HPC Modeling of the Power Grid

PNNL: Bruce Palmer, Henry Huang

ANL: Shrirang Abhyankar

LLNL: Liang Min, Slaven Peles, Steven Smith, and Carol Woodward

Lawrence Livermore National Laboratory

Argonne NATIONAL LABORATORY

Pacific Northwest NATIONAL LABORATORY

Proudly Operated by Battelle *Since 1965*

PNNL-SA-112744

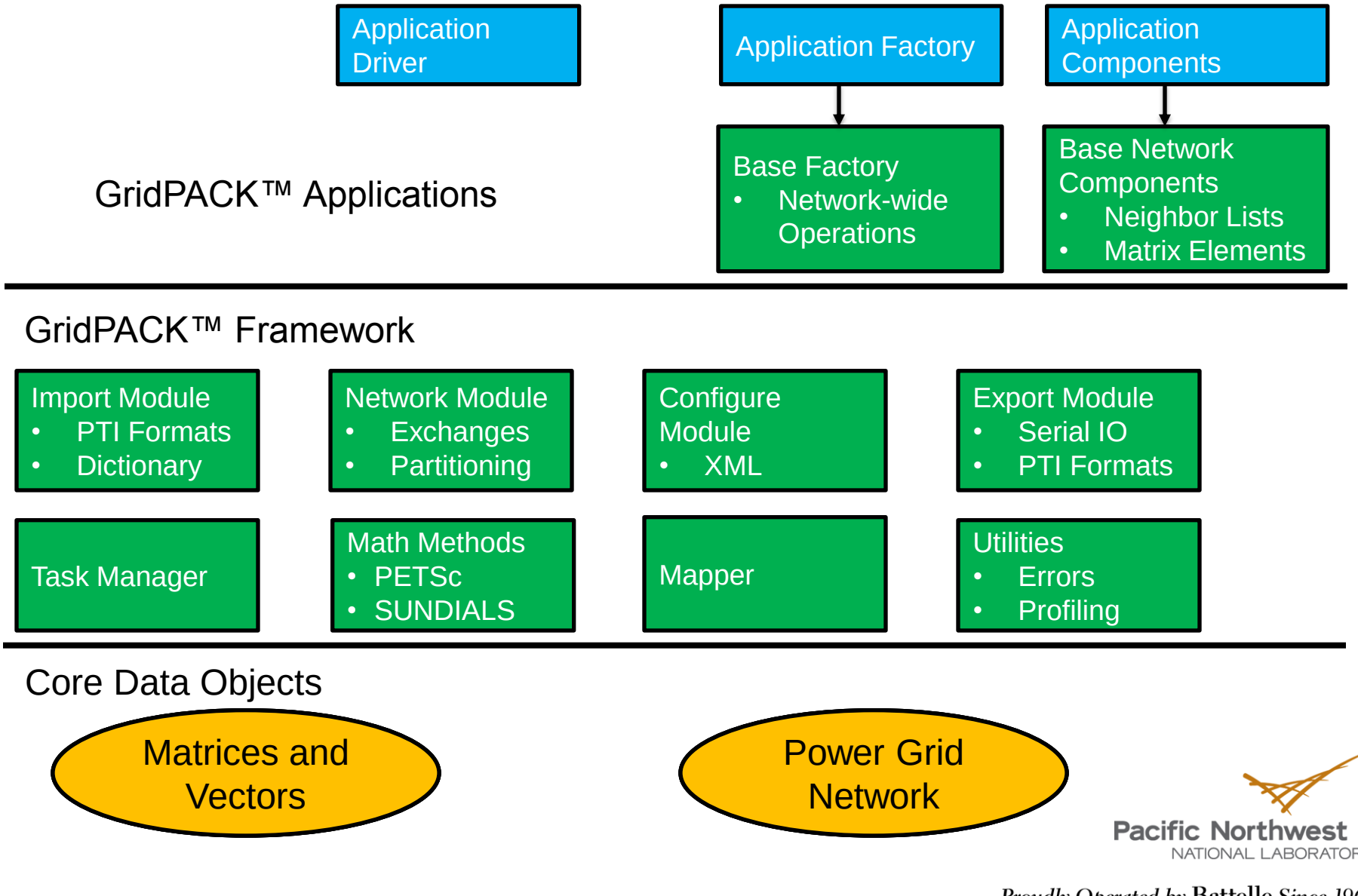
Why HPC?

- Single processor performance (clock speed) has been flat for a decade
- Serial codes are limited by memory and processor speed, and this limits the size and complexity of existing models
- Modeling large systems using small computers involves substantial aggregation and approximations
- Parallel computing can increase memory and computing power by orders of magnitude, thereby increasing the size and complexity of power grid models that can be simulated numerically
- Parallel computing can reduce time-to-solution for large and mid-sized problems

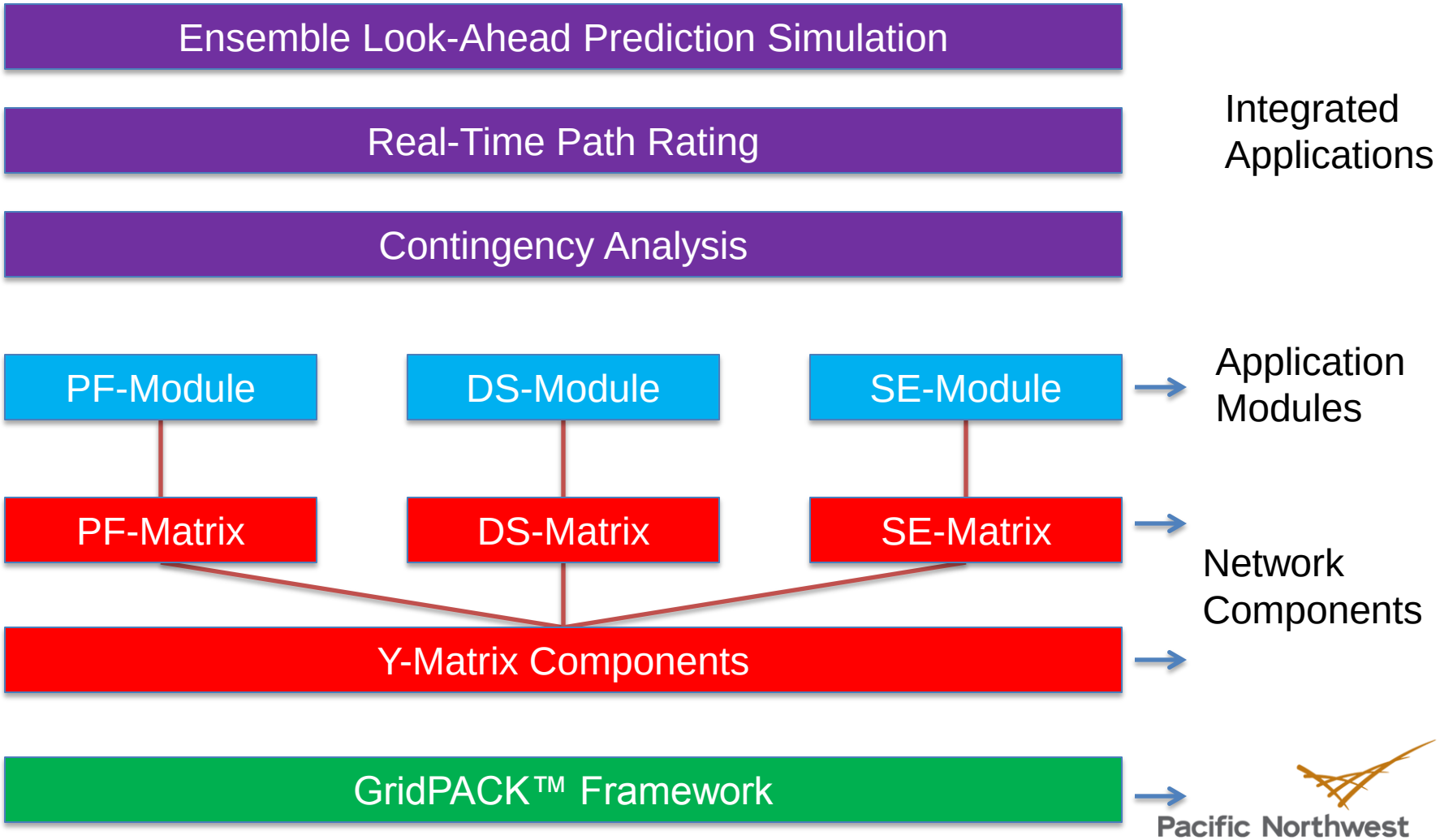
GridPACK™ Framework for Power Grid Applications

- Simplify development of HPC codes for simulating power grid
- Create high level abstractions for common programming motifs in power grid applications
- Encapsulate high performance math libraries and make these available for power grid simulations
- Promote reuse of power grid software components in multiple applications to reduce development and maintenance costs
- Incorporate as much communication and indexing calculations as possible into high level abstractions to reduce application development complexity
- Compartmentalize functionality to reduce maintenance and development costs

GridPACK™ Software Stack



GridPACK™ Application Stack

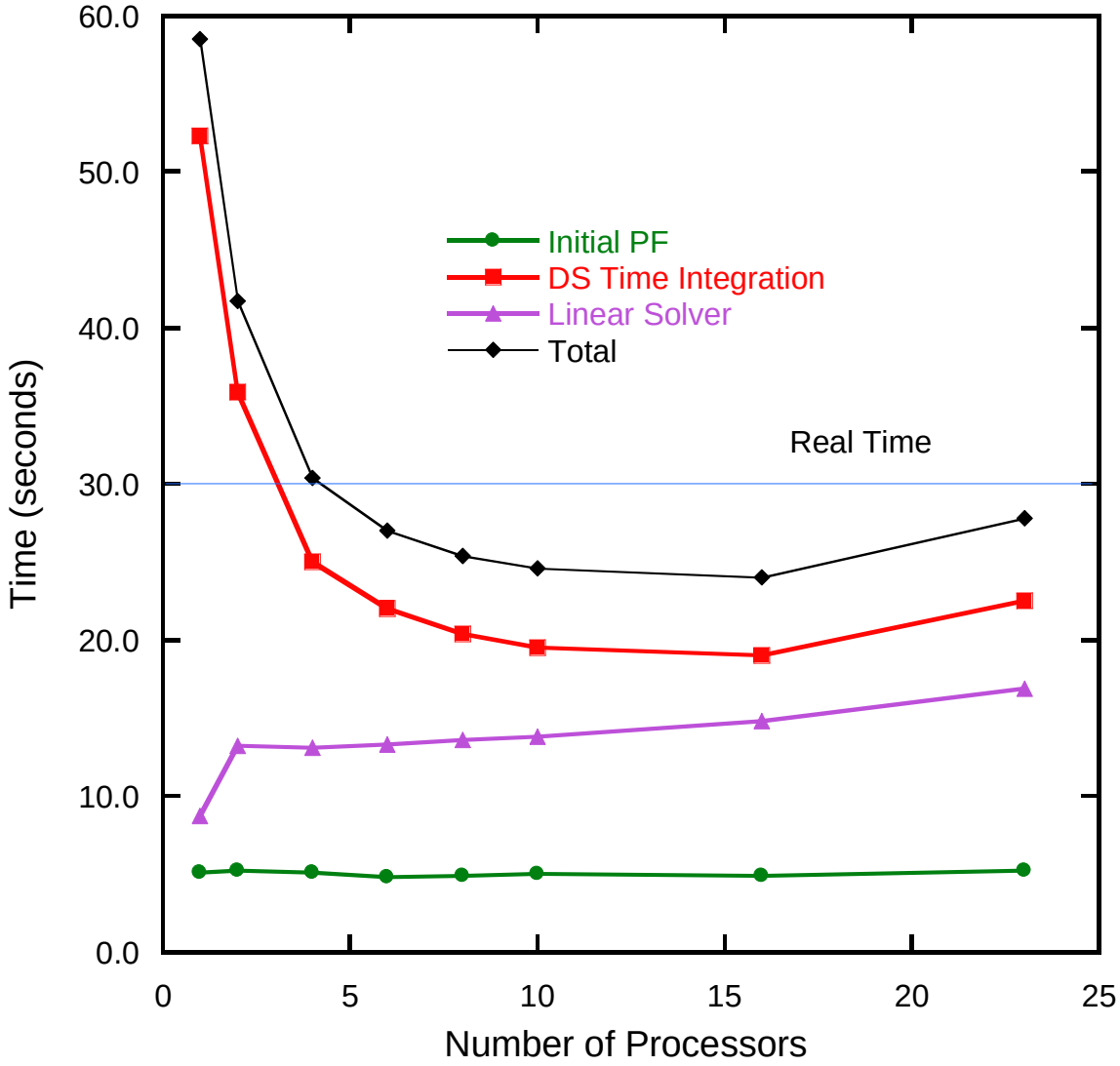


Power Flow Code

```
1  typdef BaseNetwork<PFBus,PFBranch> PFNetwork;
2  Communicator world;
3  shared_ptr<PFNetwork>
4      network(new PFNetwork(world));
5
6  PTI23_parser<PFNetwork> parser(network);
7  parser.parse("network.raw");
8  network->partition();
9
10 PFFactory factory(network);
11 factory.load();
12 factory.setComponents();
13 factory.setExchange();
14
15 network->initBusUpdate();
16 factory.setYBus();
17
18 factory.setSBus();
19 factory.setMode(RHS);
20 BusVectorMap<PFNetwork> vMap(network);
21 shared_ptr<Vector> PQ = vMap.mapToVector();
22 factory.setMode(Jacobian);
23 FullMatrixMap<PFNetwork> jMap(network);
24 shared_ptr<Matrix> J = jMap.mapToMatrix();
```

```
25 shared_ptr<Vector> X(PQ->clone());
26
27 double tolerance = 1.0e-6;
28 int max_iteration = 100;
29 ComplexType tol = 2.0*tolerance;
30 LinearSolver solver(*J);
31
32 int iter = 0;
33
34 // Solve matrix equation J*X = PQ
35 solver.solve(*PQ, *X);
36 tol = X->normInfinity();
37
38 while (real(tol) > tolerance &&
39        iter < max_iteration) {
40   factory.setMode(RHS);
41   vMap.mapToBus(X);
42   network->updateBuses();
43   vMap.mapToVector(PQ);
44   factory.setMode(Jacobian);
45   jMap.mapToMatrix(J);
46   solver.solve(*PQ, *X);
47   tol = X->normInfinity();
48   iter++;
49 }
```

Dynamic Simulation Application



Adding ANL solver and optimizer software to GridPACK™

- PETSc support in GridPACK™
  - Vector and Matrix classes
  - Parallel linear Solvers
  - Parallel nonlinear Solvers
  - Parallel Differential-Algebraic Equation solvers
- Additional support for GridPACK™
  - Accelerating Power Flow application
  - Accelerating Dynamic Simulation application
  - Developing small-signal stability analysis application (interfacing with SLEPc)
  - Incorporating parallel linear optimization software

Argonne NATIONAL LABORATORY

Adding interfaces to the LLNL SUNDIALS package

- Adding the SUNDIALS integrator for DAE systems, IDA, to GridPACK™
  - Error-based time step and order control
  - Multistep integration with Newton nonlinear solver
  - Possible extension to IDAS for sensitivities
- New DAE integrator allows for longer steps when system dynamics are slow and smaller steps when dynamics are fast
- Helping to generalize the DAE integrator API

Lawrence Livermore National Laboratory

sundials

Websites

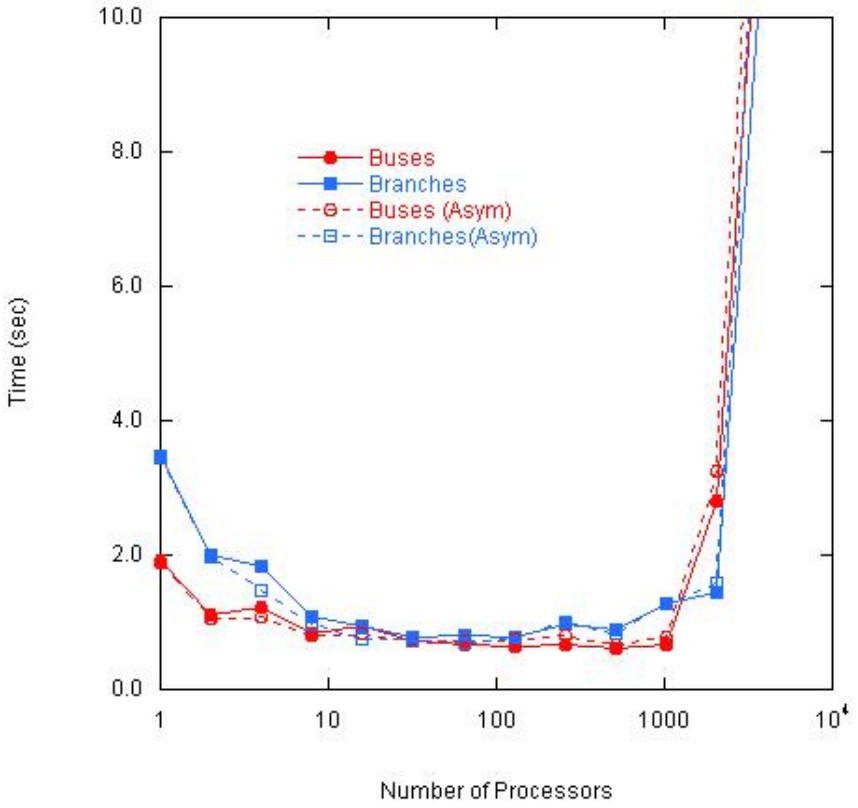
- GridPACK™ software located at https://www.gridpack.org
- SUNDIALS: https://computation.llnl.gov/casc/sundials

Issues for HPC

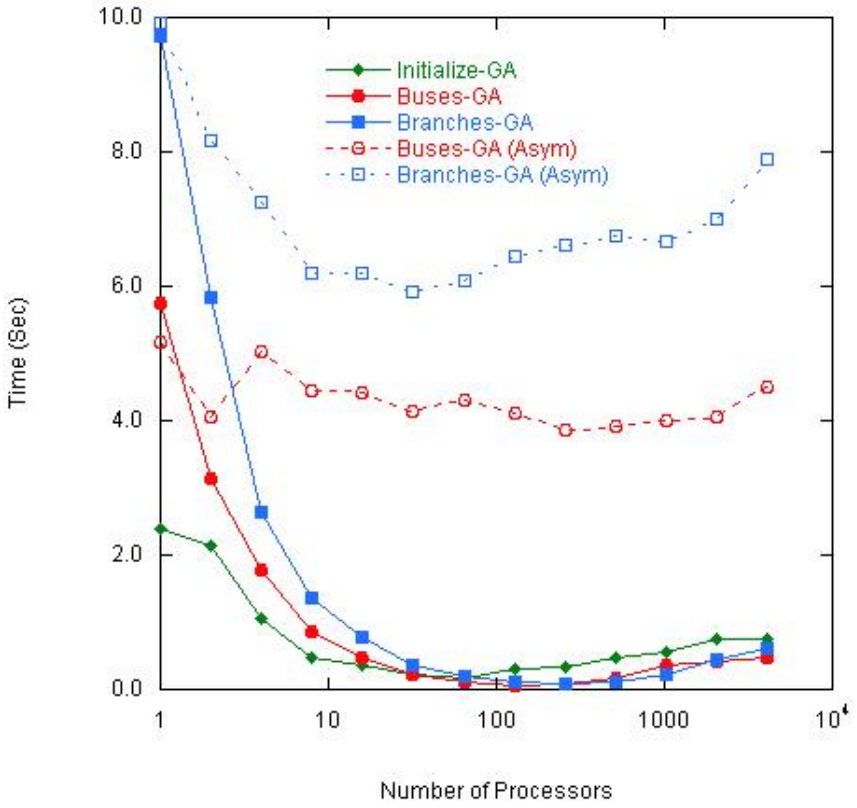
- Need open source data sets for large, realistic problems that can be shared publicly
- Need extensible data formats that can support HPC oriented I/O models (e.g. parallel ingestion from multiple files by multiple processors)
- Determining the best HPC architectures for power grid simulation (e.g., processor configuration, I/O support)

Hash Distribution Performance

Cyclic Communication



One-sided Communication



Pacific Northwest
NATIONAL LABORATORY

Proudly Operated by Battelle *Since 1965*