Extremely High-Resolution Weather Model Simulation, Data Processing, and Visualization

Si Liu
Texas Advanced Computing Center

95th American Meteorological Society Annual Meeting
January 8, 2015

Team members

TACC

Si Liu
John Cazes
Greg Foss
Greg Abram

Raytheon

Don Cook
Craig Stair

Outline

- Background
  - Project objectives
  - Target domain and expected resolution
  - Weather models: WRF and ARPS
  - WRF nested runs
- High-resolution simulation
  - Memory requirement
  - IO workflow
  - Data processing and visualization
- Achievements and conclusion

Outline

- Background
  - Project objectives
  - Target domain and expected resolution
  - Weather models: WRF and ARPS
  - WRF nested runs
- High-resolution simulation
  - Memory requirement
  - IO workflow
  - Data processing and visualization
- Achievements and conclusions

Background and objectives

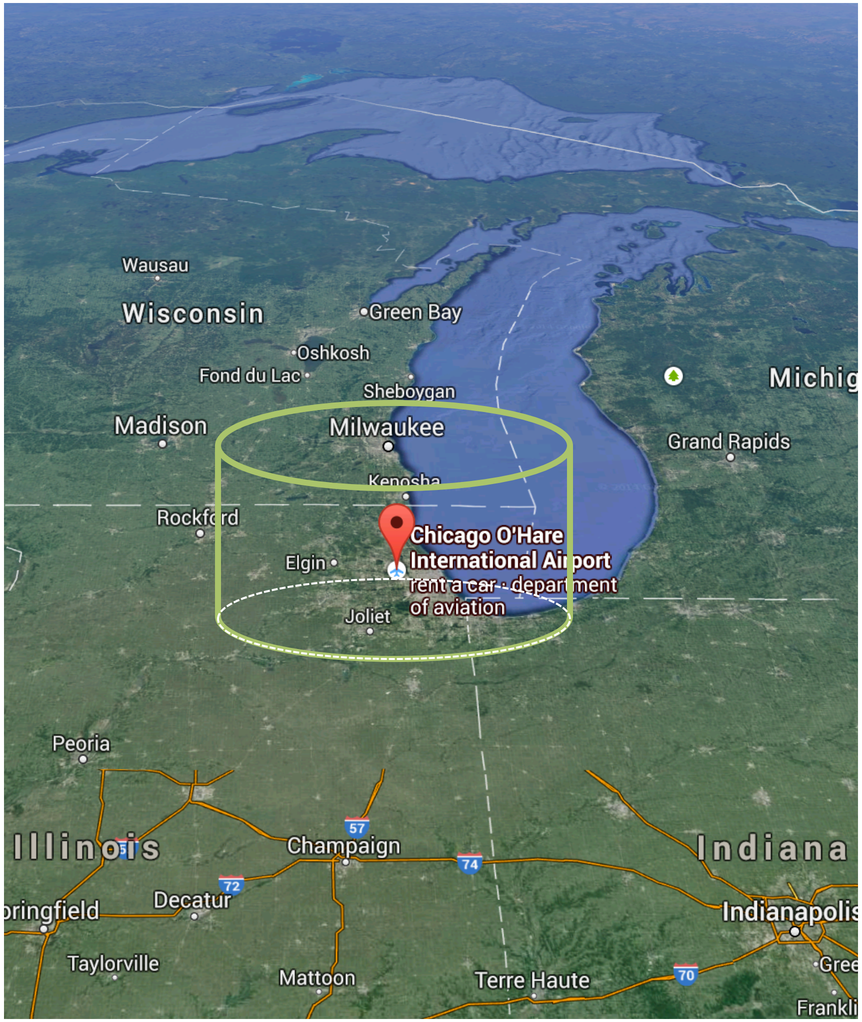
- Raytheon R&D project (2013 – now)

- Domains of interest
  - Chicago O'Hare International Airport
  - Highly localized weather modeling

- Severe weather simulation
  - Extremely high spatial resolution
  - Extremely high temporal resolution
  - Large-scale data processing for animated demonstration

Target domain and expected resolution

- O'Hare International Airport, Chicago, Illinois
  - Longitude: 87.90 W
  - Latitude: 41.98 N
- Cover the cylinder area
  - Diameter: 224 kilometers (over 120 nautical miles)
  - Height: 21 kilometers (about 70,000 feet)
- Expected resolution
  - Horizontal: 167 meters in average
  - Vertical: about 91 meters in average (300 feet)



Weather models

Weather Research and Forecasting

- Open source community software
- Developed and supported by NCAR and collaborative partners
- Parallel mesoscale weather model
- Used for both research and operational forecasts
- A large worldwide community of users (over 20,000 in over 130 countries)
- Mainly used for simulation in this project

The Advanced Regional Prediction System

- Comprehensive regional to stormscale atmospheric modeling / prediction system
- Developed at the Center for Analysis and Prediction of Storms (CAPS) at the University of Oklahoma
- Mainly used for data post processing in this project

WRF nested runs

- A fine-grid run based on the parent coarse-grid run
- Cover only a portion of the parent domain
- Lateral boundaries driven from the parent domain
- Why nested runs
  - High-resolution model running over a large domain
    extraordinarily expensive (memory, storage, computing)
  - High-resolution simulation for a very small domain with mismatched time and spatial lateral boundary conditions

One-way nested runs

A fine-grid run is made as a subsequent run after a coarser-grid run

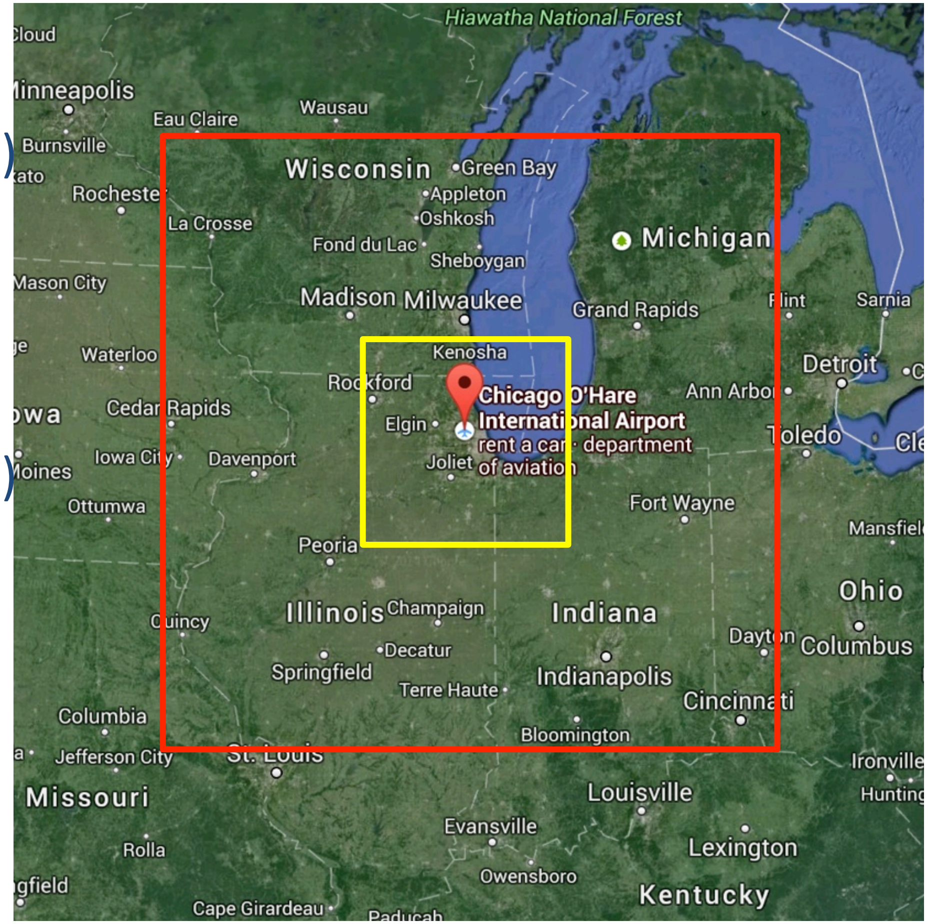
- Make a complete coarse-grid run (500 m horizontal)
- Collect output data
- Create initial and lateral boundary conditions for the fine-grid run with the WRF ndown.exe program
- Run the fine-grid simulation (167 m horizontal) with the input files generated in the previous step

Vertical refinement

- Original design:
  100 vertical levels (parent domain)
  → around 100/200/500 vertical levels (child domain)
- Practical implementation:
  234 vertical levels (parent domain)
  → 234 vertical levels (child domain)
- Vertical refinement is limited in existing WRFV models
  - Bugs have been fixed and reported to WRF developers in 2014
  - Source code changes are required!

Nested domains sketch map

- Outer domain
  - 1345 x 1345 grid cells (500 meters)
  - 234 vertical level (~91 meters)
- Inner domain
  - 1345 x 1345 grid cells (167 meters)
  - 234 vertical levels (~91 meters)
- Nested ratio
  - Horizontal: 3:1
  - Vertical: 1:1



WRF workflow

- Obtain the Global Forecast System (GFS) model data
- Run geogrid.exe, ungrib.exe, and metgrid.exe in WRF Preprocessing Systems (WPS)
- Run real.exe to generate the initial and lateral boundary condition files for the coarse-grid run
- Make a coarse-grid run (only a few output files are necessary)
- Re-run geogrid.exe and metgrid.exe for both parent and child domains
- Re-run real.exe for both parent and child domains
- Execute ndown.exe to generate fine-grid initial and lateral boundary conditions
- Make the fine-grid run and produce output files frequently as required

TACC Stampede system

- Dell Linux Cluster with CentOS
- 6,400+ Dell PowerEdge server nodes
  - 2 Intel Xeon E5 (Sandy Bridge) processors
  - 1 or 2 Intel Xeon Phi Coprocessor (MIC Architecture)
- The aggregate peak performance
  - Xeon E5 processors: 2+ PF; Xeon Phi processors: 7+ PF
- Login nodes, large-memory nodes, graphics nodes
- Global parallel Lustre file system + local disk

Outline

- Background
  - Project objectives
  - Target domain and expected resolution
  - Weather models: WRF and ARPS
  - WRF nested runs
- High-resolution simulation
  - Memory requirement
  - IO workflow
  - Data processing and visualization
- Achievements and conclusions

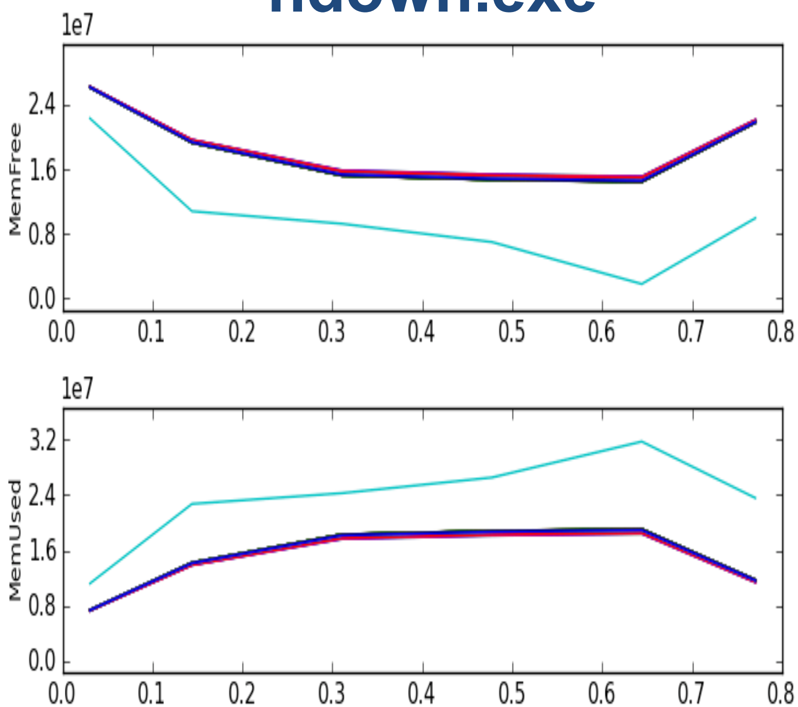
Memory issues

- What's the main problem?
  - Out of memory!
    Prevailing and critical problem in high-resolution simulations
- Why does it happen?
  - The problem size is too huge!
- How to fix it?
  - More/larger memory is a possible solution
  - Use what we have wisely!

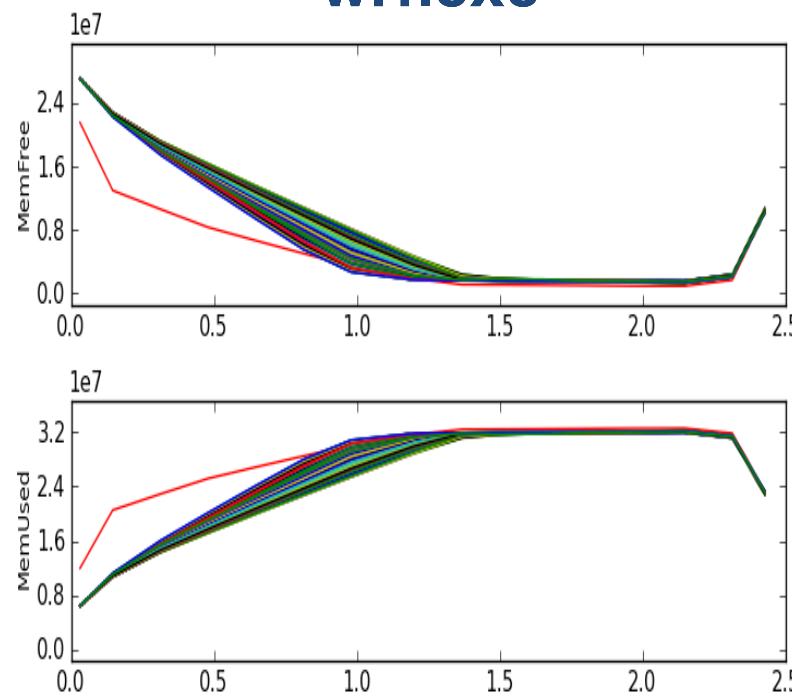
Monitor memory usage

- Each MPI task needs a huge amount of memory
- Task zero may require more memory than others
- TACC Stats: http://tacc-stats.tacc.utexas.edu/

ndown.exe

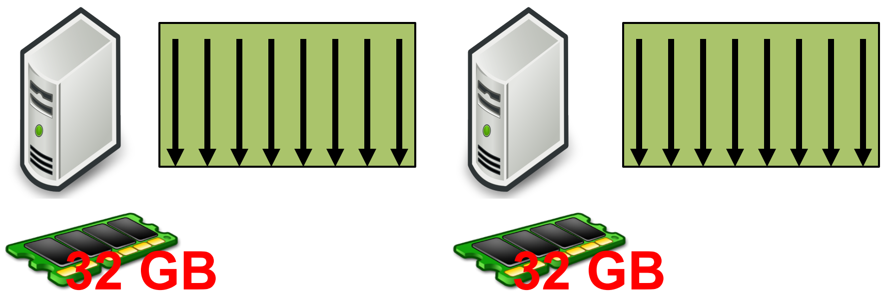


wrf.exe

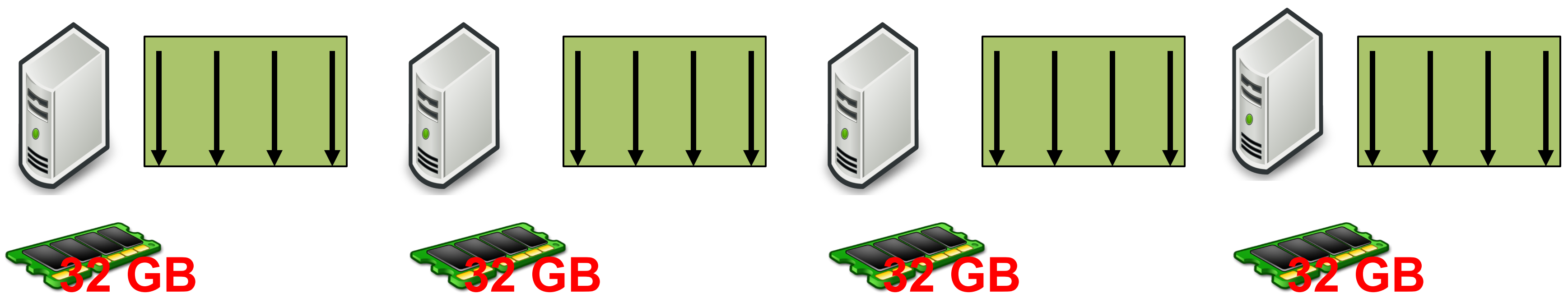


Memory management

- Original/basic settings

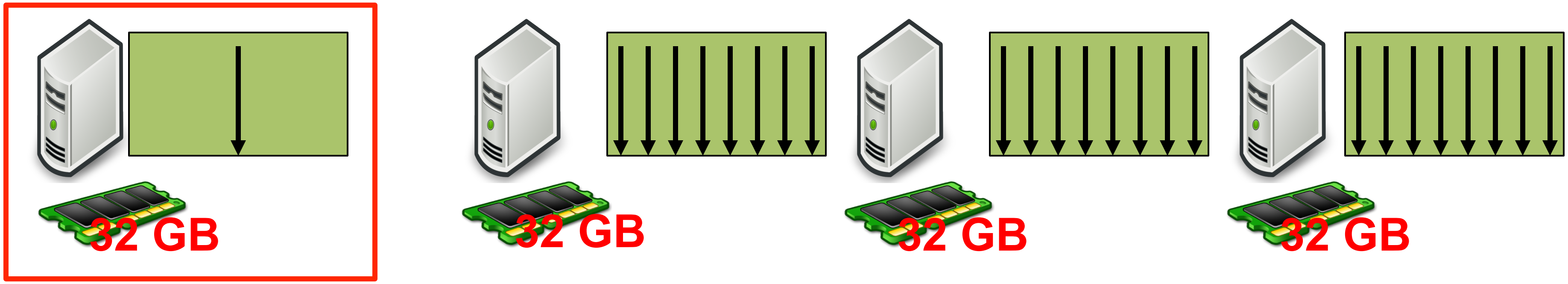


- Fewer MPI tasks per node -> More memory per task

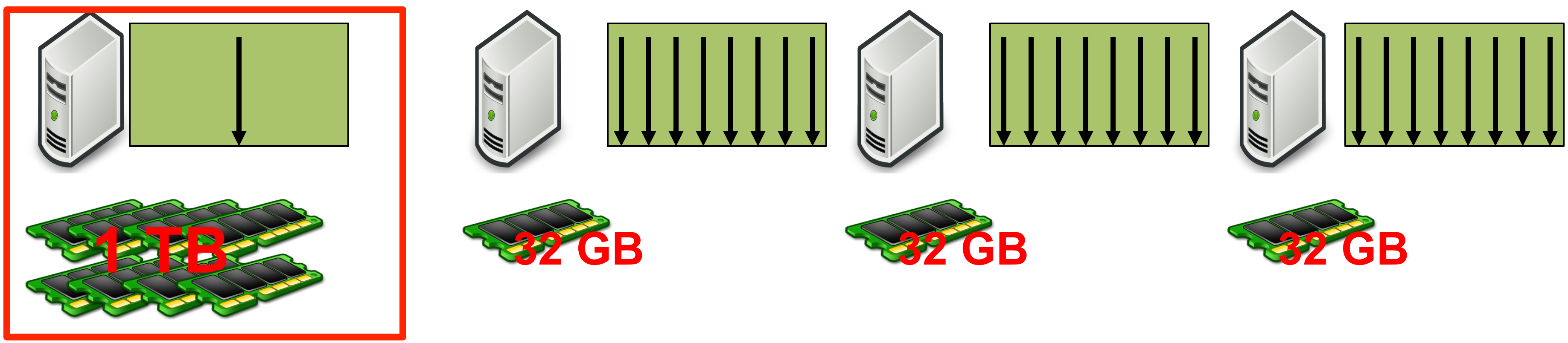


Memory management (continued)

- One dedicated node for Task zero



- One dedicated large-memory node for Task zero



SLURM reconfiguration is required!

IO workload

- Each data file is huge
  - More than 400 million grid points, about 200 variables
  - Over 11 GB per file
- Record the output every 3 model seconds
  - Generate 20 files per model minute
  - About 1200 files per model hour
- Serial I/O
  - "Spokesman": wasting a lot of computing resources
  - Independent file per core: so many IO requests

    Slow down/crash the file system
- MPI collective I/O
  - See our other paper

    A User-Friendly Approach for Tuning Parallel File Operations (SC14)

IO techniques

- Use local hard drive to temporarily keep the output
  - Local /tmp space (about 64 GB available disk space per node)
- WRF output files and WRF restart files partition
  - About 10 minutes → about 0.5s per output step
- Restrict output variables
  - Modified Registry.EM_COMMON

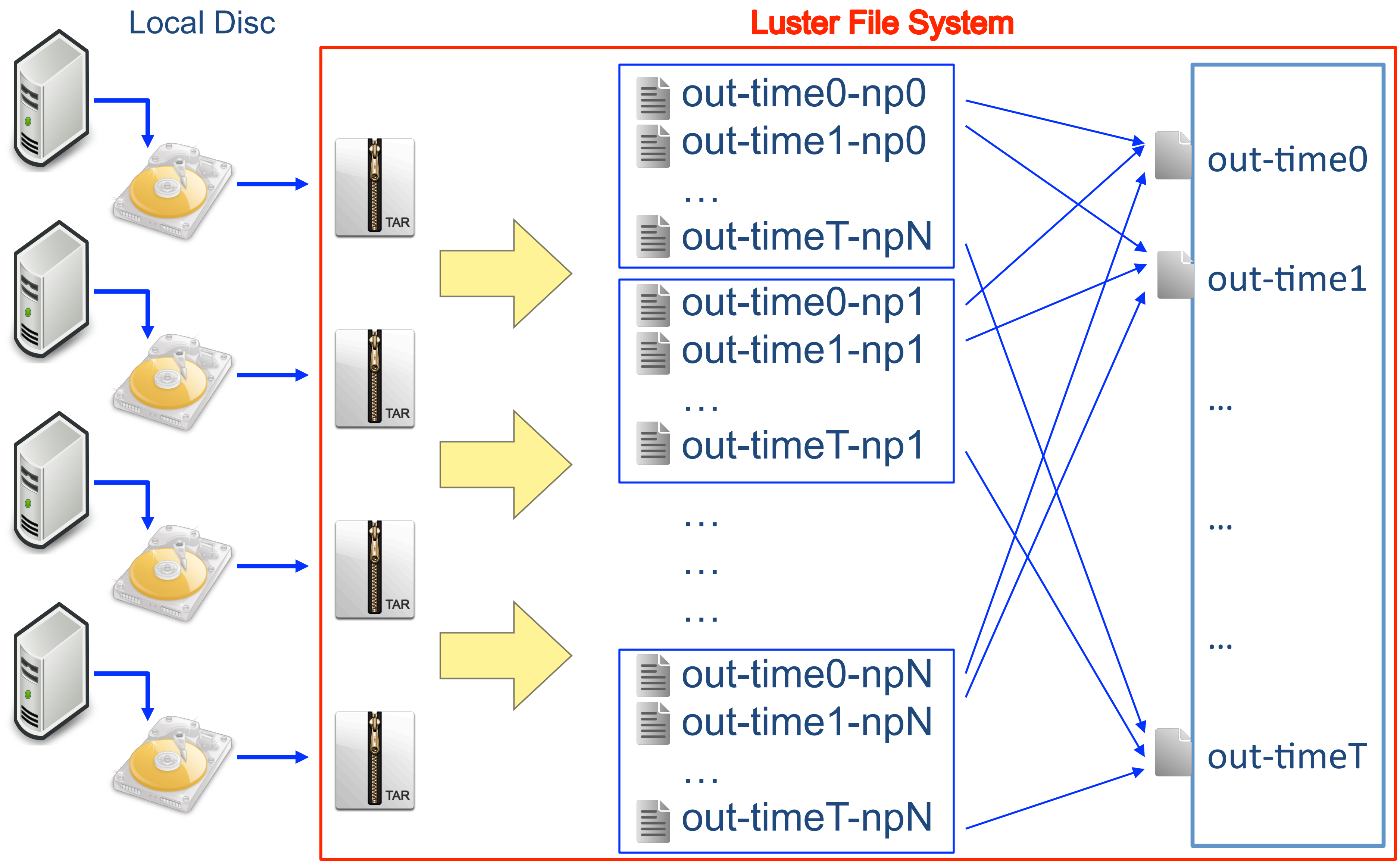
    Re-compiling the source code is required!
  - Reduce the output file size by 30-50%
- WRF checkpoint and restart mechanism
  - Complete jobs within the wallclock limit
  - Validate the output data after every single run
  - Reduce the risk of job failure and data loss

Merge split WRF output

- Regroup split WRF output files
  - Task-based → Time-step-based
  - Several "tar/untar" work to reduce the Metadata Server workload of the Lustre file system
- Merge split WRF output data files
  - Advanced Regional Prediction System (ARPS)
  - Thousands of sequential jobs
  - Large-memory nodes
  - TACC Parametric Job Launcher Utility

    utility for submitting multiple serial applications simultaneously

IO workflow



Local Disc
Luster File System
TAR
TAR
TAR
TAR
out-time0-np0
out-time1-np0
...
out-timeT-npN
out-time0-np1
out-time1-np1
...
out-timeT-np1
...
...
...
out-time0-npN
out-time1-npN
...
out-timeT-npN
out-time0
out-time1
...
...
...
out-timeT

I/O time comparison

Based on a typical run with 1201 time steps on 1024 cores

	Traditional workflow	Our advanced workflow
Time per step	About 10 minutes	0.4-0.5s on average (1024 cores)
Total Time	**More than 8 days!**	**About 8-10 minutes**
Time for extra data processing	0	8-10 hours depending on the computing resources, only when necessary
Target data files	1201 wrf-out files, 11 GB each 13 TB in total for one-hour run	1201 wrf-out files, 11 GB each 13 TB in total for one-hour run
Extra space required	0	Hundreds of tar ball files, about 10 TB extra in total, temporary files can be removed

Data analysis and visualization

- WRF output uses geopotential values to identify altitude, whereas visualization software requires coordinate values in the height axis
- Translate WRF output files to VTK files (Python)
  - Convert geopotential values into Z coordinates
  - Irregular grid
- Resulting VTK files are read into ParaView
- For a generalized aviation reference, an aviation map provided by Raytheon is included for background in the animation

Outline

- **Background**
  - Project objectives
  - Target domain and expected resolution
  - Weather models: WRF and ARPS
  - WRF nested runs
- High-resolution simulation
  - Memory requirement
  - IO workflow
  - Data processing and visualization
- Achievements and conclusions

Achievements and conclusions

- Special design for high-resolution simulation on modern supercomputers
- A specific time frame and region to provide meteorological data with extremely high spatial and temporal resolution
- The resolution is well beyond almost all similar weather simulations as we are aware of
- Almost all techniques are applicable
  - Memory-intensive programs
  - I/O-intensive applications
  - High-resolution simulations
  - Other supercomputer platforms

Future work

- Compare with other observed or experimental data and validate the results
- Perform similar high-resolution severe weather simulation over other areas
- Improve the performance of memory-intensive and/or I/O intensive WRF programs with Xeon Phi
- Investigate optimized I/O workflow with MPI collective I/O

Acknowledgement

Special thanks to:

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

MMM

Ming Chen, Dave Gill, Jordan Powers

TACC

Bill Barth, Doug James, Tommy Minyard, Todd Evans

CAPS

Yunheng Wang

Si Liu
Texas Advanced Computing Center
siliu@tacc.utexas.edu

For more information:
www.tacc.utexas.edu