Exploring EC-Earth 3.2-Beta performance on the new ECMWF Cray-Broadwell

Souhail Boussetta, Cristian Simarro, Dominique Lucas

Research Department & Forecast Department

May 2016
Abstract

EC-Earth is a state-of-the-art earth system model composed with multiple elements. In this report we focus on the climate system parts of the 3.2beta release which consists of the IFS atmospheric model, NEMO version 3.6 as ocean circulation model which is interfaced with the LIM3 sea-ice model and a river Runoff mapper. These components are linked via the OASIS3-MCT coupler and use the XIOS for the IO management. The EC-Earth 3.2beta is ported to the new Broadwell-based High Performance Computing Facility of the ECMWF using both the Intel and the Cray compilers. The performance of the coupled system is assessed and shows that, in general the use of the Cray compiler can result in an increase in the system performance up to 34%. Recommendable configurations for the number of MPI tasks to use for each components are suggested based on scaling experiments. It is also shown that compared to the Ivy-Bridge-based machine, the new Broadwell-based machine would allow the EC-Earth consortium to benefit from more extensive resources without losing in term of running speed.

1 Introduction

EC-Earth 3.2beta is the latest release of the global state-of-the-art Earth system model EC-Earth (Hazeleger et al., 2011), which is built based on the atmospheric prediction model IFS (Integrated Forecasting System) of the European Centre for Medium-range Weather Forecasts (ECMWF). This release is the base for the tuning exercise towards the CMIP6 project simulations. On July 2016, the ECMWF High Performance Computing Facility (HPCF) will be fully switched to a new Cray system based on Broadwell processors. In this reports we attempt to describe how to port and run the EC-Earth3.2beta coupled system to the new machine, assess its performance and suggest eventual appropriate configurations to run it based on the available testing nodes.

2 System components

EC-Earth global climate system consists of the following main components:

*The OASIS coupler:*

The OASIS coupler is a software allowing synchronized exchanges of coupled information between the different models representing the climate system (interpolation and exchange of coupled fields). EC-Earth 3.2beta relies on OASIS3-MCT (S. Valcke et al., 2015), which consists of the OASIS3 interfaced with the Model Coupling Toolkit (MCT). The advantage of this new version is that it is fully parallel and works as a library linked with the main components of the system. OASIS is used by approximately 30 climate modelling and weather prediction groups in Europe, USA, Canada, India, Australia and China. It is developed and maintained by CERFACS (France), DKRZ (Germany), and CNRS (France) under the framework of the EU FP7 IS-ENES with a main focus on portability and flexibility.

*The atmospheric model IFS:*

The Integrated Forecasting system (IFS) is the atmospheric model of the ECMWF which also includes the HTESSEL land-surface model (Balsamo et al., 2009; Boussetta et al., 2013). EC-Earth 3.2beta is based on CY36R4 release of the IFS (IFS doc, 2012) in addition to few important physical adjustments.
mainly related to the non-orographic gravity wave drag parameterization (latitudinal and resolution dependency of the launch momentum flux) and the diurnal cycle of convection (Bechtold et al. 2014).

*The Ocean model NEMO*

NEMO (Nucleus for European Modelling of the Ocean) is a general ocean circulation model (Madec et al., 2012) based on Navier-Stokes equation, having as Prognostic variables: the three-dimensional velocity field, a linear or non-linear sea surface height, the temperature and the salinity. The ocean within NEMO is interfaced with the LIM sea-ice model (Vancoppenolle et al., 2009) and with PISCES (Aumont et al., 2015) the biogeochemical model. EC-Earth 3.2beta uses NEMO version 3.6 and version 3 of the sea-ice model LIM3. NEMO is developed and maintained by a consortium composed with members from CNRS and Mercator-Ocean (France) and UKMO and NERC (United Kingdom).

*The XIOS server*

XIOS stands for Xml IO Server, it is a dedicated Library to manage I/O for climate codes (Meurdesoif Y., 2015). It allows management of output diagnostic and history files, it also permits temporal and spatial post-processing operations (averaging, max/min, instant, etc…).

Version 3.6 of NEMO relies on the XIOS server to manage its I/O. This new IO server can take advantage of the parallel I/O functionality of NetCDF4 to create a single output file. This can greatly reduce the post-processing burden associated with using large numbers of NEMO processors. The XIOS server is developed by Yann Meurdesoif from LSCE-IPSL (France).

*Runoff-MAPPER*

For mass conservation issues, a fourth component is used to distribute runoff from land over the ocean using river basins. This process relies on a gather-scatter operation with intermediate steps conversions between flux and mass.

### 3 Machine specification:

At the time of writing, the HPCF at ECMWF is being upgraded to use Intel Broadwell processors. Table 1 hereafter includes the configuration of the new system (Phase 2), together with the configuration of the existing system (Phase 1). The experiments described in this study are implemented on the first Broadwell nodes available for testing on one of the Cray clusters.

The Ivy-Bridge-based machine is a Cray XC30 having a node architecture of 2x12 Intel Ivy-Bridge processors with 64 GB of DDR3 memory. While the Broadwell-based machine is a Cray XC40. This new architecture uses not only a newer Intel family processor but also more dense nodes: 2x18 Intel Broadwell processors with 128 GB of DDR4 memory. Both systems use the same Cray Aries interconnect network and LUSTRE parallel files system.
Table 1: Comparison between the actual Cray machine and the new Broadwell-based Cray machine

<table>
<thead>
<tr>
<th>Phase</th>
<th>Phase 1</th>
<th>Phase 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sustained Performance (teraflops)</td>
<td>200</td>
<td>320</td>
</tr>
<tr>
<td>Peak performance (teraflops)</td>
<td>3593</td>
<td>~8500</td>
</tr>
<tr>
<td>Processor technology</td>
<td>Intel Ivy-Bridge</td>
<td>Intel Broadwell</td>
</tr>
<tr>
<td>Parallel application nodes (per cluster)</td>
<td>3400</td>
<td>3510</td>
</tr>
<tr>
<td>Pre-/Post-processing nodes</td>
<td>104</td>
<td>104</td>
</tr>
<tr>
<td>Cores per node</td>
<td>24 (2x12)</td>
<td>36 (2x18)</td>
</tr>
<tr>
<td>Total compute cores (per cluster)</td>
<td>84096</td>
<td>130104</td>
</tr>
<tr>
<td>Memory per node (GiB)</td>
<td>64 (1866 MHz DDR3)</td>
<td>128 (2400 MHz DDR4)</td>
</tr>
<tr>
<td>External login nodes</td>
<td>2 x Ivy-Bridge</td>
<td>2 x Ivy-Bridge, 1 x Haswell</td>
</tr>
<tr>
<td>Clock frequency (GHz)</td>
<td>2.7</td>
<td>2.1</td>
</tr>
<tr>
<td>Storage capacity (petabytes)</td>
<td>15</td>
<td>20</td>
</tr>
<tr>
<td>Floating Point Instruction set</td>
<td>AVX</td>
<td>AVX2</td>
</tr>
</tbody>
</table>

4 Performance Assessment

A number of experiments is carried out with different nodes settings to seek the configuration that gives the best performance to the EC-Earth models in terms of runtime and cost. For parallel jobs, the ECMWF currently charges the utilization of its HPCF resources as a factor of the used number of nodes and the elapsed time. For illustration of the cost also referred as System Billing Unit (SBU), one CPUcore*hour is about 16 SBUs.

The models resolutions used in these experiments, which are the standard setting for EC-Earth3.2beta, are the following:

Spatial resolution:

IFS: The IFS uses a horizontal spectral resolution of T255 with 91 vertical levels (T255L91).

NEMO: NEMO uses the ORCA-1 configuration which has about 1 degree horizontal resolution and 75 vertical levels (ORCA1L75).

Time resolution:

IFS, NEMO and LIM3 use the same time-step of 2700 s and a coupling frequency of 2700 s. Depending on the type of experiment, 10-days and 3-months runs are performed.

4.1 IFS standalone

In order to save resources and to mimic as close as possible the operational conditions, a first assessment is performed on a set of IFS standalone 10-day runs to seek its optimum configuration and then the coupled IFS+NEMO system is run based on the outcomes of the IFS standalone experiments.
Two groups of experiments are implemented, the firsts using the Intel-compiled models and the seconds using the Cray-compiled ones. (Details on the configurations of the compilation and runs using these compilers are shown in the appendix).

Figure 1: IFS Standalone performance on the Broadwell-based nodes using the Cray compiler (solid lines) and the Intel compiler (dashed lines), runtime results are in red and system billing units in blue.

Figure 1 shows that for IFS, it is not suitable (a waste of resources) to set more than 12 nodes (432 MPI Tasks). It also shows that using 6, 8, 10 and 12 nodes could be recommendable options depending on user needs (Table 2). In addition, Table 2, shows that using the Cray compiler improves the IFS running performance by 8 to 15% compared to the Intel compiler. These results are consistent with the study by Struthers (2016) who compared the performance if the IFS on the Cray XC40 (Beskow) of the Swedish national supercomputer centre (NSC) and found that Cray-compiled IFS scales better than the Intel-compiled one.
Table 2: IFS standalone performance for 10-days run with different MPI tasks/Nodes configurations

<table>
<thead>
<tr>
<th>MPI tasks</th>
<th>nodes</th>
<th>Intel Compiler</th>
<th>Cray Compiler</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>RT (s)</td>
<td>SBU</td>
</tr>
<tr>
<td>72</td>
<td>2</td>
<td>546</td>
<td>178</td>
</tr>
<tr>
<td>108</td>
<td>3</td>
<td>399</td>
<td>195</td>
</tr>
<tr>
<td>144</td>
<td>4</td>
<td>325</td>
<td>212</td>
</tr>
<tr>
<td>180</td>
<td>5</td>
<td>277</td>
<td>225</td>
</tr>
<tr>
<td>216</td>
<td>6</td>
<td>247</td>
<td>241</td>
</tr>
<tr>
<td>252</td>
<td>7</td>
<td>230</td>
<td>262</td>
</tr>
<tr>
<td>288</td>
<td>8</td>
<td>213</td>
<td>277</td>
</tr>
<tr>
<td>324</td>
<td>9</td>
<td>214</td>
<td>314</td>
</tr>
<tr>
<td>360</td>
<td>10</td>
<td>194</td>
<td>316</td>
</tr>
<tr>
<td>396</td>
<td>11</td>
<td>189</td>
<td>338</td>
</tr>
<tr>
<td>432</td>
<td>12</td>
<td>180</td>
<td>352</td>
</tr>
<tr>
<td>468</td>
<td>13</td>
<td>179</td>
<td>379</td>
</tr>
<tr>
<td>504</td>
<td>14</td>
<td>168</td>
<td>383</td>
</tr>
<tr>
<td>540</td>
<td>15</td>
<td>178</td>
<td>435</td>
</tr>
<tr>
<td>576</td>
<td>16</td>
<td>178</td>
<td>464</td>
</tr>
<tr>
<td>648</td>
<td>18</td>
<td>189</td>
<td>554</td>
</tr>
<tr>
<td>720</td>
<td>20</td>
<td>179</td>
<td>583</td>
</tr>
</tbody>
</table>

Given the above results and taking into account the file system variability (section 4.3) and the IO profiling data (section 4.4), a set of IFS+NEMO 3-months coupled runs is performed by fixing the number of nodes for the IFS and varying it for NEMO. These coupled runs are performed for two variants of the system, the first is built with the Intel compiler and the second is built with the Cray compiler.

4.2 Coupled runs: IFS+NEMO

4.2.1 Running with Intel-compiled system:

A first set of runs is performed using the Intel-compiled models. The simulations are for the coupled system IFS+NEMO and extend for 3 months starting at 1990-01-01, with a coupling frequency and time step for both models equal to 2700 s. The runs are performed by setting the IFS MPI tasks respectively to (216, 288, 360, 432) and varying the NEMO MPI tasks from 72 to 432 tasks as shown in Table 3. In these runs the Runoff-MAPPER and the XIOS executables are assigned 1 task each.
Exploring EC-Earth 3.2-Beta performance on the new ECMWF Cray-Broadwell

Figure 2: Performance of the coupled IFS+NEMO system on the ccb-Broadwell using the Intel compiler for different IFS MPI tasks configurations: (Blue: 216 tasks, black: 288 tasks, green: 360 tasks, and red: 432 tasks) varying with the number of NEMO assigned nodes. Simulated years per day (SYPD) results are in solid lines and system billing units in dashed lines.

Figure 2 illustrates the runtimes converted to the number of simulated years per day (SYPD) and the corresponding SBU for each configuration. It shows that under these configurations it is not appropriate to set more than 8 nodes (288 tasks) to run NEMO within the coupled system, while we can pick-out the following 3 recommended configuration as highlighted in bold in Table 3. These configurations are (108,216), (144,288) and (288,360) where the first number represents the MPI tasks for NEMO and the second the MPI tasks for IFS.

Table 3: Performance of the coupled IFS+NEMO system built with the Intel compiler for 3-months run using different MPI tasks/Nodes configurations.

| NEMO MPI tasks | NEMO nodes | IFS Tasks = 216 | | IFS Tasks = 288 | | IFS Tasks = 360 | | IFS Tasks = 432 |
|---------------|------------|----------------|----------------|----------------|----------------|----------------|----------------|
|               |            | RT(s) | SBU   | RT(s) | SBU   | RT(s) | SBU   | RT(s) | SBU   |
| 72            | 2          | 2848  | 4637  | 2750  | 5373  | 2785  | 6348  | 2758  | 7184  |
| 108           | 3          | 2299  | 4117  | 2174  | 4601  | 2169  | 5297  | 2247  | 6219  |
| 144           | 4          | 2305  | 4503  | 1994  | 4545  | 2013  | 5244  | 1943  | 5694  |
| 180           | 5          | 2319  | 4908  | 2010  | 4909  | 1997  | 5527  | 1884  | 5823  |
| 216           | 6          | 2236  | 5096  | 2132  | 5554  | 1934  | 5668  | 1743  | 5675  |
| 288           | 8          | 2118  | 5517  | 1930  | 5656  | 1722  | 5607  | 2063  | 7389  |
| 360           | 10         | 2212  | 6482  | 1930  | 6284  | 1731  | 6200  | 1759  | 6873  |
| 432           | 12         | 2374  | 7730  | 1996  | 7149  | 2117  | 8272  | 1777  | 7522  |

4.2.2 Running with Cray-compiled system

Similarly to the Intel-compiled runs, 3 months simulations with the coupled IFS+NEMO system built with the Cray compiler and starting at 1990-01-01 are performed with exactly the same models settings and configurations as with the Intel-compiled case (Table 4).
Table 4: Performance of the coupled IFS+NEMO system built with the Cray compiler for 3-months run using different MPI tasks/Nodes configurations.

<table>
<thead>
<tr>
<th>NEMO MPI tasks</th>
<th>NEMO nodes</th>
<th>IFS Tasks = 216</th>
<th>IFS Tasks = 288</th>
<th>IFS Tasks = 360</th>
<th>IFS Tasks = 432</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Runtime SBU</td>
<td>Runtime SBU</td>
<td>Runtime SBU</td>
<td>Runtime SBU</td>
</tr>
<tr>
<td>72</td>
<td>2</td>
<td>2471 4023</td>
<td>2385 4659</td>
<td>2373 5409</td>
<td>2147 5593</td>
</tr>
<tr>
<td>108</td>
<td>3</td>
<td>2126 3807</td>
<td>1971 4172</td>
<td>1849 4515</td>
<td>1911 5289</td>
</tr>
<tr>
<td>144</td>
<td>4</td>
<td>1974 3857</td>
<td>1703 3882</td>
<td>1629 4243</td>
<td>1592 4665</td>
</tr>
<tr>
<td>180</td>
<td>5</td>
<td>1990 4212</td>
<td>1608 3927</td>
<td>1638 4533</td>
<td>1410 4362</td>
</tr>
<tr>
<td>216</td>
<td>6</td>
<td>1988 4531</td>
<td>1594 4152</td>
<td>1637 4797</td>
<td>1283 4178</td>
</tr>
<tr>
<td>288</td>
<td>8</td>
<td>1983 5165</td>
<td>1647 4827</td>
<td>1618 5268</td>
<td>1361 4875</td>
</tr>
<tr>
<td>360</td>
<td>10</td>
<td>2010 5890</td>
<td>1603 5220</td>
<td>1477 5290</td>
<td>1282 5009</td>
</tr>
<tr>
<td>432</td>
<td>12</td>
<td>2010 6545</td>
<td>1641 5878</td>
<td>1641 6412</td>
<td>1285 5439</td>
</tr>
</tbody>
</table>

Figure 3 shows that for the Cray-compiled system, it is not suitable to set more than 6 nodes (216 MPI tasks) for NEMO. And in this case, three recommendable configurations arise from these experiments as highlighted in bold in Table 4, (108, 216), (180,288) and (216,432) with the latter being the most optimised one in term of runtime to SBU efficiency. Users running ensembles may opt for configuration with smaller number of nodes (108, 216) for each member, therefore being able to run more members in parallel and saving time and resources. It is also noticeable that, in average there is a 17% decrease in runtime and SBU when the system is built with the Cray compiler in comparison with the Intel built system. In such case, the runtime decrease reaches 34% for the most optimised configuration (216,432) where the coupled system could run more than 16 SYPD as depicted in Figure 3.

Figure 3: Same as Figure 2 but using the Cray compiler
4.3 File system/network time variability

One element of the runtime is the network communication and the access to the file system. In these tests, the network communication variability is limited, since the experiments are performed on the new Broadwell testing nodes which are packed together in the same frame and should not generate much communication overhead. On the other hand, running on the general purpose LUSTRE file system ($SCRATCH$) can induce some extra runtime variability which varies according to the file system load.

Table 5: Runtime variability owing to file system load and network communication, for 5 10-days IFS+NEMO coupled runs using 288 tasks for IFS and 108 tasks for NEMO

<table>
<thead>
<tr>
<th>Runtime</th>
<th>334</th>
<th>450</th>
<th>439</th>
<th>419</th>
<th>332</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Time step</td>
<td>0.639</td>
<td>0.665</td>
<td>0.633</td>
<td>0.717</td>
<td>0.645</td>
</tr>
<tr>
<td>Average Time step standard dev. among the runs</td>
<td>0.097</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5 shows different runtimes using exactly the same run configuration for 5 10-days coupled runs. The runtime variability in this case exceeded 25% and the IFS average time step standard deviation among the 5 runs was on the order of 13%. Figure 4 shows this variability of the IFS time steps for the 5 runs. However, it should be mentioned that the machine load has a relatively regular daily variability, therefore the same period of the day was used for assessing the node configuration performance in order to limit the influence of this variability on the overall results.

![Figure 4: Time variability of the IFS steps for a coupled run using 108 tasks for NEMO and 288 tasks for IFS. The colours represent different runs with exactly the same configuration leading to different total runtimes.](image)
4.4 IO profiling

Table 6: Time spent by the OASIS "notroot" debugging files compared with the total runtime for different simulation periods.

<table>
<thead>
<tr>
<th>Run Period</th>
<th>IFS debug file Time (s)</th>
<th>% of Runtime</th>
<th>NEMO debug file Time (s)</th>
<th>% of Runtime</th>
<th>Total Runtime (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 days</td>
<td>67</td>
<td>29.9</td>
<td>27</td>
<td>12.2</td>
<td>225</td>
</tr>
<tr>
<td>10 days</td>
<td>70</td>
<td>20.2</td>
<td>30</td>
<td>8.9</td>
<td>344</td>
</tr>
<tr>
<td>20 days</td>
<td>63</td>
<td>11.7</td>
<td>25</td>
<td>4.6</td>
<td>539</td>
</tr>
<tr>
<td>1 month</td>
<td>90</td>
<td>10.1</td>
<td>28</td>
<td>3.2</td>
<td>888</td>
</tr>
<tr>
<td>3 months</td>
<td>68</td>
<td>3.1</td>
<td>28</td>
<td>1.3</td>
<td>2199</td>
</tr>
<tr>
<td>1 year</td>
<td>53</td>
<td>0.6</td>
<td>26</td>
<td>0.3</td>
<td>8379</td>
</tr>
</tbody>
</table>

The IO profiling via the Darshan software (Carns et al., 2011) showed that Oasis is generating two debugging file related to NEMO and IFS which use substantial amount of running time especially when the running period (or the restart frequency) is less than 3 months. In that case, the time spent on those files can vary between 13% and 42% of the total runtime as shown in Table 6. However for 1 year running period and more, the time used by these files drops to 0.6% of the total runtime. It should be noted that there is no option to switch off the generation of those file within the OASIS namelist file “namcouple”.

4.5 Efficient MPMD (Forking)

With the standard MPMD (multiple program multiple data) configurations (as used above), XIOS and the Runoff–MAPPER are assigned a full node each, while they only use one MPI task out of the available 36 per node. To optimize this configuration, an efficient MPMD, also referred here as “Forking”, is being tested. The Forking consists of assigning a total number of MPI tasks to all the executables and to allow for one node to be shared by more than one executable instead of assigning a specific number of MPI tasks for each program without having the option to share tasks within a non-fully used node. In this case (Table 7), the IFS was assigned 2 MPI tasks less than in the regular configuration described above, and these 2 tasks were assigned to the XIOS and Runoff-MAPPER respectively. In such way, we would have a minor increase in the runtime, but we would gain 2 nodes.
Table 7: Comparison of the MPI Tasks and nodes for the standard MPMD configuration and the Forking configuration used in this experiment.

<table>
<thead>
<tr>
<th>Standard Configuration</th>
<th>Forking Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>XIOS: 1 task (1 node: 36 tasks)</td>
<td>XIOS: 1 task</td>
</tr>
<tr>
<td>Runoff-MAPPER: 1 task (1 node: 36 tasks)</td>
<td>Runoff-MAPPER: 1 task</td>
</tr>
<tr>
<td>IFS: 288 tasks (8 nodes)</td>
<td>IFS: 286 tasks</td>
</tr>
<tr>
<td>NEMO: 180 tasks (5 nodes)</td>
<td>NEMO: 180 tasks (5 nodes)</td>
</tr>
<tr>
<td><strong>Total: 470 tasks; 15 nodes</strong></td>
<td><strong>Total: 468; 13 nodes</strong></td>
</tr>
</tbody>
</table>

It should be emphasized that this new optimization feature is under testing progress and shows promising benefits by freeing 2 nodes and decreasing the SBU by about 7% for the considered case (average over 5 experiments using the above Forking configuration). However more extensive testing is needed especially to check whether possible communication issues could happen between 2 executables hosted by the same node.

4.6 Ivy-Bridge vs Broadwell

In order to check the performance of EC-Earth system on the Broadwell nodes (new) compared to the Ivy-Bridge nodes (current), the following experiments were carried out. Using the same executables built with the Cray compiler, 10 identical 3-months coupled IFS+NEMO runs were executed on each of the Ivy-Bridge and Broadwell nodes. For all the runs NEMO used 144 tasks (6 nodes on Ivy-Bridge, 4 nodes on Broadwell) and IFS used 288 tasks corresponding to 12 nodes on Ivy-Bridge and 8 nodes on Broadwell.

Table 8 shows the comparison between the same runs on the two types of nodes. The results illustrate that EC-earth runs at about the same speed on Broadwell as on Ivy-Bridge with the same number of MPI tasks. Although the CPU is slower in terms of clock speed, the new Broadwell architecture can execute more floating point instructions per cycle than Ivy-Bridge. However, like other NWP models tested on Broadwell, EC-earth doesn’t seem to benefit much from the new floating point instruction set, but would use less nodes and therefore benefit from more extensive resources. In fact, the slower clock speed on Broadwell is compensated by the faster access to memory which improved from 1866 MHz to 2400 MHz.
Table 8: Performance of the IFS+NEMO coupled system for 10 identical 3-months runs on the Broadwell machine compared to similar runs on the Ivy-bridge machine. IFS uses 288 tasks and NEMO 144 tasks.

<table>
<thead>
<tr>
<th></th>
<th>Ivy-Bridge</th>
<th></th>
<th>Broadwell</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RT(s)</td>
<td>SBU</td>
<td>RT(s)</td>
<td>SBU</td>
</tr>
<tr>
<td>1843</td>
<td>1872</td>
<td>4001</td>
<td>4267</td>
<td></td>
</tr>
<tr>
<td>1863</td>
<td>1870</td>
<td>4044</td>
<td>4262</td>
<td></td>
</tr>
<tr>
<td>1731</td>
<td>1684</td>
<td>3758</td>
<td>3838</td>
<td></td>
</tr>
<tr>
<td>1783</td>
<td>1633</td>
<td>3662</td>
<td>3722</td>
<td></td>
</tr>
<tr>
<td>1687</td>
<td>1940</td>
<td>3703</td>
<td>4422</td>
<td></td>
</tr>
<tr>
<td>1693</td>
<td>1714</td>
<td>3675</td>
<td>3907</td>
<td></td>
</tr>
<tr>
<td>1719</td>
<td>1613</td>
<td>3732</td>
<td>3676</td>
<td></td>
</tr>
<tr>
<td>1691</td>
<td>1729</td>
<td>3671</td>
<td>3941</td>
<td></td>
</tr>
<tr>
<td>1678</td>
<td>1607</td>
<td>3643</td>
<td>3663</td>
<td></td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>1739</strong></td>
<td><strong>3776</strong></td>
<td><strong>1733</strong></td>
<td><strong>3951</strong></td>
</tr>
<tr>
<td><strong>STDV</strong></td>
<td><strong>61</strong></td>
<td><strong>132</strong></td>
<td><strong>108</strong></td>
<td><strong>246</strong></td>
</tr>
</tbody>
</table>

4.7 Summary and discussion

The EC-Earth 3.2beta climate system is composed of the IFS atmospheric model, NEMOv3.6 ocean circulation model, LIM3 sea-ice model and a river Runoff mapper. These components are linked via the OASIS3-MCT coupler and use the XIOS as an IO server for NEMO3.6 and LIM3.

This system is ported to the new Broadwell-based machine of the ECMWF using both the Intel and the Cray compilers. The assessment of the performance of this coupled system shows that the runtime variability caused by the file system and/or network communication (jitter) could reach 25% of the average runtime. Therefore a recommendable configuration range is adopted rather than a fixed one. This evaluation also shows that, in comparison with the Intel compiler, building the models with the Cray compiler, can result in an increase in the runtime performance up to 34% allowing the coupled system to reach more than 16 SYPD. Accordingly, depending on the user needs and the type of application, when running the standard configuration of EC-Earth3.2beta built with the Cray compiler on the ECMWF Broadwell-based nodes the following MPI tasks settings are recommended: 288 IFS MPI tasks versus 144 or 180 NEMO MPI tasks, and 432 IFS MPI tasks versus 180 or 216 NEMO MPI tasks.

Furthermore, an efficient MPMD method is proposed to save the two nodes used by the Runoff-MAPPER and XIOS, this method shows a potential benefit of about 7% on the SBU, however it would needs more comprehensive testing.

In addition, the EC-Earth3.2beta performance comparison when using the actual Ivy-Bridge-based nodes and the new Broadwell-based nodes shows that the new HPCF system would benefit from more extensive resources without losing in term of runtime speed.

Although this study resulted in recommendable range of configurations for EC-Earth 3.2beta, some caveats on the generality of these results especially when using other modelling system should be considered and more extensive studies supported by specific performance analysis tools are still needed. This could be achieved for instance within the ESiWACE Horizon 2020 project and the ECMWF scalability project.
Acknowledgement:

This work has been achieved under the EC-Earth framework, the authors would like to thank: Peter Towers, Iain Miller, Kristian Mogensen, Michael Sleigh, Paul Burton, Will Weir, the ECMWF User and HPC support groups, Hamish Struthers, Uwe Fladrich, Philippe Le Sager, and Ralf Doscher for their fruitful suggestions, discussion and comments that allowed the accomplishment of this work.

References


Appendix

1 Compiling and Running:
The branch used in this study and which should be checked-out from the EC-Earth restricted svn repository is:


1.1 Compiling

A. Prepare the Environment to compile with

1) Load required environment modules (step a for Intel or b for Cray compiler)

a) For the intel compiler run the following script

./load_cca_intel_env.ksh

#the first dot(.) is important to load the module on your actual shell

#Check your loaded environment

module list

#The output of the list command should look like:

Currently Loaded Module files:

1) modules/3.2.10.3 19) verbose/true(default)
2) eswrap/1.1.0-1.020200.1130.0 20) cray-libsci/12.2.0
3) switch/1.0-1.0502.60522.1.61.ari 21) udreg/2.3.2-1.0502.10518.2.17.ari
4) intel/15.0.1.133 22) ugni/6.0-1.0502.10863.8.29.ari
5) craype-broadwell 23) pmi/5.0.3-1.0000.9981.128.2.ari
6) craype-network-aries 24) dmapp/7.0.1-1.0502.11080.8.76.ari
7) craype/2.5.3 25) gni-headers/4.0-1.0502.10859.7.8.ari
8) pbs/12.2.401.141761 26) xpmem/0.1-2.0502.60530.1.62.ari
9) cray-mpich/7.3.2 27) dvs/2.5_0.9.0-1.0502.2188.1.116.ari
10) cdt/16.03 28) alps/5.2.4-2.0502.9774.31.11.ari
11) verbose/false 29) rca/1.0.0-2.0502.60530.1.62.ari
12) ecfs/2.2.1-rc2(prodn:new:default) 30) atp/1.7.2
13) jasper/1.900.1(default) 31) PrgEnv-intel/5.2.82
14) grib_api/1.12.3(old) 32) python/2.7.5-01(default)
15) fftw/3.3.4.5 33) udunits/2.2.17(default)
16) emos/394-r64(old) 34) hdf/4.2.10(default)
17) sms/4.4.13(default) 35) cray-hdf5-parallel/1.8.12
18) batch_utils/1.4(default) 36) cray-netcdf-hdf5parallel/4.3.1
b) For the Cray compiler run the following script
   `. /load_cca_cray_env.ksh`

   # Check your loaded environment
   module list

   # The output of the list command should look like:
   Currently Loaded Module files:
   1) modules/3.2.10.3
   2) eswrap/1.1.0-1.020200.1130.0
   3) switch/1.0-1.0502.60522.1.61.ari
   4) craype-broadwell
   5) craype-network-aries
   6) craype/2.4.0
   7) cray-libsci/13.0.4
   8) ugrid/2.3.2-1.0502.10518.2.17.ari
   9) ugni/6.0-1.0502.10863.8.29.ari
   10) pmi/5.0.7-1.0000.10678.155.25.ari
   11) dmapp/7.0.1-1.0502.11080.8.76.ari
   12) gni-headers/4.0-1.0502.10859.7.8.ari
   13) xpmem/0.1-2.0502.64982.5.3.ari
   14) dvs/2.5_0.9.0-1.0502.2188.1.116.ari
   15) alps/5.2.4-2.0502.9774.31.11.ari
   16) rca/1.0.0-2.0502.60530.1.62.ari
   17) atp/1.8.2
   18) PrgEnv-cray/5.2.82
   19) pbs/12.2.401.141761
   20) cray-mpich/7.2.2
   21) cce/8.3.12
   22) verbose/false
   23) ecfs/2.2.1-rc2(prodn:new:default)
   24) jasper/1.900.1(default)
   25) fftw/3.3.4.5
   26) sms/4.4.13(default)
   27) batch_utils/1.4(default)
   28) verbose/true(default)
   29) grib_api/1.12.1
   30) emos/394-r64(old)
   31) cdt/15.06
   32) craype-haswell
   33) cray-netcdf-hdf5parallel/4.3.3.1
   34) cray-hdf5-parallel/1.8.14

2) Set the source directory parameter ECEARTH_SRC_DIR correctly in the configuration xml file
   (config-build-cca-cray.xml)

3) Invoke the Configuration files (option a for Intel or b for Cray)
   a) ec-conf -p cca-intel-mpi config-build-cca-cray.xml
   b) ec-conf -p cca-cray-mpi config-build-cca-cray.xml

B. Start the compilation

1) OASIS
   cd <base>/sources/oasis3-mct/util/make_dir
   make BUILD_ARCH=ecconf -f TopMakefileOasis3

2) XIOS
Exploring EC-Earth 3.2-Beta performance on the new ECMWF Cray-Broadwell

```
cd <base>/sources/xios-1.0
./make_xios --arch ecconf --use_oasis oasis3_mct --netcdf_lib netcdf4_seq
```

3) Runoff-Mapper
```
cd <base>/sources/runoff-mapper/src
make
```

4) NEMO
```
cd <base>/sources/nemo-3.6/CONFIG
./makenemo -n ORCA1L75_LIM3 -m ecconf
```

5) IFS
```
cd <base>/ifs-36r4
make BUILD_ARCH=ecconf -j 6 lib
make BUILD_ARCH=ecconf master
```

1.2 Running

1) Move to the running configuration directory
```
cd <base>/runtime/classic/
```

2) Set the appropriate directories in the config xml file (config-run-cca.xml):
   a) The source directory parameter ECEARTH_SRC_DIR
   b) The running directory parameter RUN_DIR
   c) The initial data directory parameter INI_DATA_DIR: (/fwsm/lb/project/ecearth/ece3.2b)

3) Edit the NUMPROC parameters in the configuration xml file according to your need.
   (Check the scaling results above for indication on the appropriate configuration)

4) Check the cfg file: <base>/runtime/classic/platform/cca.cfg.tmpl

5) Copy and edit the submission script cca.job
```
cp platform/cca.job.tmpl ./cca.job
```

In cca.job mainly edit which script you would run (ece-ifs+nemo.sh for example) and modify EC_total_tasks (for example = 1:216:1:432 ) according to your need. These numbers of tasks should be consistent with the NUMPROC settings in the config-run.xml. This example sets the following 1 XIOS task, 216 NEMO tasks, 1 Runoff-MAPPER task, 432 IFS tasks

6) Invoke the configuration files
   a) For Intel compiled: ec-conf -p cca-intel config-run-cca.xml
b) For Cray compiled: ec-conf -p cca cray-bdw config-run-cca.xml

7) Now you can submit your job:
   qsub cca.job

2 Compilation flags

The compilation flags are part of the configuration build file (config-build-cca.xml) and are mentioned here for the sake of completeness.

2.1 For the Intel built system

General F90 flags for compiling:
   -02 -g -traceback -r8 -fp-model strict -xHost
General C flags for compiling:
   -02 -g -traceback -fp-model strict -xHost
NEMO specific flags:
   -check pointers -check uninit -fpe0 -init=arrays,zero
CFLAGS flags for XIOS:
   -ansi -w
OASIS specific flags:
   -132 -check pointers -check uninit
Preprocessor defs for IFS source:
   linux LINUX LITTLE LITTLE_ENDIAN POINTER_64 BLAS
General flags for linking:
   -02 -g -traceback -fp-model strict -xHost -L/usr/local/apps/jasper/1.900.1/lib

2.2 For the Cray built system

General F90 flags for compiling:
   -sreal64 -em -hnoomp -02
General C flags for compiling:
   -03
CFLAGS flags for XIOS:
   -DMPI_SKIP_MPICXX -h msglevel_4 -h zero -h gnu
NEMO specific flags:
   -em -sinteger32 -sreal64 -02 -e0 -eZ -Ktrap=fp -R b
Preprocessor defs for IFS sources:

    linux LINUX LITTLE LITTLE_ENDIAN POINTER_64 BLAS

General flags for linking:

    -Wl,--as-needed -L/opt/cray/hdf5-parallel/1.8.14/cray/83/lib