Scientific Computing: Performance and Efficiency in Climate Models

Sandra Schröder*, Michael Kuhn, Nathanael Hübbe,
Julian Kunkel, Petra Nerge, Florens Wasserfall, Thomas Ludwig

University of Hamburg

*8schroed@informatik.uni-hamburg.de

1. Introduction

With the rising use of high performance computing in the natural sciences, performance analysis of numerical models is a very interesting topic in scientific computing. Using the corresponding application you can verify the model with different parameter and analyse the behaviour of the model. This requires a large amount of time consuming numerical calculations (*number crunching*). It is therefore important to find potential optimizations to make the best use of the available memory and processor resources in these applications. From a scientist's point of view, optimization is usually less important when developing scientific software, because the main intention is to find the solutions of the numerical equations. Additionally, once scientific applications are used in a productive environment, scientists are cautious regarding optimizations, because they might influence the accuracy of the results.

However, there are many tools available to easily analyse performance and efficiency of scientific software. With these tools, the program's internal workings can be analysed in detail, possibly highlighting sources of inefficiency. Using this information, it is possible to identify code sequences for potential optimizations.

In this paper we will present an overview of our research group's (Scientific Computing, Department of Informatics) on-going work with the climate model *GETM* (General Estuarine Turbulence Model). The usage of analysis tools will be described and their application will be shown exemplarily with tools like *Sunshot* and *gprof*.

2. Climate Model Optimizations

Climate Models are a good example for number crunching applications. They are not only challenging in terms of parallelization and optimization of the number crunching routines, they also produce huge amounts of data, which needs to be handled efficiently. We are tracing runs of climate models using our *HDTrace* environment to analyse communication and I/O patterns within all processes of the parallel programs.

"GETM [...] describes an idealised 3D numerical model for the Eastern Scheldt, The Netherlands, for simulating the effect of vertical mixing of nutrients on benthic filter feeders growth rates." [1]

2.1. High-Level Optimizations

For inspecting the communication between several processes, Sunshot has proven to be a useful tool. It visualises process communication and I/O in an intuitive way.

Figure 1 shows Sunshot's user interface, visualising a parallel program with four processes. The program's process topology can be seen in the left pane. The four processes are represented by four coloured so-called *timelines*, showing the activities of the respective processes. Each timeline contains *events* for all traced functions. These events can be analysed in detail by clicking on them, showing additional information like the amount of data transferred for communication functions. More information about the usage and features HDTrace can be found in [2].

Since GETM uses the *NetCDF* library for Fortran, HDTrace and Sunshot have been expanded to support the NetCDF functions and underlying I/O system calls.

By analysing GETM with Sunshot, several potential starting points for optimizations were found. For example, GETM uses a large number of `nf90_sync` calls to flush the output files to the disk. Since these synchronisations are usually not necessary and represent a very high percentage of the whole I/O (nearly 96%), we are investigating whether they can simply be omitted or, if this is not possible, how they can be reduced.

Moreover, `MPI_WAITALL` accounts for 85% of the duration of all MPI communications. `MPI_WAITALL` is used for asynchronous communication in MPI and simply waits for all outstanding operations to complete. Since it dominates the communication time in this way, we are investigating how the communication could be handled more efficiently.

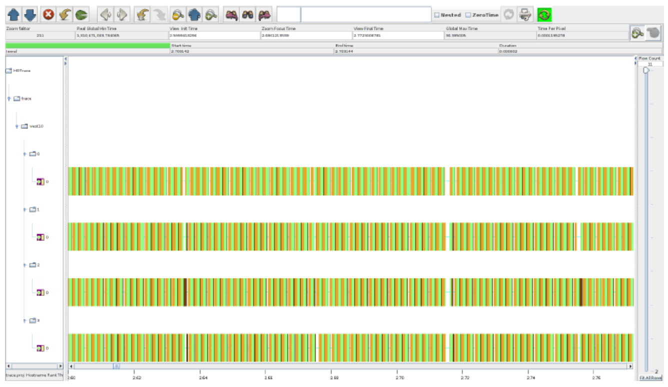

Figure 1. Timeline of Sunshot with several MPI communication and I/O routines

2.2. Low-Level Optimizations

Even though an increase of efficiency is desirable, the accuracy of the results also has to be considered. This is especially true for climate models where the developers want to be able to compare future runs with existing runs to verify their results. For example, executing a floating point division is significantly slower than a multiplication. The division $a/b/c$ can be expressed as a $a/(b \cdot c)$. These expressions are equivalent, but produce different results due to the limited-precision arithmetic. Consequently, this operation is only acceptable if the accuracy is still sufficient for the intended purpose.

Separate code segments were analysed with gprof and many sequences for optimizations could be found. For example, the code segment shown in listing 1, which is performing multiple calculations in a loop, was analysed in detail. Due to the structure of the loops, for each point (i, j) all quotients $A(i, j)/B(i, j)$ are calculated four times.

Listing 1. Calculation loop in GETM

```
do j=jmin,jmax
  do i=imin,imax
    C(i,j)=1/4*(A(i  ,j  )/B(i  ,j  )&
               +A(i+1,j  )/B(i+1,j  )&
               +A(i  ,j-1)/B(i  ,j-1)&
               +A(i+1,j-1)/B(i+1,j-1))
  end do
end do
```

The number of calculations can be easily reduced by performing every calculation only once and buffering the results for the later use. Listing 2 shows a possible way to optimize this particular loop. However, this is a typical time-memory tradeoff and the advantages and disadvantages have to be considered and discussed with the developers, since memory usage is often an important factor in climate models.

Listing 2. Avoiding multiple calculations

```
do j=jmin,jmax
  do i=imin,imax
    D(i,j)=A(i,j)/B(i,j)
  end do
end do

do j=jmin,jmax
  do i=imin,imax
    C(i,j)=1/4*(D(i  ,j  )&
               +D(i+1,j  )&
               +D(i  ,j-1)&
               +D(i+1,j-1))
  end do
end do
```

3. Conclusion and Work in Progress

Potential optimizations can be found easily when using analysis tools. They are useful to detect performance problems of (parallel) scientific applications. In this extended abstract we have presented the climate model GETM and used it as an example for performance analysis. The optimizations which are presented here are only a small amount of all the potential ones that would have been available.

Regarding I/O, we consider modifying GETM to use *ADIOS* (ADaptable IO System) and measure its impact on I/O performance. ADIOS is a new interface providing a simplified API for I/O operations, which allows optimizations especially suited for iterative applications. Previous work with ADIOS has shown great potential to improve I/O throughput.

During a previous meeting with the GETM developers we have presented our optimizations and discussed future work. We will continue this in the future to strengthen the information exchange between informatics and the natural sciences. We hope that this will provide advantages for both sides.

References

[1] GETM – General Estuarine Turbulence Model. http://getm.eu/.

[2] T. Minartz, D. Molka, J. Kunkel, M. Knobloch, M. Kuhn, and T. Ludwig. *Tool environments to measure power consumption and computational performance*. Chapman and Hall/CRC Press Taylor and Francis Group LLC, January 2012.