Efficient Implementation of Combustion Chemistry in High Speed Turbulent Flows

AbdoulAhad Validi, Michigan State University, validiab@msu.edu
Shalabh Srivastava, Michigan State University, srivas18@egr.msu.edu
Abolfazl Irannejad, Michigan State University, iranneja@egr.msu.edu
Farhad Jaberi, Michigan State University, jaberi@egr.msu.edu

High fidelity computational simulations of turbulent combustion in high speed flows often require complex and computationally expensive chemical kinetics models. At these flow speeds, the chemical times are comparable to mixing and flow time scales. Also, at these speeds, the flame ignition and extinction are very important and have to be accurately predicted. Turbulent-combustion interactions are also very important and have to be properly modeled. Large eddy simulations (LES) conducted with the Filtered Mass Density Function (FMDF) [1] is a reliable methodology capable of handling turbulent-combustion interactions with complex chemical kinetics mechanisms with no subgrid combustion model. However, the computational time of reactive flow calculations conducted with LES/FMDF and multistep reaction models are still very significant. To make these calculations possible, here, a novel approach to combustion chemistry modeling is developed based on the direct integration of chemical kinetics and parallel load optimization.

The FMDF calculations generally involve direct integration of the chemical reaction equations for millions of Lagrangian Monte Carlo (MC) particles [2,3]. These particles are almost uniformly distributed throughout the domain and the chemical reaction equations are integrated for them in all regions of the flow. However, not all of the particles lie in chemically active regions (e.g., colder and/or very lean/rich fuel regions). This leads to an imbalance in the computational load on various processors in the case of uniform domain decomposition and to inefficient scaling due to lower-load processors remaining idle while higher-load processors are still at work. The imbalance in the computational load leads to a much lower efficiency in the case of large-scale parallel computations, especially when complex chemical kinetics models are used.

The Partially-Stirred Reactor (PaSR) is a reliable configuration for the analysis of chemical kinetics models [4] and is used in this work for testing of load balancing methods for high speed ethylene combustion. To cover subsonic and supersonic flows, the residence time scales $\tau_{res}$ and the mixing time scales $\tau_{mix}$ are varied. There are several PaSRs, each with N particles ($i$th particle composition is $\phi^i(t)$). The particles are arranged in pairs where the composition of each pair (p and q) evolves by mixing and reaction in fractional steps as:

$$\frac{d\phi^p}{dt} = -\frac{(\phi^p - \phi^q)}{\tau_{mix}}$$

$$\frac{d\phi^q}{dt} = -\frac{(\phi^q - \phi^p)}{\tau_{mix}}$$

$$\frac{d\phi}{dt} = S(\phi)$$

where $S$ is the compositional rate of change obtained by the ODE solver. The detailed UCSD mechanism [5], consisting of 50 species and 244 reversible reactions, is used to describe the Ethylene-air combustion chemistry. For one class of test cases considered here, the composition distributions are made disjoint among the processors by diluting the three inflowing streams, air, fuel and pilot, by a specified amount of Argon equal to $(\alpha - 1)/(\alpha - 1 + 14.42)$, where $\alpha$ is the number of the reactors ($\alpha = 1, ..., N_r = 40$). This study is carried out for M (e.g. 40) number of PaSR’s distributed on M processors. Since different compositions of the various PaSR’s lead to reaction load imbalance, it is required to balance the computational load. The basic steps of the load balancing procedure [7] applied in this study are as follows:

1. The load information of each processor is utilized to calculate the load transfer matrix containing information about the particle communication to be carried out between different processors to achieve a balanced load. The primary load information of the MC particles is obtained from prior knowledge of combustion in zero-dimensional simulations for different compositions. The composition is taken to be
the particle species mass fractions and temperature. In the next step, if the particle lies in chemically active region, the condition number (maximum Eigen value over minimum Eigen value) of the Jacobian matrix of the related system of Ordinary Differential Equations (ODEs) [6] of the particle composition is calculated. This value indicates the stiffness of the ODE system and consequently the reaction cost (load) of each particle.

2. This matrix is then communicated to each processor and MC particle data transfer takes place using the load transfer matrix to achieve nearly uniform load on all processors.

3. Once load balance is achieved, the combustion calculation is carried out and the updated MC particle information is sent back to the originating processor.

4. The procedure is repeated again at the next time step.

This procedure significantly reduces the total simulation time by reducing the idle time of the lower load processors. Various methods can be used to calculate the load transfer matrix, e.g., Cut-Paste Repartitioning, Flow-Diffusion, Multilevel Flow/Diffusion, and Modified Cut-Paste Repartitioning methods. An important consideration in all these algorithms is that since the MC particles have to be communicated back to their originating processors, the algorithms should not only achieve load balancing but also seek to minimize the data redistribution and communication costs. The performance of the load balancing approaches developed in [7] will be studied in PaSR calculations.

The performance of the strong scaling obtained by applying the cut-paste repartitioning and the flow-diffusion load balancing methods is studied for different number of processors (8-128) and MC particles (8 million – 64 million) with a randomly generated load imbalance. Strong scaling is defined as the scaling obtained when the problem size is kept fixed and the number of processors is increased. Ideal speedup in this case is defined as the speedup obtained when computation time decreases linearly with increase in the number of processors.

![Strong Scaling](image.png)

**Figure 1.** Strong scaling for the Cut-paste repartitioning and Flow-Diffusion methods. NMC denotes the total number of MC particles, distributed equally among processors [7].

**Acknowledgments**
The support for this work was provided by the U.S. Office of Secretary of Defense under contract FA8650-12-C-2247. The authors are also grateful to the Air Force Research Laboratory staff for program management support. Computational resources were provided by the High Performance Computing Center at Michigan State University.

**References**


