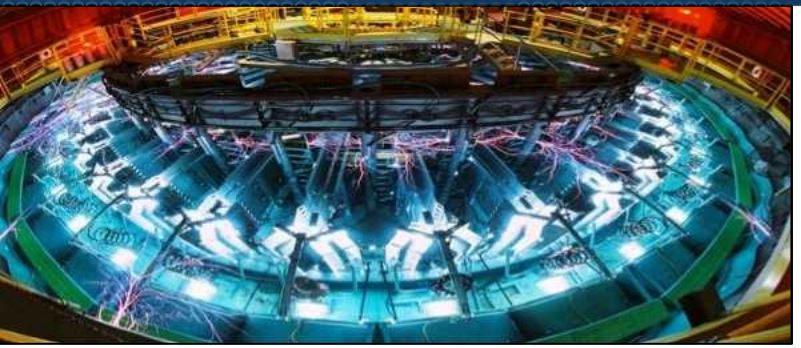
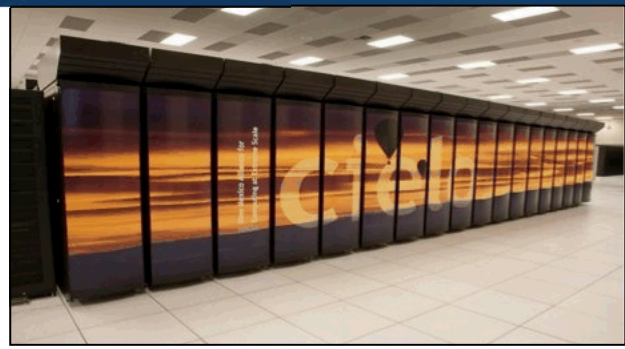
SAND2017-12469C



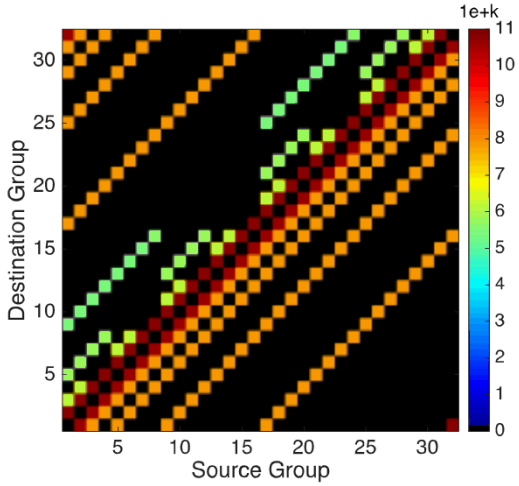


Overview of the Structural Simulation Toolkit (SST)

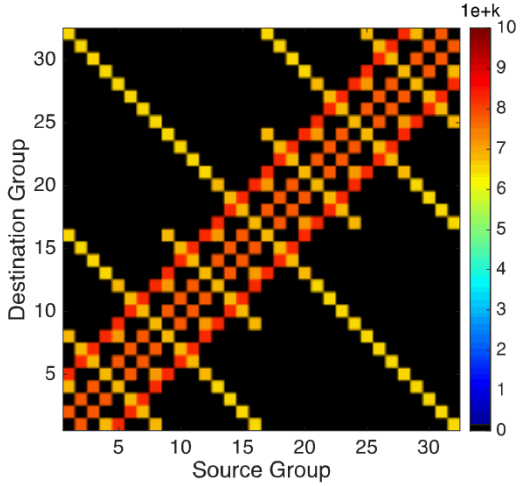
Jeremiah Wilke
Scalable Modeling and Analysis, Sandia National Labs, Livermore CA
Simulation BOF, Supercomputing 2017, Denver, CO
Unclassified Unlimited Release (UUR)

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

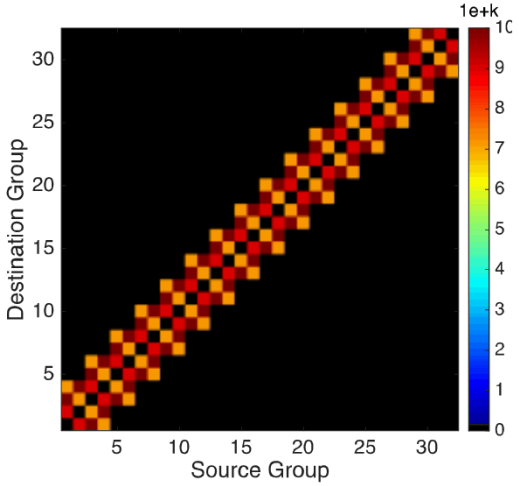
Is network simulation anything more than just injecting a known traffic pattern?



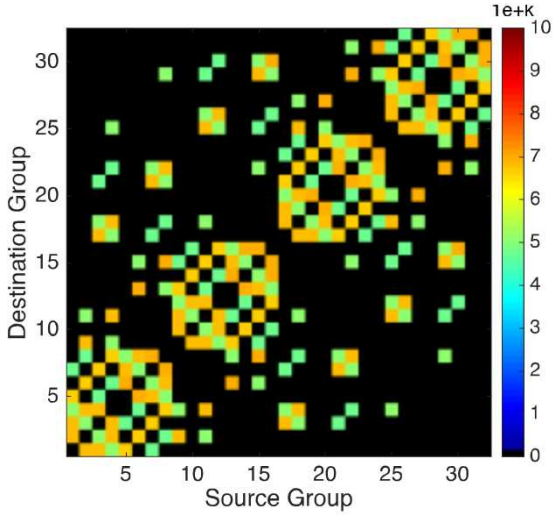
GTC



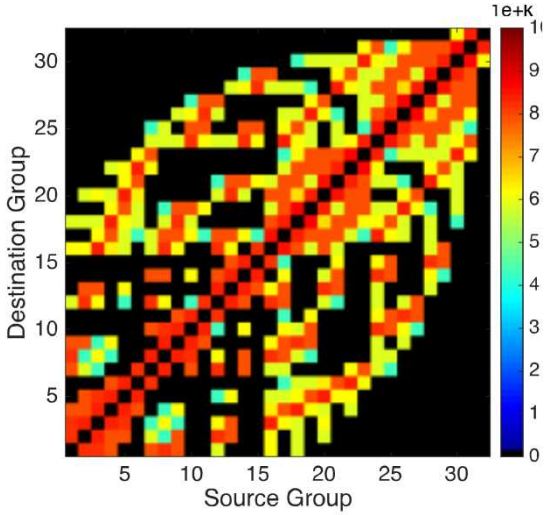
Nekbone



Lulesh

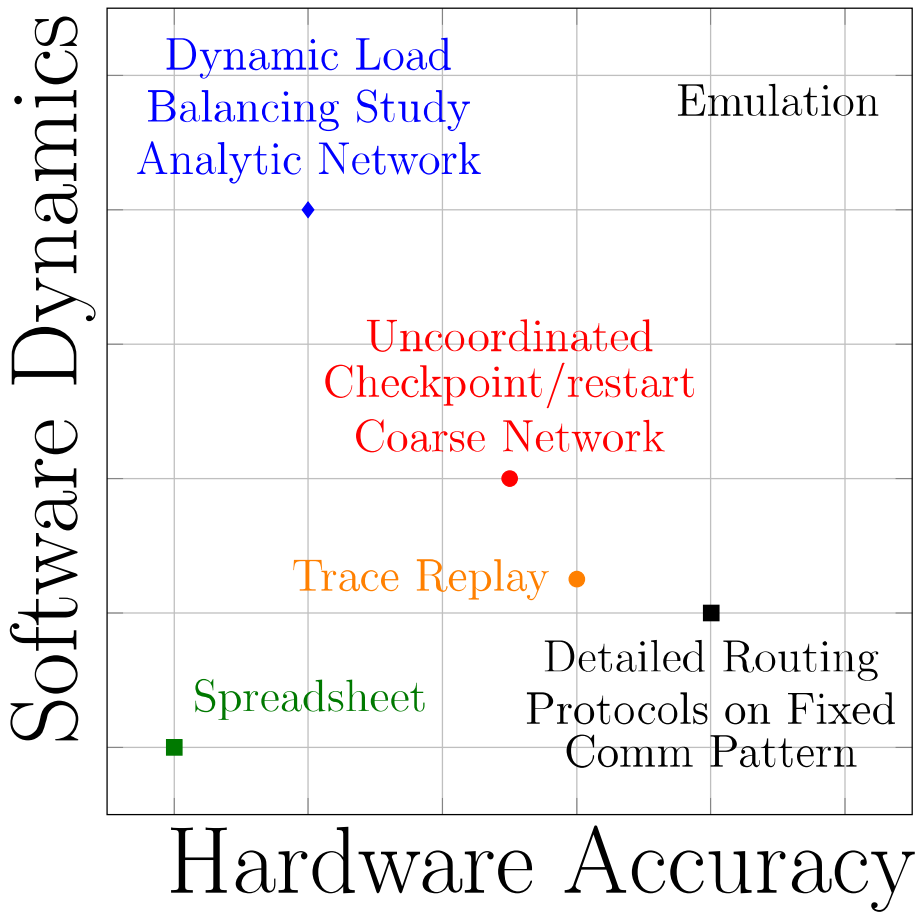


MiniFE



FillBoundary

Selection of endpoint/hardware model depends on the experiment you want to perform

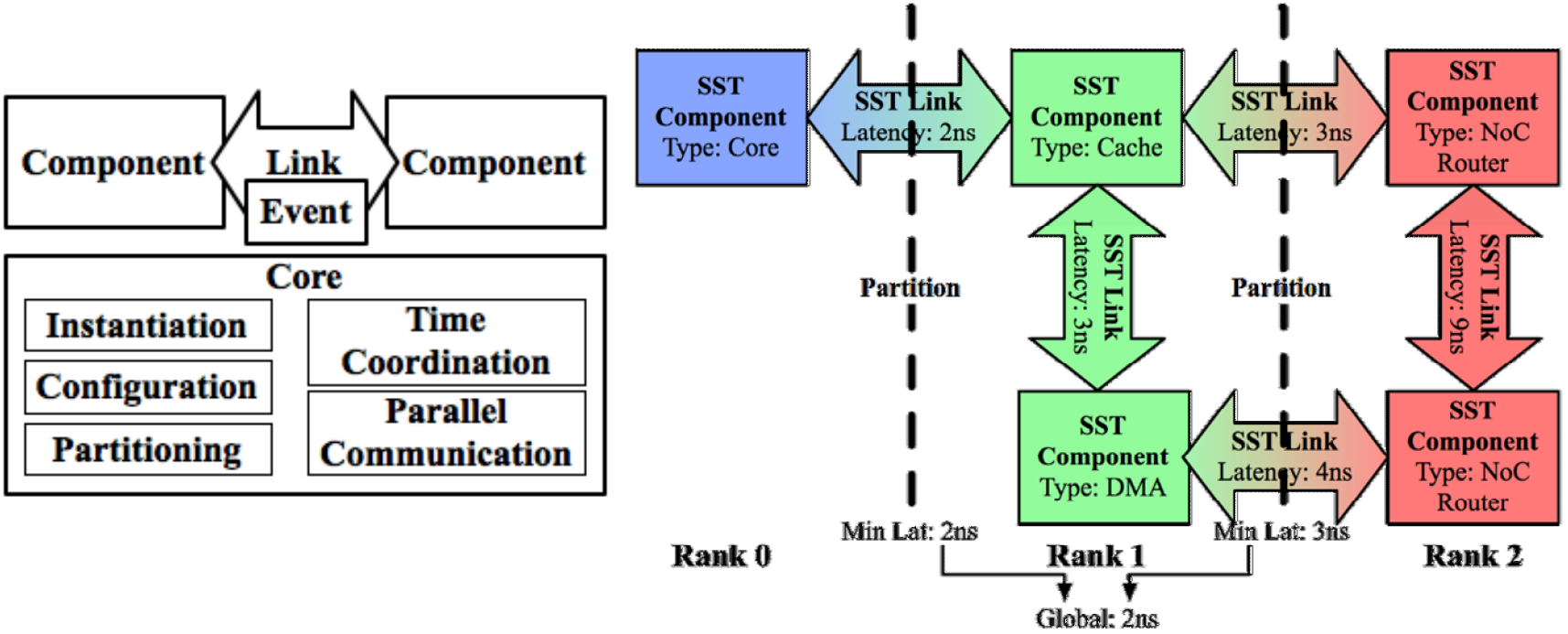


Tools available in SST span spectrum ease of use/flexibility/efficiency

- Trace replay
  - (+) Easy to setup, reasonable accuracy
  - (-) Not flexible, need large compute resources to collect trace
- Ember motif: state-machine generating compute/comm calls
  - (+) Computationally efficient, flexible
  - (-) Work to setup, data-dependent control flow difficult
- Skeleton app: compile and link MPI codes directly into simulation
  - (+) No interface to learn (just MPI code), most flexible
  - ( ) If auto-skeletonizing compiler, not difficult to generate
  - (-) Work to write from scratch, work to skeletonize existing code

Method	Human setup difficulty	Flexibility	Computational efficiency
Traces	Lowest	Lowest	Lowest
Motifs	Medium	Medium-High	Highest
Skeleton	Medium	High	High
Dynamic Skeleton/Motif	High	Highest	Medium-High

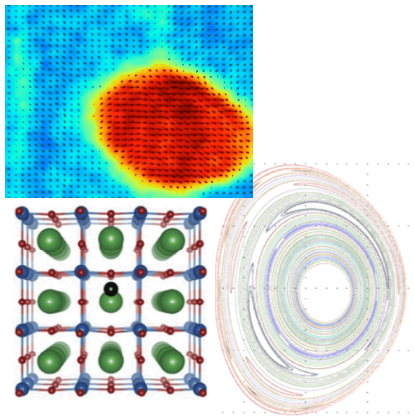
SST is designed from ground-up to enable parallel simulation of interchangeable components



- PDES is a challenging problem – no need to repeat it for every problem domain
  - Partitioning
  - Lookahead optimization/computation
  - Synchronization/event delivery
- SST aims for optimal balance of human time versus computer time
  - Focus on conservative (easier model development) over massively parallel (optimistic)
  - Python input file that builds components and connects them with links

Model for industry/academic collaboration with SST has interchangeable components

Workload models:



Endpoint Interface

SST Core

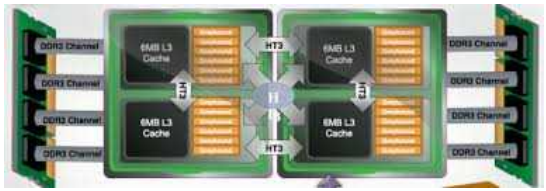
Component | Link / Event | Component

Core

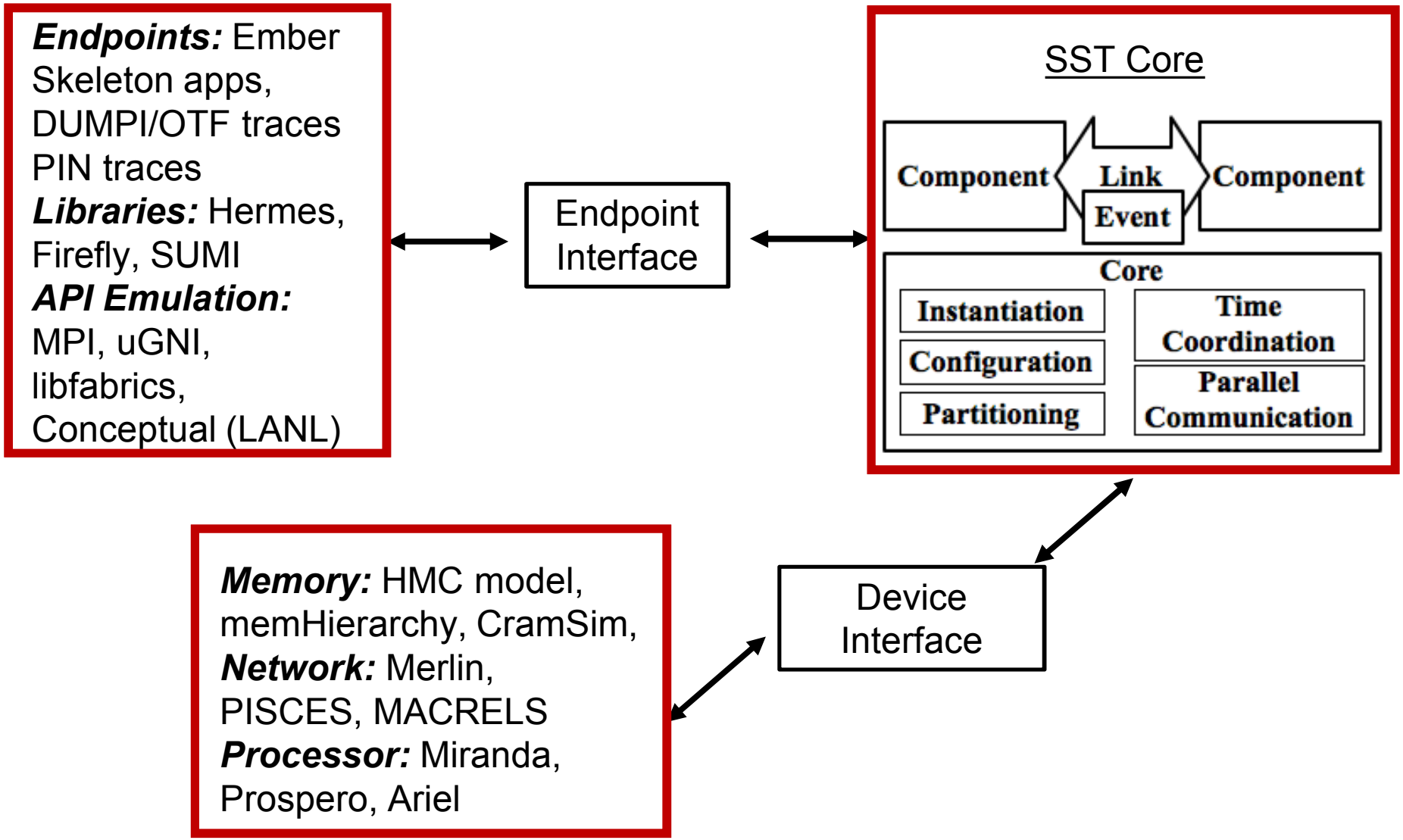
Instantiation | Time Coordination | Configuration | Parallel Communication | Partitioning

Device Interface

Device Models:



Many models with varying accuracy/cost in SST element libraries



The basics of skeleton apps in SST/macro

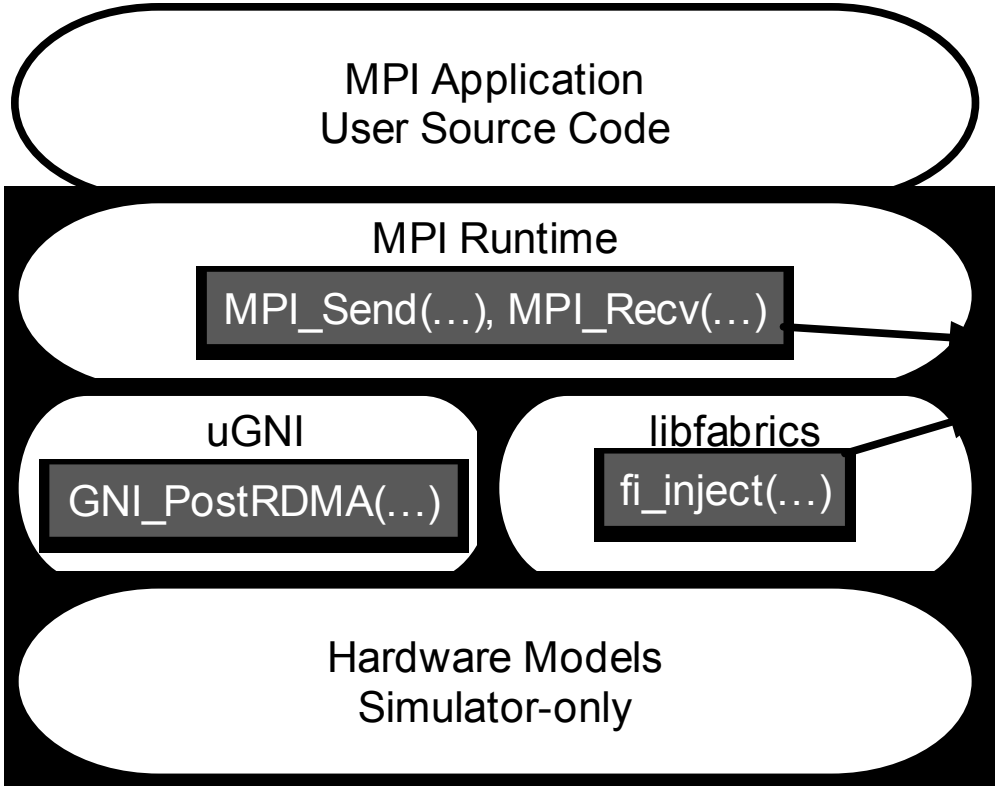
Sample from Lulesh code

```
if (planeMax) {
   /* contiguous memory */
   int fromRank = myRank + domain.tp()*domain.tp() ;
   int recvCount = dx * dy * xferFields ;
   MPI_Irecv(&domain.commDataRecv[pmsg * maxPlaneComm],
             recvCount, baseType, fromRank, msgType,
             MPI_COMM_WORLD, &domain.recvRequest[pmsg]) ;
   ++pmsg ;
}
if (rowMin && doRecv) {
   /* semi-contiguous memory */
   int fromRank = myRank - domain.tp() ;
   int recvCount = dx * dz * xferFields ;
   MPI_Irecv(&domain.commDataRecv[pmsg * maxPlaneComm],
             recvCount, baseType, fromRank, msgType,
             MPI_COMM_WORLD, &domain.recvRequest[pmsg]) ;
   ++pmsg ;
}
```

Intercept MPI calls via compiler wrapper

$>sst++ lulesh-comm.cc –c –o lulesh-comm.o

Software stack simulation leverages low-cost context switching to provide ”virtual” development environment

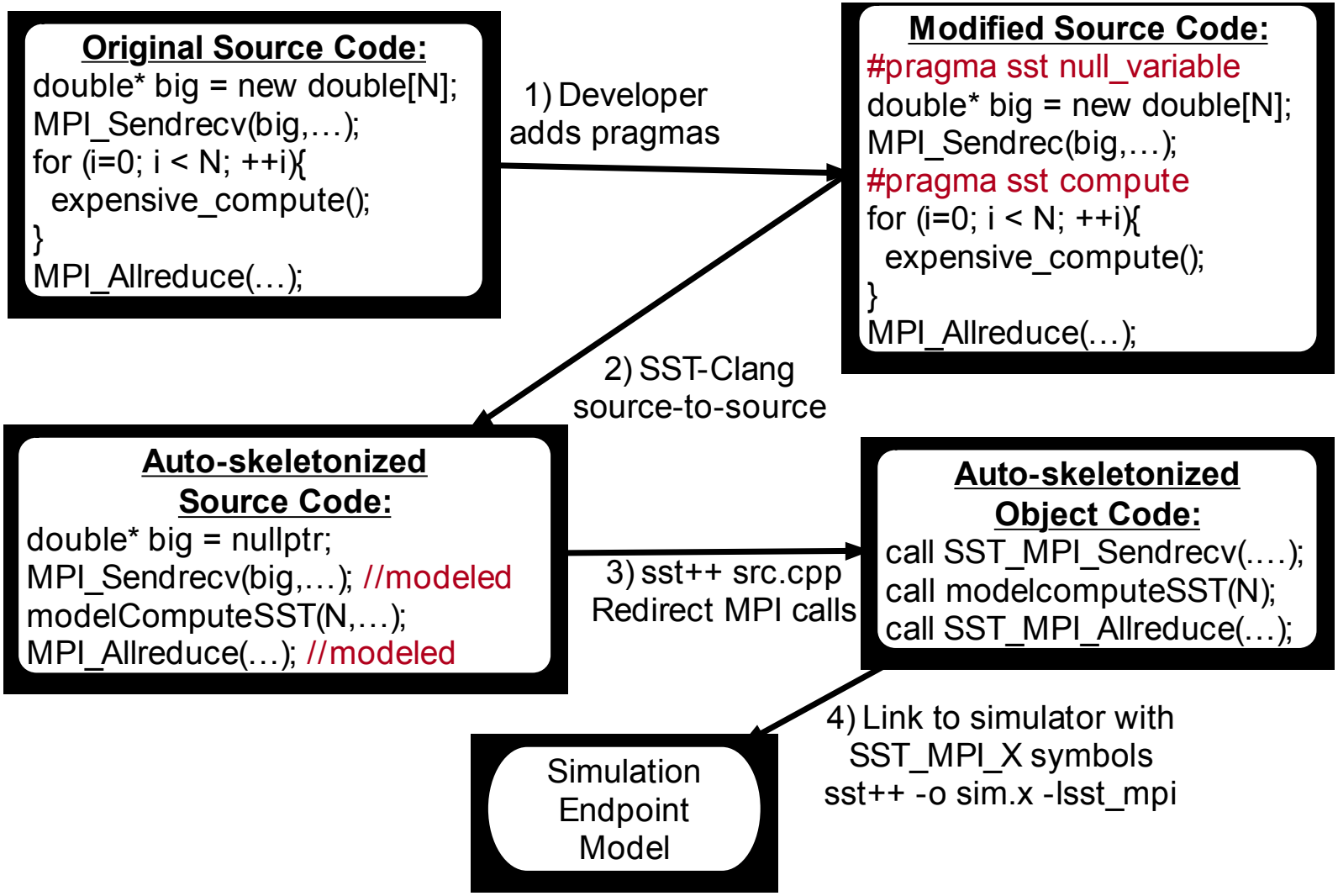


(Mostly) unmodified application code

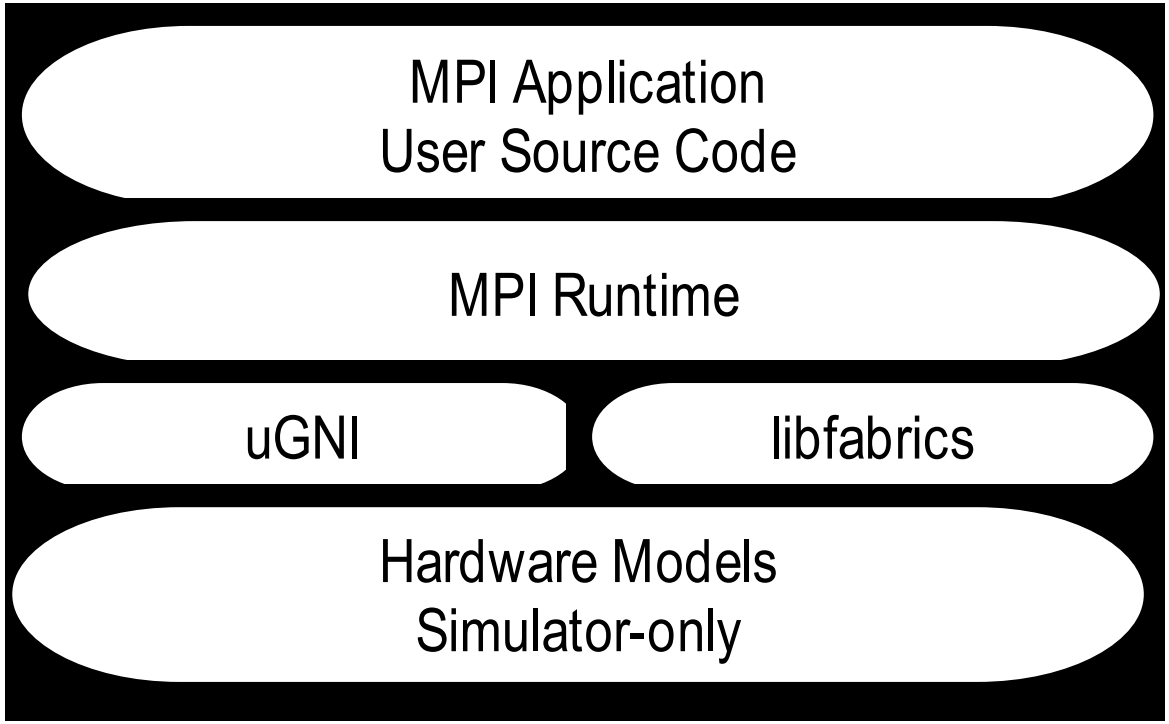
Symbols to intercept with SST/macro

SST middleware and and hardware models

Auto-skeletonizing SST compiler (mostly auto)



Enables experimentation, direct software development of different layers of network stack



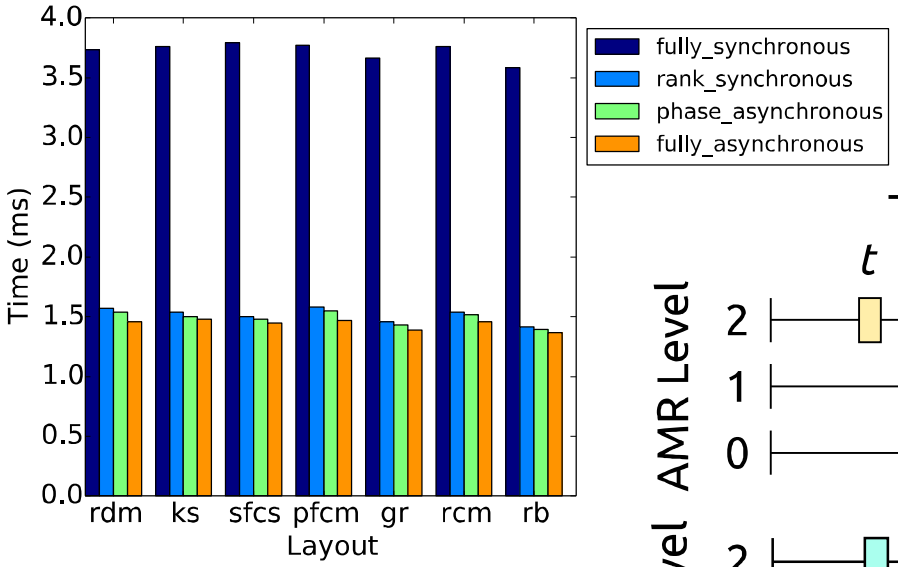
Applications/algorithms

Message order, higher-level semantics

Session layer, message-transfer layer (MTL

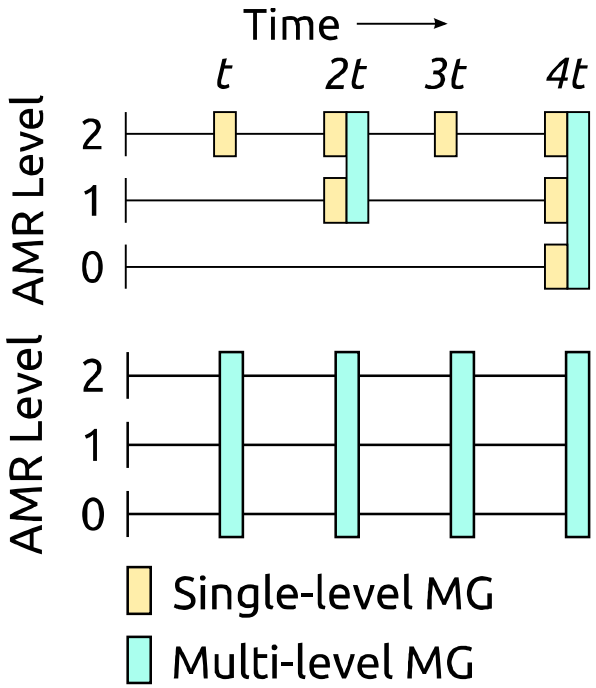
Packetization, byte-transfer layer (BTL), physical network

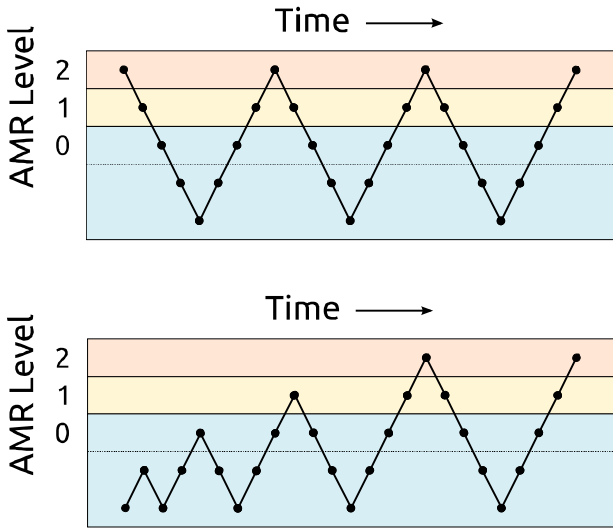
Assesses algorithmic refactoring without needing to overhaul/debug code



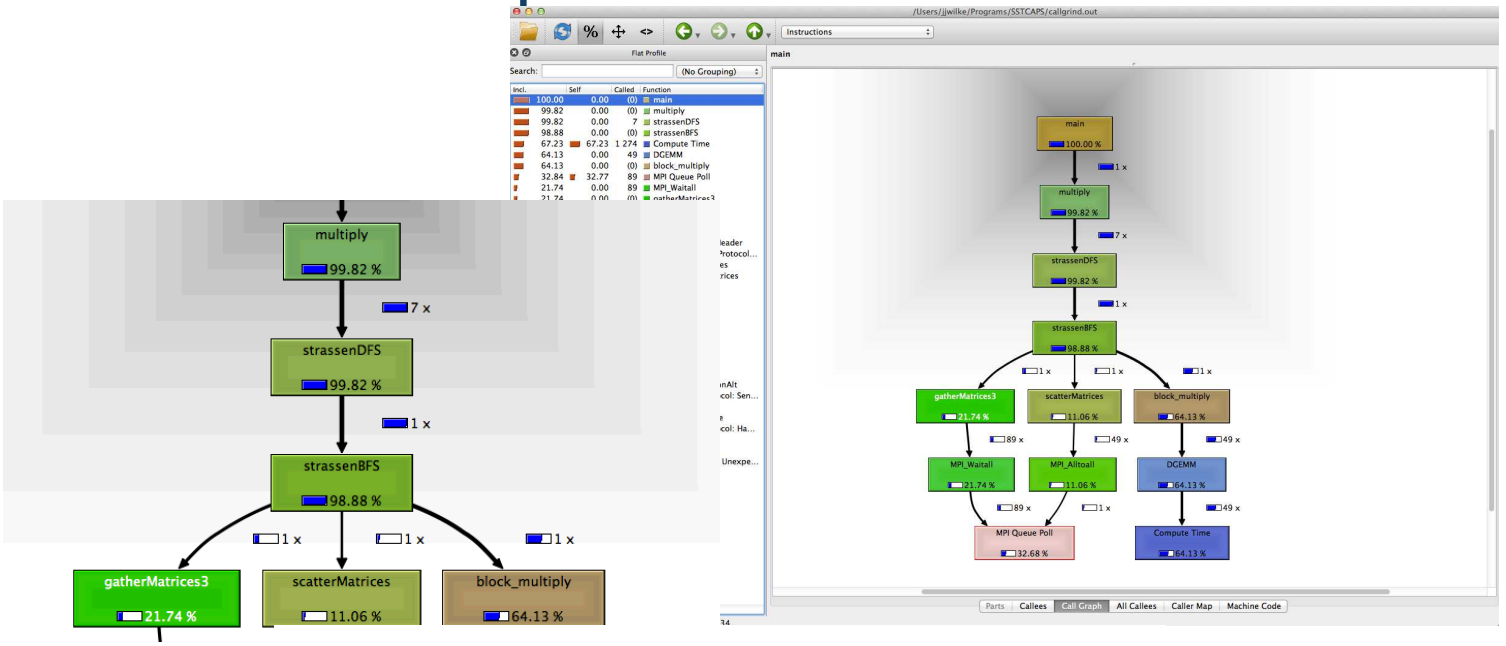
Allreduce → Iallreduce
Waitall → Waitany

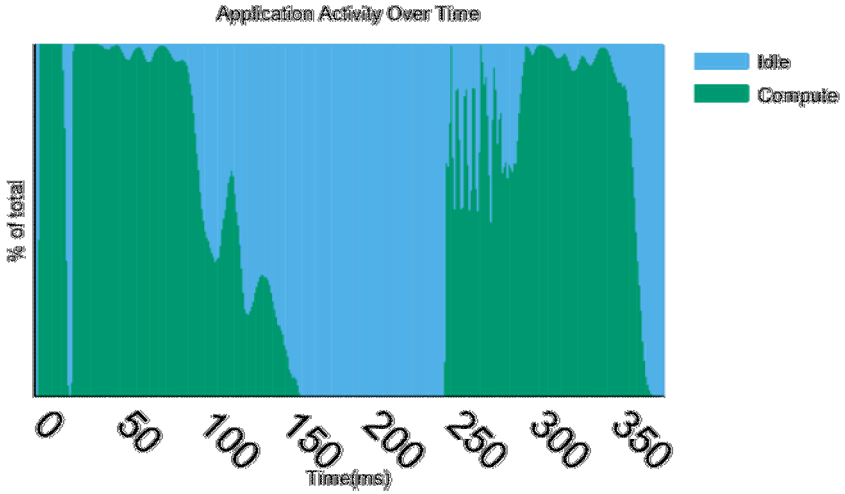
Removing blocking collectives drastically reduces times, other levels of asynchrony give limited performance gains





Extract statistics from virtual software stacks just as done in real performance tools

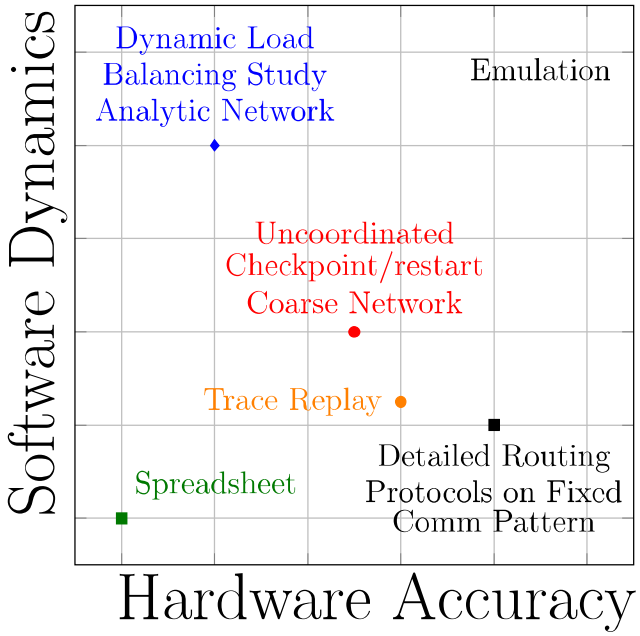




Conclusions

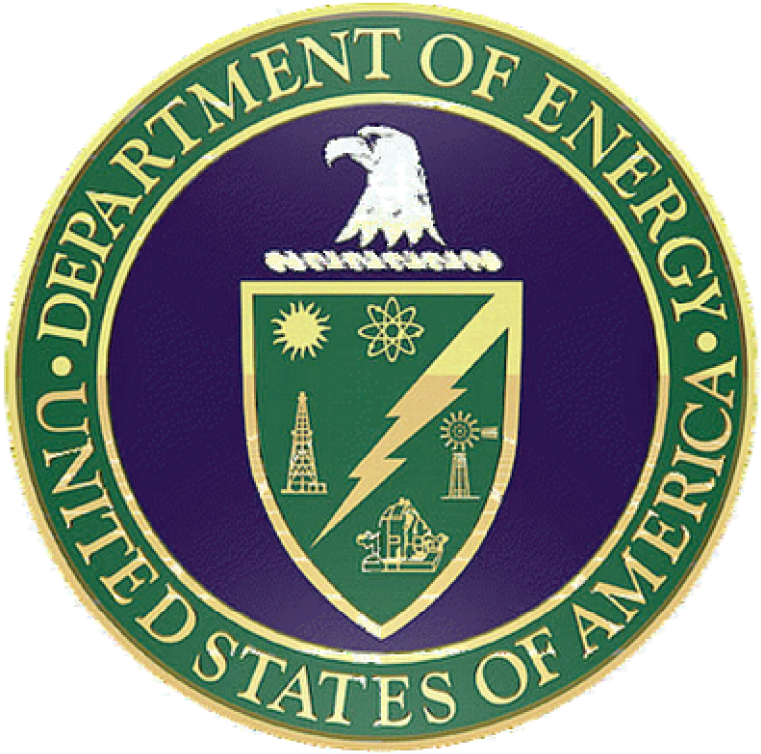
- Large hardware/software design space available to simulation
- Choice of endpoint model/network model depends on research focus/system scale/human time available

Method	Human setup difficulty	Flexibility	Computational efficiency
Traces	Lowest	Lowest	Lowest
Motifs	Medium	Medium-High	Highest
Deterministic Skeleton	Medium	High	High
Dynamic Skeleton	High	Highest	Medium-High



Acknowledgments

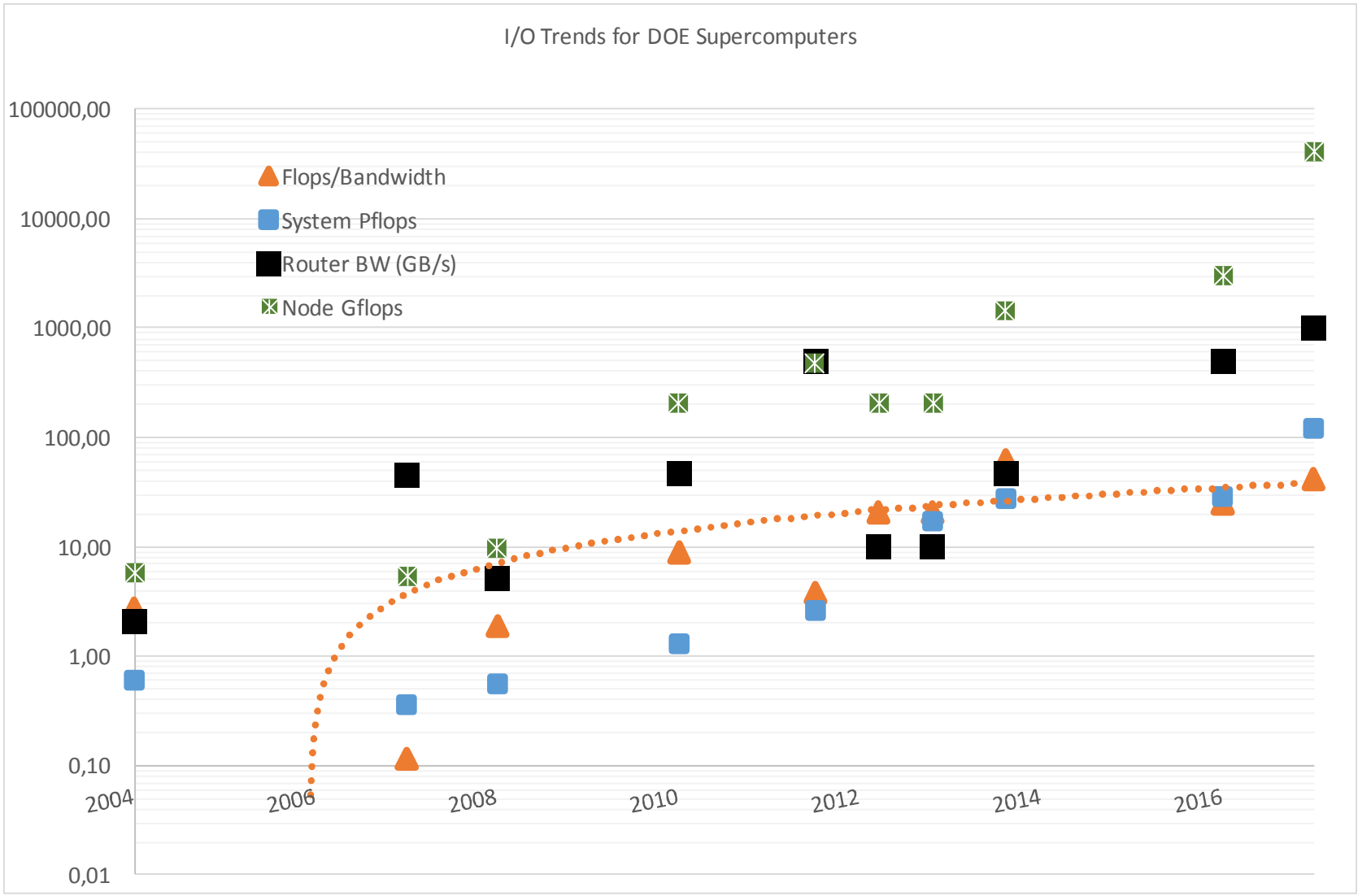
Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.



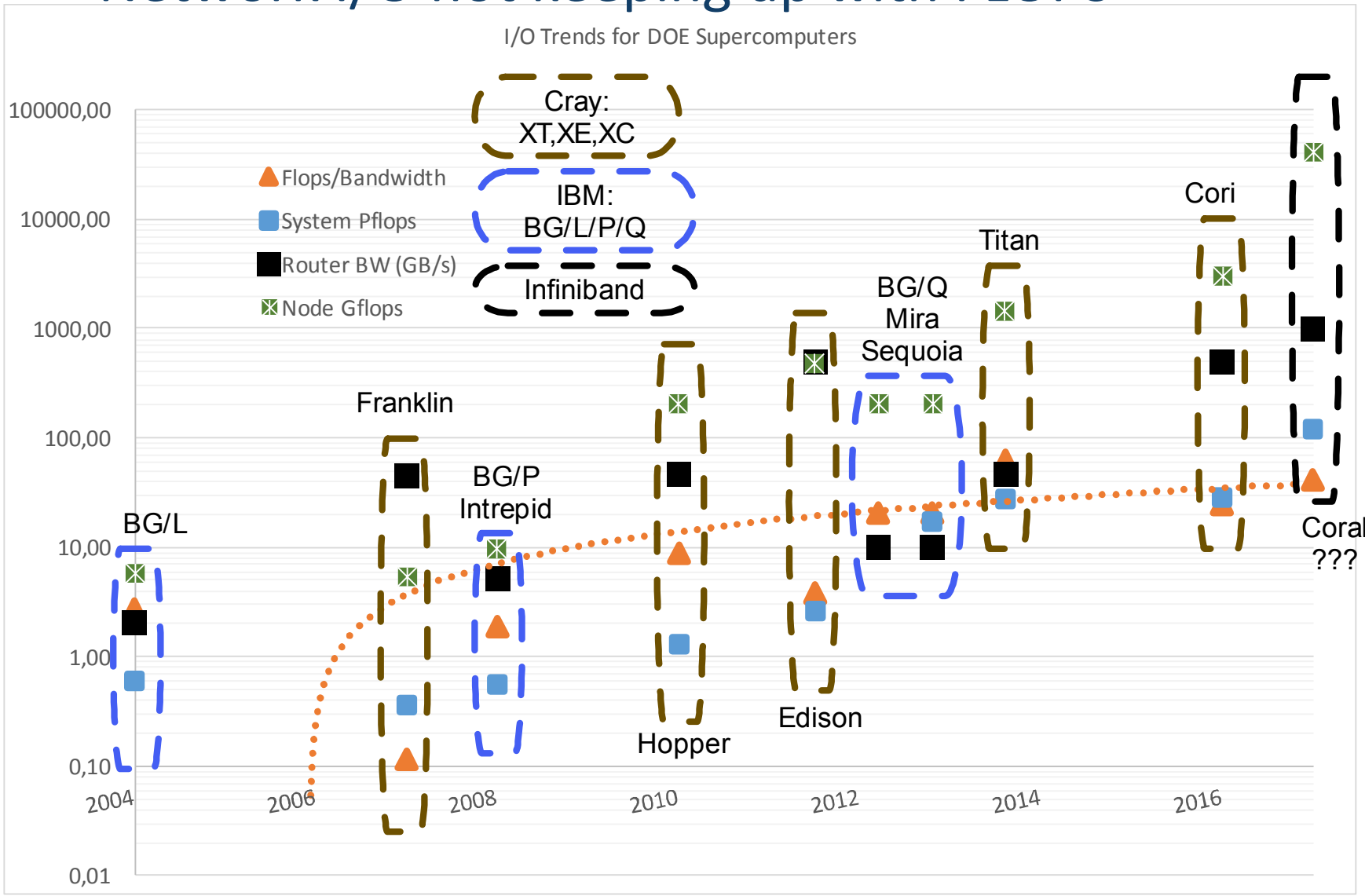
National Nuclear Security Administration

Sandia National Laboratories

System-level problems that need simulation: Network I/O not keeping up with FLOPS



System-level problems that need simulation: Network I/O not keeping up with FLOPS



System level problems that need simulation: File I/O not keeping up with FLOPS