End-to-end optimization potentials in HPC applications for NWP and Climate Research

Luis Kornblueh
and
Many Colleagues

Max-Planck-Institut für Meteorologie and DKRZ
... or a guided tour through the jungle ...
Joint DKRZ/MPIM initial brainstorming

Courtesy of Joachim Biercamp, DKRZ
Rationals

Why do end-to-end management?

- Scientific responsible experimentation support
Rationals

Why do end-to-end management?

- Scientific responsible experimentation support
- *Reduce workload of all members of the numerical experimentation community* …
An experimentation howto

- Title, statement of problem, and hypothesis
An experimentation howto

- Title, statement of problem, and hypothesis
- Site, hardware, software stack, used programs (versioned), sources, and documented input and boundary data
An experimentation howto

- Title, statement of problem, and hypothesis
- Site, hardware, software stack, used programs (versioned), sources, and documented input and boundary data
- Step-by-step description in such a way that the experiment can be repeated

Max-Planck-Institut für Meteorologie

Kornblueh et al.
Optimization potentials
An experimentation howto

- Title, statement of problem, and hypothesis
- Site, hardware, software stack, used programs (versioned), sources, and documented input and boundary data
- Step-by-step description in such a way that the experiment can be repeated
- Run the experiment, NO, not three times only one time!
An experimentation howto

- Title, statement of problem, and hypothesis
- Site, hardware, software stack, used programs (versioned), sources, and documented input and boundary data
- Step-by-step description in such a way that the experiment can be repeated
- Run the experiment, NO, not three times only one time!
- Observations made during and analysis of the results
An experimentation howto

- Title, statement of problem, and hypothesis
- Site, hardware, software stack, used programs (versioned), sources, and documented input and boundary data
- Step-by-step description in such a way that the experiment can be repeated
- Run the experiment, NO, not three times only one time!
- Observations made during and analysis of the results
- Discuss possible errors that could have occurred in the collection of the data (experimental errors)
An experimentation howto

- Title, statement of problem, and hypothesis
- Site, hardware, software stack, used programs (versioned), sources, and documented input and boundary data
- Step-by-step description in such a way that the experiment can be repeated
- Run the experiment, NO, not three times only one time!
- Observations made during and analysis of the results
- Discuss possible errors that could have occurred in the collection of the data (experimental errors)
- Publication of the results
The context

What is our context?

- complex, non-standardized workflows and toolchains
- various processing steps by various actors
- in climate research as well very often changing workflows

*Important to note - we are not doing mainstream computing!*
Target: provenance of the whole data life cycle

Focus on adding:

▶ primary data generation
▶ primary data processing

Already available to a large extent:

▶ data publishing
▶ secondary data processing
▶ data distribution
▶ further not controllable data processing
Define experiment easily

- should organize their individual experiments workflow easily
- should be enabled to program their individual tasks
- should not care on collecting provenance data
Scientific programmers

- define experiment easily
- should organize experiment workflows easily
- should be enabled to program individual tasks
- should not care on collecting provenance data
- should be enabled to query provenance data for bug tracking, performance improvements, . . .
Computing center staff

- should not care on collecting provenance data
- should be enabled to query provenance data for failure analysis, performance improvements, ...
Components of the basic systems

Frameworks

Setup
GUI or CLI

Domain Specific Language
for
Task Scripting
(Provenance Collecting Enabled)

Cylc Meta-scheduling

Preparation

Post-processing

Model run

Archiving
Tools

Packages in use:

- python as scripting language
- postgres - provenance data collection
- subversion/git (migration to git for parts or all later, if model developers get convinced)
- cmake (migration from autotools and self-maintained makefile generator)
- Web interface for site and compiler dependencies; dependencies versioned in line with model code
- namelist migration to xml as model source code, user interfaces: a kind of namelist and a GUI.
Single task component

Scripting

- user friendly modeling language
- restartability
- exception and error handling in scientist understandable form
- full support of modern programming concepts
- use the python hype to change from shell to python
Model build requirements

Standardized, convenient, and fast tools

▶ out-of-source build
Model build requirements

Standardized, convenient, and fast tools

▶ out-of-source build
▶ dependency resolution
Model build requirements

Standardized, convenient, and fast tools

- out-of-source build
- dependency resolution
- make, with hooks for actions at certain steps
Model build requirements

Standardized, convenient, and fast tools

- out-of-source build
- dependency resolution
- make, with hooks for actions at certain steps
- make test
Model build requirements

Standardized, convenient, and fast tools

- out-of-source build
- dependency resolution
- make, with hooks for actions at certain steps
- make test
- make install
Use of optimized codes: mpiesm and icon

### In use

- Vectorization (we never gave up on this!), hand gather, scatter, merge instead of standard conditionals, and exposing transcendental functions.
- nproma blocking for different architectures.
- OpenMP orphaning - whole physics including radiation run in an single OpenMP directive.
- MPI implementation constantly revisited including building up static load-balancing strategies.
- Real asynchronous parallel I/O.
- Invest in optimization of libraries.

*Maybe give up on bit-reproducability for production, but not for development!*
Data compression

On top compression for grib2: AEC (CCSDS algorithm).

A standardization exercise

- got NASA US patent released
- reimplement from scratch to go around commercial copyright
- define grib2 template for WMO
- validate with independent software stacks (Q4 2014)

- Average reduction in data size over 4-byte float: factor 5, and for grib2 2.5.
- Encoding and decoding are really, really fast.
- Get this ported to netcdf4 (hdf5), started but process is slow.

Joint work of Mathis Rosenhauer, DKRZ, Shahram Najm, ECMWF, Uwe Schulzweida, and Luis Kornblueh
Ocean scaling improvements: no land points, gmres restart, hybrid MPI/OpenMP, code rewrite.

*Courtesy of Leonidas Linardakis*
Change from serial (cdi) to parallel output (cdi-pio, using RDMA, I/O time on compute nodes is essentially zero).

*Courtesy of Irina Fast, Thomas Jahns, DKRZ, and Deike Kleberg*
Use of optimized post-processing tools

Extend and improve our toolchain (cdi and cdo)

### Available
- basic optimized code
- compute intensive operators are OpenMP parallelized
- processing of data can be handled by a threaded pipelining method

### Future
- data streaming instead of file storage transfer
- DAG based processing for highest possible parallel efficiency
- database information system for online data
Experiment organization

cylc - the Meta-Scheduler

- design your own distributed suites of inter-dependent cycling tasks efficient, modular, and reusable
- validate and visualize workflows on the fly
- control your running suites
- diagnose failures (easily!?)
- simplify failure recovery
- benefit from expert experience with a specialized tool for meteorological forecasting systems

Courtesy of Hilary Oliver, NIWA and contributors
A task modeling framework (provenance data collection)

Cytc controled tasks and provenance collection

- high level programming language — python
- embedded abstraction layer for file operations
- abstract task description
- embedded provenance data collection (database stored)
- tightly connected to cytc
- connect provenance enabled workflow to ESGF data distribution

Remark: introduces complexity reduction methods to all users

Courtesy of Deike Kleberg, MPIM and Pavan Siligam, DKRZ
A cylc optimization step

- connect cylc from inside application (in C)
- use curl for submitting http/POST request to a WebServer
- POST triggers CGI as interface to cylc
- Later: server part integrated into cylc

Do not poll! It is expensive!
What is coming next?

▶ Primary target: Get an optimized workflow system working AT OUR SITE!
▶ Keep being conservative in model source code adaptation: concentrate on what we have
▶ But: Outsource exploration of GPU handling (CSCS) and take over necessary changes
▶ Constantly observe development directions
▶ Explore CS concepts in the easier context of post-processing
▶ Do: Tutorials and training, training and tutorials, tutorials and training, . . .