Predicting the Impact of Fat-tree Configurations on Applications and Multi-job Workloads


nikhil@llnl.gov

Summer of CODES ♦ July, 2017
Computational vs communication resources

• Capabilities of networks are improved commensurately with computational power

• Otherwise, deficiency in one can offset the benefits of the other
Computational vs communication resources

• Total budget available for procurement and operation of supercomputers is fixed and limited
  • Need to strike a balance between expenditure on the two types of resources
Computational vs communication resources

• Increasing cost of networks presents a challenge
  • Desirable to buy as much computation resources as possible
  • Without letting the reduced network capabilities affect science throughput
Imperative to develop prediction methodologies

- To evaluate the performance of workloads on future alternatives

- We focus on *relatively* detailed simulations based prediction methods
  - Existing systems offer limited options for such an evaluation
  - Difficult to capture dynamic behavior and congestion effects caused by real world applications using simple models

- Can be leveraged in other use cases too
  - Performance analysis on current systems
  - Preparing applications for next-generation systems
  - Design of future networks
Need for simulation tools that can…

• Be practically deployed at full system scale
  • Because communication effects are prominent at large scale

• Be used for production applications and libraries
  • As synthetic patterns fail to capture complexities of real world codes

• Be validated to build confidence in the results
  • Otherwise, what’s the point!
**TraceR-CODES simulation framework has been shown to scale well**

- Be practically deployed at full system scale

---

Partitioning low-diameter networks to eliminate inter-job interference. IPDPS’17.

Enabling parallel simulation of large scale HPC network systems. TPDS’17.

Evaluating HPC networks via simulation of parallel workloads. SC’16.

Modeling a million-node dragonfly network using massively parallel discrete event simulations. SCC’12.
Focus and contributions of this work

- Improve the TraceR-CODES framework to reduce the gap between simulations and real world scenarios
- Study the impact of available fat-tree network configurations on applications and multi-job workloads
- Explore the effect of changing runtime characteristics in simulations, e.g. computation to communication ratio
Background: fat-tree networks

- Used in small clusters as well as large systems, e.g. Cab, Stampede
  - To be used in Sierra and Summit

- Tree-based topology, practically deployed as a folded Clos network
Fat-tree networks: available with a few configurable choices

a) Single rail single plane (full)
b) Single rail single plane (tapered)
c) Dual rail single plane
d) Dual rail dual plane
Applications and workloads

• For typical HPC data centers:
  • a significant fraction of time is spent in running a few applications
  • job sizes are predominantly between 5-30%

• We focus on such a representative set of production applications and libraries, many of which are routinely run at Lawrence Livermore National Laboratory

• For each application, we worked with developers/users to find a representative real world problem (details in the paper)
## Applications: a quick look

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>P2P</th>
<th>Collectives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypre</td>
<td>parallel solver</td>
<td>light</td>
<td>light allreduce, bcast</td>
</tr>
<tr>
<td>Atratus</td>
<td>radiation diffusion</td>
<td>light</td>
<td>light allreduce, bcast</td>
</tr>
<tr>
<td>Mercury</td>
<td>monte carlo</td>
<td>light</td>
<td>light a2a, gather; heavy allreduce, bcast, reduce</td>
</tr>
<tr>
<td>MILC</td>
<td>QCD</td>
<td>heavy</td>
<td>light allreduce</td>
</tr>
<tr>
<td>ParaDiS</td>
<td>dislocation dynamics</td>
<td>medium</td>
<td>light allgather; heavy allreduce</td>
</tr>
<tr>
<td>pF3D</td>
<td>laser plasma</td>
<td>heavy</td>
<td>light allreduce, bcast, allgather; heavy a2a</td>
</tr>
<tr>
<td>Qbox</td>
<td>DFT</td>
<td>heavy</td>
<td>light a2a; heavy allreduce, bcast, reduce</td>
</tr>
</tbody>
</table>
Simulations with TraceR-CODES

- TraceR simulates application and runtime environment using traces on top of CODES, a framework for network simulation.
- In development since 2014
- [https://github.com/LLNL/tracer](https://github.com/LLNL/tracer)

Trace collection for codes:
- automatic, low-overhead

Multi-job replay of traces:
- dynamic, configurable

Traffic flow on many networks:
- detailed, parameterizable

Discrete event simulation:
- parallel, scalable
Salient features of TraceR-CODES

• Provides topology variants of practical importance: e.g. user-defined dragonfly, incomplete/tapered/multi-rail/multi-plane fat-trees, Slim Fly, express-mesh/HyperX

• Simulation of MPI internals: eager/rendezvous protocols, collective algorithms such as bruck, multi-pair, ring for different operations

• Incorporates several factors that affect performance in real scenarios, e.g. NIC scheduling policies, congestion control mechanisms, routing schemes, software overheads, etc.

* Green text shows features added as part of this work
Salient features of TraceR-CODES

- **Supports standardized OTF2-trace format**: makes it easy to use traces from third-party tools.
- **Simultaneous multi-job simulations with user-defined mapping and job placement**.
- **Compute region scaling**, message size scaling, etc. during simulation.
- **Use of optimistic discrete event simulations (ROSS)** that enables good strong scaling in many scenarios.

* Green text shows features added as part of this work. 
Predicted vs Observed runtime

(a) Ping-pong

(b) Ping-pong relative error

(c) All-to-all

(d) 3D Stencil
Predicted vs Observed runtime

### Atratus (strong scaling)

- **Observed Runtime**
- **Predicted Runtime**

![Graph showing predicted vs observed runtime for Atratus](image)

- Time (seconds) vs Core count (32, 64, 128, 256, 512, 1K)
- Observed: -0.4%, -0.8%, -1.9%, -0.8%, -2.6%
- Predicted: -0.8%, -0.8%, -0.8%, -0.8%, -0.8%

### pF3D

- **Observed**
- **Predicted**

![Graph showing predicted vs observed runtime for pF3D](image)

- Time (seconds) vs Different configurations (A, B, C, D, E, F)
- Observed: -0.5%, 0.0%, 0.0%, -0.6%, 3.5%
- Predicted: 3.1%, -0.5%, 0.0%, -0.6%, 3.5%
Consider ~4,500 node fat-tree systems
And a workload with representative applications

<table>
<thead>
<tr>
<th>Config</th>
<th>Link bandwidth</th>
<th>#rails</th>
<th>#planes</th>
<th>Tapering</th>
</tr>
</thead>
<tbody>
<tr>
<td>SR-EDR</td>
<td>100 Gbps</td>
<td>1</td>
<td>1</td>
<td>1:1</td>
</tr>
<tr>
<td>DRP-EDR</td>
<td>100 Gbps</td>
<td>2</td>
<td>2</td>
<td>2:1</td>
</tr>
<tr>
<td>DRP-EDR</td>
<td>100 Gbps</td>
<td>2</td>
<td>2</td>
<td>1:1</td>
</tr>
<tr>
<td>SR-HDR</td>
<td>200 Gbps</td>
<td>1</td>
<td>1</td>
<td>1:1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>P2P</th>
<th>Collectives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypre</td>
<td>parallel solver</td>
<td>light</td>
<td>light, allreduce, bcast</td>
</tr>
<tr>
<td>Atratus</td>
<td>radiation diffusion</td>
<td>light</td>
<td>light, allreduce, bcast</td>
</tr>
<tr>
<td>Mercury</td>
<td>monte carlo</td>
<td>light</td>
<td>light a2a, gather; heavy allreduce, bcast, reduce</td>
</tr>
<tr>
<td>MILC</td>
<td>QCD</td>
<td>heavy</td>
<td>light allreduce</td>
</tr>
<tr>
<td>ParaDiS</td>
<td>dislocation dynamics</td>
<td>medium</td>
<td>light allgather; heavy allreduce</td>
</tr>
<tr>
<td>pF3D</td>
<td>laser plasma</td>
<td>heavy</td>
<td>light allreduce, bcast, allgather; heavy a2a</td>
</tr>
<tr>
<td>Qbox</td>
<td>DFT</td>
<td>heavy</td>
<td>light a2a; heavy allreduce, bcast, reduce</td>
</tr>
</tbody>
</table>

ScoreP, BigSim: Trace collection for codes: automatic, low-overhead
TraceR: Multi-job replay of traces: dynamic, configurable
CODES: Traffic flow on many networks: detailed, parameterizable
ROSS: Discrete event simulation: parallel, scalable

Predicting the Impact of Fat-tree Configurations on Applications and Multi-job Workloads
ParaDiS: computation heavy, low impact of network
pF3D: higher bandwidth is good, but tapering is low impact
Qbox: bandwidth and tapering matter

![Graph showing execution times for different configurations and workloads. The graph compares execution times across four configurations: SR-EDR, DRP-EDR, DRP-T-EDR, SR-HDR, DR-T-HDR, and DR-HDR. The y-axis represents execution time in seconds, ranging from 1 to 10, and the x-axis represents workloads 2x, 10x, 25x, and 50x. The results show significant differences in execution times, with percentage improvements from 9% to 67% across different configurations and workloads.]
Workload with 17 jobs (256 nodes each)

(a) Application performance in multi-job workloads

(b) Impact of inter-job interference
Multijob workload: aggregate output

Total performance of workloads

<table>
<thead>
<tr>
<th>Network Option</th>
<th>Normalized Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>SR-EDR</td>
<td>1</td>
</tr>
<tr>
<td>DRP-T-EDR</td>
<td>1.1</td>
</tr>
<tr>
<td>DRP-EDR</td>
<td>1.79</td>
</tr>
<tr>
<td>SR-HDR</td>
<td>1.4</td>
</tr>
<tr>
<td>DR-T-HDR</td>
<td>1.67</td>
</tr>
<tr>
<td>DR-HDR</td>
<td>2.81</td>
</tr>
</tbody>
</table>

Relative cost of different network options

13.5% for SR-EDR
33.5% for DRP-T-EDR
34.7% for DRP-EDR
36.7% for SR-HDR
53.7% for DR-T-HDR
Concluding remarks

• We show that it is feasible to make predictions about future networks for complex scenarios that can assist in procurement process

• Good accuracy for micro-benchmarks and applications is shown for the TraceR-CODES simulation framework

• For the networks considered, only when significant increase in computational capacity is made, effect of configurations is noticeable

• Dual rail EDR and single rail HDR are able to provide a major fraction of performance benefits possible with changing configurations
Predicting the Impact of Fat-tree Configurations on Applications and Multi-job Workloads


nikhil@llnl.gov

Summer of CODES ♦ July, 2017