Geophysical Research Abstracts
Vol. 16, EGU2014-4468, 2014
EGU General Assembly 2014

Performance Optimization of NEMO Oceanic Model at High Resolution

Italo Epicoco (1,2), Silvia Mocavero (2), Giovanni Aloisio (1,2)

(1) University of Salento, Dep. Engineering for Innovation, Lecce, Italy (italo.epicoco@unisalento.it, +390832297235), (2) Euro Mediterranean Center on Climate Change, Lecce, Italy

The NEMO oceanic model is based on the Navier-Stokes equations along with a nonlinear equation of state, which couples the two active tracers (temperature and salinity) to the fluid velocity. The code is written in Fortan 90 and parallelized using MPI. The resolution of the global ocean models used today for climate change studies limits the prediction accuracy. To overcome this limit, a new high-resolution global model, based on NEMO, simulating at 1/16° and 100 vertical levels has been developed at CMCC. The model is computational and memory intensive, so it requires many resources to be run. An optimization activity is needed. The strategy requires a preliminary analysis to highlight scalability bottlenecks. It has been performed on a SandyBridge architecture at CMCC. An efficiency of 48% on 7K cores (the maximum available) has been achieved. The analysis has been also carried out at routine level, so that the improvement actions could be designed for the entire code or for the single kernel. The analysis highlighted for example a loss of performance due to the routine used to implement the north fold algorithm (i.e. handling the points at the north pole of the 3-poles Grids): indeed an optimization of the routine implementation is needed.

The folding is achieved considering only the last 4 rows on the top of the global domain and by applying a rotation pivoting on the point in the middle. During the folding, the point on the top left is updated with the value of the point on bottom right and so on.

The current version of the parallel algorithm is based on the domain decomposition. Each MPI process takes care of a block of points. Each process can update its points using values belonging to the symmetric process. In the current implementation, each received message is placed in a buffer with a number of elements equal to the total dimension of the global domain. Each process sweeps the entire buffer, but only a part of that computation is really useful for the process' sub-domain, resulting in a bunch of redundant operations that can be avoided. The parallel time of the algorithm is still proportional to the dimension of the global domain. An optimization action could reduce the buffer length only to the number of elements actually needed by the receiving process.

The iso-efficiency analysis before and after the optimization shows that the algorithm becomes cost optimal after the optimization. The number of elements needed is exactly equal to the dimension of the sub-domain. The optimized algorithm avoids redundant operations and makes the parallel time proportional to the problem size and inverse proportional with the number of processes.