Noble Truths of HPC

Arun Rodrigues
SMTS
Sandia National Labs
Four Noble Truths

1. HPC is Suffering
   – HPC is hard (by definition)
   – Bigger scale
   – Higher Performance
   – Uncharted

2. Suffering (in part) arises from attachment to legacy codes, programming models, and paradigms
   – "flops focused"
   – BSP / MPI
   – Procurement Paradigm: COTS-dominated
   – Design Paradigm: We don’t design
More Suffering

3. Suffering ceases when attachment to legacy codes, programming models, hardware, and paradigms ceases

– To reach exascale, we need dramatic restructuring of how we build sw and hw codes will need to be rewritten

– Paradigms must change
  • performance-aware (where perf != flops, but is closer to perf == data movement)
  • power-aware (and energy aware)
  • fault-aware (fault avoiding and fault detecting and fault tolerant)
  • codesign (we must optimize both HW & SW)
A Path Forward

4. Freedom from suffering is possible through simulation and modeling
   – Need tools to guide us
   – Must address multiple audiences
     • multiple needs for guidance
     • asking different questions
   • Ergo, we need better simulation & modeling tools to guide us.
View of the Simulation Problem

Scale.....
- Many Cores + Memory
- Many Many Nodes
- Many Many Threads

Multiple Audiences.....
- Network Processor System
- Application writers purchasers designers
- System procurement algorithm co-design architecture research language research
- Present systems future systems

Complexity.....
- Multi-Physics Apps Informatics Apps
- Communication Libraries Run-Times OS Effects
- Existing Languages New Languages

Constraints.....
- Performance Cost
- Power Reliability
- Cooling Usability
- Risk Size
## SST Simulation Project Overview

### Goals
- Become the standard architectural simulation framework for HPC
- Be able to evaluate future systems on DOE workloads
- Use supercomputers to design supercomputers

### Technical Approach
- **Parallel**
  - Parallel Discrete Event core with conservative optimization over MPI
- **Holistic**
  - Integrated Tech. Models for power
  - McPAT, Sim-Panalyzer
- **Multiscale**
  - Detailed and simple models for processor, network, and memory
- **Open**
  - Open Core, non viral, modular

### Status
- **Current Release (2.1) at**
  [code.google.com/p/sst-simulator/](https://code.google.com/p/sst-simulator/)
- Includes parallel simulation core, configuration, power models, basic network and processor models, and interface to detailed memory model

### Consortium
- “Best of Breed” simulation suite
- Combine Lab, academic, & industry

![Consortium Logos](image)
Message Handling

• SST core transparently handles message delivery
• Detects if destination is local or remote
• Local messages delivered to local queues
• Remote messages stored for later serialization and remove delivery
  – Boost Serialization Library used for message serialization
  – MPI used for transfer
• Ranks synchronize based on partitioning
Multi-Scale

• Goal: Enable tradeoffs between accuracy, flexibility, and simulation speed
  – No single “right” way to simulate
  – Support multiple audiences
• High- & Low-level interfaces
  – Allows multiple input types
  – Allows multiple input sources
    • Traces, stochastic, state-machines, execution...

<table>
<thead>
<tr>
<th></th>
<th>High-Level</th>
<th>Low-Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detail</td>
<td>Message</td>
<td>Instruction</td>
</tr>
<tr>
<td>Fundamental Objects</td>
<td>Message, Compute block, Process</td>
<td>Instruction, Thread</td>
</tr>
<tr>
<td>Static Generation</td>
<td>MPI Traces, MA Traces</td>
<td>Instruction Trace</td>
</tr>
<tr>
<td>Dynamic Generation</td>
<td>State Machine</td>
<td>Execution</td>
</tr>
</tbody>
</table>

Multiscale Parameters
• Design space includes much more than simple performance
• Create common interface to multiple technology libraries
  – Power/Energy
  – Area/Timing estimation
• Make it easier for components to model technology parameters
Case Study: Reliability vs. Power
Hidden cost of DVFS

• Dynamic voltage/frequency Scaling reduces power
  • ➔ Reduces temperature
  • ➔ Causes thermal cycling
  • ➔ Reduces reliability

• Need
  – Algorithms to balance temperature, lower power, & maintain performance
  – Arch: Sensors and feedback
  – Runtime: Scheduler changes
  – App: Awareness

(Coskun 2011)
Open Simulator Framework

Simulator Core will provide...
- Power, Area, Cost modeling
- Checkpointing
- Configuration
- Parallel Component-Based Discrete Event Simulation

Components
- Ships with basic set of open components
- Industry can plug in their own models
  - Under no obligation to share

Open Source (BSD-like) license
- SVN hosted on Google Code
Need to Push Simulation into New Areas

• Simulation & Modeling historically for architects
• Need new audiences
• Tools for guiding investment
• Tools for procurement
  – Ideally, trusted simulation models would be required part of bid
• Tool for application developers
Simulation as CoDesign Tool

• Simple tool using fast simulators to provide fast feedback to application and algorithm developers
• Input: Application, Machine Model
• Output: Power/Energy, Performance, MTBF, Cost
• Allows design space exploration for future architectures
• Makes power 1st class concern
Ideal Model (a Model of Modeling)

- Problem translated to some representation of software
- Some generic (but sufficiently detailed) specification of hardware
- Universal model to provide performance, energy, and cost
- Run once for each execution model/architecture
- Pick the best
- Build it
The Problem

- What level of proof would you need to agree that your favorite programming model / execution model / HW component is not worth further investment?
- Proving a negative
Less Ideal, More Realistic

• Iterative series of models
• Feedback between models
• Refinement of models and of HW&SW representations
• Co-Design through modeling

• Goal should not be a universal artifact (e.g. a generic automated model)

• Goal should a process/methodology
<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
<th>Tools</th>
</tr>
</thead>
<tbody>
<tr>
<td>High Level</td>
<td>Back of the Envelope</td>
<td>SST/ Macro</td>
</tr>
<tr>
<td></td>
<td>High level state machines + Abstract network, memory models</td>
<td></td>
</tr>
<tr>
<td>Medium Level</td>
<td>Processor, Network &amp; Memory: one of three is detailed, others are simplified.</td>
<td>Disksim, Booksim</td>
</tr>
<tr>
<td>Behavioral Model</td>
<td>Cycle accurate models of processor, network, and memory</td>
<td>M5, IRIS/ Phoenix, DRAMSim</td>
</tr>
<tr>
<td>Mixed Simulation</td>
<td>C/C++ Software simulator + HDL or FPGA</td>
<td>ModelSim FLI, FAST</td>
</tr>
<tr>
<td>Hardware Prototype</td>
<td>Packaged die on a board</td>
<td></td>
</tr>
</tbody>
</table>
Thanks!