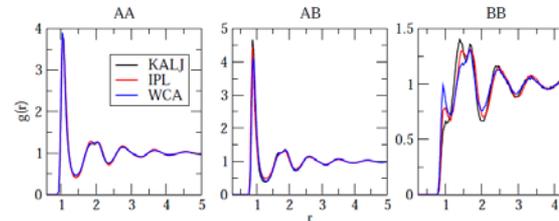
Swift Parallel Scripting Language

- **Swift** is a parallel scripting language
 - *Composes applications linked by files and data*
 - *Captures protocols and processes*
- Easy to write: a simple, high-level language
 - *Small Swift scripts can do large-scale work*
- Easy to run: on clusters, clouds, supercomputers and grids
 - *Sends work to national, campus, and Amazon resources*
- Automates solutions to four hard problems
 - *Implicit parallelism*
 - *Transparent execution location*
 - *Automated failure recovery*
 - *Provenance tracking*

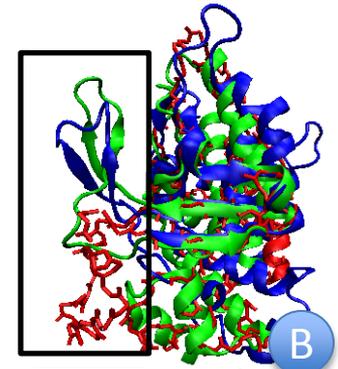


Numerous many-task applications

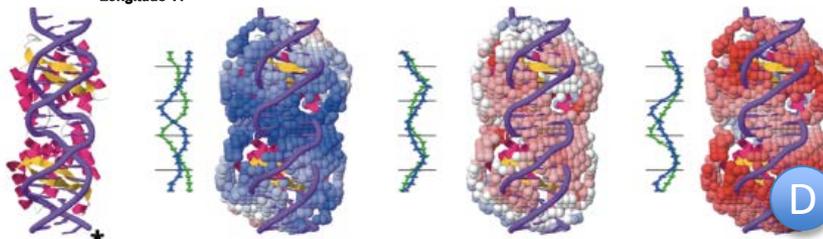
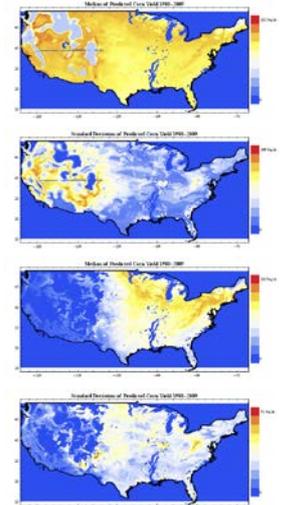
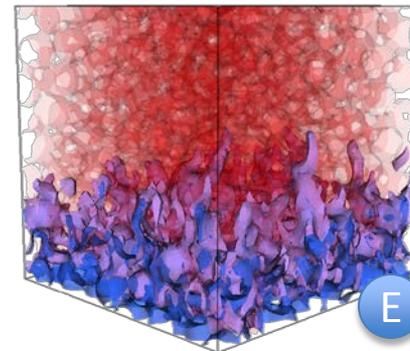
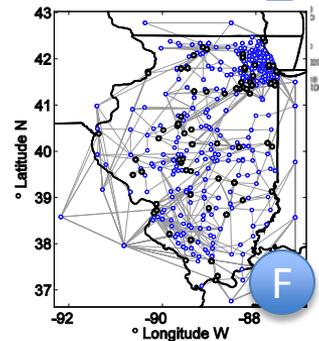
- A Simulation of super-cooled glass materials
- B Protein folding using homology-free approaches
- C Climate model analysis and decision making in energy policy
- D Simulation of RNA-protein interaction
- E Multiscale subsurface flow modeling
- F Modeling of power grid applications
- > All have published science results obtained using Swift



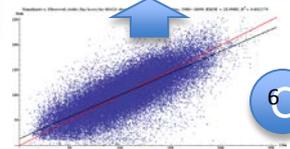
T0623, 25 res., 8.2Å to 6.3Å (excluding tail)



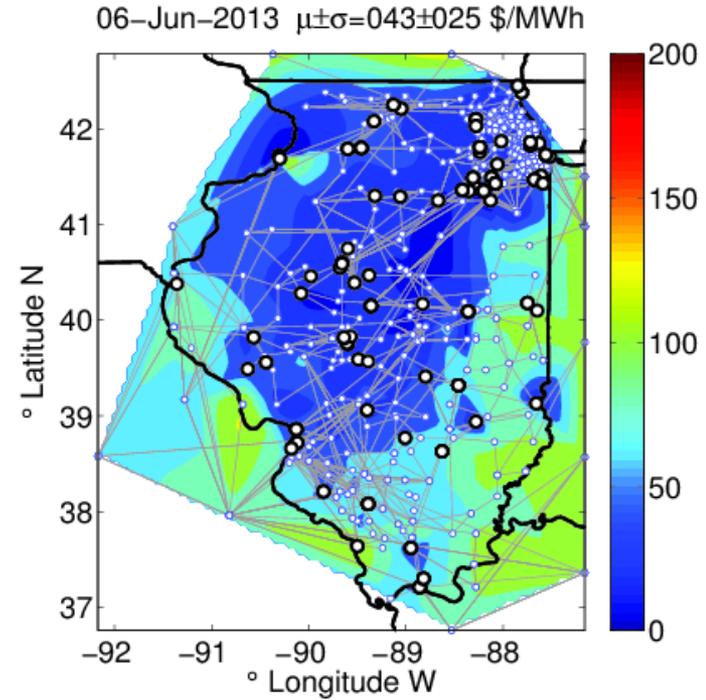
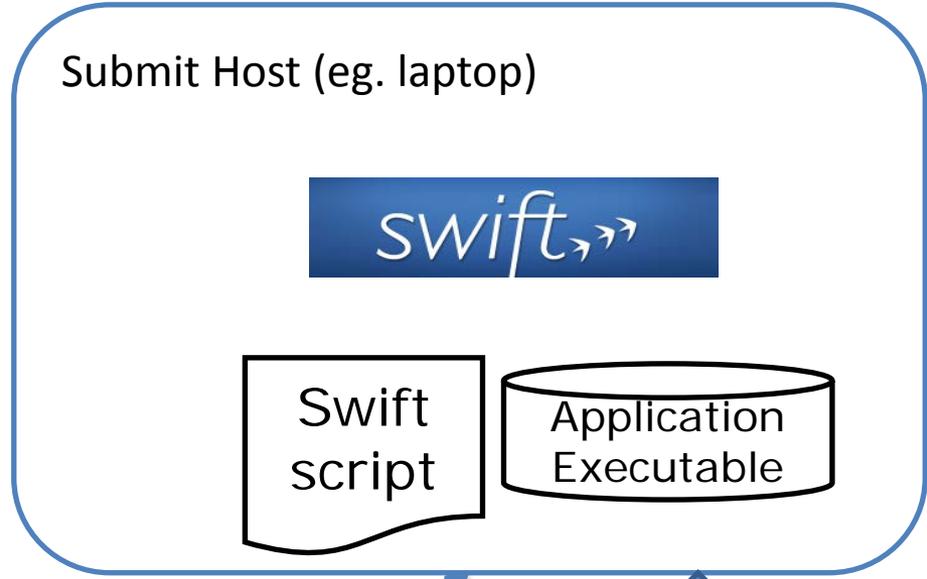
Initial
Predicted
Native



Protein loop modeling. Courtesy A. Adhikari



Power grid applications on HPC Systems via Swift



Uchicago UC3 and Midway Clusters

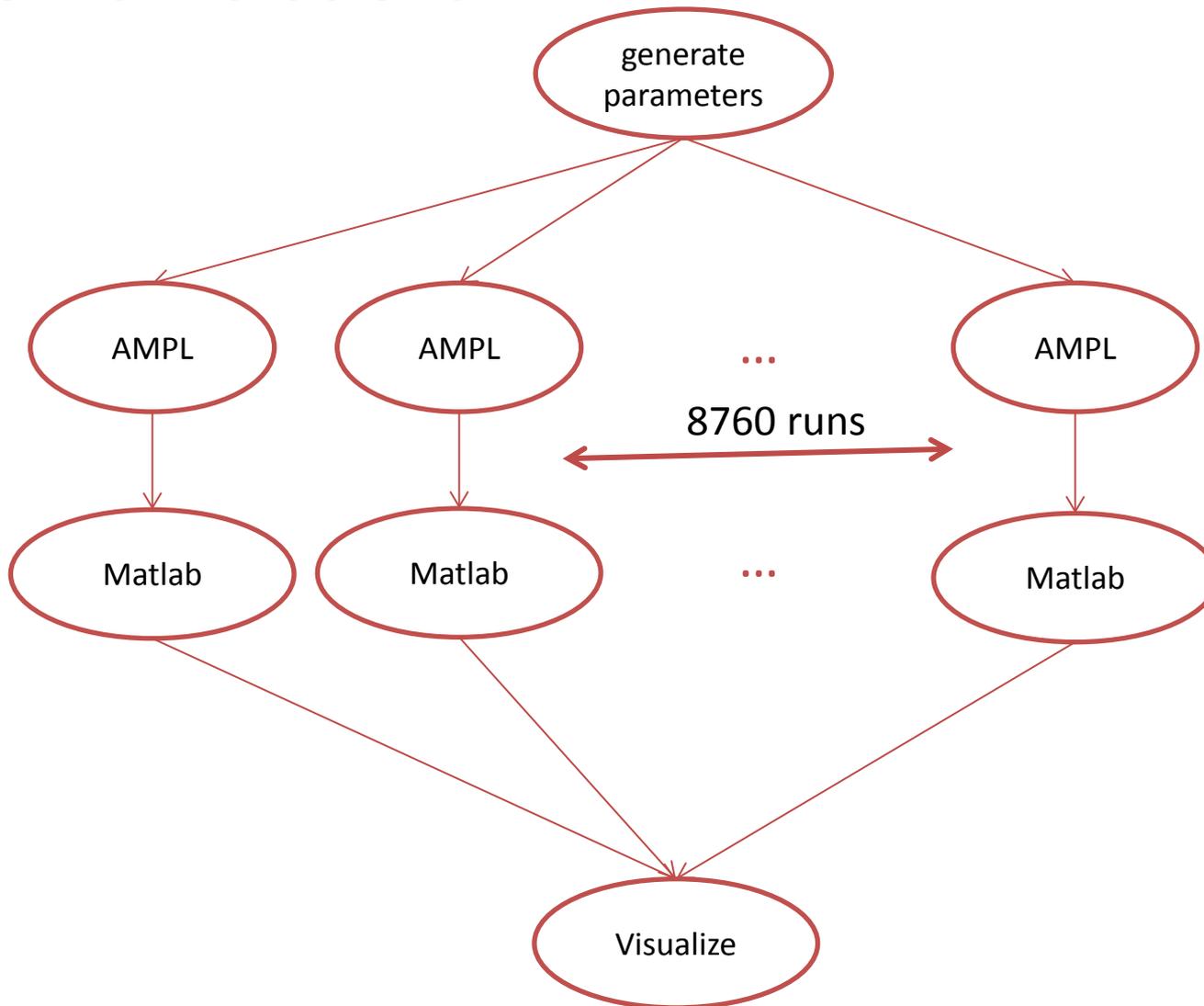


Argonne's HPC resources



Cloud resources

Application#1: Large-Scale Optimal Power Flow Simulations



Application#1: Sequential (shell script) and Parallel (Swift) code

```
for i in `seq 1 8760`  
do  
  echo $i>idx_data.dat  
  ampl optdcflow.run  
  mv lmp_res.dat /scratch/lmp_res.$i.dat  
  matlab -nodesktop -nosplash -r plot_lmp.m  
done
```

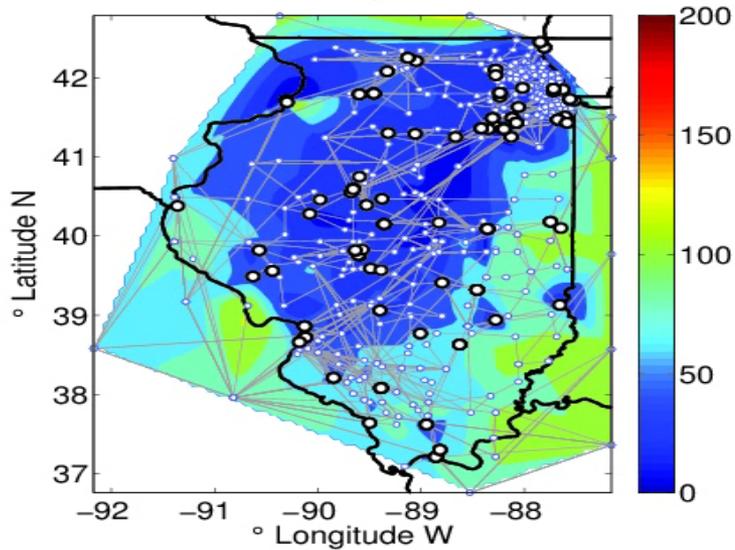
sequential shell script

```
import appdef, file;  
  
foreach i in [1:8760]{  
  file t<sprintf("lmp_res.%i.dat", i)> =  
    runampl (model, bundle, i);  
  amplout[i] = t;  
  runmatlab(t);  
}
```

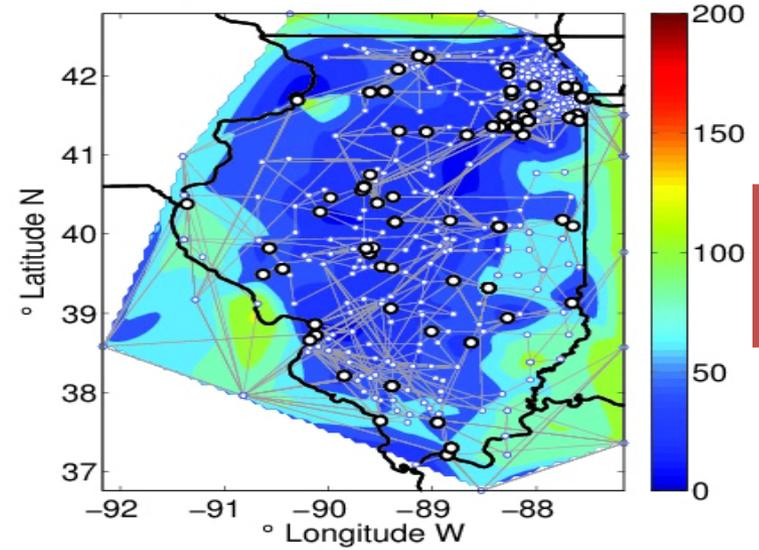
parallel swift script

Results: Visualization

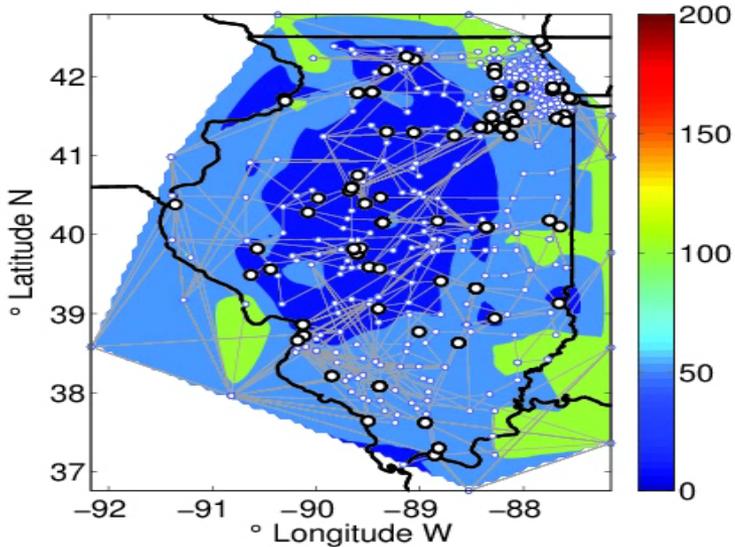
01-Jan-2000 01:00:00 $\mu \pm \sigma = 043 \pm 022$ \$/MWh



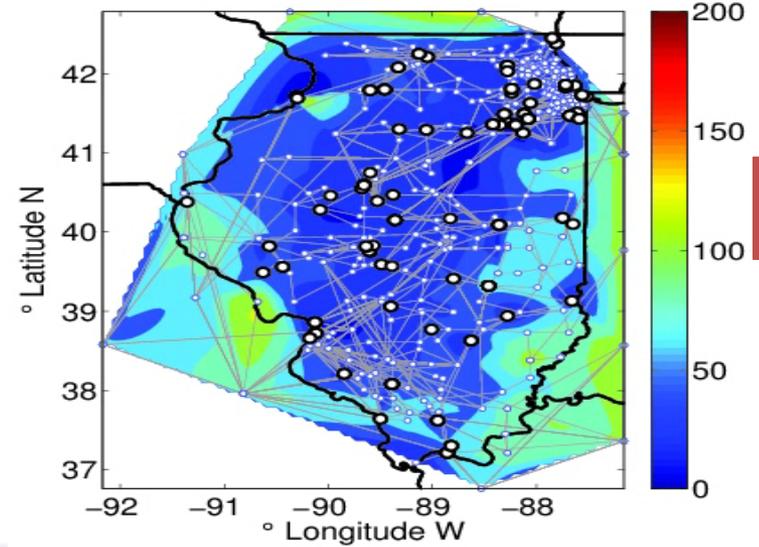
30-Apr-2000 $\mu \pm \sigma = 039 \pm 020$ \$/MWh



29-Jun-2000 $\mu \pm \sigma = 063 \pm 035$ \$/MWh

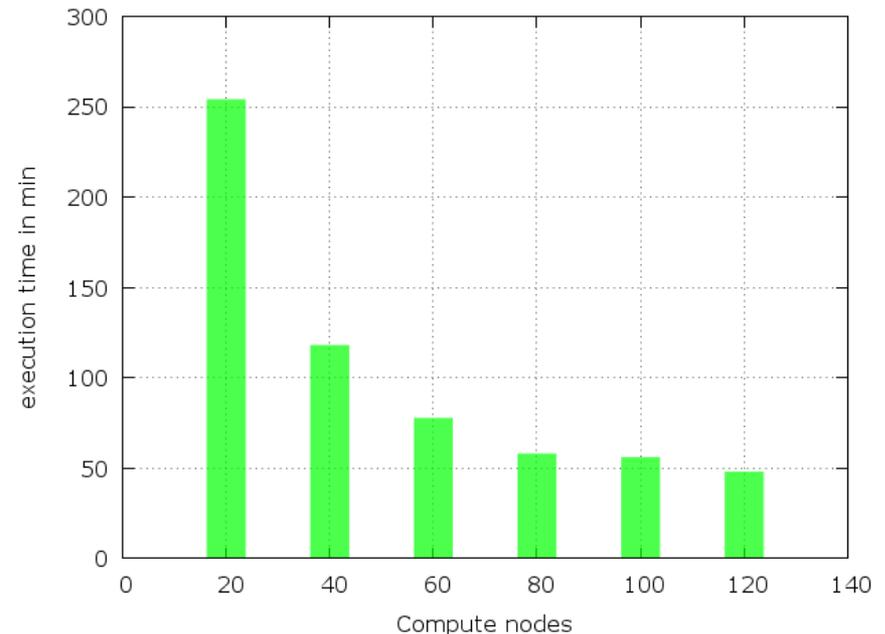


27-Sep-2000 $\mu \pm \sigma = 038 \pm 019$ \$/MWh

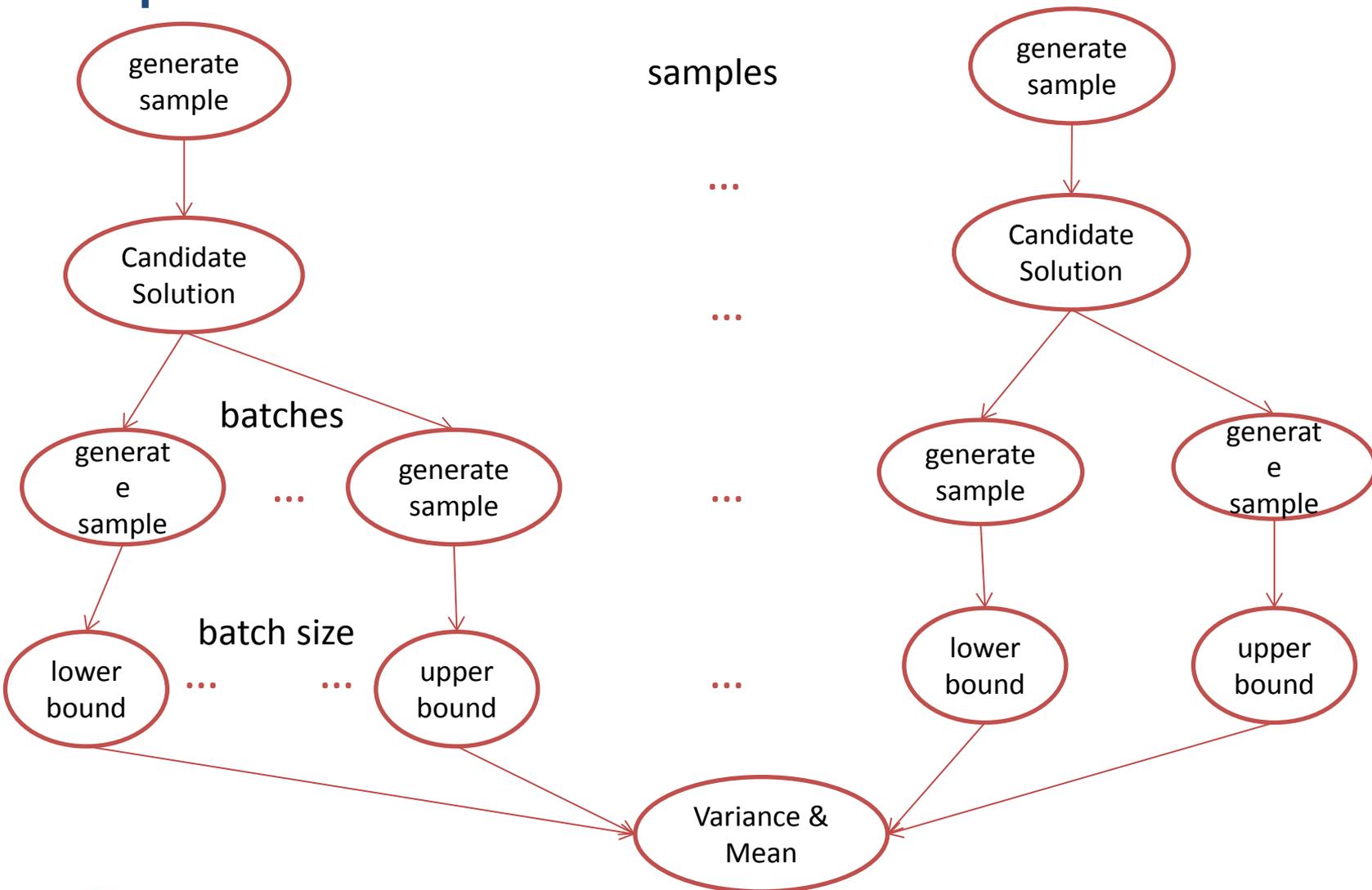


Parallel Scalability for Optimal Power Flow

- Enable rapid discovery and analysis through large-scale simulations (Do in Minutes Instead of days)
- Run optimal power flow over Illinois system with resolution of 1 hour (8760 total runs)
- An impractical execution time of 50 hours (2 days) on a single machine
- In 120 Nodes, required ~50 min



Application#2: Inference for Stochastic Optimization



Swift Code for Inference for Stochastic Optimization

```
import "mappings";
import "apps";
type file;
int nS[] = [10, 100, 1000, 10000, 100000]; //experiment with scenarios
int nB[] = [10, 20, 30]; // experiment with batches

foreach S, idxs in nS{
    o = gensample (wind_data);
    obj_out[idxs] = ampl_app (...params...);

    foreach B, idxb in nB{
        foreach k in [0:B]{

            o = gensample (S, wind_data);
            obj_out_l[idxs][idxb][k] = ampl_app_L(...params...);

            o = gensample (S, wind_data);
            obj_out_u[idxs][idxb][k] = ampl_app_U(...params...);

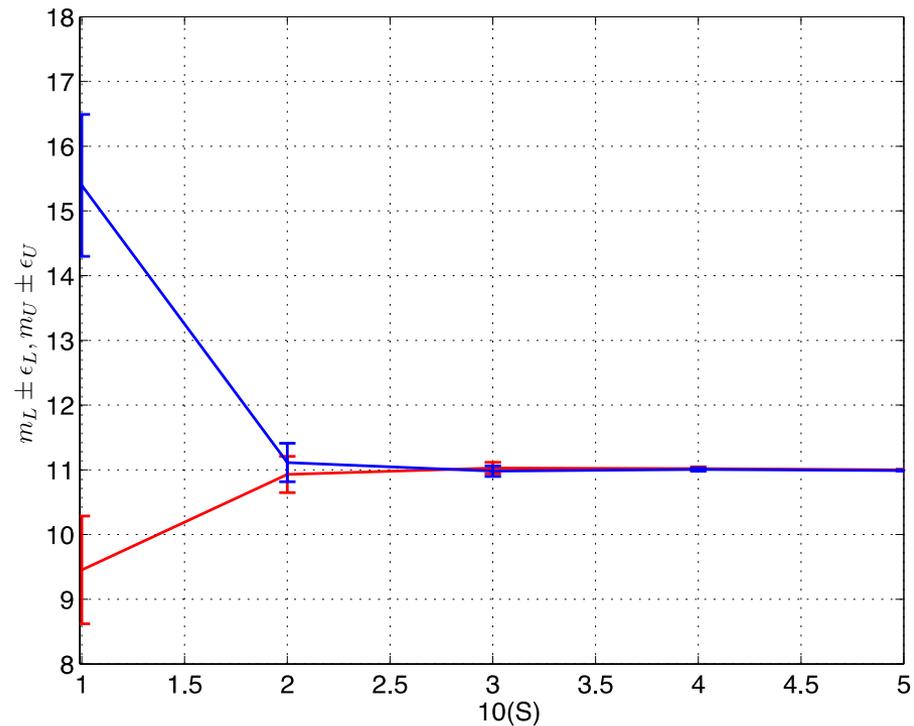
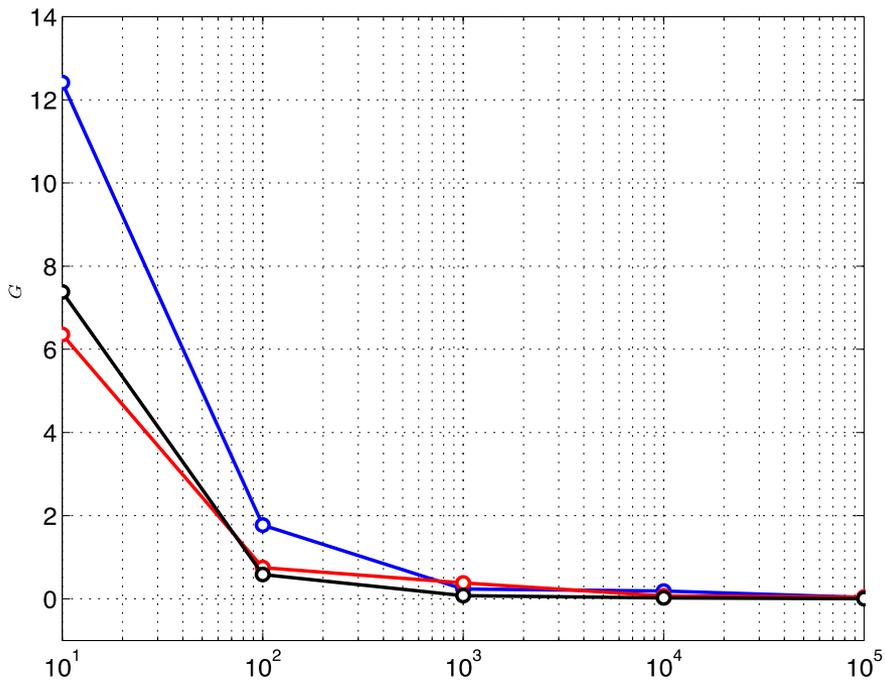
        }
    }
}
```

```
}}}
```



Results: Upper and lower bounds

- Samples=(10,100,1000,10000,100000), Batches=(10 20 30)
- Resulting in ~100K parallel tasks



Summary

- Parallel execution of ordinary programs is key to efficient usage of large systems for rapid execution of complex computational tasks
- Usability and ease of adaptation is key to overcoming many hurdles in efficiently scaling power grid apps to large parameter sizes
- Swift takes up ordinary programs and runs them in parallel on different kinds of computational infrastructure
- Swift bridges the gap between applications and systems by providing suitable interfaces
- Swift has been shown to work successfully for many scientific and engineering application areas including power grid design and simulation problems



Thank you! Questions and Comments are welcome.

ketan@mcs.anl.gov

Acknowledgments

- Swift team: Mihael Hategan, Ben Clifford, David Kelly, Yadu Nand, Jon Monnette, Alan Espinosa, Ian Foster, Dan Katz, Zhao Zhang, Yong Zhao, and others
- The Power Grid project at Argonne is supported by the funding from the Department of Energy's Office of Electricity Delivery and Energy Reliability
- Swift is supported in part by NSF grants OCI-1148443 and PHY-636265, and the UChicago SCI Program
- Part of this work was performed at Cornell under ARPA-e grant for the GridCloud project
- Scientific application collaborators and usage described in this talk:
 - U. Chicago Open Protein Simulator Group (Karl Freed, Tobin Sosnick, Glen Hocky, Joe Debartolo, Aashish Adhikari)
 - U.Chicago Radiology and Human Neuroscience Lab, (Dr. S. Small)
 - SEE/CIM-EARTH: Joshua Elliott, Meredith Franklin, Todd Muson
 - ParVis and FOAM: Rob Jacob, Sheri Mickelson (Argonne); John Dennis, Matthew Woitaszek (NCAR)
 - PTMap: Yingming Zhao, Yue Chen

