Performance Analysis of an Earth Science Application

George S. Markomanolis, Oriol Jorba, Kim Serradell, Enza Di Tomaso

University of Athens, Department of Physics

Athens, 27 March 2014
Outline

- Overview of BSC
- Introduction to Earth Sciences Modeling
- Preprocess
- Performance Analysis of NMMB/BSC-CTM Model
- OmpSs Programming Model
- Data Assimilation
- Future work
Overview of BSC
Barcelona Supercomputing Center – Centro Nacional de Supercomputación (BSC-CNS) is the Spanish National Laboratory in supercomputing.

The BSC mission:
- To investigate, develop and manage technology to facilitate the advancement of science.

The BSC objectives:
- To perform R&D in Computer Sciences and e-Sciences
- To provide Supercomputing support to external research.

BSC is a consortium that includes:
- the Spanish Government – 51%
- the Catalan Government – 37%
- the Technical University of Catalonia – 12%
www.bsc.es
BSC Current Resources

- **MareNostrum 2013**
  - 48448 Intel SandyBridge-EP cores
  - 1 PFlops

- **MinoTauro 2011**
  - 128 compute nodes
  - 182 TFlops

- **HPC Storage and Backup:**
  - 2.5 PB disk
  - 6.0 PB tapes Robot
Introduction to Earth Sciences Modeling
Research in the Earth Sciences area is devoted to the development and implementation of regional and global state-of-the-art models for short-term air quality forecast and long-term climate applications.

ES maintains two daily operational systems: AQF CALIOPE and MD forecasts: BSC-DREAM8b and NMMB/BSC-CTM.
Earth Sciences research lines

Air Quality Forecast

Climate change modelling

Transfer technology (EIA and AQ studies)

Mineral dust transport: BSC-DREAM8b

Atmospheric modelling: development of NMMB/BSC-CTM

WMO SDS WAS [AEMET-BSC]

- To enhance the ability of participating countries to establish and improve systems for forecasting and warning to suppress the impact of Sand and Dust Storms by
- Establishing a coordinated global network of Sand and Dust Storm forecasting centers delivering products useful to a wide range of users in understanding and reducing the impacts of SDS

North Africa, Middle East and Europe

Asia

BSC-CNS, AEMET, Spain

China Meteorological Administration (CMA)
Xiao Ye Zhang
xiaoye@cams.cma.gov.cn
Severo-Ochoa Earth Sciences Application

**Development of a Unified Meteorology/Air Quality/Climate model**
- Towards a global high-resolution system for global to local assessments

**International collaborations:**
- Meteorology
  - National Centers for Environmental Predictions (NCEP)
- Climate Global aerosols
  - Goddard Institute Space Studies (NASA)
- Air Quality
  - Uni. of California Irvine (UCIrvine)

**EARTH SCIENCES**
- Extending NMMB/BSC-CTM from coarse regional scales to global high-resolution configurations
- Coupling with a Data Assimilation System for Aerosols

**COMPUTER SCIENCES**
- Data management
  - Use of persistent storage DB
- Preprocess
  - static DB: orography, landuse, vegetation, ...
- Chemistry emissions
  - Energy, industry, on-road transport, ports, airports, vegetation
- Unified model
  - On-line integration: meteo/AQ/climate (feedbacks)
- Code analysis and OpenMP programming
- CPU/GPU HPC
- PBs of data

**Preprocess**

**Chemistry emissions**

**Unified model**

**Data management**

**COMPSs programming**

**Code analysis and OpenMPs programming**
Not a new problem:

- As far back as the 13th century, people started complaining about coal dust and soot in the air over London, England.

- As industry spread across the globe, so did air pollution.

- The worst air pollution happened in London when dense smog (a mixture of smoke and fog) formed in December of 1952 and lasted until March of 1953. 4,000 people died in one week. 8,000 more died within six months.

A picture is worth a thousand words
Not a local problem, wide regions with air pollution problems
Air Pollution: Europe, South China, the Earth

Effects:
- It can cause illness and even death.
- It damages buildings, crops, and wildlife.
- It has a strong impact in visibility
- Impact on climate system
Where do we solve the primitive equations? Grid discretization

High performance computing resources:
If we plan to solve small scale features we need higher resolution in the mesh and so more HPC resources are required.
Unified models: meteorology – chemistry – climate

Embedding chemistry processes within a meteorological core driver

NMMB/BSC-DUST_CHEM

CHEMISTRY MODULE

- Emission tendency
- Photolysis computation
- Dry deposition
- Convective transport
- Wet deposition
- Chemical reactions

DYNAMICS
(DUST, CHEM hadv, vadv and hdiff)

Dynamics variable to physics
T, U, V, Q, CW, Q2, OMGALF, DUST, CHEM

Physics variable to dynamics
T, U, V, Q, CW, Q2, DUST, CHEM

PHYSICS
(Dust
Emission, vdiff, deposition, chem emission, vdiff, photolysis, chemical mechanism, deposition)

ATMOSPHERIC DRIVER

DUST MODULE

CHEM MODULE
Global aerosol simulation

Source: NASA GSFC
Types of simulations

Climate Simulations
- Global scale
- Large periods
- Huge amount of data created
- Execution time is not a critical constraint
- Example: EC-EARTH model for 1900 to 2100, year simulation

Operational Simulations
- Global/Regional Scale
- Small periods
- Data created is smaller but postprocess products are more important
- Execution time and reliability are very critical
- Example: Daily weather forecast
Setting up a model

- A model is a collection of source codes
- We need to compile to build an executable
- The executable will run and produce results

- Usually, models have a building procedure
  - Configure
  - Makefiles
  - Scripting…
Computational demands

Which domains are we simulating ¿?
- Barcelona
- Spain
- World

Which resolution ¿?
- 1 km²
- 4 km²
- 12 km²
- 50 km²

How many variables we want to compute ¿?
- T2
- U10, V10
- QRAIN, QVAPOR

Increasing this parameters, increases the system constraints
- Computation Needs (CPU's, Memory Bandwith…)
- Data Storage

Define this parameters in function of your hardware and time to serve forecast.
We need to be able to run this models in Multi-core architectures.

Model domain is decomposed in patches

Patch: portion of the model domain allocated to a distributed/shared memory node.
What is the role of a coupler?

- Exchange and transform information through two or more different models.
- Manage the execution and synchronization of the code.
- Example: couple an ocean model and atmosphere.
Post-processing

Once the model is run successfully, we need to post-process results to visualize data:

- Maps
- Plots
- Text files
- 3D Animations
Models at BSC

Mineral Dust Modeling
- BSC-DREAM8b V2: Dust REgional Atmospheric
  - Model
  - Fortran Code
  - Not parallel

NMMB/BSC-CTM
- Meteorology-Chemistry coupled model
  - Meteo. Driver: Nonhydrostatic Multiscale
  - Model on the B grid (NMMB)
  - Fortran Code
  - MPI

Climate Change
- EC-EARTH
  - Fortran, C
  - MPI, OpenMP
3D Outputs
Execution diagram: Focus on the Preprocessor

Tested in two cases: Global domain $1^\circ \times 1.4^\circ$ resolution
Global domain 12km x 12km
COMPSs programming model intends to maximize the programmability of Java applications running on parallel and distributed infrastructures.

COMPSs is fully developed at BSC.
Preprocess is divided in two main tasks:

- **Fixed**: which is only done once, when configuring the model
- **Variable**: is done each run, as takes daily meteorological and surface sea temperature inputs.
- Fixed and Variable are now run separately.

**Totally sequential, synchronous, ignore data dependencies between subprocesses.**

```
#FIXED
../exe/smmount.x
../exe/landuse.x
../exe/landusenew.x
../exe/topo.x
../exe/stdh.x
../exe/envelope.x
../exe/topsoiltype.x
../exe/botsoiltype.x
../exe/toposeamask.x
../exe/stdhtopo.x
../exe/deeptemperature.x
../exe/snowalbedo.x
../exe/vcgenerator.x
../exe/roughness.x
../exe/gfdlco2.x
../exe/lookup_aerosol.x
```

```
#VARIABLE
ln -s ../meteo_data/wafs.00.0P5DEG.13042400.grib1
../output/gfs.t002.pgrb2f00
ln -s ../meteo_data/sst2dvar_grb_0.5.13042400.grib1
../output/sst2dvar_grb_0.5

./degribgfs_generic_05.sh 00 00 03 pgrb2f ../output
../exe/gfs2model_rrtm.exe 00
../exe/inc_rrtm.x
../exe/cnv_rrtm.x
../exe/degribsst.x
../exe/albedo.x
../exe/albedorrtm1deg.x
../exe/vegfrac.x
../exe/z0vegustar.x
../exe/allprep_rrtm.x
../exe/read_paul_source.x
../exe/dust_start.x
```
The executions are done in MareNostrum3.

Compiled with ifort compiler,
- `FFLAGS="-mcmodel=large -shared-intel -convert big_endian -traceback -assume byterecl -O3 -fp-model precise -fp-stack-check"`

9.3 Gb statical data required (geodata and GTOPO30 databases)

Runtime for the global operational domain:
- Fixed: 7m30s
- Variable: 0m32s
Preprocess is a collection of Fortran codes.
In order to port to COMPSs, we need to modify sources to manage files as arguments instead of being hardcoded.

Example:
- `smmount` creates two files, `seamaskDEM` and `heightDEM`.
- With COMPSs, `smmount` is executed with files as arguments
  - `./smmount ../output/seamaskDEM ../output/heightDEM`

Fortran source code is modified to handle arguments.
Each executable is wrapped in a Java method and selected as a task.
This method is not hard to code, but `allprep` executable in variable, manages more than 44 files !!!
Then, three files are written in JAVA:

- *Fixed.java*: main program of the application, contains task calls.
- *FixedBinaries.java*: implementation of each task with the call to the executable.
- *FixedItf.java*: selection of tasks, providing the necessary metadata about their parameters.

The same files are written for Variable.
We implemented a Fortran/MPI application only for the Fixed preprocess, using 5 cores of one node based on the dependency graph acquired from CompSs.

Runtime for the global domain, 24 km:
- Fixed: 2m30s.
We used 5 cores and achieved a speedup of 2.7 times!
The serial part *allprep* consumes a lot of time, we should investigate a hybrid solution because of memory issues.

Need to be improved for higher resolution forecasts.
We applied this method to generate 12km global resolution input files (more than 6GB output files)
Execution Remarks

Data dependencies between tasks are automatically detected, thus exploiting the inherent concurrency of the application when executing the tasks.

In the Fixed application, 8 tasks are free of dependencies at the beginning, and therefore they can be sent for execution immediately.

Performance

- Fixed: the exploitation of task parallelism speeds up the process.
- Variable: it has little computation and parallelism, which does not compensate the overhead of task processing and distribution (e.g. dependency analysis, file transfer, task submission), hence incrementing the execution time.
Performance Analysis of NMMB/BSC-CTM Model
Execution diagram – Focus on the Model

Study domain: Global domain 24km x 24km resolution
One hour simulation of NMMB

Last four processes are used for I/O
It seems that previously there was noise during the execution
Issue with I/O

There is no parallel I/O implemented!

Last hour
With I/O
Without I/O
Issue with the last binary file

"Last binary is written with delay."
"Example regional 11km resolution"

4778176548 Dec 15 09:25 nmmb_hst_01_bin_0000h_00m_00.00s
4778176548 Dec 15 09:28 nmmb_hst_01_bin_0001h_00m_00.00s
4778176548 Dec 15 09:31 nmmb_hst_01_bin_0002h_00m_00.00s
4778176548 Dec 15 09:34 nmmb_hst_01_bin_0003h_00m_00.00s
4778176548 Dec 15 09:38 nmmb_hst_01_bin_0004h_00m_00.00s
4778176548 Dec 15 09:41 nmmb_hst_01_bin_0005h_00m_00.00s
4778176548 Dec 15 10:42 nmmb_hst_01_bin_0006h_00m_00.00s
Initial mapping for an experiment with 64 cores where the last 4 ranks are the write tasks

Final mapping
Issue with the last binary file solved

The instrumented execution has no issue...

4778176548 Dec 15 11:14 nmmb_hst_01_bin_0000h_00m_00.00s
4778176548 Dec 15 11:17 nmmb_hst_01_bin_0001h_00m_00.00s
4778176548 Dec 15 11:21 nmmb_hst_01_bin_0002h_00m_00.00s
4778176548 Dec 15 11:24 nmmb_hst_01_bin_0003h_00m_00.00s
4778176548 Dec 15 11:27 nmmb_hst_01_bin_0004h_00m_00.00s
4778176548 Dec 15 11:30 nmmb_hst_01_bin_0005h_00m_00.00s
4778176548 Dec 15 11:33 nmmb_hst_01_bin_0006h_00m_00.00s
The new mapping improved the execution time between 2.73 and 3.85 times.
Processor affinity improved the execution time between 2.8% and 10% (some colleagues reported 20% improvement)
Decomposition (X,Y)

- Usually we use a square decomposition or something close to square.
- It is better to use values to a more rectangular decomposition (i.e. X<<Y). This leads to longer inner loops for better vector and register reuse, better cache blocking, and more efficient halo exchange communication pattern.
New decomposition improved the execution time till 6.5%
Throttling mechanism

An application is developed for many years and sometimes the scientists are not located anymore in the department.

Use gprof (-pg) to figure out number of calls and duration of functions.

Use Intel Fortran compiler with “-g -finstrument-functions” option and create a function list with the following rule, do not instrument the functions that are executed more than 10,000 times and the duration of each call is less than 1ms or 0%.

For example:
00000000008c0230  # module_dynamics_routines_mp_hdiff_
One hour simulation of NMMB, global, 24km, 64 layers

- **meteo**: 9 tracers
- **meteo + aerosols**: 9 + 16 tracers
- **meteo + aerosols + gases**: 9 + 16 + 53
Paraver – Useful computation - Meteo
### One hour simulation of NMMB, global 24km

#### Meteo

<table>
<thead>
<tr>
<th>Functions</th>
<th>Percentage</th>
<th>IPC</th>
</tr>
</thead>
<tbody>
<tr>
<td>rrtm</td>
<td>13.7% - 52%</td>
<td>2.18 - 2.38</td>
</tr>
<tr>
<td></td>
<td>(31.3%)</td>
<td></td>
</tr>
<tr>
<td>gather_layers</td>
<td>8.26% - 13.7%</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>(11.1%)</td>
<td></td>
</tr>
<tr>
<td>scatter_layers</td>
<td>10.6% - 14.1%</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>(12.1%)</td>
<td></td>
</tr>
</tbody>
</table>

#### Meteo + aerosols + chemistry

<table>
<thead>
<tr>
<th>Functions</th>
<th>Percentage</th>
<th>IPC</th>
</tr>
</thead>
<tbody>
<tr>
<td>run_ebi</td>
<td>14% - 20.3%</td>
<td>0.71-1.11</td>
</tr>
<tr>
<td></td>
<td>(16.55%)</td>
<td></td>
</tr>
<tr>
<td>rrtm</td>
<td>3.97% - 15.07%</td>
<td>2.17 – 2.37</td>
</tr>
<tr>
<td></td>
<td>(9.05%)</td>
<td></td>
</tr>
<tr>
<td>gather_layers</td>
<td>12.37% - 24.55%</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>(16.93%)</td>
<td></td>
</tr>
<tr>
<td>scatter_layers</td>
<td>14.65% - 26.58%</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>(19%)</td>
<td></td>
</tr>
</tbody>
</table>
Simulation: 02/12/2005
Paraver – Global – 24km – Meteo – between radiations
Paraver – Global – 24km – Meteo – radiation
Communication matrix
Paraver – (useful) user functions

<table>
<thead>
<tr>
<th>THREAD 1.1.1</th>
<th>rrtm</th>
<th>coszmn</th>
<th>rdtemp</th>
<th>sedimentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>100.167,178.73 us</td>
<td>21.056.63 us</td>
<td>108.056.07 us</td>
<td>22.548.991.18 us</td>
<td></td>
</tr>
<tr>
<td>88.414,567.01 us</td>
<td>21.016.15 us</td>
<td>116.538.77 us</td>
<td>23.014.879.62 us</td>
<td></td>
</tr>
<tr>
<td>99.061,084.26 us</td>
<td>21.050.96 us</td>
<td>118.806.56 us</td>
<td>22.422.152.93 us</td>
<td></td>
</tr>
<tr>
<td>99.296,557.47 us</td>
<td>21.144.29 us</td>
<td>113.197.49 us</td>
<td>22.703.669.54 us</td>
<td></td>
</tr>
<tr>
<td>101.360,926.82 us</td>
<td>21.000.36 us</td>
<td>114.855.45 us</td>
<td>22.767.577.96 us</td>
<td></td>
</tr>
<tr>
<td>101.527,185.54 us</td>
<td>20.899.77 us</td>
<td>112.238.29 us</td>
<td>23.099.842.97 us</td>
<td></td>
</tr>
<tr>
<td>105.942,158.85 us</td>
<td>20.972.37 us</td>
<td>108.866.92 us</td>
<td>22.602.703.19 us</td>
<td></td>
</tr>
<tr>
<td>100.998,572.75 us</td>
<td>21.024.79 us</td>
<td>120.283.78 us</td>
<td>22.297.857.18 us</td>
<td></td>
</tr>
<tr>
<td>99.923,115.69 us</td>
<td>21.138.73 us</td>
<td>114.058.65 us</td>
<td>23.000.976.69 us</td>
<td></td>
</tr>
<tr>
<td>89.672,261.36 us</td>
<td>20.526 us</td>
<td>112.094.26 us</td>
<td>22.811,064.18 us</td>
<td></td>
</tr>
<tr>
<td>84.678,315.30 us</td>
<td>20.813.17 us</td>
<td>113.146.01 us</td>
<td>22.699,483.99 us</td>
<td></td>
</tr>
<tr>
<td>103.738,349.58 us</td>
<td>21.091.18 us</td>
<td>117.928.27 us</td>
<td>22.577,006.32 us</td>
<td></td>
</tr>
</tbody>
</table>

![User function x thread](new_dust_salt.filter1.prv)

1, 128, 620, 277 us
Paraver – (useful) user functions
Computation load imbalance
Tracer Monotonization

“This routine is designed with a not efficient approach, the serialization can be observed"
Zoom between radiation calls for dust/sea-salt
The execution time with 65 cores is increased by 60% at least (without I/O) but the functions gather/scatter are improved by 5.2 - 5.8 times.
For the extra datapoint we use a domain of 16 x 128 processors instead of 32 x 64
Code vectorization

% Vectorized code to add two vectors
a = rand(1,4);
b = rand(1,4);
c = a + b;

% Non-vectorized version
a = rand(1,4);
b = rand(1,4);
for k = 1:length(a)
  c(k) = a(k) + b(k);
end
<table>
<thead>
<tr>
<th>Rank(s)</th>
<th>Type</th>
<th>Message</th>
<th>From</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-35</td>
<td>Warning</td>
<td>Argument 2 (n) is zero, which is correct but unusual!</td>
<td>Representative location: call MPI_Group_excl (1st occurrence)</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>Error</td>
<td>Argument 4 (source) specifies a rank that is greater then the size of the given communicator. (source=24, communicator size:4)!(Information on communicator: Communicator created at reference 1 size=4, is an intercommunicator remote group has size=32)</td>
<td>Reference 1 rank 35: call MPI_Intercomm_create (1st occurrence)</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>Error</td>
<td>Argument 4 (source) specifies a rank that is greater then the size of the given communicator. (source=8, communicator size:4)!(Information on communicator: Communicator created at reference 1 size=4, is an intercommunicator remote group has size=32)</td>
<td>Reference 1 rank 33: call MPI_Intercomm_create (1st occurrence)</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>Error</td>
<td>Argument 4 (source) specifies a rank that is greater then the size of the given communicator. (source=16, communicator size:4)!(Information on communicator: Communicator created at reference 1 size=4, is an intercommunicator remote group has size=32)</td>
<td>Reference 1 rank 34: call MPI_Intercomm_create (1st occurrence)</td>
<td></td>
</tr>
</tbody>
</table>
OmpSs Programming Model
OmpSs Introduction

Parallel Programming Model
- Build on existing standard: OpenMP
- Directive based to keep a serial version
- Targeting: SMP, clusters, and accelerator devices
- Developed in Barcelona Supercomputing Center (BSC)

Mercurium source-to-source compiler
Nanos++ runtime system
void foo ( int *a, int *b )
{
    for ( i = 1; i < N; i++ ) {
        #pragma omp task in(a[i-1]) inout(a[i]) out(b[i])
        propagate(&a[i-1], &a[i], &b[i]);

        #pragma omp task in(b[i-1]) inout(b[i])
        correct(&b[i-1], &b[i]);
    }
}
Roadmap to OmpSs

NMMB is based on the Earth System Modeling Framework (ESMF)

The current ESMF release (v3.1) is not supporting threads. However, the development version of NMMB uses ESMF v6.3

Post-process broke because of some other issues but it was fixed

The new version of NMMB with OmpSs support has been compiled and is ready to apply and test OmpSs

Current work to be presented at PRACE Scientific and Industrial Conference 2014
Improved I/O (future work)

“Parallel NetCDF written to single files by all MPI tasks.”
Future work

- Use OmpSs programming model
  - Study GPU case
  - Explore Xeon Phi

- Prepare NMMB model for higher resolutions, first milestone is the global model for 12km
  - Improve performance and scale NMMB for thousands of cores

- Fix I/O issue
  - IS-ENES Exascale Technologies & Innovation in HPC for Climate Models workshop
  - Possible collaboration across the community to focus on a global solution
Atmospheric models are far from being perfect. A considerable amount of accurate earth observations is available.

Data assimilation 'optimally' combines models and observations.

http://sds-was.aemet.es/forecast-products/dust-forecasts/compared-dust-forecasts

http://www.wmo.int/pages/prog/gcos/
Data Assimilation – Workflow

Ensemble background

- Ensemble analysis
- Observations

Kalman filter*

short-term forecast

Mean analysis

long-term forecast

* In collaboration with N. Schutgens (Uni. Oxford, UK)
BASH script starts the submission of the assimilation job

- We want all the ensembles to be executed in parallel
- We have 40 ensembles, we provide 20 cores for each execution and one ensemble for long-forecast. We should need totally 82 nodes (1,312 exclusive cores)

Now, we need 52 nodes (832 cores), ~36% less resources
Thank you!

For further information please contact
georgios.markomanolis@bsc.es