Use of Parallel and High Performance Computing in Power Systems Simulations and Its Challenges

John Craig Muller, Rajendra Singh and Eric Mellick
PSCAD Development Group
Manitoba HVDC Research Centre
Winnipeg, MB, Canada
Outline

• Background
• Situation
• Motivation for improvements
• Experimental developments
• Future targets
Background

• Engineers have always strived for methods for speeding-up the simulation computation time
  – by writing efficient algorithms
  – or, often have compromised and lived with coarse grained analysis due to large computation times

• Multiple-core computers at Engineer’s disposal
  – have added to frustration
  – simulation software haven’t caught-up with it yet
Background

The first 20 years

- For the first two decades of the software all implementations considered only a single process PSCAD and a single process EMTDC. This is for all versions up until v4.4 in 2011.
- Quad core computers were not in general use until 2008.

Frequency of communication is very low. Messages are exchanged only at the beginning and end of a run.
Motivation

Need for speed...

- PSCAD has been performing sequential computing for quite some time (since 1993).
- Lately, technology and developments have steered PSCAD to utilize parallel and distributed computing models of computation
  - Recent developments enable
    - Task Parallelism (MPMD)
    - Data Parallelism (SPMD)
  by using Message Passing over TCP/IP Inter-process communication (IPC) techniques
2011

- Utilizing existing implementations, changes were made to allow the application to accept multiple simulations to execute at the same time. All projects had to be independent and unique.
- Each had to be launched on an individual basis.

Frequency of communication is very low. Messages are exchanged only at the beginning and end of a run.
2012

- Data parallel approach allows for one root, or master project, to control multiple slave projects, where both master and slaves must be part of the same Simulation Set.

Frequency of communication is very low. Messages are exchanged only at the beginning and end of a run.

Creates a parallel version of the multi-run interface
Background

2015 (Data Parallel Simulation)

- Significant refactoring of the run interface enabled the ability to execute coordinated sets of simulations.
- The launch system was expanded to enable a single project to spawn multiple copies of itself to execute under different conditions.

Frequency of communication is very low. Messages are exchanged only at the beginning and end of a run.

Creates a parallel version of the multi-run interface
Background

**2015 (Task Parallel Simulation)**

- A single electric network may be split so that each electric subsystem is represented by a separate project, and thereby runs using separate processes.
- Each process is linked together via a *Communication Interface* to form a cohesive simulation that is run from within a single workspace.

Frequency of communication is *high*. Messages are exchanged every time step.
Using the transmission line interface, multiple PSCADs can be tethered together to co-simulate a larger network on separate computing platforms.
Using the transmission line interface, multiple PSCADs can be tethered together to co-simulate a larger network on separate computing platforms.
PSCAD v4.6
Parallel and Distributed computing software

• Task Parallel and Data Parallel approach
  – Utilizes all cores on Localhost
  – Extends it to utilizing LAN

Benefits
  – Performance is increased many fold
  – Task Parallel
    • Orders of magnitude improvements (large cases – Province wide power network)
    • Very large networks handled easily
  – Data Parallel
    • Parametric studies that took months can be done in days or hours
    • Fine grained studies could be performed in reasonable time
Situation

Need for speed...

- To run simulations with greater fidelity in the same time or less, we need
  - Parallel computing capabilities
  - Less interference from operating system actions
  - Many cores on a single CPU with high clock speed
    - Typical desktop has 4 or 6 cores per CPU with one or two slots
    - Blade computers provide many cores, with 2 to 4 slots. (Slower Clock)

- Task Parallel
  - High frequency of communication, not a high volume.
  - Communication backplane is standard TCP/IP network
    - Relatively high communication latency on across LAN.
    - Relatively low communication latency on local host, still not so efficient.
Several equal sized IEEE bus system kernels were connected using transmission-lines to create a large hypothetical bus system

- for e.g. 273 bus system created by connecting 7 IEEE 39 bus system kernels
- Experiments were performed with IEEE 14, 39, 78, 118 and 300 bus systems
  - largest hypothetical bus system being 300 X 7 = 2100 bus system
Experimental setup – impact of communication and latency

IEEE 39 Bus

[Diagram of IEEE 39 Bus network with t-line highlighted]
Experimental setup – impact of communication and latency

Contrived balanced system

Synthetic system designed to measure communication performance
Impact of communication and latency on Task Parallel

IEEE 14 Bus

IEEE 39 Bus
Impact of communication and latency on Task Parallel ...
Impact of communication and latency on Task Parallel ...

IEEE 300 Bus

Execution Time(s)

- Execution Time (s)-single core
- Total Execution Time (s)-multiple core
- Communication Overhead

# of 300 Bus Systems Used

Speed-up

- Speed-up Factor

# of 300 Bus Systems Used
Impact of Communication and Computation Grain on Task Parallel ...

\[
\left( \frac{T_{sol}}{X} + (T_{comm} \times Z)Y \right) \approx T_{total}
\]

and

\[
(X - 1) \leq Y \leq \left( \frac{X(X - 1)}{2} \right)
\]

Where:
- \( T_{sol} \) is computation time per sub-network
- \( T_{comm} \) is communication time per message
- \( X \) is number of decoupled sub-networks
- \( Y \) is number of connections
- \( Z \) is number of transmission lines
- \( T_{total} \) is the total computation time of entire network
- \( \left( \frac{X(X - 1)}{2} \right) \) is the maximum edges in a undirected graph
Motivation
For further improvements

• The key for performance gain is smaller computation grains
  – That is why we break larger cases into smaller cases
  – But granularity of the case is limited by the communication speed
    • If communication speeds are higher we can also speed-up smaller cases
• Clearly, communication becomes a bottleneck with smaller size of cases.
  – 78 Bus is a significant size
    • Can this be reduced?
If we only have TCP/IP as a communication method, speed is heavily impacted by transport across nodes.

- Communication latencies are less within a node (20 us on TCP/IP) – Can we reduce this?
- Communication latencies over multiple nodes are (230 us on TCP/IP) – Can we reduce this?

We would like to avoid transport, but we do not have enough cores in a single compute node to fit all pieces of a simulation!
Motivation...

Multiple inter-connected multi-core

Assumptions

- It is possible to achieve extremely fast communication speeds – *both local-host and inter-host*
- It is possible to break a large simulation into 100 smaller sub-systems
  - But typical desktops have only 12 or less cores
    - *Impossible to include all sub-systems*
  - We *need* multiple compute nodes connected to each other.

Will the combination of super-computing network hardware give us this capability?
High Performance Computing (HPC)

- The current design supports *Parallel Computing* through Task and Data Parallel approaches
  - Achieving performance figures never seen before in offline EMT tools.
  - But, we still use standard TCP/IP as communication backplane

- Further, performance can be improved using HPC methods.
  - Communication using a *Shared Memory* software model.
  - Communication using *InfiniBand* ultra fast networking hardware. Fourteen Data Rate (FDR), 14.0625 Gbps
  - A hybrid solution using a combination of both.

*On going experimental developments...*
Experimental Test Rig

TITANUS (Quad-Slot) 64x Core
Experimental Test Rig

P900 Single Compute Node (Dual-Slots) 40x Core
Experimental Test Rig

InfiniBand Switch/Cables and FDR Host Adaptors
Experimental Test Rig
Shared Memory Model

- Multiple processes share memory blocks in the *local system*
  - Inter-process communication is fast due to low memory access latencies
  - Methods cannot communicate between nodes.
  - Processes can communicate using a lightweight trade mechanism.
  - Channels are multiplexed into and out of shared storage.

Communication latencies are in the range of 0.001 to 1us
Vectored I/O / Scatter-Gather

Buffer

Buffer

Buffer

Communication Channel

Gathering Writes

Scattering Reads

Buffer

Buffer

Buffer
Communication IPC

Memory IPC
Read/Write Control
Application level semaphores

Network Direct IPC

Read/Write Control
Recv CQ
Send CQ

Send CQ
Recv CQ

P1

P2
Remote Direct Memory Access

- **Remote**
  - Memory data transfers *between nodes in a network*

- **Direct**
  - Operating System Kernel by-passed in transfers
  - Transfer off-loaded onto Network Card processor.

- **Memory**
  - Transfers between user space application’s virtual memory
  - Zero extra copying or buffering

- **Access**
  - send, receive, read, write, atomic operations
Motivation…

Hybrid Solution

- Shared memory communication latencies are low
  - *Varies between 1 – 2 us*
- RDMA communication latencies are low
  - In the range of *1 – 3 us*

<table>
<thead>
<tr>
<th>Packet Size</th>
<th>Protocol</th>
<th>Localhost</th>
<th>Inter-machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 B</td>
<td>IB</td>
<td>0.9 us</td>
<td>2.9 us</td>
</tr>
<tr>
<td></td>
<td>TCP</td>
<td>20.0 us</td>
<td>230.0 us</td>
</tr>
<tr>
<td></td>
<td>PACS</td>
<td>0.9 us</td>
<td>N/A</td>
</tr>
<tr>
<td>8 KB</td>
<td>IB</td>
<td>4.7 us</td>
<td>7.3 us</td>
</tr>
<tr>
<td></td>
<td>TCP</td>
<td>20.0 us</td>
<td>400.0 us</td>
</tr>
<tr>
<td></td>
<td>PACS</td>
<td>1-2 us</td>
<td>N/A</td>
</tr>
</tbody>
</table>
RDMA Transfer Map

Interoperability Laboratory & Computer Science Department University of New Hampshire
InfiniBand Speed

Performance of InfiniBand

- Port to Port communication latency is 0.19 us using the IB fabric with node-node latency around 1-2 us.
- TCP/IP can range between 20-200 us
- The performance tests show that messages of size 64 bytes exchanged between server and client over IB network using RDMA gives an average latency of ~2 us

Performance verified on a Mellanox 2-node IB cluster using RDMA
Advanced Communication Fabric
Multi-Node Fabric Architecture

Switched Fabric Architecture for De-Centralized Computing Cluster
Experiment Split DC Link (all-in-one)

Monopolar HVDC Link representing a balanced pairing
On the server side, the transmission line modeling is included. For this reason the actual line configuration is included.
On the client side the distributed line model is not modeled since this is already taken into consideration on the server side.
A SIMPLE AC SYSTEM

A 230 kV transmission line system with a passive load. This demonstrates the use of the single line Bergeron model of a transmission line directly connected with the sending and receiving ends as opposed to subpages in simpleac.psc and simpleac_sld1.psc. Double click on FLAT230 to see the basic parameters of the transmission line, go to ‘Edit’ to see more.

The sending end currents are measured on the transformer secondary windings inside the transformer component.

A timed phase C to ground fault is applied at 0.25 secs and lasts 50 msec. The timed breaker logic is set to trip at 0.26 secs and reconnect at 0.31 secs.
Experimental Single Split

Single Source WTG

[Diagram of DFIG Wind Farm Average Model with values P = 4.655, Q = 2.062, V = 1.108, 550 [MVA], Q = 6.1148, V = 1.108, and connections to TLine_19 and RRL 2]
Experimental Mixed VG Network
Experimental Wind Park
Each machine is modelled in full detail with complete control and protections systems. Wind speed is converted to torque on them machine based on standard specifications. Back to back converters are modeled in full detail.
## Experimental Results

### 1 Million Communications

<table>
<thead>
<tr>
<th>Case Name</th>
<th>All-in-one Single Process (sec)</th>
<th>Protocol</th>
<th>Single Node Localhost (sec)</th>
<th>Multi-Node Across Hosts (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple AC Network</td>
<td>4</td>
<td>IB</td>
<td>--</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PACS</td>
<td>4</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TCP</td>
<td>15</td>
<td>252</td>
</tr>
<tr>
<td>P2P HVDC (CIGRE Benchmark)</td>
<td>21</td>
<td>IB</td>
<td>--</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PACS</td>
<td>15</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TCP</td>
<td>30</td>
<td>242</td>
</tr>
<tr>
<td>Source to Wind Park Average Model</td>
<td>48</td>
<td>IB</td>
<td>--</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PACS</td>
<td>44</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TCP</td>
<td>60</td>
<td>236</td>
</tr>
<tr>
<td>14 BUS to Wind Park Average Model</td>
<td>82</td>
<td>IB</td>
<td>--</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PACS</td>
<td>51</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TCP</td>
<td>62</td>
<td>232</td>
</tr>
<tr>
<td>39 BUS to 118 BUS</td>
<td>320</td>
<td>IB</td>
<td>--</td>
<td>188</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PACS</td>
<td>210</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TCP</td>
<td>224</td>
<td>332</td>
</tr>
</tbody>
</table>
## Experimental Results

1 Million Communications

<table>
<thead>
<tr>
<th>Case Name</th>
<th>All-in-one (sec)</th>
<th>Protocol</th>
<th>PSCAD Localhost (sec)</th>
<th>PSCAD Across Hosts (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wind Park Type 3 DFIG x10</td>
<td>679</td>
<td>IB</td>
<td>--</td>
<td>130</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PACS</td>
<td>117</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TCP</td>
<td>195</td>
<td>1570</td>
</tr>
<tr>
<td>Wind Park Type 3 DFIG x22</td>
<td>1017</td>
<td>IB</td>
<td>--</td>
<td>214</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PACS</td>
<td>160</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TCP</td>
<td>319</td>
<td>1827</td>
</tr>
<tr>
<td>Wind Park Type 3 DFIG x38</td>
<td>1850</td>
<td>IB</td>
<td>--</td>
<td>270</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PACS</td>
<td>259</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TCP</td>
<td>587</td>
<td>2127</td>
</tr>
<tr>
<td>2500 Bus Transmission Network (40 splits)</td>
<td>18206</td>
<td>IB</td>
<td>--</td>
<td>410 (x44)**</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PACS</td>
<td>604 (x30)**</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TCP</td>
<td>1520 (x12)</td>
<td>838 (x21)</td>
</tr>
</tbody>
</table>
MCL Application and View

Markov Chain Clustering Algorithm (MCL) applied to 40 Way Concurrent EMTDC.
150 Way Split of Across 4 Nodes
Future Targets

Breaking large power system simulation into smaller tasks is difficult

- Inherent inter-dependence in mesh configuration
- Breaks reduce computational burden per process, but increase number of processes and interconnects
- Optimally mapping large number of broken apart sub-networks onto a fixed number of processors is challenging
- Intelligent graph theory methods may be required to
  - Optimize loading on processes
  - Optimize the use of processors
Conclusions

- High Performance Communication Fabric Advantages
  - Conventional TCP/IP kernel overheads limit scaling.
  - Zero mutex IPC with shared ring buffers offers exceptionally low latency.
  - IP over IB offers nearly the same performance as local shared memory.
  - Processor affinity can be maintained using non-blocking methods.
  - Separating the communication fabric provides portability
  - Co-simulation can be achieved by matching message protocols.
  - Sub-synchronous interactions studies possible with full fidelity simulations.
Thank you!