Computational workflow of the LZ experiment

Venkitesh Ayyar,
Computation Research Division,
Lawrence Berkeley National Laboratory

CHEP 2019, Adelaide,
November 4-8, 2019

Joint work with W. Bhimji, M.E. Monzani, A. Naylor, S. Patton, Q. Riffard, C. Tull
For the LUX-ZEPLIN collaboration
Dark matter

Evidence of matter beyond SM:

★ Galaxy rotation curves
★ Cosmic Microwave Background
★ Gravitational lensing

Estimated to constitute up to 85% of all matter
Different types of dark matter proposed
One of them is WIMP (Weakly Interacting Massive Particle)

Some current experiments: LZ, XENONnT, PandaX-4T

Figure from Wikipedia
LZ experiment  LUX + ZEPLIN

Tank containing 7 tonnes liquid Xe

- Direct detection experiment
- Looking for Weakly Interacting Massive Particles (WIMPS)

Detect scintillations in liquid Xe

Sanford Underground Research Facility, South Dakota
LZ collaboration

Collaboration of 37 institutes:
US, UK, Portugal, Russia, Korea

Data centers in both UK and US

Expected timelines:
Experimental installation: 2019
Data collection start: 2020

Data Challenges
MDC2: September 2018
MDC3: October 2019
The Code

Input
Detector configuration,

Time consuming part

BACCARAT
Simulates events using Geant4

DER
Creates sensor data for the given events

LZap
Event reconstruction (Gaudi)

in C++

Runs much faster
Computing workflow

Job submission tools

Psquared Production processing Manager

PostgreSQL database Stores details of job submission and execution

RabbitMQ queuing system

GUI showing status of running jobs

Workers Running jobs on nodes

Scripts that
• Pick up jobs from the queue
• Run jobs on compute nodes
Job submission

**CLI (Command Line Interface)**
- Python scripts to submit jobs
- Tools for debugging failed jobs for re-submission
- Interacts with Postgres Database via restful-interface
- Intended for use by production team

**Web-based Graphical User Interface**
- Job Submission Interface developed by UKDC
- Developed for USDC by Andrew Naylor
- Users can request to submit jobs
- Admin approval for submission
Computing resources at US Data Center (NERSC)

### HPC

<table>
<thead>
<tr>
<th>Machine</th>
<th>Details</th>
<th>Total nodes</th>
<th>Cores per node</th>
<th>Memory per node (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edison</td>
<td>Cray XC30</td>
<td>5586</td>
<td>24</td>
<td>64</td>
</tr>
<tr>
<td>Cori-Haswell</td>
<td>Cray XC40</td>
<td>2388</td>
<td>32</td>
<td>128</td>
</tr>
<tr>
<td>Cori-Knl</td>
<td>Cray XC40</td>
<td>9688</td>
<td>68</td>
<td>96</td>
</tr>
</tbody>
</table>

PDSF and Edison decommissioned
Currently only Cori is running
New machine Perlmutter is also HPC, with GPUs

LZ workflow had to be ported to HPC
Reasons to focus on HPC:
- Large investment by DOE in HPC exascale machines
- NERSC focus on large HPC machines
- Memory shared across cores: less chance of memory overload
Improving efficiency

Before September 2018

Avg mem usage ~ 4GB
Jobs per node = 32

High memory usage

Use of profiling and monitoring tools to fix memory leaks

- Restricts number jobs per node
- Increases job-crash incidents
Process monitor

Developed by ATLAS
https://github.com/HSF/prmon

Gives memory usage and other I/O data as a function of time

<table>
<thead>
<tr>
<th>Quantity monitored</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmem</td>
<td>Virtual Memory</td>
</tr>
<tr>
<td>pss</td>
<td>Proportional Set Size</td>
</tr>
<tr>
<td>rss</td>
<td>Resident Set Size</td>
</tr>
<tr>
<td>swap</td>
<td>Swap Size</td>
</tr>
<tr>
<td>utime</td>
<td>User CPU-time</td>
</tr>
<tr>
<td>stime</td>
<td>System CPU-time</td>
</tr>
<tr>
<td>wtime</td>
<td>Wall-time</td>
</tr>
<tr>
<td>rchar</td>
<td>I/O Read (rchar)</td>
</tr>
<tr>
<td>wchar</td>
<td>I/O Written (wchar)</td>
</tr>
<tr>
<td>read_bytes</td>
<td>I/O Read (read_bytes)</td>
</tr>
<tr>
<td>write_bytes</td>
<td>I/O Written (write_bytes)</td>
</tr>
<tr>
<td>rx_packets</td>
<td>Network Received (packets)</td>
</tr>
<tr>
<td>tx_packets</td>
<td>Network Transmitted (packets)</td>
</tr>
<tr>
<td>rx_bytes</td>
<td>Network Received (bytes)</td>
</tr>
<tr>
<td>tx_bytes</td>
<td>Network Transmitted (bytes)</td>
</tr>
<tr>
<td>nprocs</td>
<td>Number of Processes</td>
</tr>
<tr>
<td>nthreads</td>
<td>Number of Threads</td>
</tr>
</tbody>
</table>

- Simple, easy to use
- Gives time map for a lot of quantities
- Lightweight
- Tacks on to pid, gives usage of all spawned processes

### Run the MCprod script
```
shifter $script_loc/../RunMCprod.sh OUTPUT=$output_loc BACCARAT_VERSION=$bac_vers BACCARAT_MAC=$mac_loc BACCARAT_CODE=$bac_code ITEM=$ITEM &
```

### Link the pid to the job above
```
$prmon_loc --pid $! --Filename $prmon_output --interval $interval
```
Further info

Memory usage in MB

Network received bytes

Has its own plotting code
Also supports GNU prof profiling or gperftools profiling
VTune
Developed by Intel

Tool for detailed profiling

• Collects a lot of data
• Runs a lot longer
• Requires full node
Find memory leaks using VTune
Average usage is lower

✓ Code modified, features added
✓ Memory leaks plugged

Peak memory usage for a set of 50 jobs with different seeds

Some jobs still take a lot of memory

Average usage is lower
Memory usage summary for a set of 50 jobs with different seeds on a single node with current version of code

<table>
<thead>
<tr>
<th></th>
<th>September 2018</th>
<th>September 2019</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average memory usage</td>
<td>~ 4 GB</td>
<td>~ 2.3 GB</td>
</tr>
<tr>
<td>Jobs per node</td>
<td>32</td>
<td>50</td>
</tr>
</tbody>
</table>

Memory usage summary for a set of 50 jobs with different seeds on a single node with current version of code.
Summary

• LZ is successfully running workflow on HPC
• Increased efficiency by reducing memory footprint using tools like VTune and Process Monitor
• GUI for job submission

Future Direction

• Data collection in 2020
• Checkpointing of jobs
• Multi-threading for GEANT4
Thank you
Back up slides
Some jobs still take high memory
Other challenges

Need a custom environment for running applications

- Custom environment for running scripts provided by NERSC
- Containers (like Docker)
- No root access necessary

- CERN Virtual File System
- Used to store compiled binaries for use
- Custom installation at NERSC to mount on cray machines