Computational Profiling Analysis for Climate and Weather

Mario C. Acosta and Xavier Yepes

Summer School on Effective HPC
Computational Profiling Analysis for Climate and Weather

• Objectives
  • Define performance analysis fundamentals (objectives, methods, metrics, hardware counters, etc.)
  • Define a methodology to study HPC performance for numerical models, know your enemy.
  • Describe the BSC performance analysis tools suite (Extrae, Paraver, Dimemas)
  • Interpret uses cases from Earth System Models (HARMONIE, IFS, NEMO, etc.) that illustrate how to identify and solve performance issues
  • Apply profiling techniques to identify performance bottlenecks in your code
  • Summarise typical performance problems
  • Discuss specific knowledge about performance analysis applied to earth system modelling
Currently, **only computational models have the potential** to provide geographically and physically consistent estimates.
Introduction
Introduction
Introduction
Introduction
Introduction

• To be able to use the computing power of modern supercomputers, applications must exploit parallelism.
• Parallelism produce overhead (extra computation and communications)
  – “Overhead does not look a problem in my model” → But if the needs increase (i.e. higher resolutions), a bad implementation will be a problem in some point.
  – We need a method to evaluate the parallelism efficiency of our computational models.
    • When the hardware change
    • When the number of resources change
    • When the model complexity change
    • When the resolution change
    • …
Introduction

Efficiencies: \( \sim [0,1] \)
Multiplicative model
Introduction

• The necessary refactoring of numerical codes is given a lot of attention and is stirring a number of discussions.
  – Computational performance analysis and new optimizations are needed for actual numerical models.
  – Study new algorithms for the new generation of high performance platforms (path to exascale).

• Several European institutions and projects working together on the same direction (ESCAPE2, ESIWACE2, IS-ENES3, ETP4HPC…)
CES-Performance Team & ES Department

- Knowledge about the mathematical and computational side of Earth System Applications
- Knowledge about the specific needs in HPC of the Earth System Applications
- Researching about HPC methods specifically used for Earth System Applications
Methodology

1. Start
2. Computational and mathematical study
3. Scalability Study
4. Performance analysis (profiling and/or tracing)
5. Optimization
6. Results verification
7. Execution time measurement
   - Is it good enough?
     - No
     - Yes: Integrate optimization
       - Is overall speedup good?
         - No
         - Yes: End
Methodology

• Mathematical study
  – Some methods could be better than others
    • Discretization used (explicit, implicit, semi-implicit…)
    • Parallel adaptation (solvers, preconditioners…)
  – How to implement new algorithms for new architectures

• Computational study
  – Achieve load balance among components
  – Reduce overhead introduced by parallel applications
  – Assure that the computational algorithm takes advantage of the architecture
Possible load balance of coupled components of an Earth System Model
Methodology

- Since 1991
- Based on traces
- Open Source: http://www.bsc.es/paraver
- Extrae: Package that generates Paraver trace-files for a post-mortem analysis
- Paraver: Trace visualization and analysis browser
  - Includes trace manipulation: Filter, cut traces
- Dimemas: Message passing simulator
Methodology

- Introducing optimizations
  - Improvement of the mathematical and/or computational algorithm
    - Apply scientific methods which are found in the literature
    - Improve the method with a new approach
    - Revolution: Create a new (and better) algorithm taking into account the research line followed
Methodology

- Reproducibility study
  - Evaluate if the accuracy and reproducibility of the model is similar using or not the optimizations proposed
  - Take into account the nature of climate models
    - How to evaluate, in parallel executions, if the differences between runs are significant or not.
Methodology

- Reproducibility study
  - Evaluate if the accuracy and reproducibility of the model is similar using or not the optimizations proposed
  - Take into account the nature of climate models
    - How to evaluate, in parallel executions, if the differences between runs are significant or not.

Kolmogorov-Smirnov differences of two 5-members ensembles
Profiling Analysis: BSC Tools

- BSC Tools
  - General description
  - Extrae
    - General description
    - How to use it
  - Paraver
    - General description
    - How to use it
    - Configurations available
  - Dimemas
    - General description
    - How to work with large traces
      - Filtering/Burst mode
      - Cutting
BSC Tools

- Since 1991
- Based on traces
- Open Source → http://www.bsc.es/paraver
- Extrae: Package that generates Paraver-trace files for a post-morten analysis
- Paraver: Trace visualization and analysis browser
- Dimemas: Message passing simulator
  - Include traces manipulation: Filter, cut traces...
**CORE TOOLS**

- **EXTRAIE**
  Instrumentation framework to generate execution traces of the most used parallel runtimes.
  Version 3.4.3 • 2.32 MB

- **PARAVER**
  Expressive powerful and flexible trace visualizer for post-mortem trace analysis.
  Version 4.6.3 • 1.56 MB

- **DIMEMAS**
  High-abstracted network simulator for message-passing programs.
  Version 5.3.0 • 0.93 MB

**PERFORMANCE ANALYTICS**

- **CLUSTERING**
  Automatically expose the main performance trends in applications’ computation structure.
  Version 2.6.6 • 1.97 MB

- **TRACKING**
  Analyze how the behavior of a parallel application evolves through different scenarios.
  Version 2.6.5 • 1.98 MB

- **FOLDING**
  Combined instrumentation and sampling for instantaneous metric evolution with low overhead.
  Version 1.3.1 • 1.28 MB

- **SPECTRAL**
  Signal processing techniques to select representative regions from ParaVer traces.
  Version 3.4.0 • 0.3 MB

- **BASIC ANALYSIS**
  Framework for automatic extraction of fundamental factors for ParaVer traces.
  Version 0.2 • 66.41 MB

---

https://tools.bsc.es/downloads
These six tutorials can be opened with wxParaver versions newer than 4.3.0, and you’ll be able to follow the steps within the tool. To install them, download and untar the package and follow the instructions of the Help/Tutorial option on the Paraver main window. You can download them in a single package either in `.tar.gz` format (127 Mb) or `.zip` format (127 Mb).

- **Paraver introduction (MPI)** Start here to familiarise with Paraver basic commands and the first steps of a performance analysis.
- **Dimemas introduction** The basic steps to learn how to configure and run the Dimemas simulator and to start looking at the results.
- **Introduction to Paraver and Dimemas methodology** This tutorial presents different ways to analyze a MPI application through well-known rules, their diagnosis and how they impact on your exploration (no traces included).
- **Methodology** This tutorial shows some examples of the analysis that can be done using the provided configuration files.
- **Tutorial on HydroC analysis (MPI, Dimemas, CUDA)** One example of performance analysis of the MPI application Hydro and further simulations with Dimemas.
- **Trace preparation** Look at this tutorial to select a representative region for a large trace that cannot be loaded into memory.
- **Trace alignment tutorial** If you identify some unexpected unalignment or backwards communications, use this tutorial to learn how to correct shifts between processors.

**Methodology of analysis**

**MPI+OpenMP Performance Analysis tips**

**Tutorial slides**

**Introduction**

<table>
<thead>
<tr>
<th>Core tools</th>
<th>Advanced features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paraver, Detailed material</td>
<td>Tools scalability</td>
</tr>
<tr>
<td>Dimemas</td>
<td>Clustering</td>
</tr>
<tr>
<td>Extrae</td>
<td>Sampling</td>
</tr>
</tbody>
</table>
BSC Tools: Extrae

- Trace generation

Instrumentation

Analysis
BSC Tools: Extrae

- Trace Generation: Set Environment
  - Module load extrae
    - load extrae 3.X.0 (PATH, EXTRAE_DIR, EXTRAE_ROOT, EXTRAE_LIB)
  - Job script → trace-fortran.sh | trace-c.sh
  - Extrae config → extraeMPI.xml | extraeMPI+OMP.xml
  - Files modified for model→ run_parallel.sh
BSC Tools: Extrae

- Job script: trace-fortran.sh

- Available loading extrae module

```bash
#!/bin/bash

# Workaround for tracing in MN3, make TMPDIR point to an existing dir
if [ ! -z "${TMPDIR}" ]; then
    export TMPDIR=${TMPDIR}/extrae
    mkdir -p $TMPDIR
#fi

EXTRAE_ROOT=/usr/local/apps/extrae

if [ -z "${EXTRAE_DIR}" ]; then
    echo "ERROR: EXTRAE_DIR not set, maybe extrae module not loaded?"
    exit 1
fi

if [ -z "${OMP_TRACE}" ]; then
    export LD_PRELOAD=${EXTRAE_DIR}/lib/libmpitracef.so
    if [ -z "${EXTRAE_CONFIG_FILE}" ]; then
        export EXTRAE_CONFIG_FILE=${EXTRAE_ROOT}/xml/MPI/extrae.xml
    fi
else
    export LD_PRELOAD=${EXTRAE_DIR}/lib/libompitracef.so # For Fortran apps
    if [ -z "${EXTRAE_CONFIG_FILE}" ]; then
        export EXTRAE_CONFIG_FILE=${EXTRAE_ROOT}/xml/MPI+OMP/extrae.xml
    fi
fi
```

True if openmp is used

Extrae config by default
Extrae config:extrae.xml

Available using nama_CY43R1_IFS_traces branch

- Activate MPI tracing and emit hardware counters at MPI calls
- Activate OpenMP tracing
- Emit call stack information (number of levels) at acquisition point
- Add instrumentation at specific user functions
- PAPI counters used

```xml
<?xml version='1.0'?>

<trace enabled="yes" home="/usr/local/apps/extrae/3.3.0rc/MPICH2-6.3.1" initial-mode="detail" type="paraver" xml-parser-id="Id: xml-parse.c 3893 2016-02-04 12:30:232 harald s"

  <mpi enabled="yes">
    <counters enabled="yes"/>
  </mpi>

  <openmp enabled="no">
    <locks enabled="no"/>
    <counters enabled="yes"/>
  </openmp>

  <PTHREAD enabled="no">
    <locks enabled="no"/>
    <counters enabled="yes"/>
  </PTThread>

  <Callers enabled="yes">
    <mpi enabled="yes">1-5</mpi>
    <sampling enabled="yes">1-5</sampling>
    <dynamic-memory enabled="no">1-3</dynamic-memory>
  </callers>

  <user-functions enabled="yes" list="/perm/rd/nama/extrae/xml/MPI/functions_for_xml.txt" exclude-automatic-functions="no">
    <counters enabled="yes"/>
  </user-functions>

  <counters enabled="yes">
    <cpu enabled="yes">sampling-set-distribution="1"</cpu>
    <set enabled="yes" domain="all" change-at-time="0">
      PAPI_TOT_INS, PAPI_TOT_CYC, PAPI_L1_DCM, PAPI_L2_DCM, PAPI_L3_TCM, PAPI_FP_INS, PAPI_BR_INST, PAPI_FP_OPS
    </set>
    <set enabled="yes" domain="all" change-at-time="0">
      PAPI_TOT_INS, PAPI_TOT_CYC, PAPI_LD_INS, PAPI_SR_INS, PAPI_BR_UA, PAPI_BR_CN, PAPI_VEC_SP, RESOURCE_STALLS
    </set>
    <set enabled="yes" domain="all" change-at-time="0">
      PAPI_TOT_INS, PAPI_TOT_CYC
    </set>
  </counters>
</trace>
```
BSC Tools: Extrae

- Extrae config: extrae.xml

- Available using nama_CY43R1_IFS_traces branch

```xml
<storage enabled="no">
  <trace-prefix enabled="yes">TRACE</trace-prefix>
  <size enabled="no">5</size>
  <temporal-directory enabled="yes">/scratch</temporal-directory>
  <final-directory enabled="yes">/gpf/scratch/bsc41/bsc41273</final-directory>
</storage>

<buffer enabled="yes">
  <size enabled="yes">500000</size>
  <circular enabled="no" />
</buffer>

<trace-control enabled="no">
  <file enabled="no" frequency="5M">/gpf/scratch/bsc41/bsc41273/control</file>
  <global-ops enabled="no" />
  <remote-control enabled="no"/>
  <signal enabled="no" which="USR1" />
</trace-control>

{others enabled="yes"}
  <minimum-time enabled="no">10M</minimum-time>
  <finalize-on-signal enabled="yes">
    SIGUSR1="no" SIGUSR2="no" SIGINT="yes" SIGQUIT="yes" SIGTERM="yes" SIGXCPU="yes" SIGPIPE="yes" SIGSEGV="yes" SIGABRT="yes"
  </finalize-on-signal>
  <flush_sampling-buffer-at-instrumentation-point enabled="yes" />
</others>

<bursts enabled="no">
  <threshold enabled="yes">5000</threshold>
  <mpi_statistics enabled="yes" />
</bursts>

<sampling enabled="no" type="default" period="50m" variability="10m" />
<alloc enabled="yes" threshold="32768" />
<free enabled="yes" />
</dynamic-memory>

<input enabled="no" />

<merge enabled="yes">
  <synchronization disabled>
    core = null
    max-memory="32768"
    joint-states="yes"
    keep-mouths="yes"
    sort-addresses="yes"
    overwrite="yes"
  </synchronization>
</merge>
```

- Emit computation burst of a minimal duration
- Plus summarized MPI events
- Merge individual traces automatically
BSC Tools: Extrae

- Files modified for run_parallel
  - If BSCTRACE=1 → Extrae is used

```bash
if [[ %BSCTRACE:0% = 1 ]]; then
  command="aprun -cc cpu $arg $N $submit_total_tasks -N $submit_tasks_per_node $D $arg -j $submit_cpus_per_compute_unit -d $omp_num_threads $submit_force_numa_memory_affinity $(whence $command) $args"
else
  command="aprun -cc cpu $arg $N $submit_total_tasks -N $submit_tasks_per_node $D $arg -j $submit_cpus_per_compute_unit -d $omp_num_threads $submit_force_numa_memory_affinity $(whence $command) $args"
fi

export EC_LD_LIBRARY_PATH=${PIFS_LD_LIBRARY_PATH}:$LIBS
if [[ %BSCTRACE:0% = 1 ]]; then
  LOAD_MODULE extrae
  export EXTRA_CONFIG_FILE=/perm/rd/nama/extrae/xml/MPI/extrae.xml
  export TRACEDIR=${WDIR}/bsctrace.${TASK}.${ECF_TRYNO}.${T}
  export EC_LD_LIBRARY_PATH=${EC_LD_LIBRARY_PATH}:${EXTRAEXE_LIB}
  if [[ %OMPTRACE:0% = 1 ]]; then
    export OMP_TRACE=1
    fi
  mkdir $TRACEDIR
  fi
  (export LD_LIBRARY_PATH=${EC_LD_LIBRARY_PATH}; eval $command)
fi
if [[ %BSCTRACE:0% = 1 ]]; then
  mv *.prv *.pcf *.row $TRACEDIR
fi
```

Parameter to activate profiling

Trace files generated
BSC Tools: Paraver

- Trace visualization/analysis + trace manipulation
- Timelines
- 2/3D tables (Statistics)
- Goal = Flexibility
  - No semantics
  - Programmable
- Comparative analyses
  - Multiple traces
  - Synchronize scales
**BSC Tools: Paraver**

- **Paraver traces:** made up from records (timestamp + event or activity) of three different kind:
  - **State records:** intervals of thread status, i.e., waiting in a barrier (either MPI or OpenMP), waiting for a message, computing...
  - **Event records:** punctual event occurred in a given timestamp, as entry & exit points of user functions, MPI routines, OpenMP parallel regions...
  - **Communication records:** relationship between two objects, as communication between two processes (MPI), task movement among threads (OpenMP/OmpSs) or memory transfers (CUDA/OpenCL).
How a trace looks like: basic overview

Timeline

MPI processes

Computation

Communication

MPI call color legend

- Outside MPI
- MPI_Recv
- MPI_Isend
- MPI_Irecv
- MPI_Wait
- MPI_Alltoallv
- MPI_Comm_size
- MPI_Waitany
BSC Tools: Paraver

- From timelines to tables

MPI calls

MPI calls profile

Useful Duration

Histogram Useful Duration
BSC Tools: Paraver

One columns per specific value of categorical Control window

Value/color is a statistic computed for the specific thread when control window had the value corresponding to the column

Relevant statistics:
Time, %time, #bursts, Avg. burst time
Average of Data window
MPI calls and profile

- Different types of MPI functions are quantified
- In this case, only the MPI_Alltoallv and MPI_Waitany functions represent a significant amount of time with 14.65% and 9.29% respectively.
Point-to-point connectivity matrix

• It indicates who communicates with whom
• Almost all point-to-point communications are locally performed between MPI processes neighbours
Collective communications

- Four calls to MPI_Alltoallv each time step
- The most significant in terms of size and duration is the second one
BSC Tools: Paraver

Columns correspond to bins of values of a numeric Control window
duration, instructions, BW, IPC, ...

Value/color is a statistic computed for the specific thread when control window had the value corresponding to the column

Relevant statistics:
Time, %time, #bursts, Avg. burst time
Average of Data window
BSC Tools: Paraver

- Semantic functionality
  - Derived windows
    - Point wise operation
      - $S = \alpha * S^a <op> \beta * S^b$
      - $<op> : +, -, *, /, ...$

![Interval between MPI events](image1)

![In MPI call](image2)

![MPI call duration](image3)
BSC Tools: Paraver

- Data handling capability
  - Original trace containing all the events
  - Filtering/Burst mode
    - Subset of records in original trace
    - By duration, time, value, event type
    - Trace filtered can be analysed in the same way
    - Also using burst mode from xml file
      - Save only computation bursts longer than a value
  - Cutting
    - All records in a given time interval
    - Only some processes
BSC Tools: Paraver

- **Filtering**

  - Filter original trace discarding most of the records, only keeping most relevant information (MPI events can be used for this purpose)
BSC Tools: Paraver

- **Cutting**
  - Cut original trace to obtain a fully detailed trace for the time interval considered representative or of interest
  - Use filtered trace to know the area of interest (remember that input must be the original trace)
  - Right click → run → cutter
BSC Tools: Paraver

- Configurations for analysis (usr/local/apps/paraver/X.X.X/cfgs)
  - General
    - Including basic views (timelines) and analysis (2D/3D profiles)
  - Counters_PAPI
    - Hardware counters derived metrics
      - Program: related to algorithm/compilation (instructions, FP ops…)
      - Architecture: related to execution on specific architectures (cache misses…)
      - Performance: metrics reporting rating per time (MIPS, IPC…)
  
  - MPI → Views and analysis of MPI events
  - OpenMP → Views and analysis of OpenMP events
  - Complete Profile (general_cfgs)
BSC Tools: Dimemas

The impossible machine: \( BW = \infty, \quad L = 0 \)

- Actually describes/characterizes intrinsic application behavior
  - Load balance problems?
  - Dependence problems?

Impact on practical machines?
Profiling Methodology

- Area of study
- Deployment efficiency
- Benchmarking
- Profiling analysis
- Validation
Profiling Methodology

• Area of study
  • Configuration used (Operational, New algorithms, Global, Parallelization paradigm...)
  • Components activated and cyclic patterns
    • IO, ICE, Radiation, MPI, OpenMP
  • Area of study
    • 1 complete time step

• Deployment efficiency
• Benchmarking
• Profiling analysis
• Validation
Types of time step for the practical example

Time steps with radiation are much more expensive due to the extra computation in the grid-point part.

Regular time step

Reguar time step plus radiation

Regular time step
Structure of a regular time step

A - Inverse transformations
B - Grid-point computations
C - Direct transformations
D - Spectral computations
Profiling Methodology

- Area of study (IFS)
  - 24 hours of simulation, T511L137 on CCA (ECMWF)
  - Selected 1 time step: 104 MPI processes + 4 IO (No OpenMP)
  - Metrics collected for large areas of computation automatically
Profiling Methodology

- **Area of study (NEMO)**
  - 1 day of simulation, ORCA025L91 on MN4 (BSC)
  - Selected the fastest time step automatically
  - 1 time step: 72 MPI processes (No IO, No OpenMP, No SI3)
  - Metrics collected for User functions manually
Profiling Methodology
Profiling Methodology

• Area of study
• **Deployment efficiency**
  • Compilation flags
    • Comparing fp options (fast, precise, strict...) and optimization options (OX, vectorization, approximations...)
  • Checking external libraries compilation
  • Debug flags (-g, Optimization reports, -f-instrument-functions...)
• Benchmarking
• Profiling analysis
• Validation
Profiling Methodology

- Area of study
- Deployment efficiency
- **Benchmarking**
  - Basic Tests to collect Hardware metrics
    - Communications (Latency, Bandwidth, CPU, Parallel Efficiency...)
  - Weak and Strong scaling (MPI, OpenMP, Block processing and Hybrid sets)
  - Comparing optimizations (Double VS Single Precision...)
  - Extrae metrics collection and trace production
- Profiling analysis
- Validation
MPI strong scaling: trace views

- 285 MPI (8 nodes)
- 576 MPI (16 nodes)
- 1008 MPI (28 nodes)
- 1800 MPI (50 nodes)
Basic Analysis: MPI Strong Scaling

- Computation and parallel efficiency factors for MPI only:
  - Good computation scalability and serialization efficiency
  - Not very good load balance neither transfer efficiency

<table>
<thead>
<tr>
<th>Global efficiency</th>
<th>75.34</th>
<th>72.04</th>
<th>68.04</th>
<th>69.82</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Parallel efficiency</td>
<td>75.34</td>
<td>71.36</td>
<td>68.37</td>
<td>67.69</td>
</tr>
<tr>
<td>- Load balance</td>
<td>88.05</td>
<td>85.38</td>
<td>87.76</td>
<td>86.87</td>
</tr>
<tr>
<td>- Communication efficiency</td>
<td>85.56</td>
<td>83.58</td>
<td>77.91</td>
<td>77.93</td>
</tr>
<tr>
<td>- Serialization efficiency</td>
<td>96.35</td>
<td>98.23</td>
<td>94.03</td>
<td>96.87</td>
</tr>
<tr>
<td>- Transfer efficiency</td>
<td>88.81</td>
<td>85.09</td>
<td>82.85</td>
<td>80.44</td>
</tr>
<tr>
<td>- Computation scalability</td>
<td>100.00</td>
<td>100.95</td>
<td>99.51</td>
<td>103.15</td>
</tr>
<tr>
<td>- IPC scalability</td>
<td>100.00</td>
<td>102.06</td>
<td>101.86</td>
<td>106.12</td>
</tr>
<tr>
<td>- Instruction scalability</td>
<td>100.00</td>
<td>99.24</td>
<td>98.00</td>
<td>97.55</td>
</tr>
<tr>
<td>- Frequency scalability</td>
<td>100.00</td>
<td>99.67</td>
<td>99.69</td>
<td>99.64</td>
</tr>
</tbody>
</table>
Basic Analysis: Double P VS Single P

Overview of the collected raw data:

<table>
<thead>
<tr>
<th></th>
<th>108</th>
<th>108</th>
</tr>
</thead>
<tbody>
<tr>
<td>Runtime (us)</td>
<td>110741508.76</td>
<td>71238767.9</td>
</tr>
<tr>
<td>Runtime (ideal)</td>
<td>105675625.64</td>
<td>68396939.23</td>
</tr>
<tr>
<td>Useful duration (average)</td>
<td>88427932.03</td>
<td>57382830.24</td>
</tr>
<tr>
<td>Useful duration (maximum)</td>
<td>94410288.2</td>
<td>61484222.58</td>
</tr>
<tr>
<td>Useful duration (total)</td>
<td>9196504931.3</td>
<td>5967814345.21</td>
</tr>
<tr>
<td>Useful duration (ideal, max)</td>
<td>94410288.2</td>
<td>61484222.58</td>
</tr>
<tr>
<td>Useful instructions (total)</td>
<td>26798422515714</td>
<td>23201423473963</td>
</tr>
<tr>
<td>Useful cycles (total)</td>
<td>21985000332874</td>
<td>14299301515415</td>
</tr>
</tbody>
</table>

Overview of the computed model factors:

<table>
<thead>
<tr>
<th></th>
<th>108</th>
<th>108</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parallel efficiency</td>
<td>79.85%</td>
<td>80.55%</td>
</tr>
<tr>
<td>Load balance</td>
<td>93.66%</td>
<td>93.33%</td>
</tr>
<tr>
<td>Communication efficiency</td>
<td>85.25%</td>
<td>86.31%</td>
</tr>
<tr>
<td>Serialization efficiency</td>
<td>89.34%</td>
<td>89.89%</td>
</tr>
<tr>
<td>Transfer efficiency</td>
<td>95.43%</td>
<td>96.01%</td>
</tr>
<tr>
<td>Computation scalability</td>
<td>100.00%</td>
<td>154.10%</td>
</tr>
<tr>
<td>Global efficiency</td>
<td>79.85%</td>
<td>124.13%</td>
</tr>
<tr>
<td>IPC scalability</td>
<td>100.00%</td>
<td>133.11%</td>
</tr>
<tr>
<td>Instruction scalability</td>
<td>100.00%</td>
<td>115.50%</td>
</tr>
<tr>
<td>Frequency scalability</td>
<td>100.00%</td>
<td>100.23%</td>
</tr>
<tr>
<td>Speedup</td>
<td>1.00</td>
<td>1.55</td>
</tr>
<tr>
<td>Average IPC</td>
<td>1.22</td>
<td>1.62</td>
</tr>
<tr>
<td>Average frequency (GHz)</td>
<td>2.39</td>
<td>2.40</td>
</tr>
</tbody>
</table>
Profiling Methodology

- Area of study
- Deployment efficiency
- Benchmarking
- **Profiling analysis**
  - MPI and OpenMP profile summary and Basic Analysis Tool
  - PAPI counters
  - MPI and OpenMP evaluation in detail
  - Clustering and Tracking Tools
  - Sampling and Folding Tools
  - Connection to the code
  - Dimemas Tool
- Validation
MPI Profile Summary

Parallel and Communication efficiency, Global load balance → less than 85%?

Parallel Efficiency

<table>
<thead>
<tr>
<th></th>
<th>Outside MPI</th>
<th>MPI_Send</th>
<th>MPI_Recv</th>
<th>MPI_Send</th>
<th>MPI_Recv</th>
<th>MPI_Wait</th>
<th>MPI_Barrier</th>
<th>MPI_Allreduce</th>
<th>MPI_Wait</th>
<th>MPI_Gather</th>
<th>MPI_Comm_rank</th>
<th>MPI_Comm_size</th>
<th>MPI_Bsend</th>
<th>MPI_Waitany</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
<td>66.578.44%</td>
<td>1.71%</td>
<td>773.76%</td>
<td>646.21%</td>
<td>239.35%</td>
<td>12.36%</td>
<td>806.93%</td>
<td>10,757.31%</td>
<td>35.56%</td>
<td>2.49%</td>
<td>448.23%</td>
<td>0.81%</td>
<td>7.746.82%</td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td>66.31%</td>
<td>0.00%</td>
<td>0.77%</td>
<td>0.64%</td>
<td>0.24%</td>
<td>12.31%</td>
<td>0.80%</td>
<td>10.71%</td>
<td>0.04%</td>
<td>0.00%</td>
<td>0.45%</td>
<td>0.81%</td>
<td>7.72%</td>
<td></td>
</tr>
<tr>
<td>Maximum</td>
<td>72.93%</td>
<td>0.01%</td>
<td>2.99%</td>
<td>1.60%</td>
<td>0.80%</td>
<td>18.56%</td>
<td>1.34%</td>
<td>25.06%</td>
<td>1.12%</td>
<td>0.01%</td>
<td>1.88%</td>
<td>0.81%</td>
<td>19.25%</td>
<td></td>
</tr>
<tr>
<td>Minimum</td>
<td>57.05%</td>
<td>0.00%</td>
<td>0.01%</td>
<td>0.08%</td>
<td>0.07%</td>
<td>3.11%</td>
<td>0.00%</td>
<td>5.25%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.16%</td>
<td>0.81%</td>
<td>0.31%</td>
<td></td>
</tr>
<tr>
<td>StdDev</td>
<td>2.03%</td>
<td>0.00%</td>
<td>0.57%</td>
<td>0.36%</td>
<td>0.06%</td>
<td>2.52%</td>
<td>0.41%</td>
<td>3.57%</td>
<td>0.12%</td>
<td>0.00%</td>
<td>0.10%</td>
<td>0%</td>
<td>3.18%</td>
<td></td>
</tr>
<tr>
<td>Avg/Max</td>
<td>0.91%</td>
<td>0.31%</td>
<td>0.26%</td>
<td>0.40%</td>
<td>0.30%</td>
<td>0.66%</td>
<td>0.44%</td>
<td>0.43%</td>
<td>0.03%</td>
<td>0.34%</td>
<td>0.24%</td>
<td>1%</td>
<td>0.40%</td>
<td></td>
</tr>
</tbody>
</table>

Global Load Balance

Communication Efficiency

IFS
PAPI Counters

- PAPI counters collected during the execution
- Some of them are based on other native PAPI counters and derived from the base metrics

<table>
<thead>
<tr>
<th></th>
<th>Derived</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instructions</td>
<td></td>
</tr>
<tr>
<td>Cycles</td>
<td></td>
</tr>
<tr>
<td>Useful Duration</td>
<td>X</td>
</tr>
<tr>
<td>Useful Instructions</td>
<td>X</td>
</tr>
<tr>
<td>Useful IPC</td>
<td>X</td>
</tr>
<tr>
<td>Loads</td>
<td></td>
</tr>
<tr>
<td>Stores</td>
<td></td>
</tr>
<tr>
<td>L3/L2/L1_Total_Misses</td>
<td></td>
</tr>
<tr>
<td>L3/L2/L1_MISS_RATIO</td>
<td>X</td>
</tr>
<tr>
<td>FP_OPS</td>
<td></td>
</tr>
<tr>
<td>FP_TOT_INS</td>
<td></td>
</tr>
<tr>
<td>INS_VEC</td>
<td>X</td>
</tr>
</tbody>
</table>
PAPI Counters
PAPI Counters
PAPI Counters

MPI Events

IPC

L1 Misses per 1000 INS
MPI evaluation

Fourier Trans.  Legendre Trans.
MPI evaluation

- IPC less than 1 for calculation areas?
- Are there load imbalance regions?
MPI evaluation

- Are MPI communications efficient according to the map affinity?
Clustering Tool

Applying Clustering for an automatic profiling analysis

- Characterizes computing bursts that are similar and groups them into clusters
- Allows to study the behavior of the clusters separately, identify patterns, etc.
Tracking Tool

• A friendly way to quantify and visualize the evolution of the clusters among several traces
• The tool has 2 parts
  • Recognition algorithm of “who-is-who”, based on heuristics
  • A visualization GUI
• Examples analyzing multiple traces
  • Scaling number of MPI/OpenMP resources (64 – 128 – 256…)
  • Testing different microarchitecture features
  • Changing the problem size
  • Trying different compiler optimizations
Tracking Tool
Tracking Tool

Tracking IFS MPI+OMP Strong Scaling
Sampling Tool

• Extrae can be configured to capture performance metrics on a periodic basis using alarm signals and specifying period and variability (10 and 2 respectively for IFS and NEMO tests).
• This means that we will capture samples every 10 ms with a random variability of 2 ms.

• Every sample contains processor performance counters (where every PAPI counter is referred at configured time) and callstack information.
Folding Tool

- Combine instrumentation and sampling to provide **instantaneous performance metrics**, source code and memory references. This mechanism receives a trace-file and generates plots showing the fine evolution of the performance.
- The samples collected are gathered from scattered computing regions into a synthetic region by preserving their relative time within their original region so that the sampled information determines how the performance evolves within the region.
- The performance evolution is connected to source code and memory references at the same time.
Folding Tool

![Graph showing normalized PAPI_{T,NS} rate over time. The graph includes a line chart with mean PAPI_{T,NS} rate, and markers for used samples, excluded samples, and unused samples. The x-axis represents time (ms) ranging from 0.00 to 2293.65, and the y-axis represents normalized PAPI_{T,NS} rate ranging from 0.00 to 1.00. The graph also includes a line indicating the fitted model using Kriger with a nugget of 1.0e-04.]
Folding Tool
Folding Tool
Folding Tool

TOT_INS

TOT_CACHE_MISSES

Connection to the code

USER_FUNCTION_LINE
DIMEMAS Tool

The impossible machine: \( BW = \infty, \quad L = 0 \)

- Actually describes/characterizes intrinsic application behavior
  - Load balance problems?
  - Dependence problems?

Impact on practical machines?
DIMEMAS Tool

Ideal Network for IFS execution

• Actual run

• Ideal network
Profiling Methodology

- Area of study
- Deployment efficiency
- Benchmarking
- Profiling analysis
- **Validation**
  - Reproducibility Test
  - Validation Test
Validation

Reproducibility Test: Are your results comparable to the EC-Earth community results?

The Test proposed:

- **20-yr long, 5-member, Forcing Fixed Cmip and Amip simulations**
  - Allows to look at impact of machine on mean state/bias (not possible in the case of 1-yr simulations)
  - Allows to measure differences due hardware as compared to internal variability
  - Working under stationary conditions removes possible dependence of hardware impact on the mean state
  - Addresses the problem from a global point-of-view; suitable to give recommendations for CMIP6

The results comparing platforms or configurations:

- **AMIP platform (Rhino;CCA) comparison**
  - Kolmogorov-Smirnov differences of two 5-members ensambles
Validation Test (NEMO)

- Initial conditions perturbed with white noise in the 3D temperature field.
- Evaluating 53 output variables.
Validation Test (NEMO)

- Initial conditions perturbed with white noise in the 3D temperature field.
- Evaluating 53 output variables.
Validation Test (NEMO)

Example: Compiling with -xHost
Examples
Examples

Border Exchange
Examples

• Diagnostic for NEMO:
  – Scalability is constrained by:
    • 1) Algorithms with too much communication
    • 2) Sub-optimal implementation

• Actions taken
  – Improve communication implementation to reduce number of point-to-point messages
  – Reduce number of collectives
Examples

- First studies showed that IFS-NEMO coupling was not a big issue
- But it seems that it is when increasing number of cores
Examples

- BSC has been working successfully with the EC-Earth Technical Working Group to improve the execution of the model.

- A success case: coupling field gathering and OPT option of OASIS coupler for global conservative transformations.

- With these optimizations, up to 90% improvement in coupling process can be achieved.

- These improvements are now in trunk EC-Earth 3.2.2, substantially benefiting our CMIP6 simulations.
- BSC has been working successfully with the EC-Earth Technical Working Group to improve the execution of the model.

- A success case: coupling field gathering and OPT option of OASIS coupler for global conservative transformations.

- With these optimizations, up to 90% improvement in coupling process can be achieved.

- These improvements are now in trunk EC-Earth 3.2.2, substantially benefiting our CMIP6 simulations.
Examples

- Synchronal point to point communication could be a bottleneck even for only one message from one master to hundreds of slaves

  - Sigcheck method

- Using one asynchronal collective communication this time is reduced almost to 0
Examples

• Hybrid Test (128 MPI+4 OpenMP, Total: 512)

One Complete Time Step

Very small granularity of the OpenMP parallel Regions
Examples

- Small OpenMP parallel Regions
- 128 MPI processes and 4 OpenMP threads per process
- Only one coarse OpenMP parallel region
- 18% of reduction

Thank you

The research leading to these results has received funding from the EU H2020 Framework Programme under grant agreement H2020 GA 675191.

The content of this presentation reflects only the author’s view. The European Commission is not responsible for any use that may be made of the information it contains.

mario.acosta@bsc.es