Parallel Computing 49 (2015) 50–65

Contents lists available at ScienceDirect

Parallel Computing

journal homepage: www.elsevier.com/locate/parco

Reducing the memory footprint in Large Eddy Simulations of reactive flows

S. Weise*, C. Hasse

Chair of Numerical Thermo-Fluid Dynamics, TU Bergakademie Freiberg, ZIK Virtuhcon, Fuchsmühlenweg 9, Freiberg 09599, Germany

ARTICLE INFO

Article history:
Received 8 July 2014
Revised 1 June 2015
Accepted 14 July 2015
Available online 29 July 2015

Keywords:
Memory manager
Database
Parallel simulation

ABSTRACT

CFD simulations of reactive flows couple the domains of flame chemistry and computational fluid dynamics. Solving the chemistry domain in-situ is extremely demanding. It is therefore calculated beforehand and stored into a Look-up table (LUT), which is loaded in each MPI-based CFD application process in a parallel simulation. These chemistry database files can become very large when including many variables and solution values, putting a limit on the number of CFD mesh points as well as the solver instances which can be kept in RAM at the same time. In the paper we approach this problem using dynamic memory managing techniques for single core and parallel applications effectively reducing the memory requirements per core, while keeping the same database resolution. Different implementation options for shared memory on compute nodes provided by the MPI-3 standard and MMAP as a POSIX-compliant system call are analyzed and evaluated using two test cases.

1. Introduction

In CFD simulations of reactive flows an appropriate description of the chemical reactions is necessary in addition to the modeling of the fluid flow. Detailed chemical reaction mechanisms usually involve a significant number of species and reactions among them and both increase with fuel complexity. For each chemical species, a partial differential equation must be solved with a source term, which is the sum of all reaction contributions described by the mechanism. In addition, an energy or enthalpy equation must be solved. The solution of the complete reactive system by itself requires optimized solvers due to its stiff character. The associated smallest chemical time scales are usually smaller than the laminar/turbulent flow time scales. Direct numerical simulation of non-reactive systems at high Reynolds numbers is currently not feasible except for special applications. Thus, a reactive direct numerical simulation fully resolving all chemical and physical time and length scales is extremely demanding (with HPC requirements significantly higher than for the non-reactive case) already for small scale systems when considering complex chemistry. For larger combustion systems, the application of DNS is currently restricted to simple chemical mechanisms.

Instead of trying to perform a DNS, the use of suitable turbulence-chemistry interaction models such as flamelet, FGM for FPI [1–5] in combination with a RANS or LES modeling approach for the fluid dynamics is very popular. These methods use pretabulated solutions of non-premixed or premixed flames. These solutions are accessed during runtime and these retrieved solutions replace the direct solution of the reactive subsystem. The parameter for these tables depend on the system and the number of physical processes to be simulated, e.g. when considering non-adiabatic flows, the enthalpy defect describing heat losses due to

* Corresponding author. Tel.: +49 3731394497.
E-mail address: steffen.weise@iec.tu-freiberg.de (S. Weise).

http://dx.doi.org/10.1016/j.parco.2015.07.004
0167-8191/

radiation or at wall, becomes such a parameter [6]. In addition to the multi-dimensionality of the table, each retrieved solution is a vector, which contains e.g. the species mass fractions and other variables such as density, temperature or heat capacities. These two factors, the multi-dimensional table and the size of the solution vector, respectively, lead to table sizes which can easily go beyond several 100 MB or more depending on the required accuracy. The database may even become larger than the RAM available on the system, after subtracting the memory required for the CFD solver, the grid instances and the operating system components. Also, other physical processes relevant for reacting and non-reacting flows such as thermal radiation might require additional databases of significant size, see e.g. [7]. In a parallel application, the multidimensional table must be available for each process, which in multi-core applications leads to a large overall memory footprint. This paper investigates techniques to reduce this memory footprint and is structured as follows. First, in Section 2 an introduction to current trends in HPC systems and software is given, which outline the targets and limitations for the memory management techniques presented in this paper. It is followed in Section 3 by a description of approaches specific to reactive flow simulations, showing the tool-chain used for look-up-table generation and access of selected values in a CFD application. Section 4 gives an introduction to memory management terminology and techniques, which are then used to describe the methods that are applied later in this paper. These tools are then applied in Section 5 to two combustion test cases. The effects and benefits of the presented memory management approaches are studied in Section 6. Finally the paper concludes with a discussion of the results in Section 7.

2. Development of HPC systems and software

High Performance Computers (HPC) today are large installations of commonly available hardware that follow computer industry trends in processor and system design closely. For years increased circuit density, known as Moore's Law, translated into higher clock frequencies. This increased the speed of existing single core applications without code modification. Due to design limits in power dissipation, the so called power wall [8], an increased number of processor cores is put on a die instead of increasing the clock rate. Programs have to be modified in order to make use of multi-threaded and multiprocessor programming paradigms.

Computational Fluid Dynamics (CFD) has a long history of using multiple processors employing techniques such as domain decomposition to decrease simulation time. The average amount of RAM per processor in specialized Symmetric Multiprocessor Systems (SMP) has traditionally been high since there was a practical limit for the total number of processors in a machine that shared the main memory. Now the number of cores translates into number of execution units, and the available RAM per core decreases drastically [9,10]. The available RAM in HPC systems increases slowly and does not match the increase in number of cores. This can become a serious issue [11] in highly-parallel MPI-based applications of combustion systems.

Modern RAM is still much slower than the CPU and this speed difference still increases [12,13]. This requires detailed attention to any memory transfers inside modern computer systems. Modern highly parallel applications are mostly MPI-based [10] and thus not aware that different instances on the same machine may use other communication and data exchange primitives than message passing. Hybrid OpenMP/MPI applications [14–17] can solve this issue efficiently. MPI is in charge of machine to machine communication hosting one OpenMP application per machine. The OpenMP application has a shared memory layout since OpenMP is a thread-based parallelization scheme. This shared memory is used to allow for communication and data exchange between threads.

There have been similar application-specific approaches to reduce replicated data in MPI parallelized programs using shared memory instead of OpenMP [18–20]. The MPI-3.0 standard [21] has recently added programming options, that makes MPI applications aware of the machine context, enabling communication groups that collect other processes which are using the same shared memory. This kind of programming is called MPI+MPI programming [15,22].

It is clearly not feasible to rewrite a large existing MPI-based CFD application such as OpenFOAM, which is used in this work, to make use of hybrid OpenMP/MPI. This paper explores two general implementation options using the MPI-3 standard and a MMAP-based implementation, respectively. Both techniques are applied and analyzed for the specific problem of database memory management in combustion simulations. This investigation builds upon a previous publication [23], where online compression algorithms were investigated as a first step to reduce the memory footprint.

3. Reactive simulations using pretabulated chemistry

3.1. Flamelet progress variable approach

A widely used method for reactive flow simulations with tabulated chemistry is the flamelet progress variable (FPV) approach [24–26]. The basic idea of the FPV approach is that the chemical state can be uniquely determined by the mixture fraction and the progress variable. It requires the solution for the non-reactive mixture fraction Z and the reactive scalar progress variable Y_C.

The mixture fraction describes the mixing of fuel and oxidizer and is bounded between 0 (oxidizer) and 1 (fuel). It is obtained by solving a suitable transport equation. These are given in the numerical setup of the test cases described in Sections 5.1.2 and 5.2.2.

The chemical state is determined by solving the flamelet equations (with Z as the independent coordinate and varying scalar dissipation rate), see Eqs. 1 and 2 for the temperature and the species mass fractions for a non-premixed system assuming unity