Accelerating NEMO: towards exa-scale climate simulation

1 Summary

The NEMO[1] model is far from having a good scalability. The model cannot even take advantage of small clusters with efficiency, and therefore it is even further from being able to take profit of the current supercomputers. In a close future where the potential of the new supercomputers will keep growing, the lack of capacity to take advantage of this technology will represent a huge loss of opportunities. It is in this context, where it becomes relevant to analyse the scalability problems that the model currently has and also the problems that can emerge when more and more processors are used.

In order to do so, we analysed the behaviour of the model with different tests using several profiling tools. These tests enabled us to see where the model was spending most of the time, and which parts of it had a bad scalability. The version of the NEMO model used was NEMO 3.6. The configuration used to perform these tests was ORCA1 with 46 vertical levels, the executions included the sea-ice model LIM3 and the number of processors used to perform these tests varies from 1 to 1152 processors.

With the above configuration, the best speed-up[2] was found at aproximately 288 processors. As the number of processors increase the speed up degraded and the total execution time of the simulations increased. And even if the speed-up keeps improving until 288 processors, the efficiency is nonetheless bad.

Taking these results into account, we identified some of the elements that limit the scalability of the model. These elements sorted by their relevance are: 1) synchronous communication 2) work Unbalance 3) Input/Outputs 4) communication routines and 5)sub-domain overlapping. From the mentioned issues, only the problems related to I/O are purely technical issues. The need of synchronous communication and the sub-domain overlapping are elements required by the algorithms used to solve the equations. The work unbalance comes basically from the sea-ice model , and this is due to the fact that the sea-ice model has different workloads in different regions of the earth. Finally, the communication process is a mix of both technical and method issues. The need of frequent communications is a problem that comes from the algorithm requirements but, how these communications are implemented is a technical issue and is far from being optimal.

The need of synchronous communication is the element that most constrains the scalability of the model, as all the processors of the model spend most of their time waiting to each other when a certain threshold considering the number of processors is exceeded[3]. The main problem of the synchronous communication is that all the processes must wait for each other in each of the small steps of the model. Since the computation time of the routines is not exactly defined, the time taken by the model to compute any routine varies at each execution when these are executed in different processors or even in the same processor. The result of increasing the number of processors is that there are more chances of finding one slower process. Although this happens when there is no use of shared resources, when all the processes are accessing the same shared resource at the same time (I/O, communication network...) some of the processes are even more delayed. As the system works in a synchronous way, all the processors are using the same shared resources massively at the same time in very short periods and this increases the effect explained before. Since all the processes need to wait for each other, even when there is only one process being delayed, the global execution slows down. It is possible to reduce these delays by using more powerful hardware but it would be a waste of resources. To minimize the problems that emerge from the need of synchronous communication, there are some possible solutions such as reducing the number of times the model needs to communicate and improving the communication routine.

[1]NEMO (Nucleus for European Modelling of the Ocean) is a state-of-the-art modeling framework for oceanographic research, operational oceanography seasonal forecast and climate studies.

[2]In the field of computer architecture, speed up is a metric for relative performance improvement when executing a task. Example: $Speedup = \frac{T_{old}}{T_{new}} = \frac{100s}{10s} = 10$

[3]Runing the model with 1152 processors, the time spent waiting is three times bigger than the time spent computing.

The work unbalance comes basically from different workloads on different regions of the earth (More Sea-ice calculations in the poles and less computations in the tropics.). It is not possible to solve the work unbalance of the sea-ice module by adapting the size of the sub-domains for two reasons: First, the differences between workloads are not the same in all the model modules and, in second place, due to the synchronisation requirement, it is not possible to find concurrency[4] between modules, One solution that would solve the work unbalance coming from the sea-ice model is to uncouple this module from the NEMO model, generate a different executable and couple it again using OASIS. This would allow the definition of different sub-domains for the sea-ice module and the other modules, the use of dedicated processors to compute the sea-ice model, and finally it would allow concurrency between the sea-ice module and the other modules.

The problems related with the Input/Output (I/O) were the main bottleneck in the NEMO model. The recent implementation that enables the model to use dedicated processors to write the outputs allows the model to write data without any degradation in the performance. This new approach is still not fully implemented in all the routines that uses I/O and it is still limiting the scalability of the model.

The problem in the communication process is that the current routine is far from being optimal and could magnify the effects caused for the need of synchronous communication. Related to that, we focused on analysing the effects of the current communication pattern on the performance and, for that purpose, some experiments have been done to compare the current communication pattern to three alternative suggested patterns[5]. The results show that the performance of the current communication pattern is worse than the three proposed what outlines the potential benefits of improving the communication routine of NEMO by changing the communication pattern. The actions to improve the communication pattern were focused on avoiding the unnecessary waiting time and also avoiding cascades of events. Since the communication is the main element that needs attention when thinking about exa-scale executions it is worth to pay attention to the optimisation of the communication routine and the communication pattern
.

The sub-domain overlapping is a requirement of the domain decomposition method used. It forces the neighbour sub-domains to share the points in the common border. When the number of sub-domains grows the number of overlapped points increases and then the maximum efficiency that the model can reach is reduced since the shared points are computed twice. The impact of the overlapping is bigger on small domains because the proportion of overlapped points is bigger. In higher resolutions the effect of the overlapping is still not so relevant.

2 Conclusions

An important conclusion that emerges from this work is that in order to face up the possibility of executing the NEMO model with thousands or even millions of processors, just adapting and modifying the old code is not enought, but it is necessary to rethink some important elements.

The work shows that the current technical limitations of the model can be reduced to improve the efficiency, but only with some hundreds or few thousands of processors. To achieve a good efficiency with many thousands of processors and in order to improve the scalability of the NEMO model it is important to study the scalability of the algorithms. The suggestion would be to be aware of the improvements done in algorithms for massive parallel applications since the problems of NEMO are probably shared with other parallel applications. The profiling has shown that the most important technical limitations are the file access and how the communications are taking place. The server approach for the I/O is promising and the recent developments shows a good improvement on the performance. The need of synchronous communication itself is a big issue in the current NEMO application. Moreover, the way the communication routine is implemented also degrades the performance as it requires even more synchrony than the one actually demanded by the algorithms. As it is demonstrated in the master thesis, it is possible to easily improve the current communication routine by changing the communication pattern. The simple implementation of the suggested improvements to the NEMO code is worthwhile, but it would be even better to join efforts to implement an optimal communication routine

[4]The module concurrency is the capacity of the model to execute different modules at the same time in parallel.
[5]For more details about the suggested patterns read the master thesis.