Simulation & Modeling Efforts

Arun Rodrigues

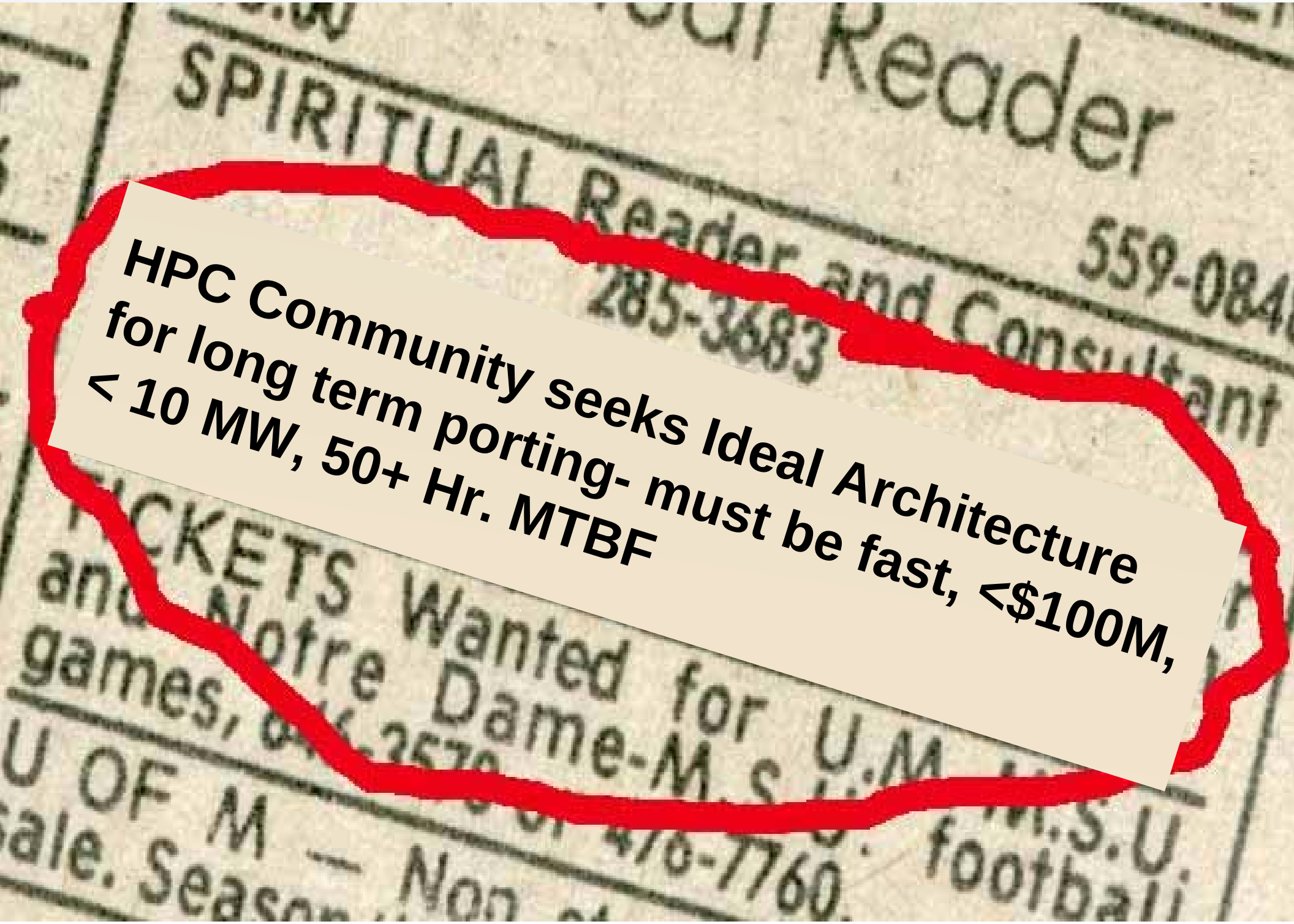
Thanks

n B. Bell (IBM)
Chiou (U.Texas)
Evensky (SNL)
ross (U.Maryland)
Hemmert (SNL)
Jacob (U.Maryland)
Janssen (SNL)
McCurdy (ORNL)
e Riley (GTech)
Roth (ORNL)
Vetter (ORNL)
kar Yalamanchili (GTech)
an Yu (ORNL)

- Brian Barrett (SNL)
- Greg Diamos (GTech)
- Elizabeth Lynch (GTech)
- Mike Levenhagen (SNL)
- Many others....

The Problem

HPC Problem



HPC Community seeks Ideal Architecture for long term porting- must be fast, <$100M, < 10 MW, 50+ Hr. MTBF

.

Many Cores + Memory

X

Many Many Nodes

X

Many Many Many Threads

Audiences.....

r

X

Application writers
purchasers
designers

X

system procurement
algorithm co-design
architecture research
language research

X

present
future

xity.....

Physics Apps
matics Apps

X

Communication Libraries
Run-Times
OS Effects

X

Existing Langu
New Languag

ints.....

Performance

Power

Cooling

Risk

HPC Simulation Main Challenges

work/Application Feedback: A static trace or simple statistic
el will not capture the causal relationships between message

ability: Many network effects only become apparent at hund
ousands of nodes.

able Processor/Memory/Network Systems: Local interacti
have global performance implications.

ty to Model Message Overheads: Overheads in the netwo
packetization, protocol overhead) and messaging library (e.g
hing, message assembly) can have a major effect on
ormance.

ty to Explore Programming Models: Novel hardware will r
l programming techniques and capabilities.

er and Economic Effects: Power and cost are the key limiti
rs on system design. Any system model must be able provid
back on the power and cost implications of new architectural

Promising and worrying industry trends

onductor industry trends

re's Law still holds, but clock speed now constrained by power and cooling limits

cessors are shifting to multi/many core with attendant parallelism

pute nodes with added hardware accelerators are introducing additional complexity of heter
itectures

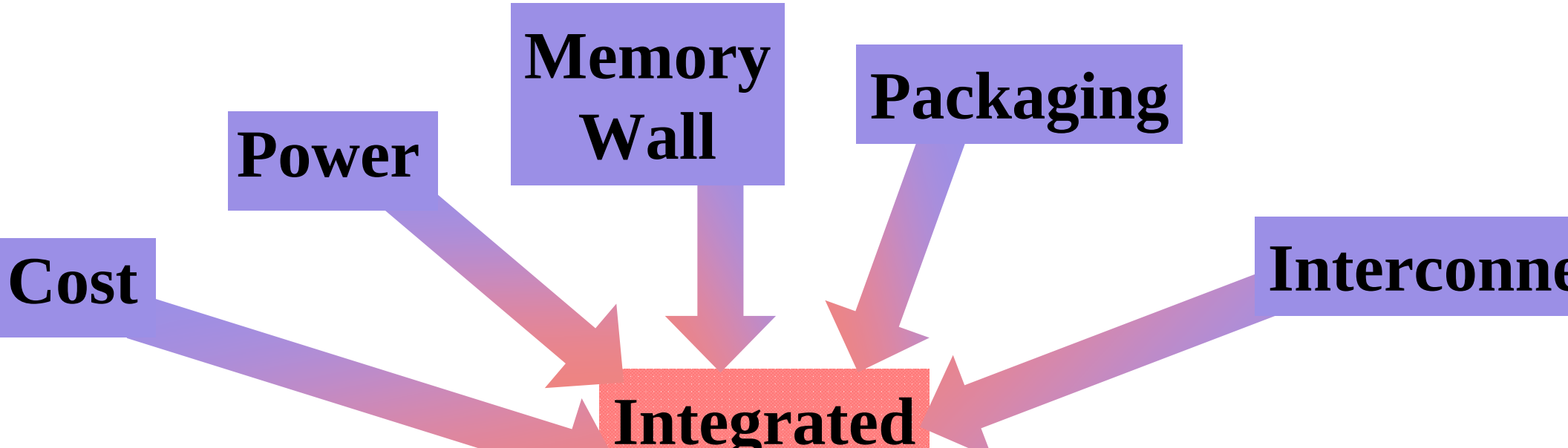
cessor cost is increasingly driven by pins and packaging, which means the memory wall is gr
portion to the number of cores on a processor socket

pment of large-scale Leadership-class supercomputers from commodity computer compone
s collaboration

ercomputer architectures must be designed with an understanding of the applications they a
nded to run

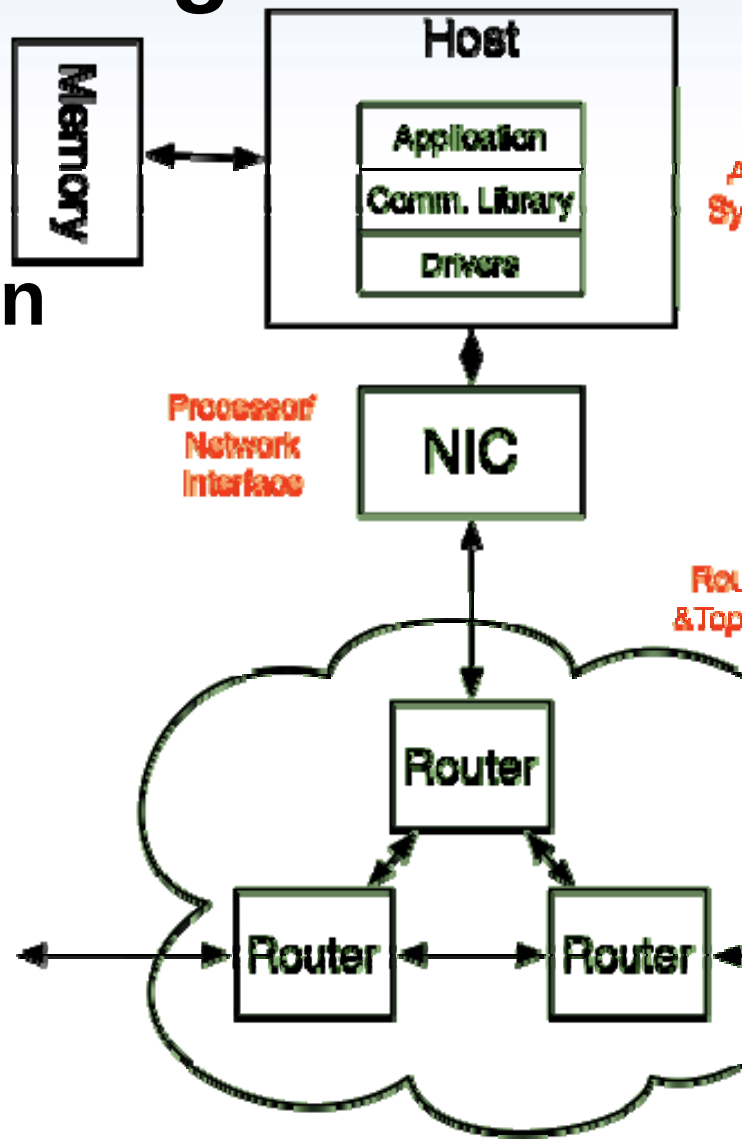
ler to integrate commodity components into a large scale massively parallel supercomputer
itecture that performs well on full scale real applications

lership-class supercomputers cannot be built from only commodity components



HPC Simulation Challenges

- Multiple User communities
  - Procurement: Architecture Evaluation
    - Is this the right machine to buy?
  - Application Writers: Optimization
    - How will code run on this machine?
  - Architects: Design
    - What should the machine look like?
- What level of detail?
- What subsystems examined?



- Current Simulators...
  - Cycle-accurate node-level (accurate, slow)
  - Stochastic network models (accurate?, lose details)
  - FPGA (detailed, fast, hard to develop/scale)

bility

orting

upport

ocumentation

onfiguration & Visualization

allel execution

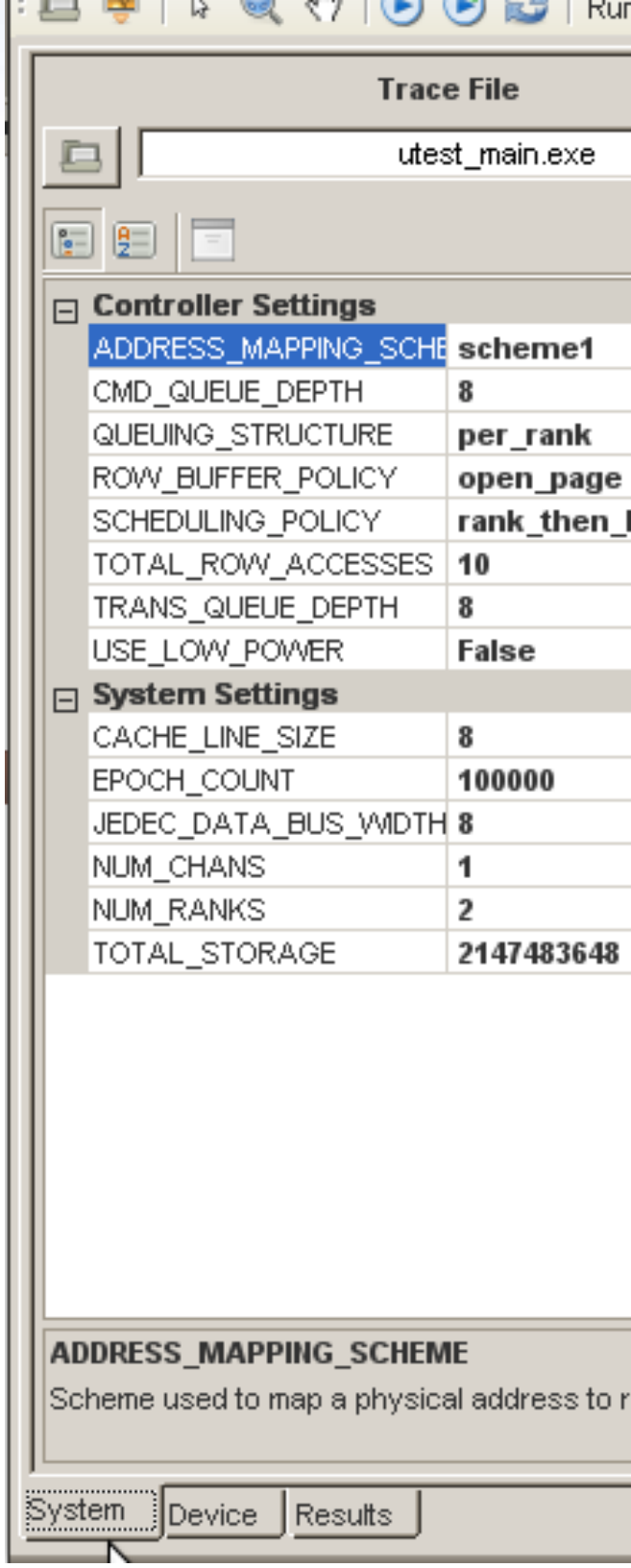
00s of simulated nodes on 100s of al nodes

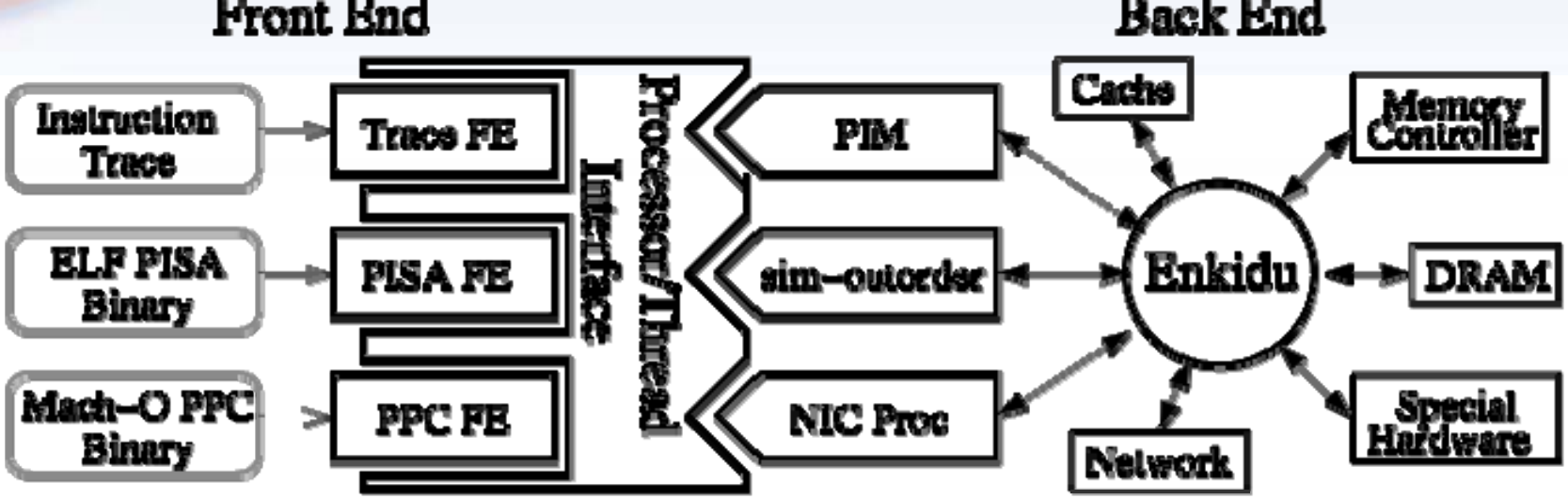
APSTONE & Gossamer Prototypes

Parallel DES simplified because of limited communication patterns

Allows use of conservative distance-based optimization

tiscale: Tradeoff between cision and speed





The SST's modular structure allows flexible simulation

he SST was developed to explore systems with novel rocessors, memories, interconnects and programming mod

lovel Architectures require novel hardware and software

pen Source: http://www.sandia.gov/SST/

ST Features

-Modular, component-based design

-Multiple architectural models

-Multi-(processor|threaded|core) simulation

Current and future S

ustomers

-Application Developers

SW

Performance

oarchitecture

er CacheLine Gather (ICGL)

con. FU (Wisc./SNL)

Aggregates

Memory Ops

cation analysis

mory Footprint

truction Usage

ork/MPI

Tradeoffs

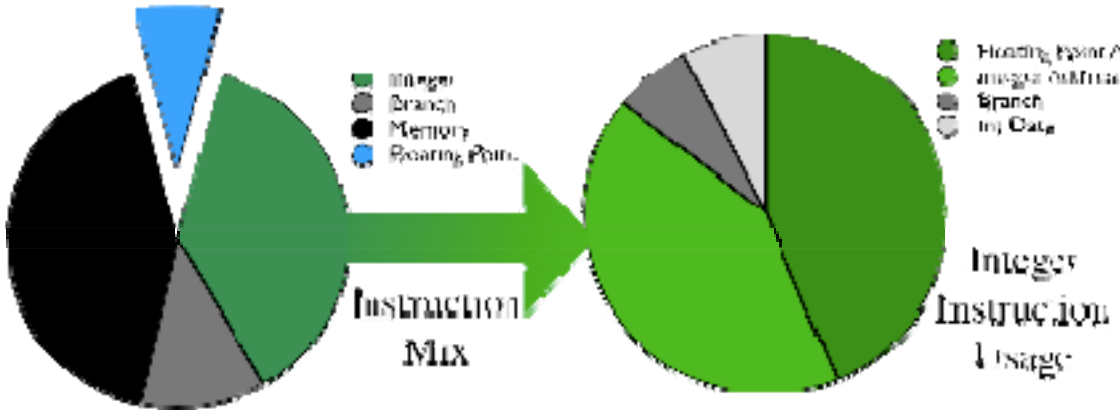
Acceleration

ramming Models

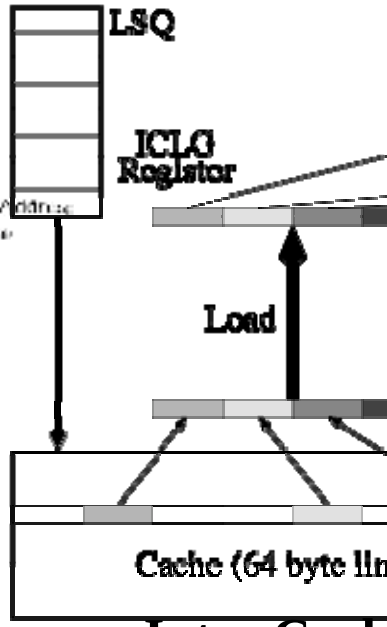
Compiler work (SNL/Rice)

allX (LSU/SNL)(FastOS)

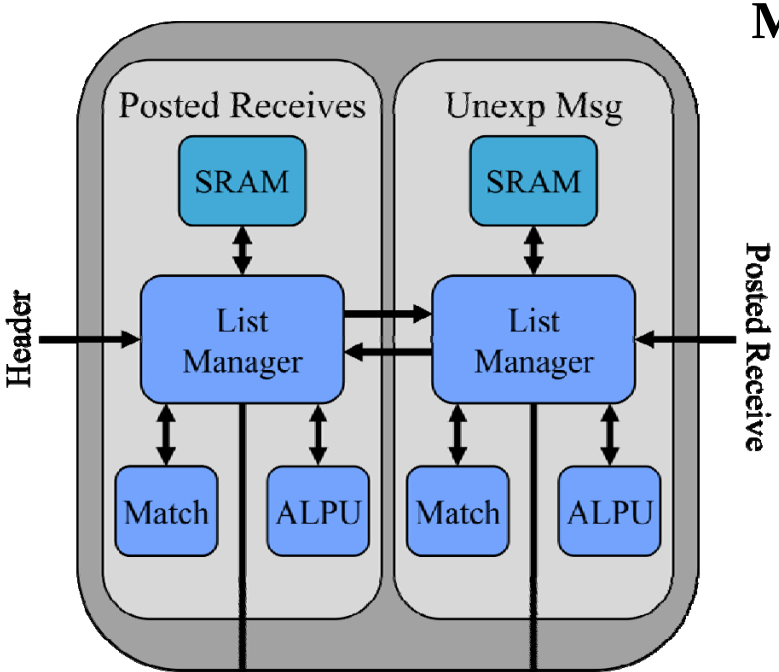
nsactional Memory (ORNL)



Instruction Mix & Usage for Sandia Applications



Inter Cach
Scatter/Ga
Microarchit



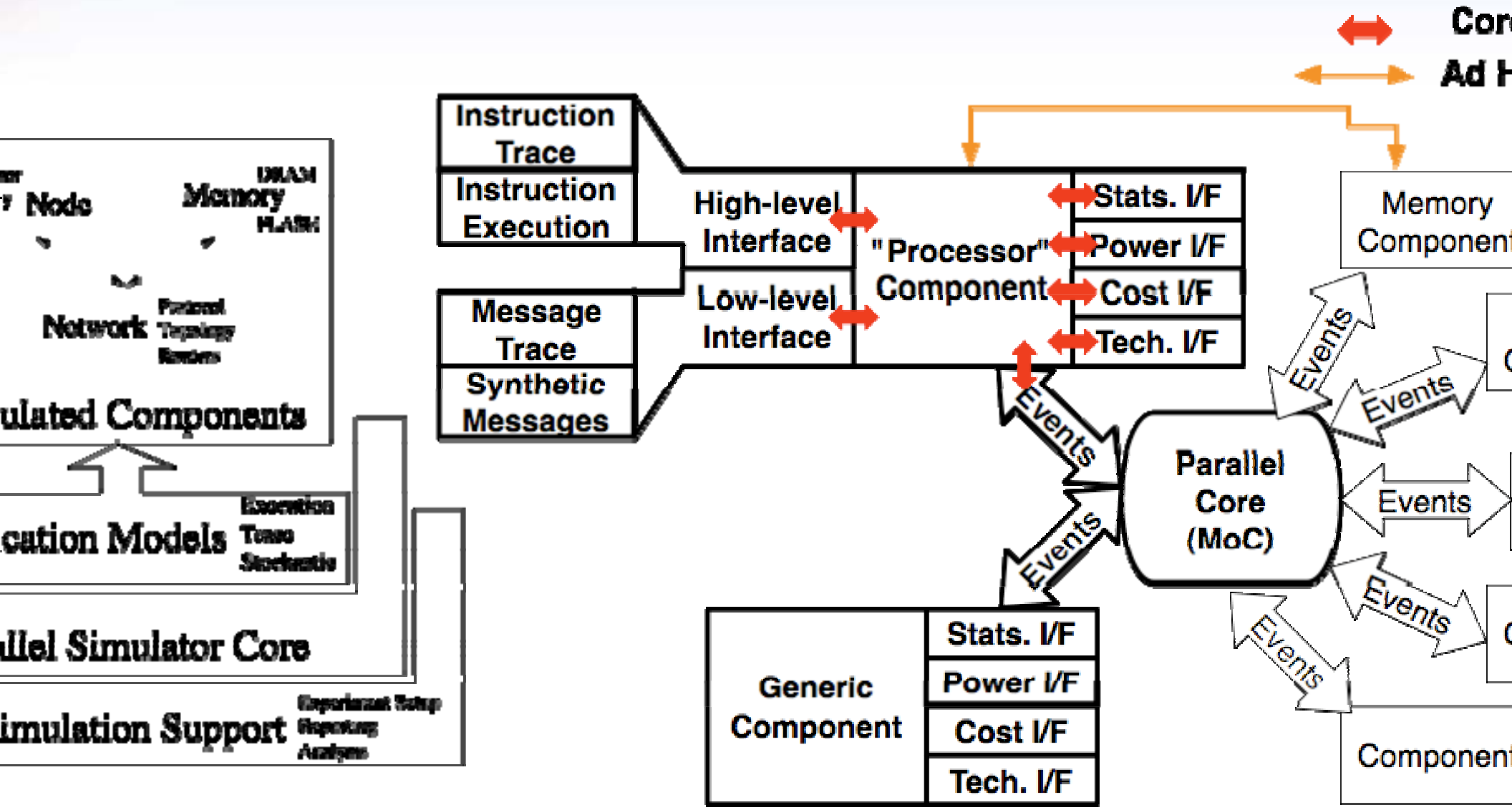
IAA Simulator

Simulation
Strategy

ong Term Vision	Become the HPC community standard simulator	• Long-term support • Community acceptance • Standard integration API • Advisory panel • User group
ong Term Goals	Multi-scale simulation	Instruction- or network-trac driven. 1:1000 to 1:100 slow
	Technology model interface	Power, area, cost models
	Highly scalable parallel	10000s simulated nodes or real nodes
ear Term Goals	Define Interfaces	
	Prototype Parallel SST	
	Open Source Release	
	Improve Component Library	

Model	Message Traces, Symbolic Workload Descriptions	Instruction Based	Execution w/ FP Acceleration
Nodes	100s-1000s	100s	1-10
ed Nodes	10000s-100000s	100s-1000s	1-10
		Cycle-level system	Circuit-level explora

Proposed Architecture



arate Software/Front-End from Hardware/Timing/Back-En

dard interfaces for power, statistics, cost?, technology?

nt-Handling mechanism to coordinate between compone

: Interface component
e at different scales

- & Low-level interfaces
e?)

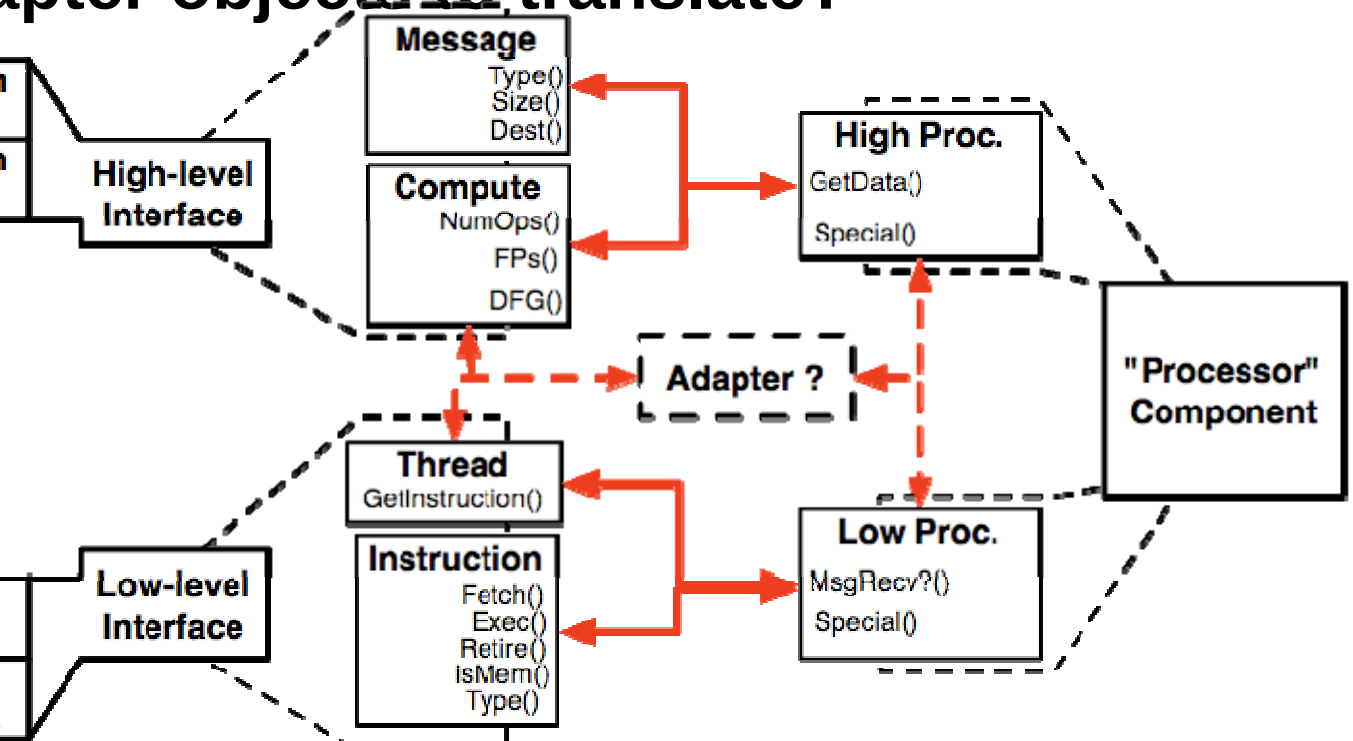
ows multiple input types

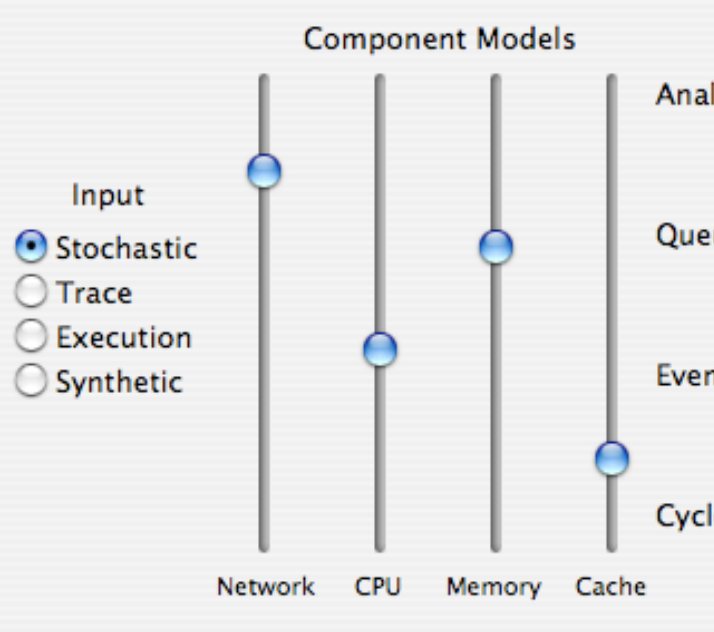
ows multiple input sources

races, stochastic, state-
nachines, execution...

apter objects to translate?

	High-Level	Low
Detail	Message	Instr
Fundamental Objects	Message, Compute block, Process	Instr Th
Static Generation	MPI Traces, MA Traces	Instr Tr
Dynamic Generation	State Machine	Exe





Multiscale Parameters

Current Status

wman

PI Testbed”

1000 lines of code

emonstrates basic functionality

- Sim startup
- Component partitioning
- Checkpointing
- Event passing

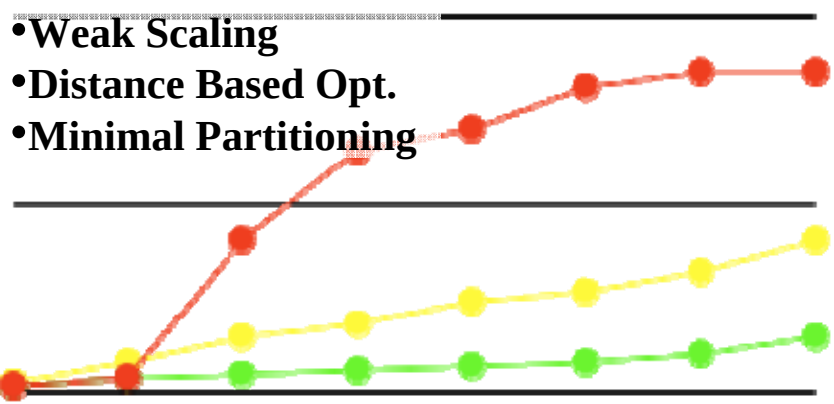
rent work

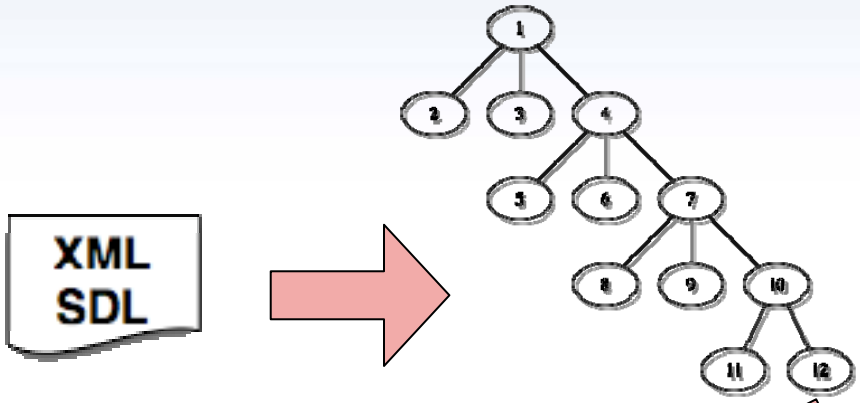
ystem Description Language

efining Parallel DES

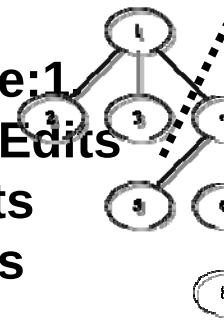
vent interfaces

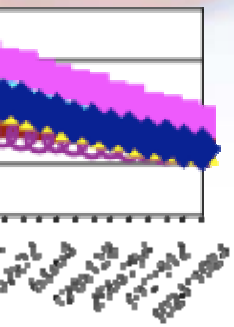
caling



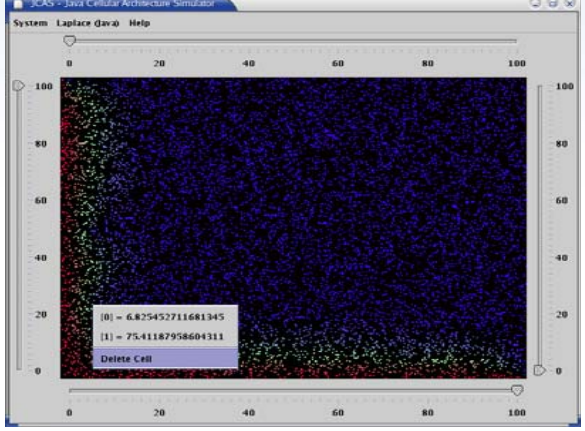


– Initial/Setup Mode:1. Load config file(s)2. Generate component graph3. Partition graph
– 4. Instantiate components on each node
  - 5. Dump initial checkpoint Run Mode:1. Read checkpoint from disk2. Apply Edits 3. Run Loop a. advance components upto time+dt b. exchange messages with neighbors c. goto (3a)





g Assertions

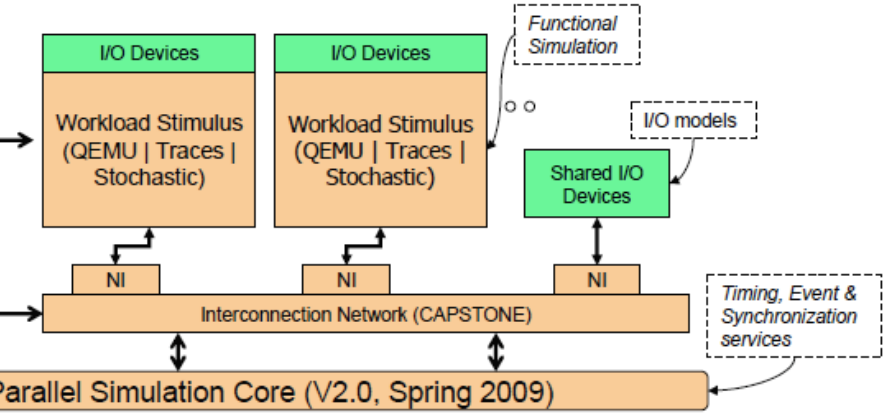


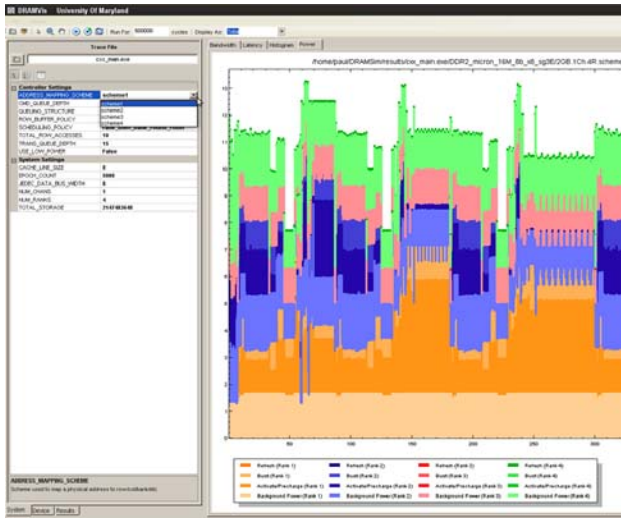
JCAS Vizualizer

- IAA Simulation effort is a community effort
- Seeking more partners...

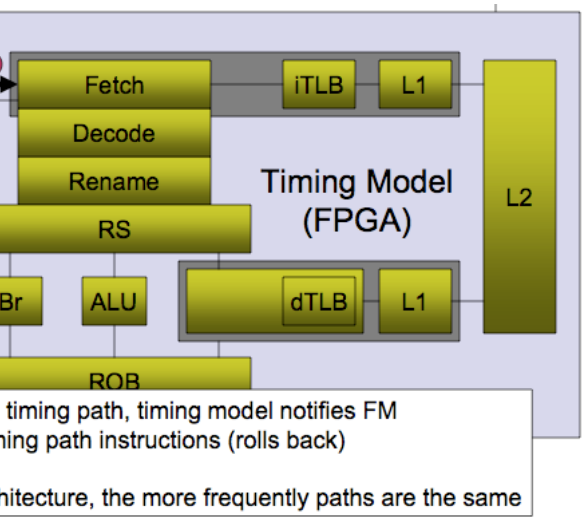
- Current consortium
  - Sandia (Structural Simulation Too
  - ORNL (Scalable application model
  - U. Maryland (DRAMSim II)
  - U.Texas-Austin (FAST)
  - Georgia Tech
  - JCAS (ORNL)
  - Seshat (SNL)

Overview

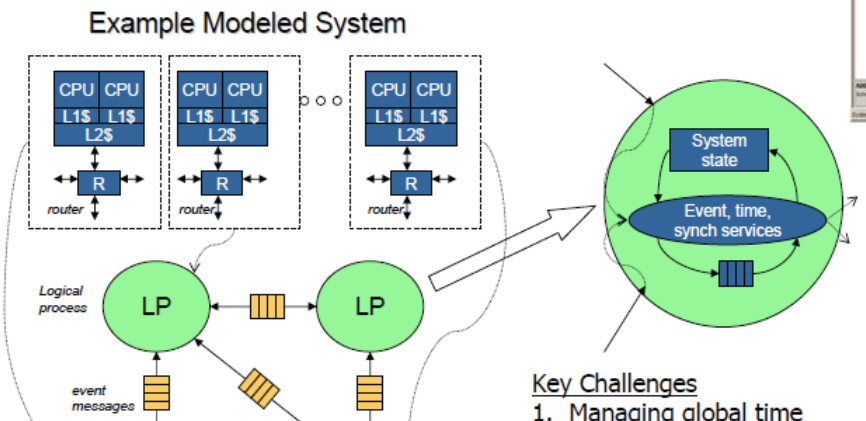




DRAMSim II



FAST



Key Challenges
1. Managing global time

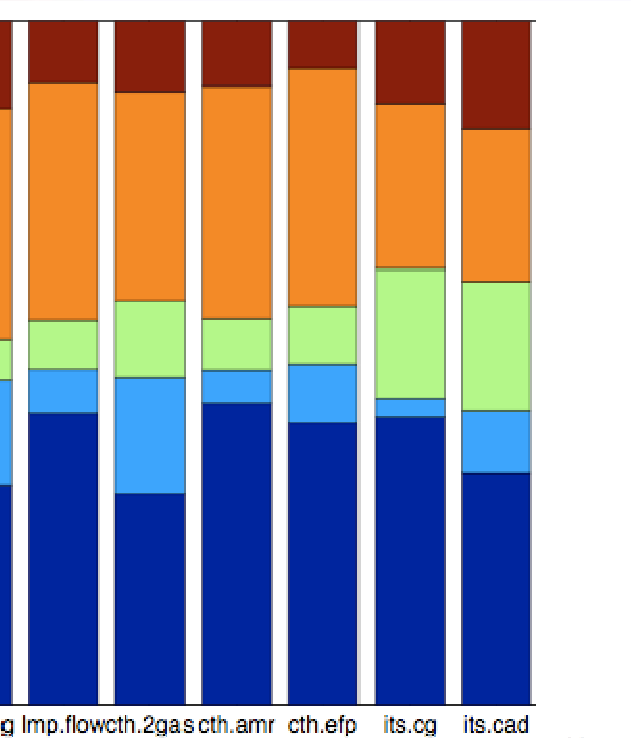
mulator aims to address the key questions in HPC system simul
lable simulation
tiscale simulation

mulator aims to be an effective tool for...
stem procurement
gorithm co-design
chitecture research

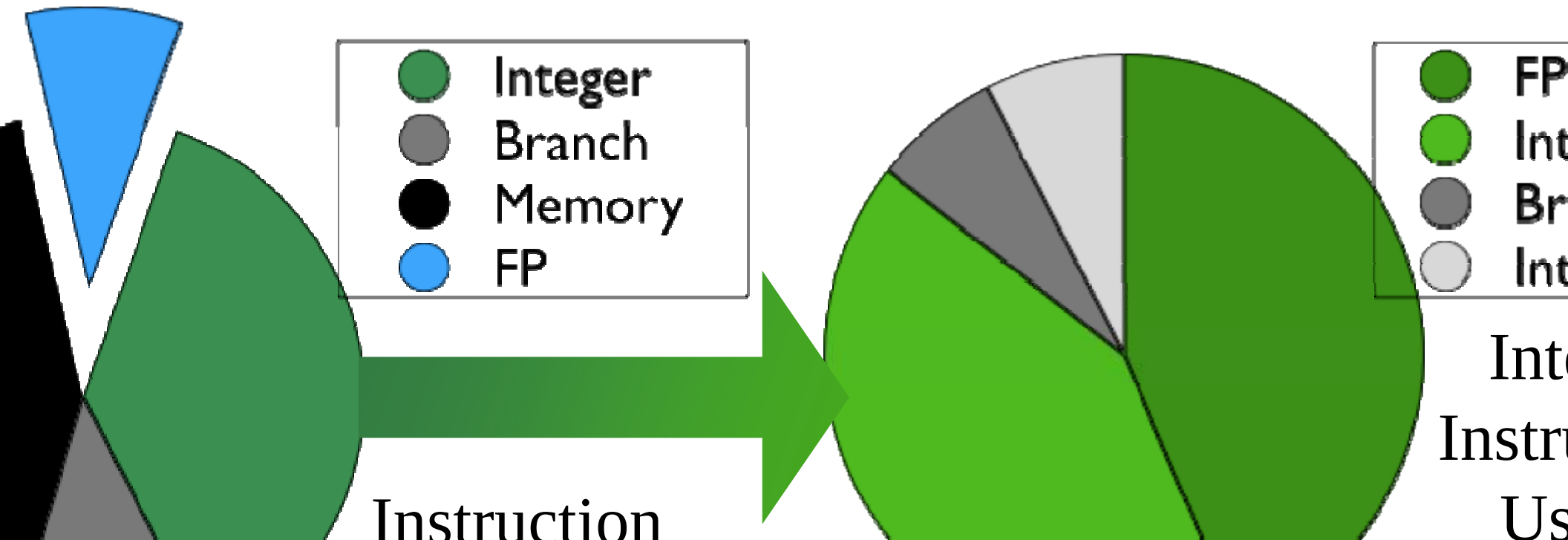
mulation Group is actively soliciting input from potential users a
ers
at are <u>your</u> requirements?
at should the simulator look like?
can we deal with your IP issues?
w would you like to be involved?

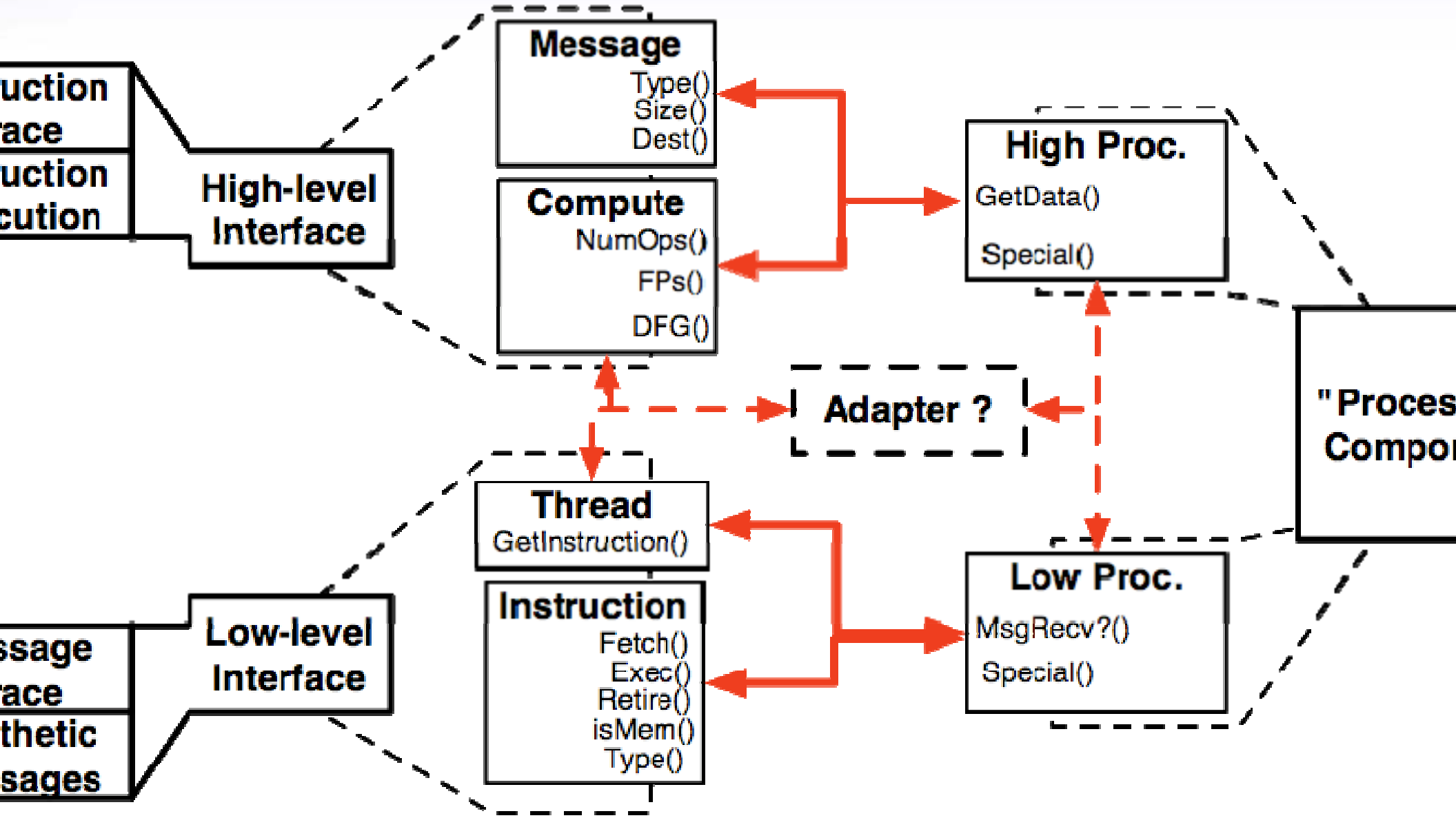
Bonus Slides

Memory Operations Dominate



- FP ops ("Real work") < 10% of Sandia c
- Several Integer calculations, loads for e FP load
- Memory and Integer Ops dominate
  - ...and most integer ops are computin memory addresses
- Theme: processing is now cheap, data movement is expensive





nponent reuse at different scales

ercomputers to design
ercomputers

cal mass of component
lators

cessors

mory

work

isolation

cal mass of application
els

mpact Apps

lable application models

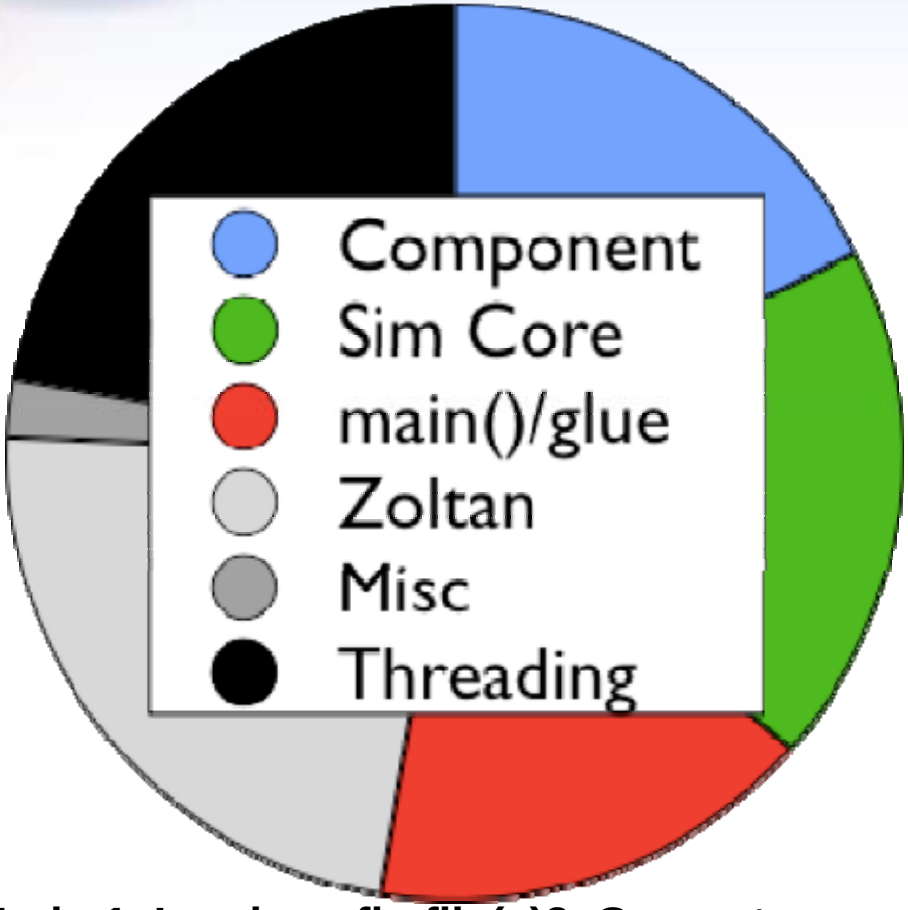
te machine models

ssage traces

simulation super-

- SimpleScalar & PTLSim
  - Popular
  - Processor oriented
  - Conventional processor/memory only
  - No Network
- simg4 simg5 Tango
  - High accuracy
  - Proprietary
  - Single model processor only
- LSE/MicroLib
  - Very detailed
  - Fine grained
  - Slow execution and developmen
- Simics / GEMS
  - Not suited to network study

Grammar modifications



Mode:1. Load config file(s)2. Generate component graph3.
ph

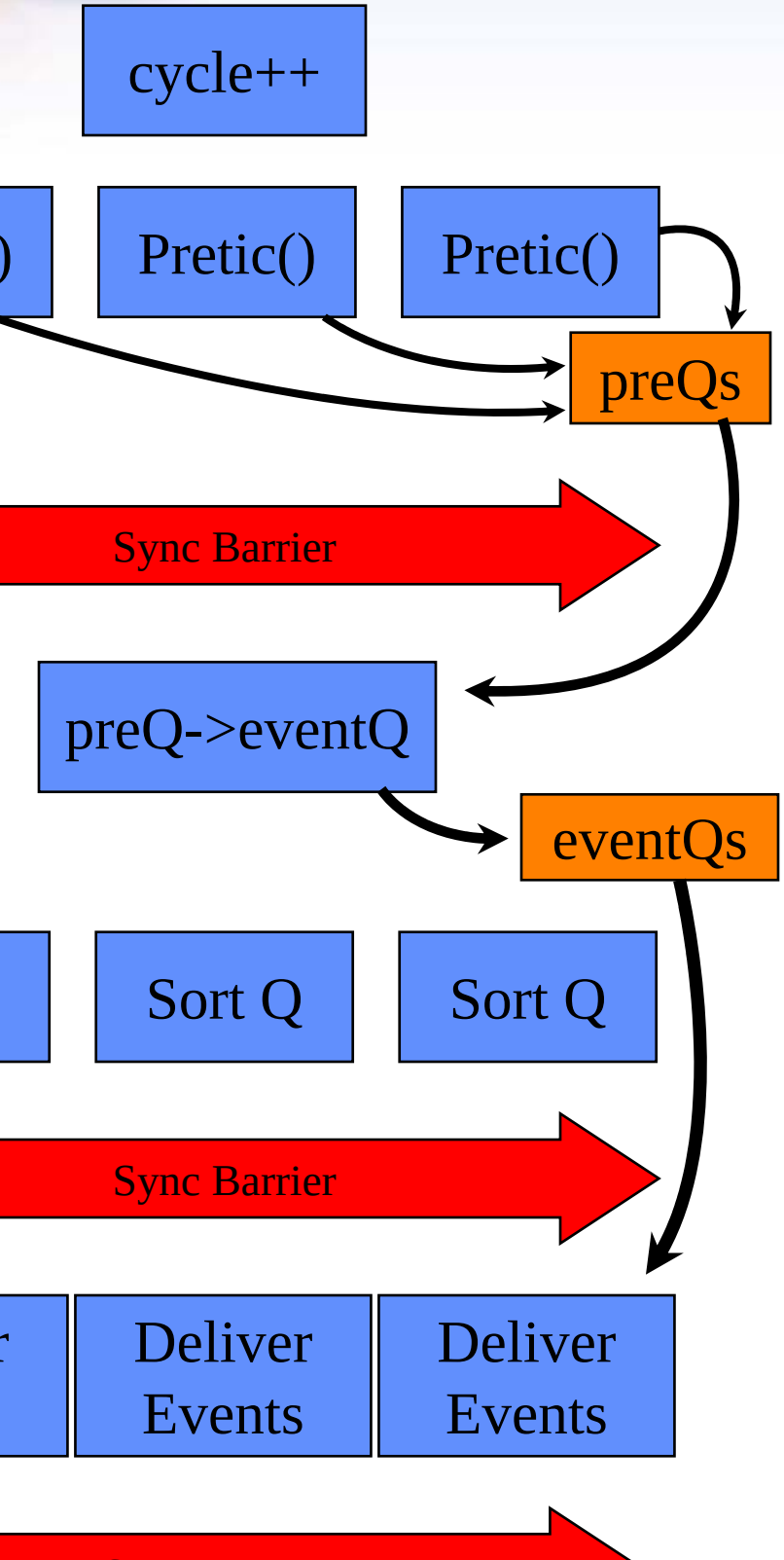
nal partition latency (dt)5. Distribute subgraphs to each
ntiate components on each node

components

initial checkpoint Run Mode:1. Read checkpoint from
pply Edits 3. Run Loop a. advance components upto
b. exchange messages with neighbors c. occasionally,
oint d. goto (3a)

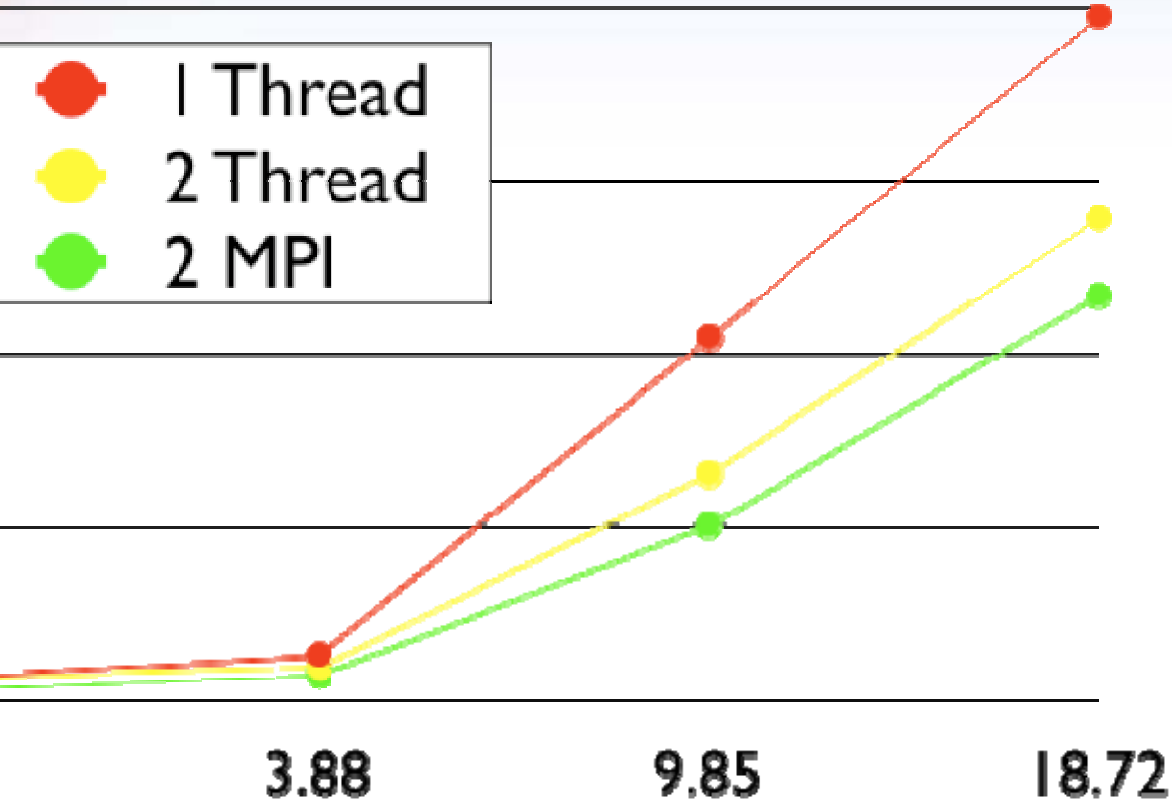
- Added ~250 LOC
  - Original ~1000
- Each Component & i
  outgoing links assig
  to a single thread
  - Minimal locking
- Three parallel operat
  - Pretic()s
  - HandleEvent()s
  - Queue sorting

Run Loop

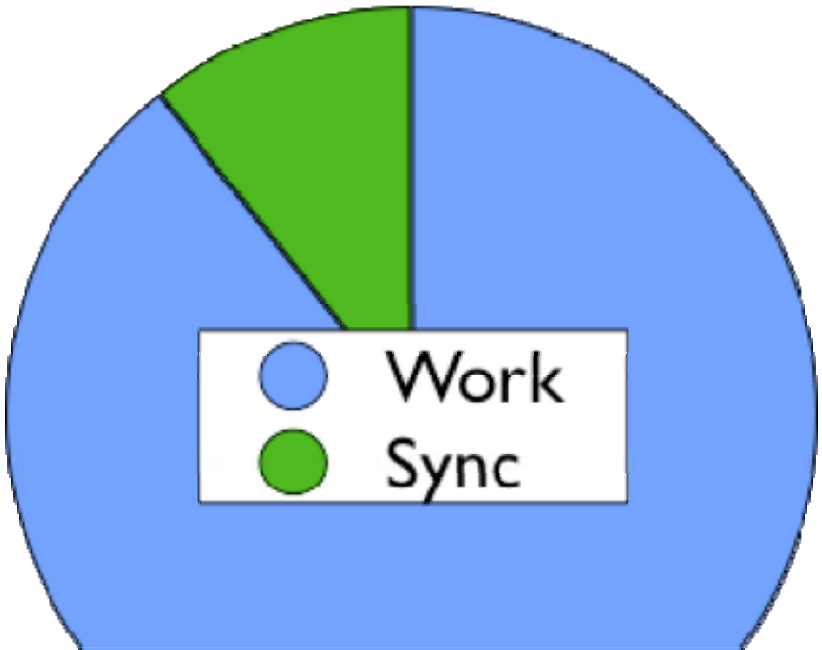


- **Parallel: Pretic()s place newly generated event into preQs**
- **Serial: preQ contents sorted into per thread eventQs**
- **Parallel: eventQs order**
- **Parallel: events deliver**

Performance



- Scaling similar to pu
  MPI
- Performance slightl
  worse than pure-MP
- Reasonable Sync vs
  work



- Improvements
  - better locking
  - improved partitionin
    (currently round robi
  - more locking → finer