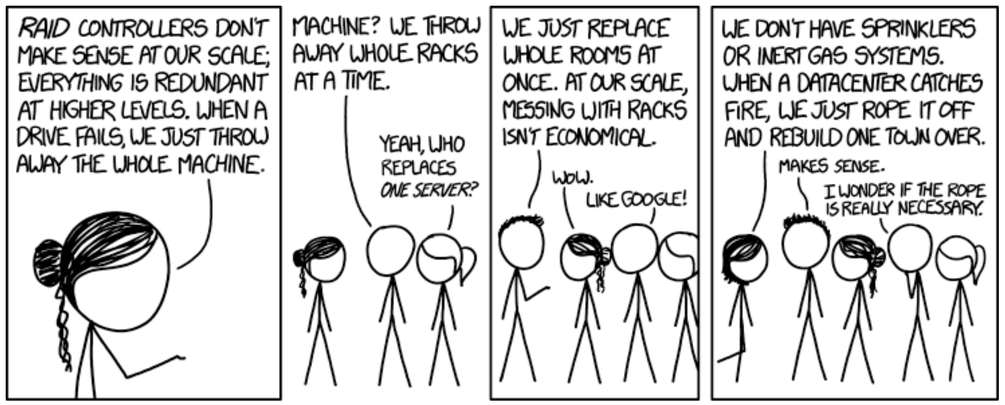
Benchmarking High Performance Computing Architectures with CMS' Skeleton Framework



Jim Amundson, Patrick Gartung, Chris Jones and Elizabeth Sexton-Kennedy

Motivation

- CMS chose to use TBB as it's underlying threading technology, based on evaluations of a number of technologies, on the computing architectures that were available in 2012.

- Computing architectures are changing, moving to many core technologies. We wanted to be sure that the TBB strategy on the new systems still made sense.

- HPCs are a window into future computing architectures. Our access to Cori P1, Mira, and a development KNL give us an opportunity to check our model.

Methodology: Emulation

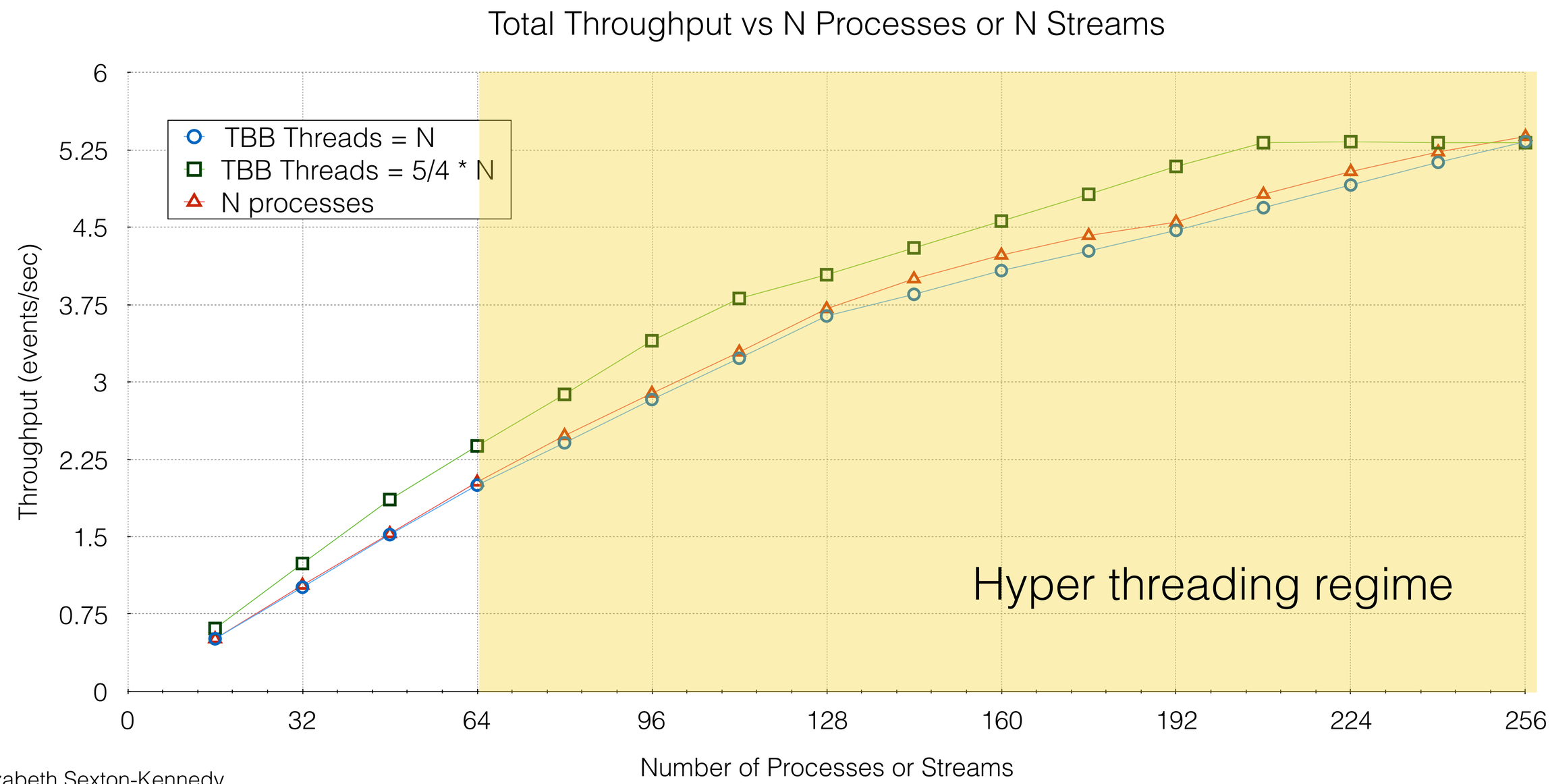
- The full reconstruction application is run. We record both the module timing per event, and the module dependencies in a data file.
  - 489 Producer (Algorithms) with 278 requested only by the OutputModule
  - A data run from 2011 with ~30 interactions per crossing=> 2016 complexity
- Feed that data into the multi-threaded skeleton framework*
  - A BusyWaitProducer is used to emulate the workload of each Producer. It's a loop which sums sin(real x) for enough iterations to match the time spent in the original Producer.
- Compare timing to a simple single threaded framework.

* https://github.com/Dr15Jones/toy-mt-framework

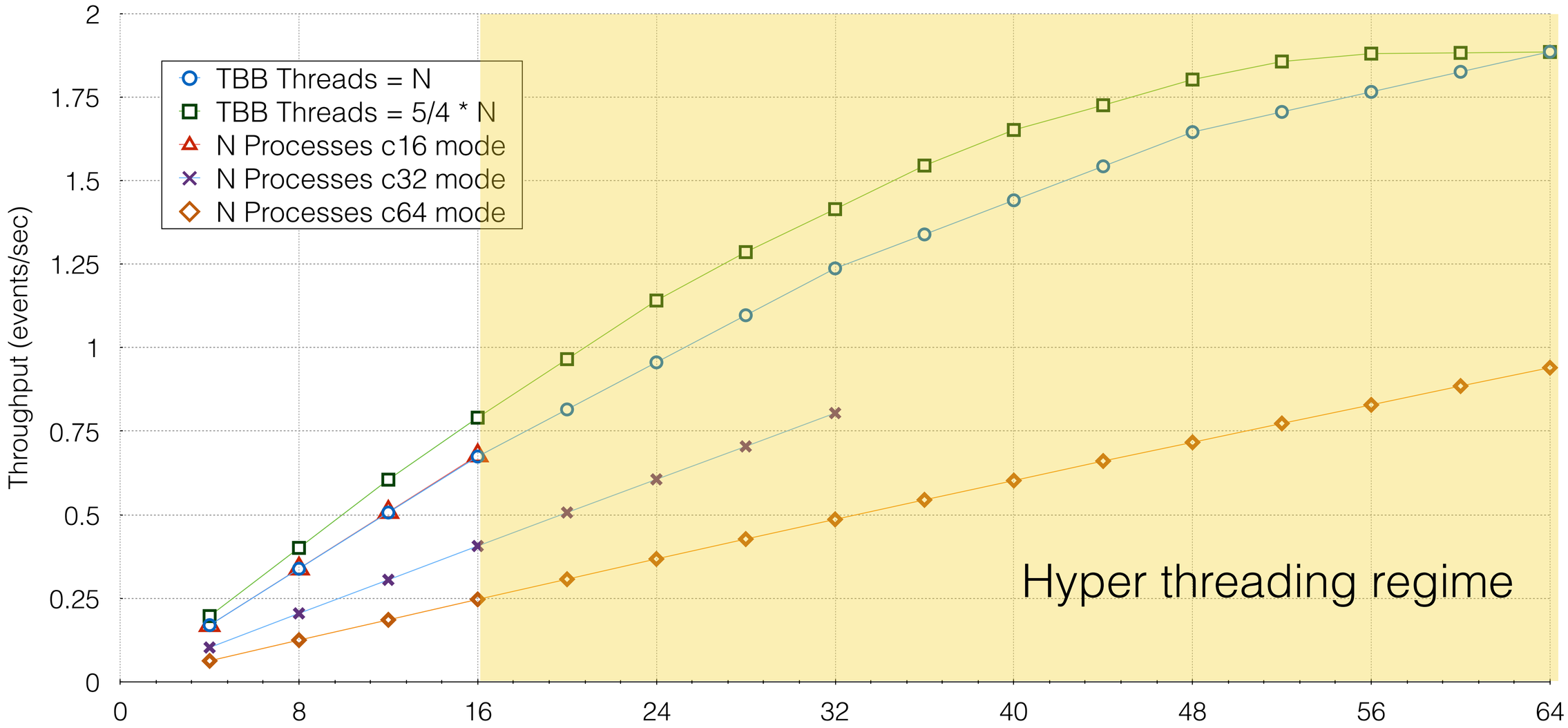
Measurement Machines

- KNL - Thanks to a loan a time from the HPC department of the FNAL SCD
  - XeonPhi 7210: 1.3GHz with 64 cores and 256 hardware threads. 96GB Memory/node or 1.5GB/core
  - PBS job control
- ALCF BG/Q - Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357
  - PowerPC A2 1.6GHz with 16 cores and 64 hardware threads. 16GB Memory/node or 0.25GB/thread
  - COBALT job control
- NERSC Edison - National Energy Research Scientific Computing Center, a DOE Office of Science User Facility, Contract No. DE-AC02-05CH11231
  - Ivy Bridge 2.4GHz with 24 cores per node, 64GB Memory/node or 2.67GB/core
  - SLURM job control

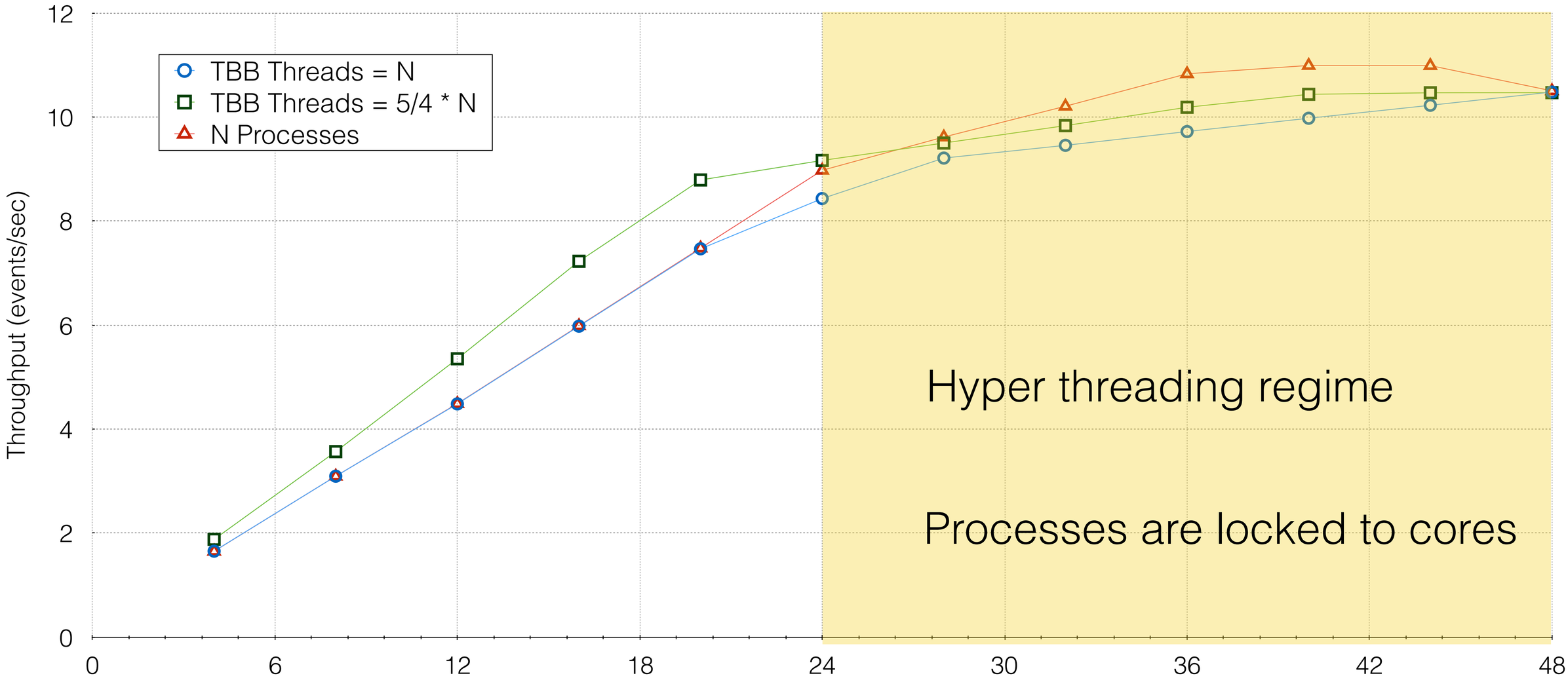
KNL Results


Total Throughput vs N Processes or N Streams
TBB Threads = N
TBB Threads = 5/4 * N
N processes
Hyper threading regime
Throughput (events/sec)
Number of Processes or Streams
0
0.75
1.5
2.25
3
3.75
4.5
5.25
6
0
32
64
96
128
160
192
224
256

ALCF BG/Q Results



NERSC Edison Results



Conclusions & Plans

- TBB scales to supercomputing architectures as expected and PowerPC machines (not so expected).
- Overcommitting threads gives a ~20% throughput benefit without any effort.
- Work has begun testing the full CMSSW stack on the KNL system. Once this is done we can investigate the optimal balance of memory consumption vs. throughput on this architecture.
- See Chris Jones slides for information about the balance of memory and throughput on traditional architectures.

This work was supported throughs the CCE funded by DOE CompHEP