Performance Analysis and Optimizations of the Weather Research and Forecasting (WRF) Model

Dixit Patel[1], Davide Del Vento[2], Akira Kyle[3] , Brian Vanderwende[2], Negin Sobhani[2]

University of Colorado Boulder[1] - National Center for Atmospheric Research[2] - Carnegie Mellon University[3]

University of Colorado Boulder

Carnegie Mellon University

NCAR

UCAR

Contents

Introduction

The Weather Research & Forecasting(WRF) Model

- Mesoscale numerical weather prediction system
- Designed for both atmospheric research and operational forecasting needs.
- Used by over 30,000 Scientists around the world.

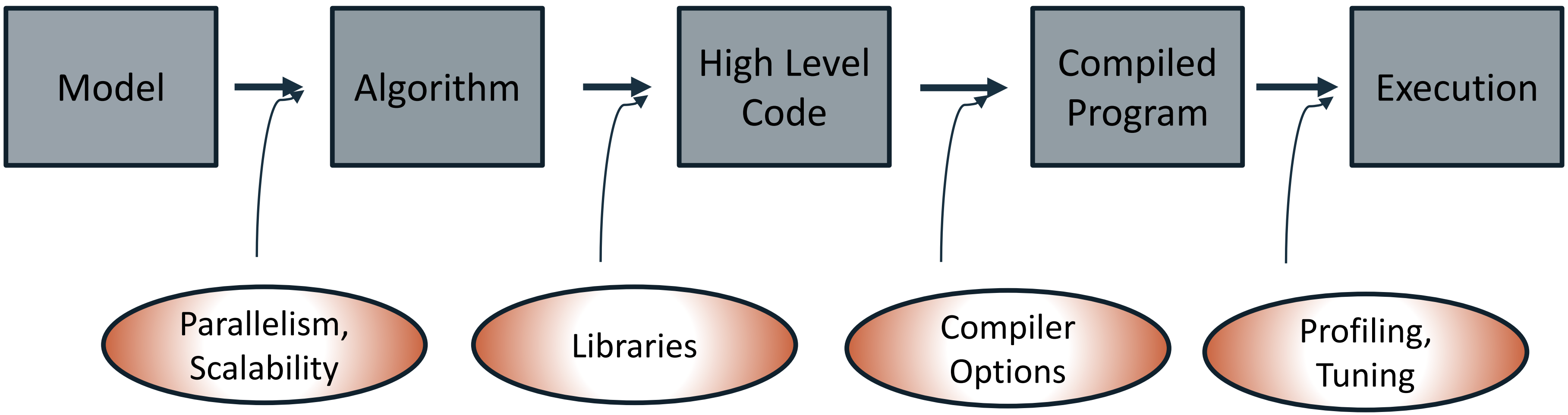
Cheyenne Supercomputer

- 4032 dual-socket nodes
- 2.3-GHz Intel E5-2697v4 processors (Broadwell)
- 18 cores/socket
- 313 TB memory
- Partial 9D Enhanced Hypercube single-plane interconnect topology (25 GBps) - Mellanox EDR InfiniBand

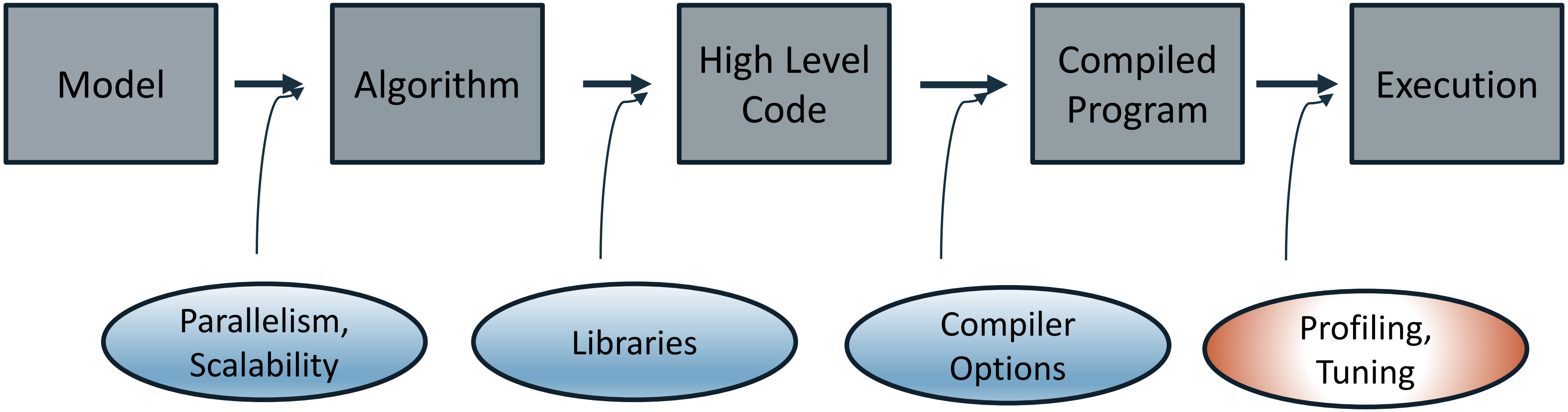
Motivation

Reproduced from : https://cvw.cac.cornell.edu/Optimization/overview

Motivation


Model
Algorithm
High Level Code
Compiled Program
Execution
Parallelism, Scalability
Libraries
Compiler Options
Profiling, Tuning

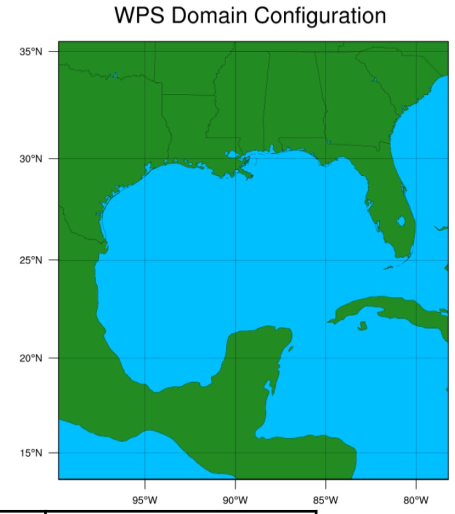
Motivation


Model
Algorithm
High Level Code
Compiled Program
Execution
Parallelism, Scalability
Libraries
Compiler Options
Profiling, Tuning

Scaling WRF

- Motivation
  - How to optimize performance ?
  - Can I solve a problem of a given size in timely manner ?
  - How many core-hours would I need for my problem size ?

Scaling Summary

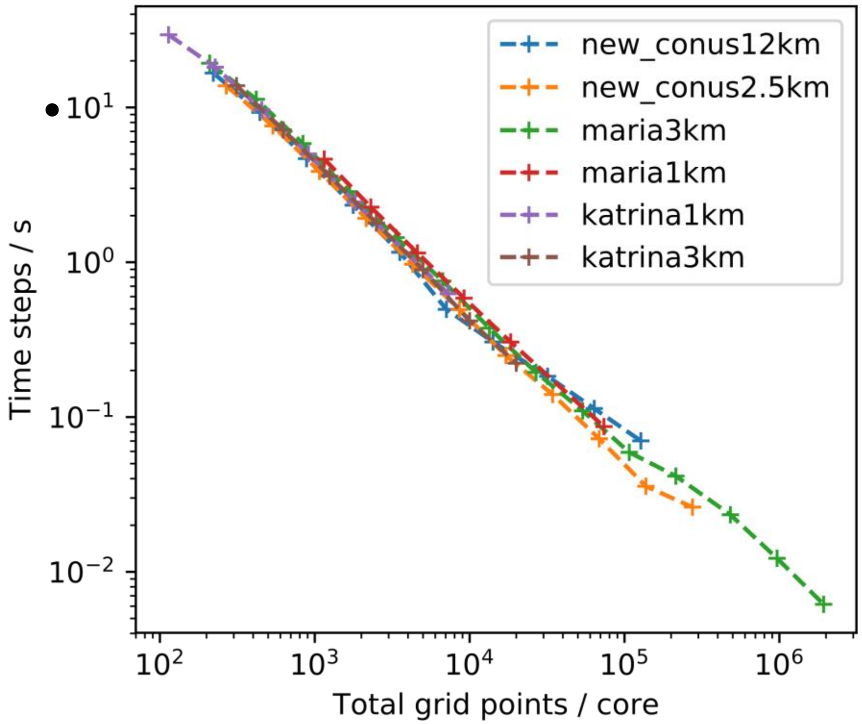


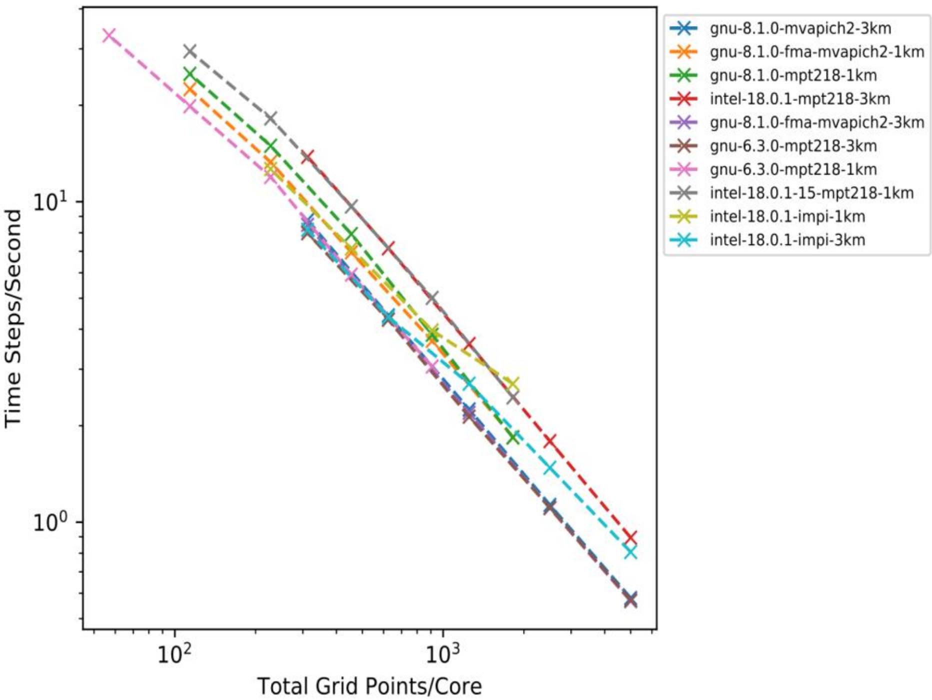
- Compilers : Intel (18,17) , GNU (6.3.0,8.1.0)
- MPI : MPT (2.18) , MVAPICH2, IMPI (2018)
- Case : Katrina 1km, Katrina 3km
- Domain :

Case Resolution	Grid Size	Total Grid Points	Time Steps (sec)	Simulation
Katrina 1km	512 x 512 x 35	262,144	6	12 Hours
Katrina 3km	800 x 900 x 35	720,000	12	12 Hours

- Physics suite = 'Conus' :
- Higher problem size makes the model unstable (CFL violations)

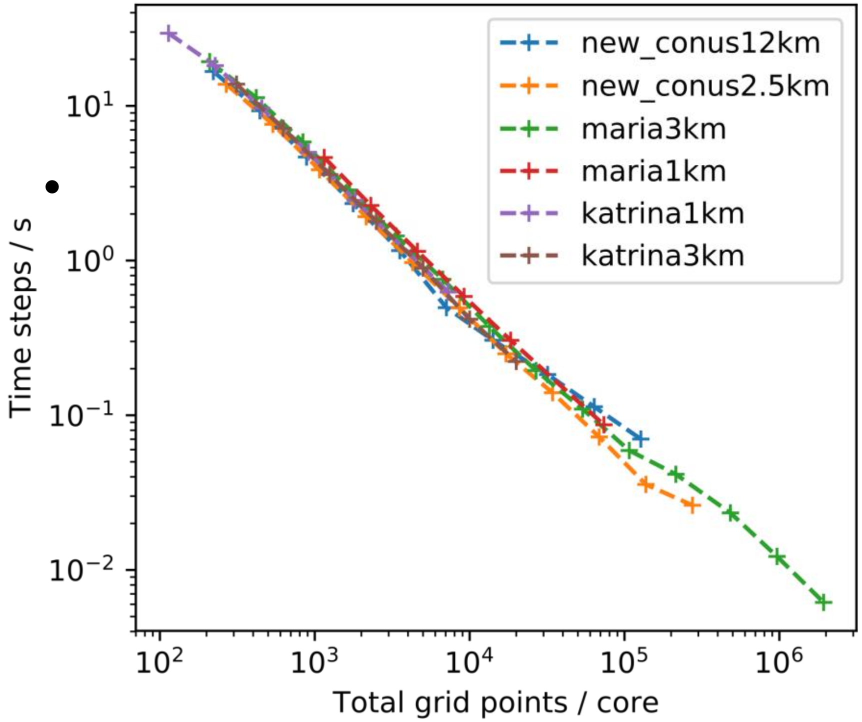
Scaling Summary



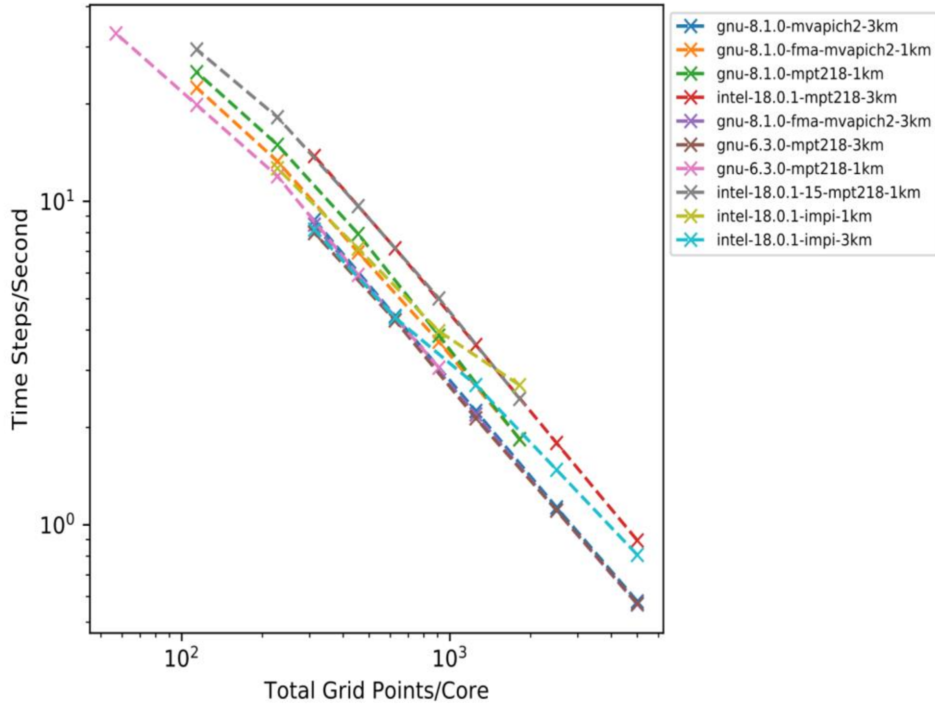


- All cases
- Katrina Cases

Scaling Summary



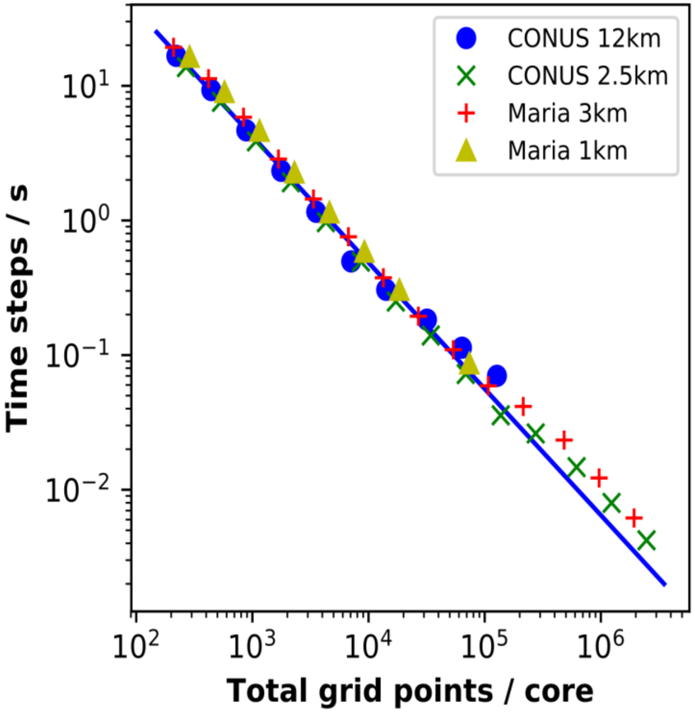
All cases



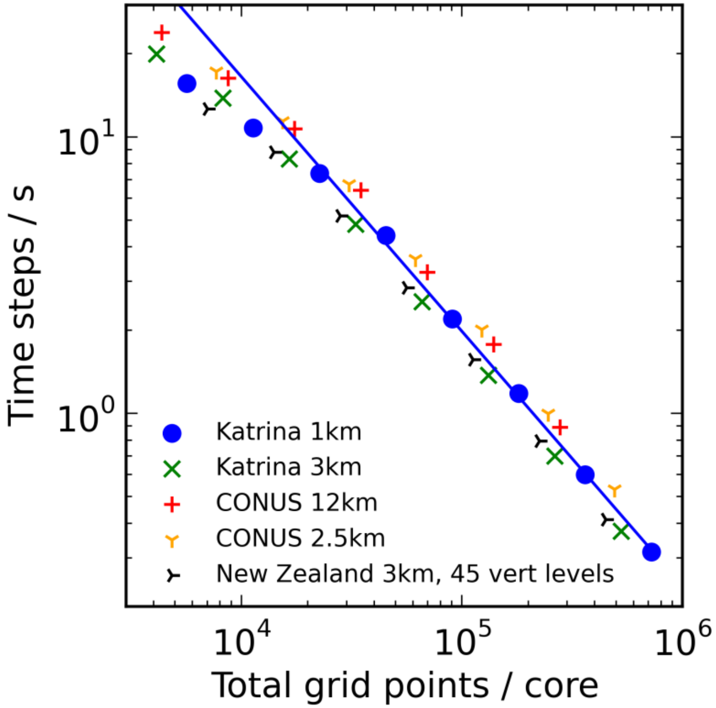
Katrina Cases

- **Intel 18.0.1 + SGI MPT 2.18 gives best performance**
- GNU + MVAPICH2 < GNU + SGI MPT
- IMPI at high node count doesn’t do well.

Scalability Summary



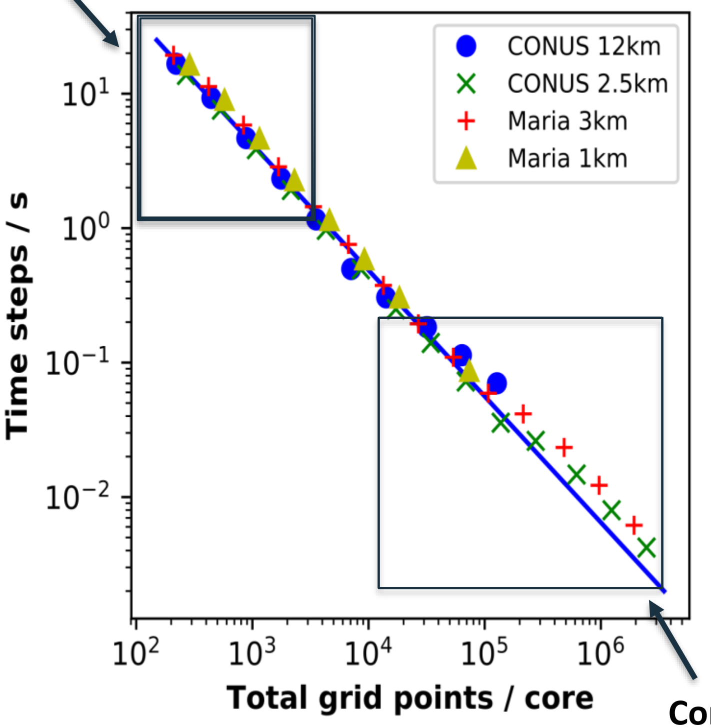
- Cheyenne



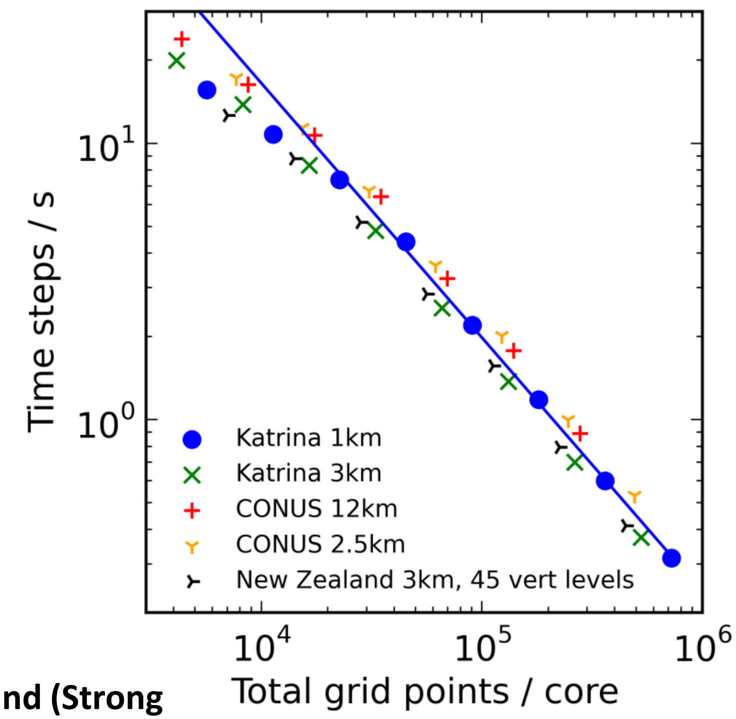
- Yellowstone

Scalability Summary

MPI Bound



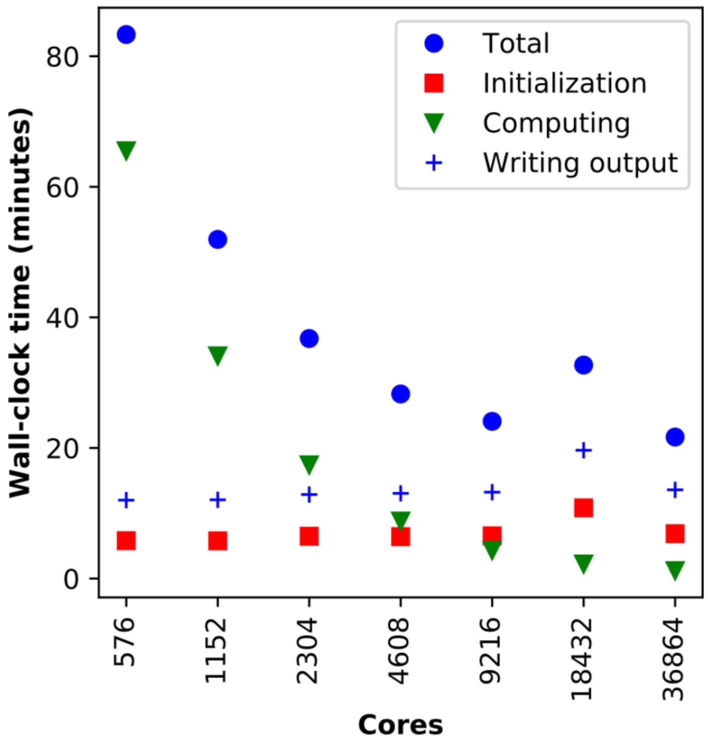
Compute Bound (Strong Scaling)



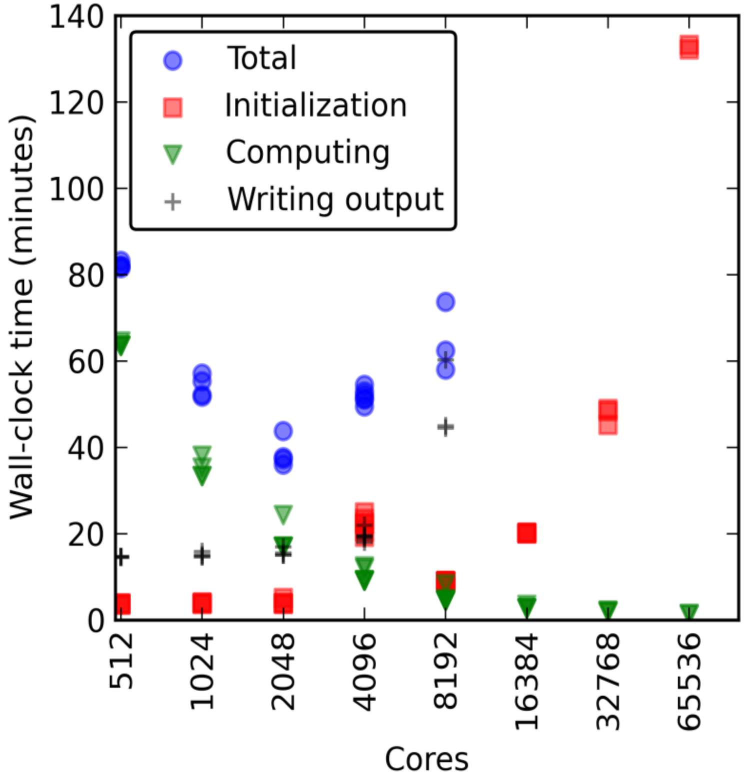
- Cheyenne
- Yellowstone

- Strong scaling : Keeping problem size fixed, increasing the core-count will increase performance, while the (approx.) same core-hours are consumed

Scalability Summary



- Cheyenne

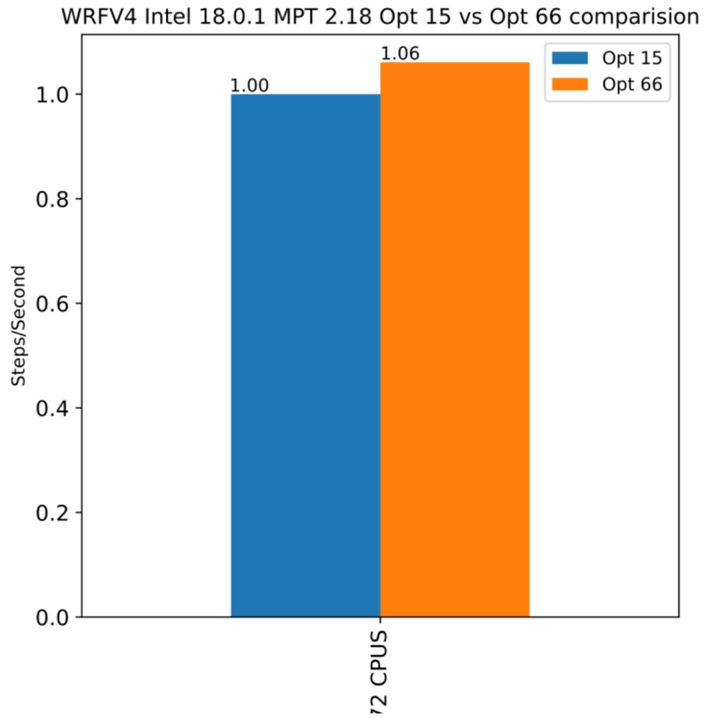


- Yellowstone

- Much better Initialization and I/O due to improvements in WRF

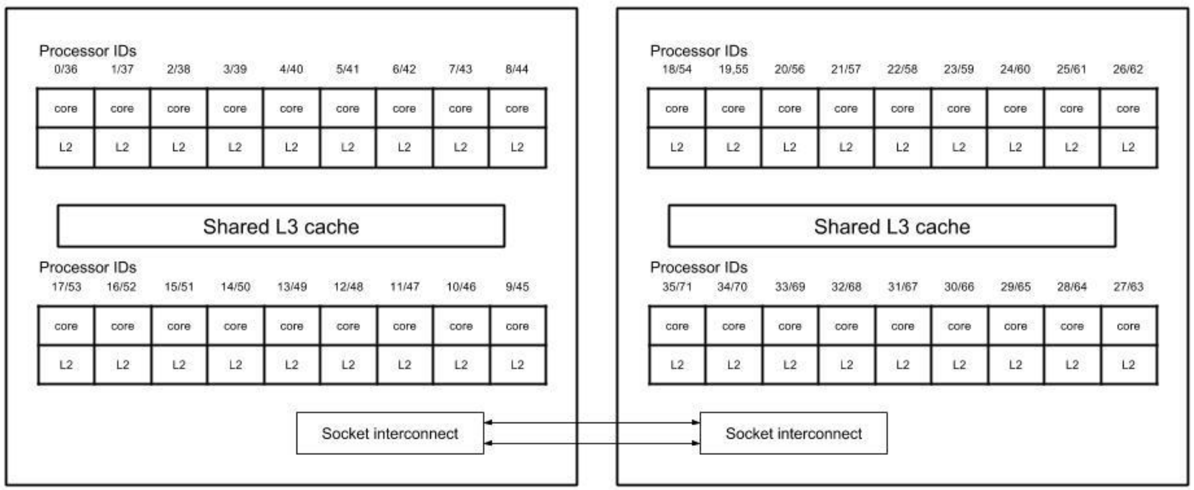
Compiler Optimizations

- GNU : -**ofast**
- Intel : **-xHost -fp-model fast =2**
- Use Option 66 or 67 for Intel Compiler :
  - **xHost -fp-model fast=2 -xCORE-AVX2**
  - -no-heap-arrays -no-prec-div -no-prec-sqrt -fno-common



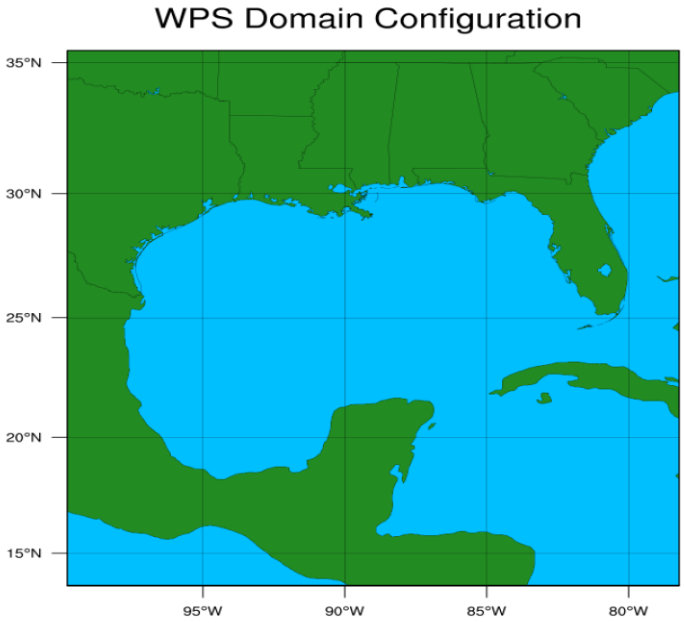
Hybrid Parallelism

- MPI (Distributed Memory) + OpenMP (Shared Memory)
- Why Hybrid ?
  - Eliminates domain decomposition at node level
  - Better Memory Coherency and less data movement within node
- Cheyenne Node Layout :



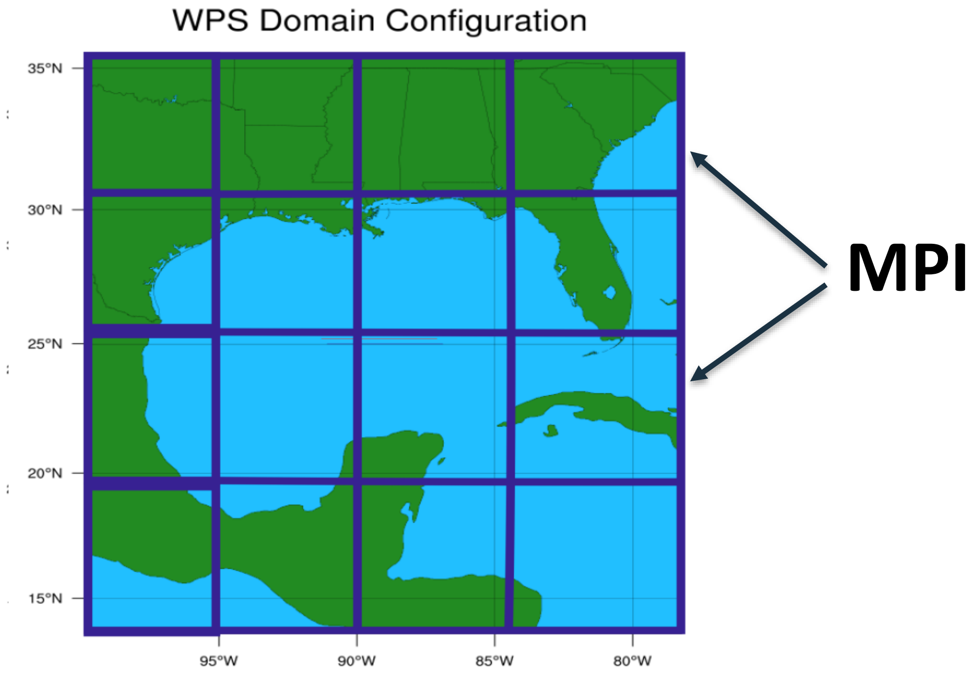
Hybrid Parallelism

- WRF Tiling :
  - Domain decomposition to divide work



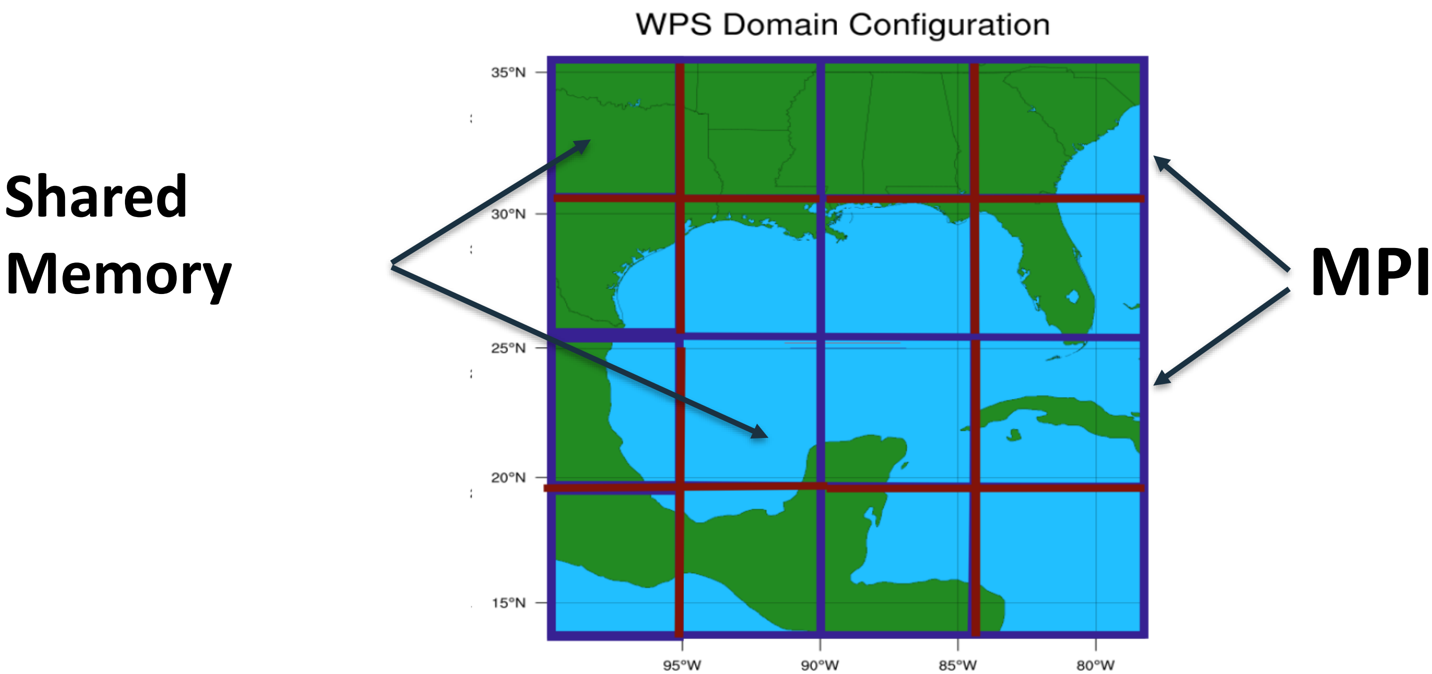
Hybrid Parallelism

- WRF Tiling :
  - Domain decomposition to divide work
  - Domain first broken into pieces called patches



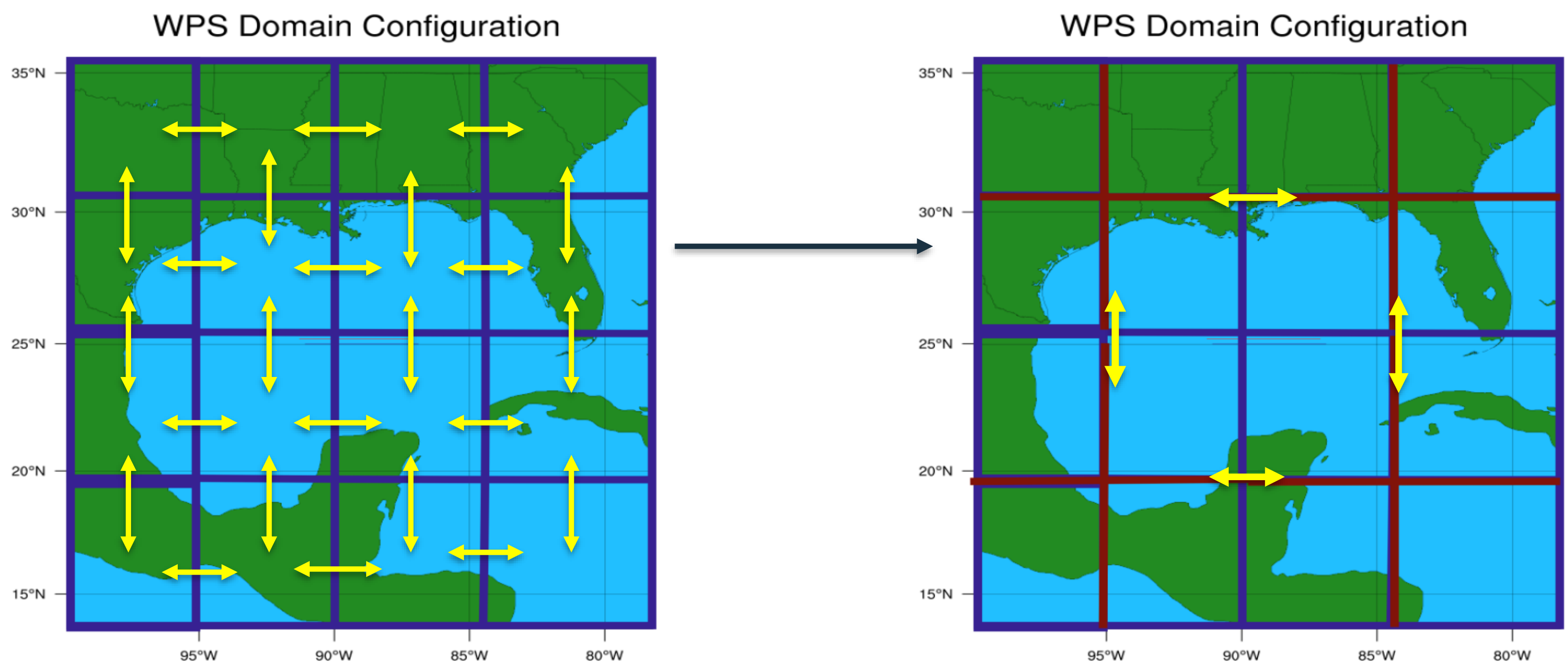
Hybrid Parallelism

- WRF Tiling :
  - Domain decomposition to divide work
  - Domain first broken into pieces called patches
  - Each can be further sub divided for shared memory parallelism



Hybrid Parallelism

- WRF Tiling :
  - Domain decomposition to divide work
  - Domain first broken into pieces called patches
  - Each can be further sub divided for shared memory parallelism

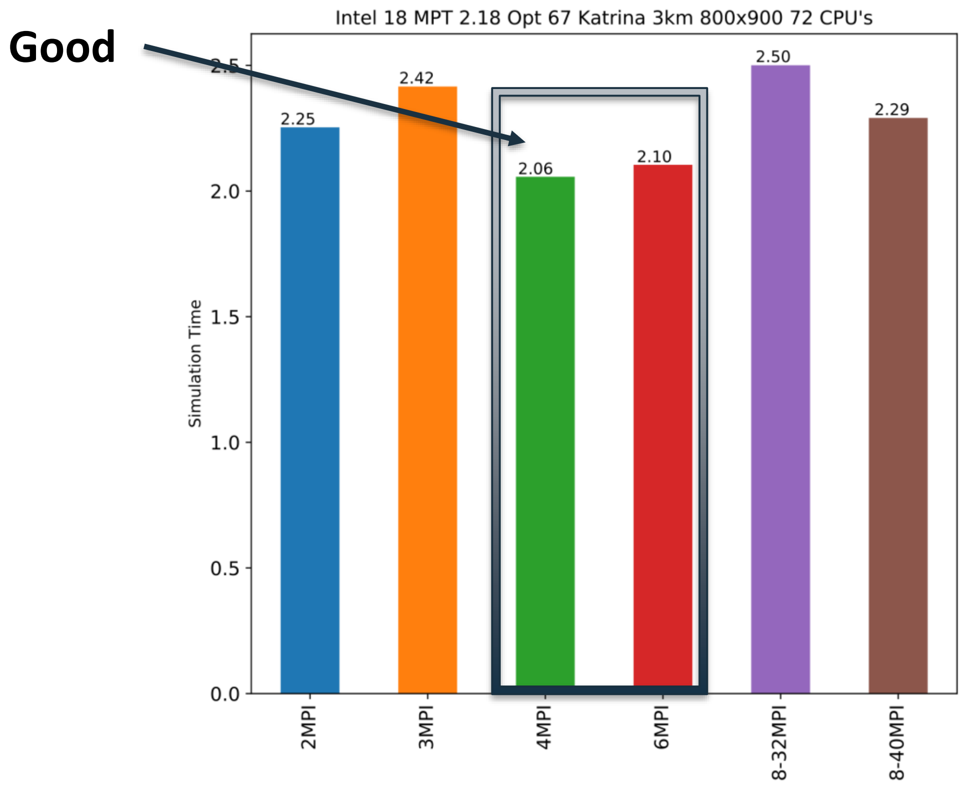


Hybrid Parallelism

- How many OpenMP threads and MPI tasks ?

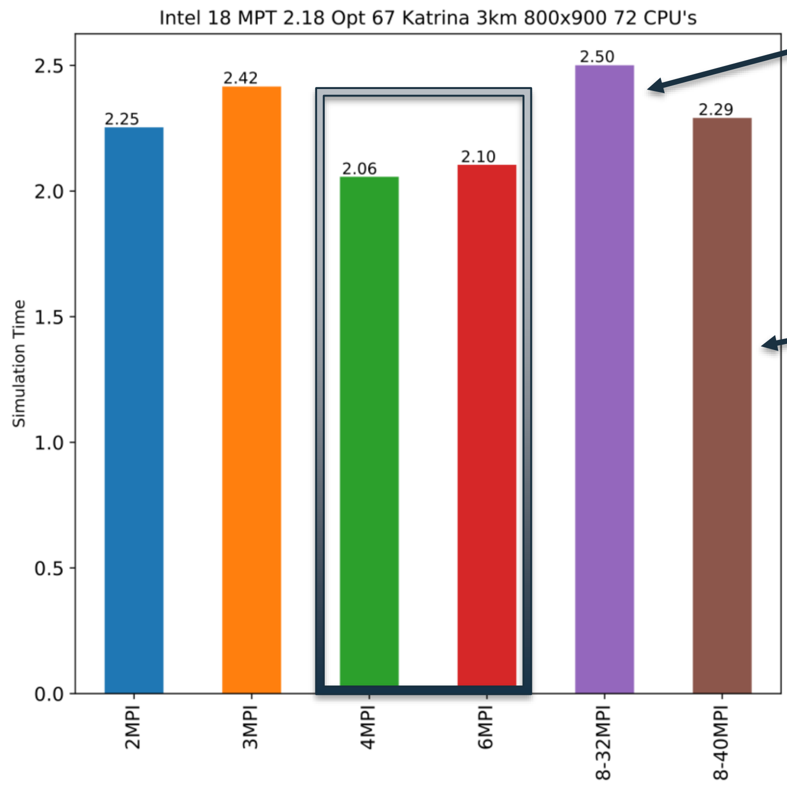
Hybrid Parallelism

- How many OpenMP threads and MPI tasks ?
  - 4MPI+9OMP or 6MPI+6OMP seems to work well



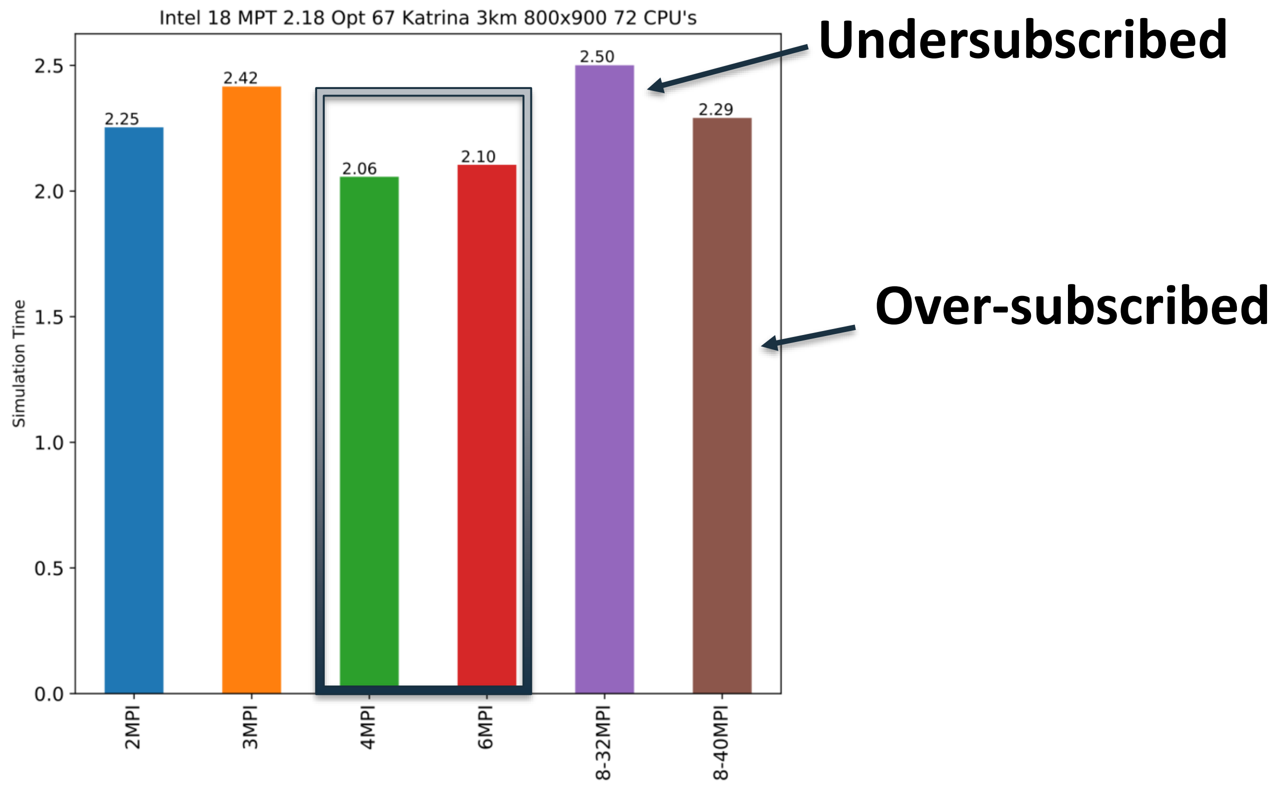
Hybrid Parallelism

- How many OpenMP threads and MPI tasks ?
  - 4MPI+9OMP or 6MPI+6OMP seems to work well



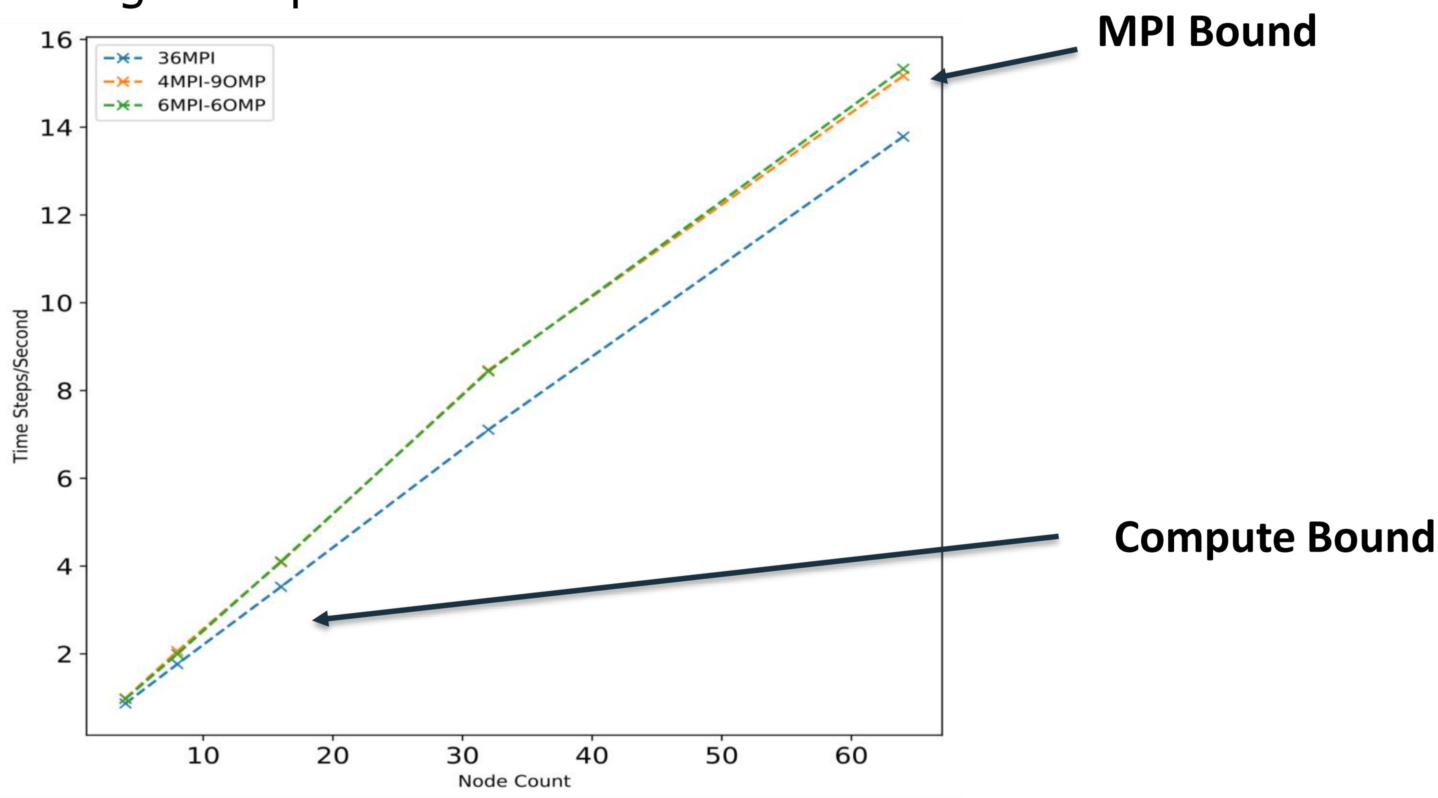
Hybrid Parallelism

- How many OpenMP threads and MPI tasks ?
  - 4MPI+9OMP or 6MPI+6OMP seems to work well



Hybrid Parallelism

- How many OpenMP threads and MPI tasks ?
  - 4MPI+9OMP or 6MPI+6OMP seems to work well
  - Better Scaling than pure MPI

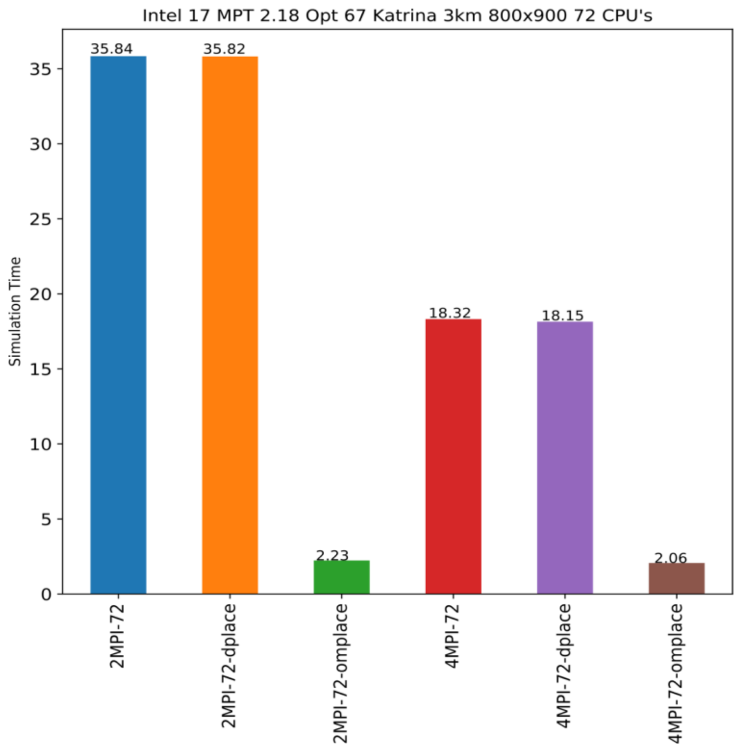


Core Affinity/Binding

- Above runs use core affinity

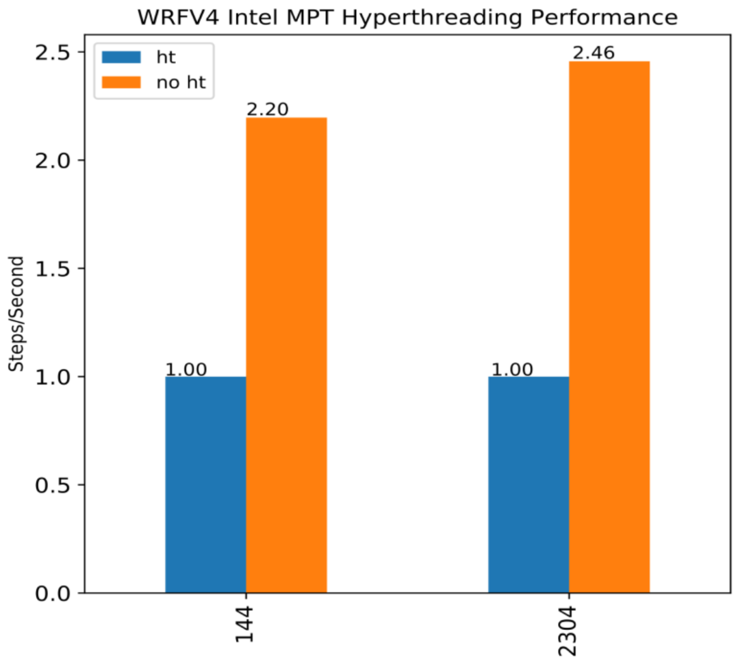
Core Affinity/Binding

- Above runs use core affinity
  - Omplace -vv



Hyperthreading Results

- Hyperthreading on Cheyenne :
  - Single Core acts like two logical cores
  - Model performance was significantly reduced
  - Do not recommend using more than 36 cores per node



Conclusion

- We recommend running WRF with the latest Intel compiler (18.0.1) and MPT (2.18) library. It consistently gave better performance than other options.
- For intel compilation's : –xHost –fp-model fast =2 –xCore-AVX2 option turned on. For GNU, use –ofast
- For hybrid runs, thread affinity significantly affects performance. Use omplace or dplace (if you want to specify explicitly the CPU mapping)
- Hyperthreaded runs lowered model performance

Future Work

1. Investigate MVAPICH Environment settings
   1. Eg : Inter-node Communication : Eager vs Rendezvous Protocol
      (MV2_VBUF_TOTAL_SIZE & MV2_IBA_EAGER_THRESHOLD & MV2_SMP_EAGERSIZE)
   2. WRF crashes around a size of **2k** & halts at **128k** (64 nodes)
2. Explore different patching/tiling strategies for domain decomposition
3. Explore different core binding strategies for hybrid runs to understand impact on performance at high node counts.
4. Personal : Tools to profile application : WRF Advection Code

Acknowledgements

- Mentors **:**
  - Davide Del Vento
  - Brian Vanderwende
  - Negin Sobhani
  - Alessandro Fanfarillo
- Project Partner :
  - Akira Kyle
- SIParCS Team
  - AJ Lauer, Jenna Preston, Eliot Foust, Shilo Hall
  - Rich Loft
  - and fellow interns !