Contents lists available at SciVerse ScienceDirect

Combustion and Flame

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

Computationally-efficient and scalable parallel implementation of chemistry in simulations of turbulent combustion

Varun Hiremath [a,*], Steven R. Lantz [b], Haifeng Wang [a], Stephen B. Pope [a]

[a] Sibley School of Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY 14853, USA
[b] Center for Advanced Computing, Cornell University, Ithaca, NY 14853, USA

ARTICLE INFO

Article history:
Received 18 November 2011
Received in revised form 24 February 2012
Accepted 25 April 2012
Available online 6 June 2012

Keywords:
Turbulent reactive flows
Large-eddy simulation
Probability density function method
In situ adaptive tabulation
Parallel function evaluation
Parallel scaling

ABSTRACT

Large scale combined *Large-Eddy Simulation* (LES)/*Probability Density Function* (PDF) parallel computations of reactive flows with detailed chemistry involving large numbers of species and reactions are computationally expensive. Among the various techniques used to reduce the computational cost of representing chemistry, the three approaches in widest use are: (1) *mechanism reduction*, (2) *dimension reduction*, and (3) *tabulation*. In addition to these approaches, in large scale parallel LES/PDF computations, we need strategies to distribute the chemistry workload among the participating cores to reduce the overall wall clock time of the computations. Here we present computationally-efficient strategies for implementing chemistry in parallel LES/PDF computations using *in situ* adaptive tabulation (ISAT) and *x2f_mpi* – a Fortran library for parallel vector-valued function evaluation (used with ISAT in this context). To test the strategies, we perform LES/PDF computations of the Sandia Flame D with chemistry represented using (a) a 16-species augmented reduced mechanism; and (b) a 38-species C_1–C_4 skeletal mechanism. We present three parallel strategies for redistributing the chemistry workload, namely (a) PLP, purely local processing; (b) URAN, the uniform random distribution of chemistry computations among all cores following an early stage of PLP; and (c) P-URAN, a Partitioned URAN strategy that redistributes the workload only among partitions or subsets of the cores. We show that among these three strategies, the P-URAN strategy (i) yields the lowest wall clock time, which is within a factor of 1.5 and 1.7 of estimates for the lowest theoretically achievable wall clock time for the 16-species and 38-species mechanisms, respectively; and (ii) for reaction, achieves a relative weak scaling efficiency of about 85% when scaling from 2304 to 9216 cores and a relative strong scaling efficiency of over 60% when scaling from 1152 to 6144 cores.

1. Introduction

Computations of turbulent combustion flows using detailed chemistry involving a large number of species and reactions are computationally expensive. Modern chemical mechanisms of real fuels involve hundreds or thousands of species and thousands of reactions [1,2]. Incorporating such detailed chemistry in the combustion flow calculations is computationally prohibitive. Among the various efforts put into reducing the computational cost of representing chemistry, the three approaches in widest use are: (1) *mechanism reduction* to reduce the number of species and reactions involved [3–5]; (2) *dimension reduction* to represent chemistry using a reduced number of variables [6–9]; and (3) *tabulation* to reduce significantly the cost of expensive evaluations of the reaction mappings involving ODE integrations [10–13]. In recent times, combined methodologies have also been developed, wherein reduced reaction mechanisms or dimension reduction methods are used in conjunction with tabulation [14–17].

Due to the high computational cost of turbulent combustion problems, most of the modern day simulations are performed in parallel on multiple cores using distributed computing. Thus, in addition to the aforementioned techniques, when performing large scale parallel LES/PDF computations, strategies are needed to efficiently distribute the chemistry workload among the participating cores to reduce the overall wall clock time of the computations [18–21]. In this paper we present parallel strategies for the implementation of chemistry using *in situ* adaptive tabulation (ISAT) [11] and *x2f_mpi* – a Fortran 90/95 library for parallel vector-valued function evaluation (used with ISAT in this context) [20]. The parallel strategies are tested by performing LES/PDF simulations of the Sandia Flame D [22].

The work here is presented mainly in the context of large scale parallel LES/PDF computations of turbulent reactive flows, in which the ISAT algorithm has proved to be particularly efficient at reducing the computational cost by more than two to three

* Corresponding author.
E-mail address: vh63@cornell.edu (V. Hiremath).

Nomenclature

Roman symbols

h	enthalpy
p	pressure
n_e	number of elements
n_s	number of species
$\mathbf{z}$	species specific moles
$\mathbf{S}$	chemical production rates
$\mathbf{R}(\mathbf{z},t)$	reaction mapping after time t starting from $\mathbf{z}$
Δt	reaction time step
N_{pc}	number of particles per cell
S	size of ISAT table
N_c	number of cores
N_t	number of time steps
$N_x \times N_y \times N_z$	LES grid size
$D_x \times D_y$	domain decomposition

Greek symbols

τ	time spent in the PLP stage in P-URAN
κ	partition size in P-URAN

Abbreviations

LES	Large Eddy Simulation
PDF	Probability Density Function
ISAT	*in situ* adaptive tabulation
PLP	purely local processing
URAN	uniform random
P-URAN	Partitioned URAN

orders of magnitude [11,15]. However, the ISAT algorithm has been successfully applied in many other fields like chemical engineering [23], control [24], and solid mechanics [25]; and has also been implemented in the commercial CFD package ANSYS Fluent [16,17]. Hence, the parallel strategies presented in this work for implementing chemistry in LES/PDF computations may have wider applications in other fields also.

The outline of the remainder of the paper is as follows: in Section 2 we describe our hybrid LES/PDF solver; in Section 3 we describe the parallel strategies implemented using ISAT and *x2f_mpi* for implementing chemistry; in Section 4 we describe the computational details for simulating Sandia Flame D; in Section 5 we present performance results for different parallel strategies; and finally in Section 6 we state our conclusions.

2. Hybrid LES/PDF solver

2.1. LES solver

In this study we use an LES solver based on a Stanford LES code [26,27]. The LES solver solves the Eulerian transport equations for mass, momentum and scalars using finite difference methods on structured non-uniform grids. It supports both Cartesian and cylindrical coordinate systems; is second order accurate in space and time; and is parallelized (using MPI) by domain decomposition in two dimensions.

2.2. PDF solver

We use the PDF solver, *HPDF*, developed at the Turbulence and Combustion Research Group at Cornell [28]. The HPDF solver has second-order accuracy in space and time; supports Cartesian and cylindrical coordinate systems; is parallelized (using MPI) by domain decomposition in two dimensions; and has a general interface to facilitate coupling with existing LES (or RANS) solvers. In this work we use the "one-way" coupling in our LES/PDF solver as described in [28], i.e., LES flow field data is used in the PDF solution, however there is no feedback of temperature and density from the PDF to LES solution. The LES solver uses its own assumed-PDF Flamelet model to obtain density and temperature. However, the thermo-chemical statistics reported in this work are collected from the PDF particle data.

In HPDF, the thermo-chemical composition of the fluid within the solution domain is represented by a large number of particles (see Fig. 1). The HPDF solver has three main components which account for:

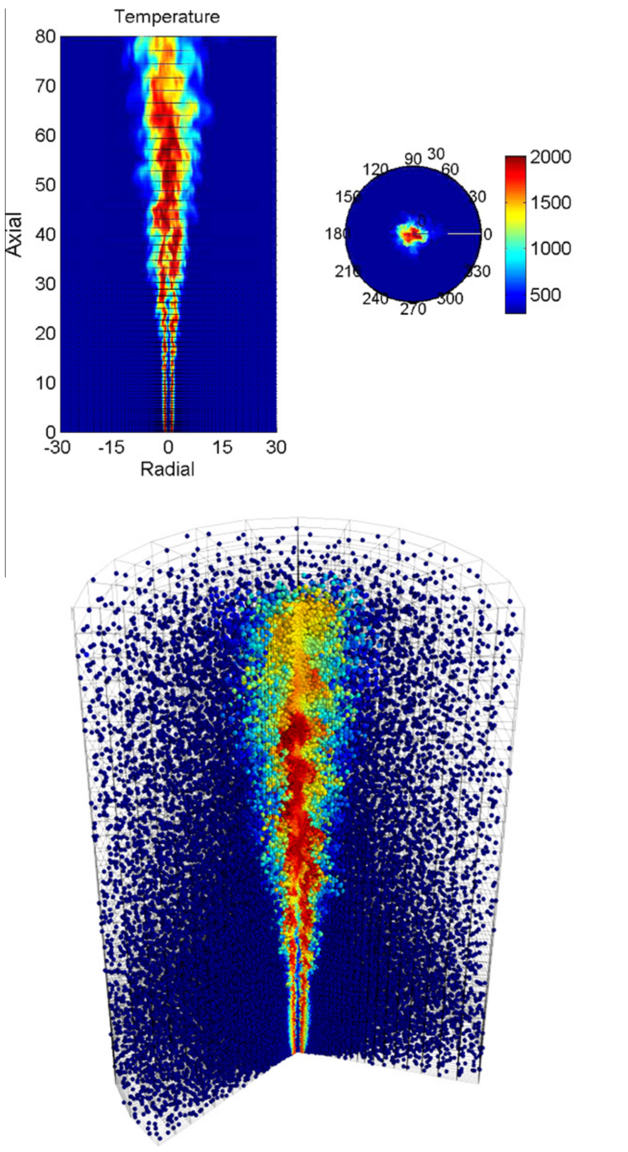


Fig. 1. LES/PDF simulation of the Sandia Flame D. Top: Instantaneous temperature distribution on a 2D cut-plane through the axis of the computational domain. Dots in the plot indicate every third grid node in the axial and radial directions. Bottom: A 3D slice-view of the PDF particle temperature distribution in the computational domain. Every fourth LES grid line is shown.

1. *transport*: the motion of particles in the physical domain, including a random-walk component to represent the effects of subgrid-scale turbulent advection and molecular diffusion;
2. *mixing*: the change in composition of a particle due to mixing with neighboring particles (which models the effects of molecular mixing); and
3. *reaction*: the change in composition of a particle due to chemical reaction.

These components are implemented in *fractional steps* using splitting schemes [29].

In this study we use the first-order $\mathbb{TMR}$ splitting scheme (which is found to perform as well as the second-order splitting scheme for jet flames [28]), which denotes taking fractional steps of *transport*, $\mathbb{T}$; *mixing*, $\mathbb{M}$; and *reaction*, $\mathbb{R}$ in this order on each time-step. The Kloeden and Platen (KP) [30] stochastic differential equation (SDE) scheme is used to integrate the *transport* equations; and the *mixing* is represented using the modified Curl [31] mixing model. In the remainder of this section, we focus on the implementation details of the *reaction* fractional step.

2.3. Chemistry representation

We consider a reacting gas-phase mixture consisting of n_s chemical species, composed of n_e elements. The thermo-chemical state of the mixture (at a given position and time) is completely characterized by the pressure p, the mixture enthalpy h, and the n_s-vector $\mathbf{z}$ of specific moles of the species.

In our LES/PDF computations, we neglect acoustic interactions and compressibility (under the "low Mach number" approximation), and assume that the thermodynamic variables are decoupled from the small variations in pressure about some fixed specified background pressure field, $p = p_0$. Thus only p_0 is coupled to the thermodynamic variables, and (given p_0) the thermo-chemical state is fully characterized by $\{\mathbf{z}, h\}$. In the HPDF solver, the particles carry the composition $\{\mathbf{z}, h\}$.

In the reaction fractional step, a particle's chemical composition $\mathbf{z}$ evolves (at constant enthalpy h) in time according to the following set of ordinary differential equations (ODEs)

$$\frac{d\mathbf{z}(t)}{dt} = \mathbf{S}(\mathbf{z}(t)), \tag{1}$$

where $\mathbf{S}$ is the n_s-vector of chemical production rates determined by the chemical mechanism used to represent the chemistry.

The *reaction mapping*, $\mathbf{R}(\mathbf{z}, t)$ is defined to be the solution to Eq. (1) after time t starting from the initial composition $\mathbf{z}$. The reaction mapping obtained by directly integrating the set of ODEs given by Eq. (1) is referred to as a *direct evaluation* (DE). We use DDASAC [32] for performing ODE integration.

Owing to the large cost of direct evaluation of reaction mappings involving large numbers of species, *in situ* adaptive tabulation (ISAT) is used in the HPDF solver to reduce the cost of chemistry calculations. In addition, we use the *x2f_mpi* library to distribute the chemistry workload efficiently among the participating cores in large scale parallel LES/PDF simulations. The details of the implementation are discussed in Section 3.

2.4. Domain decomposition

The LES computations are performed on structured non-uniform grids in Cartesian or cylindrical coordinate system. We denote the grid used for LES computations by $N_x \times N_y \times N_z$ (in the three principal directions). In performing parallel LES/PDF computations (using the hybrid LES/HPDF solver) on N_c cores, the computational domain is decomposed into N_c sub-domains and each core performs the computations of one sub-domain. The domain decomposition is done in the first two principal directions, and is denoted by $D_x \times D_y$, where $D_x D_y = N_c$. The domain decomposition is done such that N_x and N_y are exact multiples of D_x and D_y, respectively. In addition, the domain decomposition in the LES solver is restricted by the grid size such that $D_x \leqslant N_x/2$ and $D_y \leqslant N_y/2$, i.e., each slice in a given dimension must contain at least two grid points. The HPDF solver has the capability to use its own domain decomposition independent of the LES solver, but in the current study, we use the same domain decomposition in both the LES and HPDF solvers.

2.5. Compute cluster architecture and parallelization

For the sake of consistency, all the results presented in this work are obtained on the TACC Ranger cluster. Each node on Ranger contains four AMD Opteron Quad-Core 64-bit processors, i.e., 16 cores in all, with 32 GB of memory (2 GB per core).

In our LES/HPDF solver, each sub-domain is assigned to one core (independent of the cluster architecture), and the inter-core communication is implemented using MPI. From the implementation point of view, the intra-node communication between two cores on a single node is not treated differently from the inter-node communication between two cores on two different nodes. This ensures the code is highly robust and portable, properties that would be difficult to achieve with a hybrid parallel implementation blending multithreading with message passing [33]. Obviously, though, the actual MPI core-to-core communication time is affected by details of the cluster architecture and connectivity; the influence of these factors will be discussed in Section 5.4.

3. Parallel strategies for implementing chemistry

In performing parallel LES/PDF computations with chemistry tabulation using ISAT, each core has its own ISAT table for tabulating the chemistry. On the reaction fractional step, the reaction mappings for all the particles in the computational domain need to be evaluated. A particle whose reaction mapping has been evaluated is called a *resolved* particle; and the act of resolving a particle by successfully retrieving a linear approximation to the reaction mapping from an ISAT table is called a *retrieve*. In parallel computations, given a particle on a core, the following ISAT operations can be invoked using *x2f_mpi* in an attempt to resolve the particle:

1. attempt to retrieve from the local ISAT table (also referred to as a "quick try");
2. if "quick try" fails, make one (or more) attempt (s) to retrieve from remote ISAT table (s);
3. if all the retrieve attempts fail, do a direct evaluation (followed by addition to the ISAT table) on the local core or on a remote core.

The goal is of course to resolve all the particles in the minimum possible wall clock time by redistributing the chemistry workload among all the cores. But this is not a trivial task because:

1. the time required to resolve a given particle is unknown ahead of time;
2. the time to resolve a particle may vary by 4 orders of magnitude, as the retrieve time from an ISAT table is typically $\mathcal{O}(10)$ μs while a direct evaluation may take $\mathcal{O}(10^5)$ μs (using DDASAC [32] for mechanisms involving 100 or more species);
3. furthermore, the probability of retrieving from an ISAT table depends on the history and duration of the run.

The *x2f_mpi* library [20] is used as an interface between the HPDF solver (for the *reaction* fractional step) and the ISAT tables to redistribute the chemistry workload (of resolving all the particles) efficiently among the participating cores to reduce the overall wall clock time of the computations.

A thorough discussion of the use of *x2f_mpi* library in conjunction with ISAT to redistribute chemistry workload is provided in [20]. Two of the strategies, namely (i) Purely Local Processing (PLP) and (ii) Uniform Random Distribution (URAN), presented in [20], are described here in the context of LES/PDF computations. Additionally, here we present a new strategy, Partitioned URAN (P-URAN), which is shown to perform better than the PLP and URAN strategies, and which scales well to large number of cores.

1. Purely Local Processing (PLP): In this strategy, all the particles on a core are resolved (i.e., the reaction mapping is evaluated) using the local ISAT table without any message passing or load redistribution. This in some sense is the same as invoking ISAT directly from HPDF on each core without using the *x2f_mpi* interface. The main advantages of this strategy are:
   (a) ease of implementation;
   (b) no communication cost;
   (c) higher probability of retrieving particles from the local table;

   and the main disadvantage is:
   (a) load imbalance, especially between cores handling computation of sub-domains in the reactive zone versus cores handling computation of sub-domains in the coflow/air, leading to relatively high wall clock time.
2. Uniform Random Distribution (URAN): This strategy aims at achieving statistically ideal load balancing by evenly distributing the chemistry workload among all the participating cores. The strategy involves one initial HPDF step of PLP to initialize the local ISAT tables. In the subsequent steps, on each core, a "quick try" is first made to attempt to resolve particles by retrieving from the local ISAT table; following this, there is a uniform random distribution of all the unresolved particles to all the cores. This strategy thus ensures that every core receives (approximately) the same number of particles to resolve, with a similar distribution of particles from the reactive and non-reactive zones of the computational domain. The main advantage of this strategy is:

   (a) close to ideal load balancing, after the initial "quick try" lookup;

   and the disadvantages are:
   (a) relatively costly all-to-all communication;
   (b) lower probability of retrieving particles (due to the random distribution of unresolved particles over all the cores);
   (c) poor scaling (to large number of cores) due to all-to-all communication.
3. Partitioned Uniform Random Distribution (P-URAN): This is a new strategy which is a combination of the previous two, PLP and URAN, strategies. This strategy works in two stages: in stage 1 (for a specified duration of time) the PLP strategy is used to resolve particles on all cores at each time step; then in stage 2 (for the remainder of the time steps), the participating cores are partitioned into smaller groups, and within each partition the URAN strategy is used to uniformly distribute the chemistry workload among the cores in that partition. The advantages of this strategy are:
   (a) relatively higher probability of retrieving from the local tables due to the initial PLP stage;
   (b) reduced communication cost compared to URAN (communication restricted to within smaller partitions);
   (c) good load balancing within partitions;
   (d) good scaling to large number of cores;

   and some of the disadvantages are:

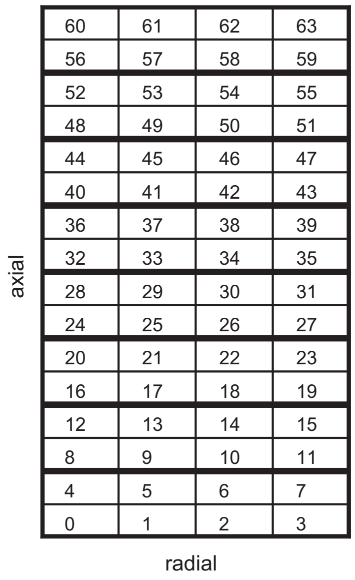


Fig. 2. A schematic showing LES/HPDF domain decomposition of 16 × 4 for $N_c = 64$ cores (indicated by cores ranked from 0 to 63), and formation of partitions of size $\kappa = 8$ (indicated by thick lines) for applying the P-URAN[$\tau, \kappa = 8$] strategy.

   (a) load imbalance among different partitions;
   (b) the need to determine additional parameters: (i) duration of the PLP stage; and (ii) size of the partitions.

To specify the parameters used in the P-URAN strategy, in the remainder of the text we use the notation: P-URAN[τ, κ], where τ denotes the time (in hours) spent in the PLP stage (in addition to the first initialization time step); and κ denotes the size of the partitions, i.e., partitions of κ cores are formed from the overall N_c cores used in the computations (which means that the number of partitions used is N_c/κ). Since the dominant load imbalance is caused in the radial direction in the simulation of jet flames, in this study we choose κ to be a multiple of the domain decomposition in the radial direction, D_y, and form partitions in the axial direction by grouping together the domains in the radial direction as shown in the schematic Fig. 2 for applying P-URAN[$\tau, \kappa = 8$] strategy with $N_c = 64$, $D_x = 16$, $D_y = 4$. (As discussed later in Section 5.4, it is also desirable to have D_y and κ to be exact multiples of the number of cores per node on the compute cluster (for example 16 on Ranger)).

In addition to the PLP and URAN strategies, in [20] two more strategies are presented: (i) the preferential distribution (PREF) strategy; and (ii) an "on the fly" *adaptive* distribution strategy, which blends PLP, URAN and PREF. In [20], the PREF and *adaptive* strategies are tested using the Partially-Stirred Reactor (PaSR) using up to 64 cores, and are found to perform better than the PLP and URAN strategies. However, in the current study, we do not find the PREF and *adaptive* strategies performing any better than the PLP or URAN strategy when applied to the LES/PDF simulation of the Sandia Flame D using more than 1000 cores, presumably due to one or more of the following reasons:

1. LES/PDF simulation of the Sandia Flame D exhibits significantly more load imbalance (among the jet, pilot and coflow regimes) compared to the PaSR setup used in [20], and so efficient redistribution of chemistry workload is harder for Sandia Flame D;
2. the PREF and *adaptive* strategies involve significantly more MPI communication than other strategies, and the communication may not scale well to a large number of cores (more than 1000) used in the current study.

We are still investigating these reasons, but nonetheless, the new P-URAN strategy presented here is shown to perform within a factor of 1.5–1.7 of estimates for the best that can be achieved (in terms of simulation wall clock time); and scales well up to 9216 cores.

4. LES/PDF simulation of Sandia Flame D

To test the chemistry implementation we perform LES/PDF simulations of the Sandia Flame D.

4.1. Sandia Flame D

The Sandia Flame D is a piloted CH_4/Air jet flame operating at a jet Reynolds number, Re = 22,400. All the details about this flame and the burner geometry can be found at [22]. Here we mention only some of the important aspects of this flame.

The jet fluid consists of 25% CH_4 and 75% air by volume. The jet flows in at 49.6 m/s velocity at 294 K temperature and 0.993 atm pressure. The jet diameter, D = 7.2 mm. The pilot is a lean (equivalence ratio, Φ = 0.77) mixture of C_2H_2, H_2, air, CO_2, and N_2 with the same nominal enthalpy and equilibrium composition as that of CH_4/Air at this equivalence ratio. The pilot velocity is 11.4 m/s. The coflow is air flowing in at 0.9 m/s at 291 K and 0.993 atm.

4.2. Computational details

We perform LES/PDF simulation of the Sandia Flame D using the coupled LES/HPDF solver. The simulation is performed in a cylindrical coordinate system. A computational domain (see Fig. 1) of $80D \times 30D \times 2\pi$ is used in the axial, radial and azimuthal directions, respectively. A non-uniform structured grid of size $192 \times 192 \times 96$ (in the axial, radial and azimuthal directions, respectively) is used for the LES solver. In the HPDF solver (for the base case), the number of particles per LES cell (N_{pc}) used is $N_{pc} = 40$. With a total of $192 \times 192 \times 96 \approx 3.5 \times 10^6$ LES cells, an overall 140×10^6 particles are used in the computational domain.

To represent the chemistry we consider two different mechanisms:

1. 16-species augmented reduced mechanism (ARM1) [34]; and
2. 38-species C_1–C_4 skeletal mechanism [35]. (This mechanism is developed especially for ethylene combustion, but is also applicable to methane flames.)

In this study, we are not interested in comparing the accuracy of representing chemistry using these two mechanisms, but are only interested in studying the parallel performance of chemistry implementation using these two mechanisms involving different numbers of species.

A fixed ISAT error tolerance, $\epsilon_{tol} = 10^{-4}$ (which yields less than 3% tabulation error for both the mechanisms), is used in this study. In addition, we specify a maximum allowed ISAT table size, S, per core. In the simulations, when an ISAT table on a core becomes completely filled, then subsequent unresolved queries on that core are resolved using direct evaluation. We typically specify a maximum ISAT table size of $S \leqslant 1$ GB because for tables of size over 1 GB, the *search* and *add* times in ISAT become large and are sometimes comparable to direct evaluation time. For the 16-species mechanism we specify a maximum ISAT table size of S = 600 MB per core; and for the 38-species mechanism we specify S = 1000 MB per core. In simulations with the 16-species mechanism, none of the tables become filled over the duration of the runs covered in this report, however with the 38-species mechanism some tables (near the flame front with the PLP or P-URAN strategy) reach the maximum specified size limit. (For mechanisms involving over 40 species, we typically use a dimension reduction method like the rate-controlled constrained-equilibrium (RCCE) [6,15] to reduce the number of tabulated variables to 20–30, thereby reducing the ISAT table size [36].)

5. Results

The LES/PDF simulation tests are performed in three phases (more computational details are given in the later sections):

- Phase 1: Base case. In this phase we perform an LES/PDF simulation of the Sandia Flame D to obtain a statistically-stationary flame.
- Phase 2: Comparison of parallel strategies. In this phase, starting from the statistically-stationary base case, we compare the performance of various parallel strategies implemented using *x2f_mpi*.
- Phase 3: Scaling studies. In this phase we perform weak and strong scaling studies with different *x2f_mpi* strategies using up to 9216 cores.

All the simulations are performed on the Texas Advanced Computing Center (TACC) Ranger cluster.

5.1. Base case

We perform separate LES/PDF simulations of the Sandia Flame D with chemistry represented using the 16-species and 38-species mechanisms on 1024 cores (using 64×16 domain decomposition) until a statistically-stationary state is reached.

After reaching the statistically-stationary state, for the simulation using the 38-species mechanism, we collect statistics for thermo-chemical quantities from the PDF particle data time-averaged over 10,000 time steps (which corresponds to about three flow through times based on the jet inlet velocity). Figures 3 and 4 show comparisons of radial profiles of azimuthally-averaged and time-averaged density-weighted mean and standard-deviation statistics with the experimentally the measured statistics [22] at axial locations x/D = 15, 30, 45. A good qualitative agreement between the simulated and experimentally measured statistics is observed, which is adequate for the current study as the main focus is on the efficient parallel implementation of chemistry. There have been many previous studies of the Sandia Flame D using PDF and LES based methods [37–41]. A slightly better prediction for the peak value of the mean temperature at x/D = 15 is obtained in [37,38], however overall a similar level of agreement for species mass fractions is observed in these studies.

5.2. Comparison of parallel strategies

Starting from the respective base cases for the 16-species and 38-species mechanisms, we employ the PLP, URAN and P-URAN parallel strategies, and compare the overall wall clock time for running a fixed number of simulation time steps on 1024 cores. To test P-URAN, we consider the P-URAN[0.2 h,32] strategy for the 16-species mechanism, and the P-URAN[0.1 h,32] strategy for the 38-species mechanism. P-URAN sensitivity results to changes in the time spent in the PLP stage (τ) and the partition size (κ) are presented in the next section.

For the 16-species and 38-species mechanisms we perform N_t = 2000 and N_t = 1000 simulation time steps, respectively. (We perform fewer simulation time steps with the 38-species mechanism due to relatively expensive chemistry and limited availability of compute hours on the TACC Ranger cluster.) The timing results

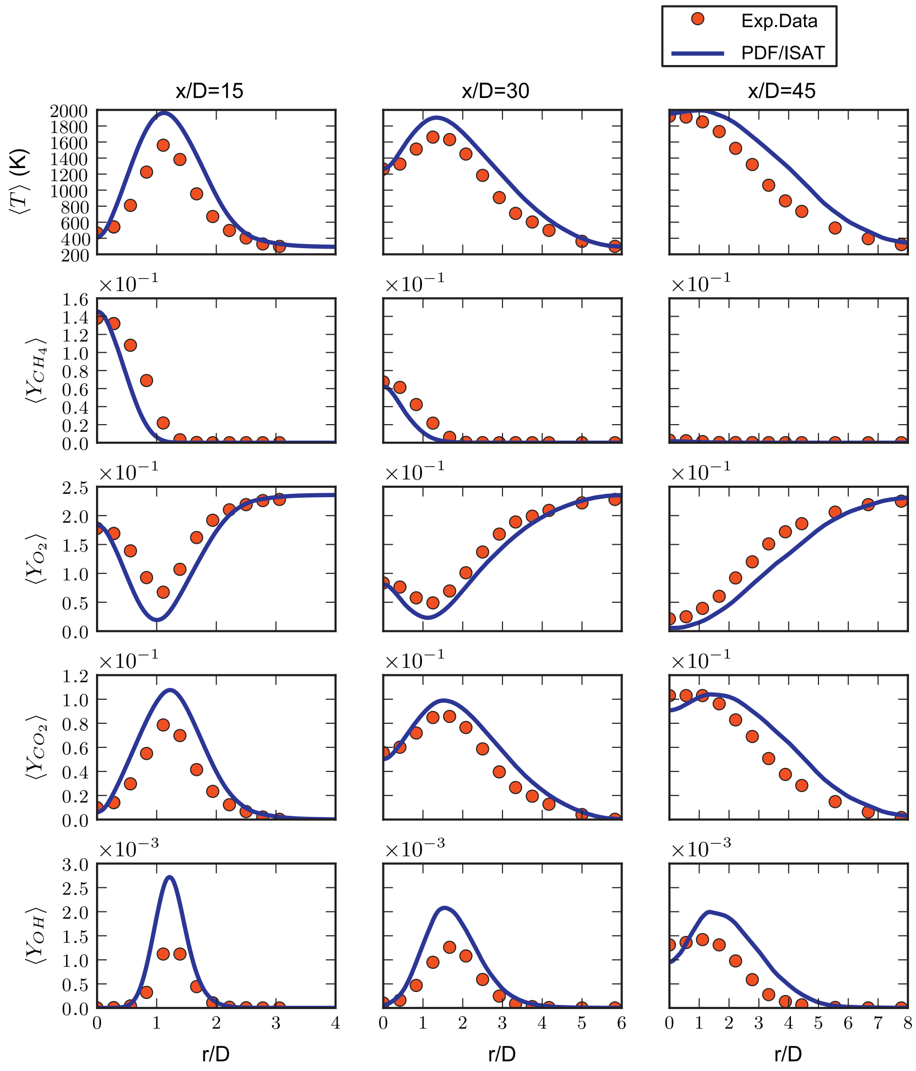


Fig. 3. Radial profiles of time-averaged density-weighted mean temperature *T*, and mass fraction of species CH_4, O_2, CO_2, *OH* at axial locations x/D = 15, 30, 45 obtained from experimental data and an LES/PDF simulation using the 38-species mechanism.

using different strategies are shown in Figs. 5 and 6 using bar charts for the 16-species and 38-species mechanisms, respectively. The bars show the breakdown of time spent in LES, HPDF (outside reaction) and Reaction (including *x2f_mpi* communication time, if any). Also shown is the Waiting time, which is indicative of the load imbalance caused by reaction, with a lower bound of zero indicating perfect reaction load balancing, and an upper bound equal to the Reaction time for the extreme case where the complete reaction load is concentrated on a single core at each time step. The method used to compute these wall clock time statistics is explained in Appendix A. In these figures, for comparison, we also show the wall clock time for the case where the chemistry in the LES/HPDF simulation is represented using a single scalar (mixture-fraction) based flamelet implementation (without using ISAT). Additionally we show two estimates of the best wall clock time that can be achieved using ISAT/*x2f_mpi* if (i) all the cores have pre-built ISAT tables, and all the particles can be resolved by retrieving reaction mappings from the local tables; and (ii) the communication cost is zero, and the chemistry workload, allowing for a typical fraction of direct evaluations in addition to retrieves, is perfectly balanced among all the cores. The method used to make the best wall clock time estimates is explained in Appendix B.

A summary of relative wall clock times required for simulating the Sandia Flame D with the chemistry represented using different methods is given in Table 1.

Based on these results, we can draw the following conclusions:

1. for both the mechanisms, the waiting time, which is indicative of the extent of the load imbalance, is maximum for PLP, minimum for URAN (due to near-ideal load balancing), and moderate for P-URAN (mainly due to load imbalance across partitions);
2. in the P-URAN strategy, for the 16-species and 38-species mechanisms respectively, more than 40% and 60% of the overall wall clock time is spent on reaction, confirming that reaction is the most expensive part of these computations;
3. the P-URAN strategy yields the lowest wall clock time for both the mechanisms: more than 25% less than PLP or URAN for the 16-species mechanism; and about 10% and 50% less than URAN and PLP, respectively for the 38-species mechanism;
4. the wall clock time with P-URAN is within a factor of 1.5 and 1.7 of the best wall clock time estimates (based on no communication) for the 16-species and 38-species mechanisms, respectively;

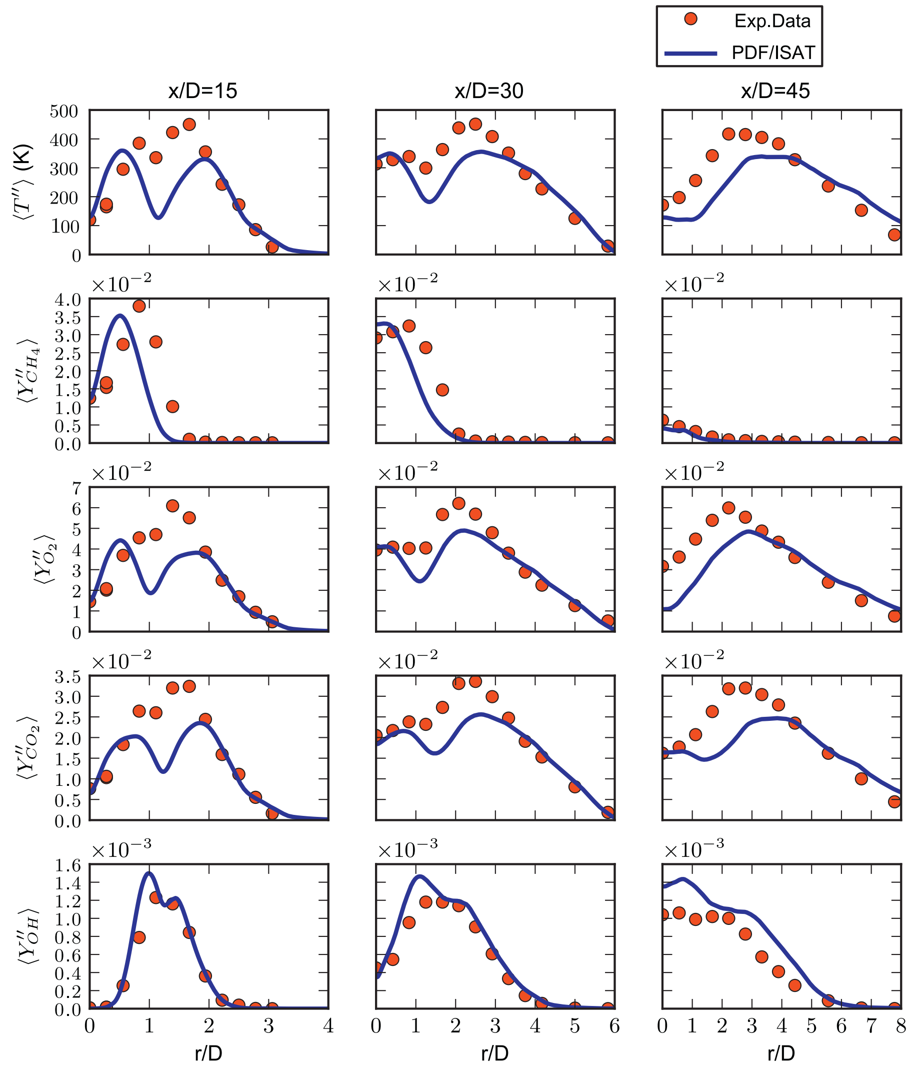


Fig. 4. Radial profiles of time-averaged density-weighted standard deviation of temperature T, and mass fraction of species CH_4, O_2, CO_2, OH at axial locations $x/D = 15, 30, 45$ obtained from experimental data and an LES/PDF simulation using the 38-species mechanism.

5. the P-URAN strategy, compared to the simple single scalar based flamelet representation, is more expensive by a factor of only 2.7 for 16-species and 5.4 for 38-species.

In short, we have shown that the P-URAN strategy performs much better than the PLP and URAN strategies, and yields wall clock time within a factor of 1.5 and 1.7 of estimates for the lowest theoretically achievable wall clock time for the 16-species and 38-species mechanisms, respectively.

Here we have compared the relative performance of the three strategies after the flame has reached a statistically-stationary state, and we find that the P-URAN strategy performs the best. A great deal of computational time can be expended reaching the statistically-stationary state, but even during these initial computations the P-URAN strategy is expected to perform the best. A relatively brief time should be spent in the PLP stage (in P-URAN) during these initial computations, because the chemistry in the domain is evolving quickly, which reduces the chances of a local retrieve.

5.2.1. P-URAN: sensitivity tests

In the previous section we considered specific strategies P-URAN[0.2 h, 32] (for the 16-species mechanism) and P-URAN[0.1 h, 32] (for the 38-species mechanism) for testing P-URAN. Here we perform sensitivity studies to see how P-URAN performs with changes in the time spent in the PLP stage (τ) and the partition size (κ).

By definition, the P-URAN strategy has the following limits in which it reduces to the PLP or URAN strategy:

- P-URAN[$\tau = \infty, \kappa$] = PLP
- P-URAN[$\tau, \kappa = 1$] = PLP
- P-URAN[$\tau = 0, \kappa = N_c$] = URAN

In our tests with the P-URAN strategy, we typically use a value of τ less than 0.5 h, and choose a partition size, κ, which is a multiple of the domain decomposition in the radial (or lateral) direction, D_y, and is preferably closer to $\sqrt{N_c}$. In the sensitivity results presented here, we show that the wall clock time with the P-URAN strategy shows very little sensitivity to changes in the values of τ and κ in the typical range of values that we might use in our computations, and consistently performs better than the PLP and URAN strategies. We have not tried to study how the P-URAN strategy approaches the aforementioned limits for extreme values of τ and κ.

First, we fix the partition size, $\kappa = 32$, and vary the time spent in the PLP stage, τ, from 0 to 5 h, and compute the overall wall clock

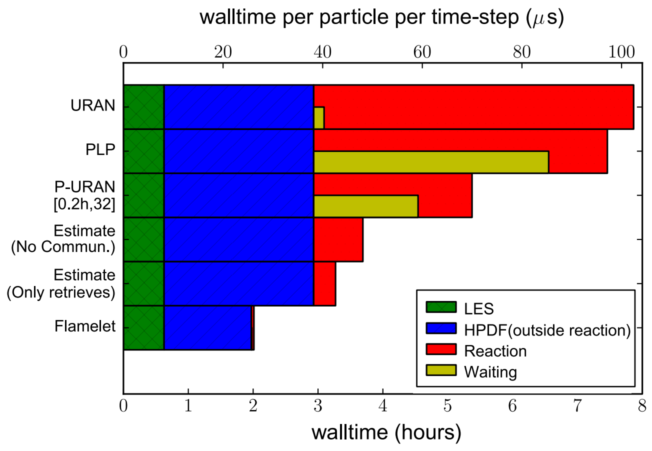


Fig. 5. LES/PDF simulation of Sandia Flame D with the 16-species mechanism on $N_c = 1024$ cores. Wall clock time for 2000 time steps along with breakdown of time spent in LES, HPDF (outside reaction), Reaction (including *x2f_mpi* communication) and Waiting (average idle time) using different parallel strategies. From bottom to top: (1) Flamelet – using mixture-fraction based flamelet representation of chemistry; (2) Estimate (only retrieves) – estimate based on performing only local retrieves using pre-built ISAT tables; (3) Estimate (no commun.) – estimate based on perfect load balancing with no communication cost; (4) P-URAN[0.2 h, 32]; (5) PLP and (6) URAN.

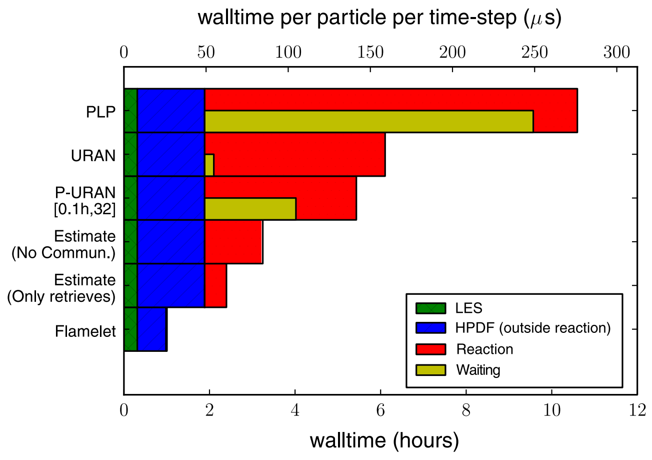


Fig. 6. LES/PDF simulation of Sandia Flame D with the 38-species mechanism on $N_c = 1024$ cores. Wall clock time for 1000 time steps along with breakdown of time spent in LES, HPDF (outside reaction), Reaction (including *x2f_mpi* communication) and Waiting (average idle time) using different parallel strategies. From bottom to top: (1) Flamelet – using mixture-fraction based flamelet representation of chemistry; (2) Estimate (only retrieves) – estimate based on performing only local retrieves using pre-built ISAT tables; (3) Estimate (no commun.) – estimate based on perfect load balancing with no communication cost; (4) P-URAN[0.1 h, 32]; (5) PLP and (6) URAN.

time for running $N_t = 2000$ time steps for the 16-species mechanism. The wall clock time with different strategies is shown in Fig. 7 along with breakdown of time spent in LES, HPDF (outside reaction), Reaction and Waiting in the P-URAN tests. We see that the P-URAN strategy shows very little sensitivity to changes in the time spent in the PLP stage, and the lowest wall clock time is achieved near $\tau = 0.3$ h. The overall wall clock time for the simulation increases by only about 1 h as the time spent in the PLP stage is increased from 0 to 5 h. And in this entire range, P-URAN consistently performs better than PLP and URAN strategies. The reason for this can be understood by studying the breakdown of the wall clock time for the P-URAN tests presented in Fig. 7. We see that as more time is spent in the PLP stage, the Waiting time increases due to the initial load imbalance in the PLP stage. However, a longer time spent in PLP in stage 1 increases the chances of retrieving from the local ISAT tables in stage 2, and so reduces the Reaction time in the stage 2 of P-URAN, thereby yielding approximately the same overall wall clock time for the simulation. The relatively slow approach of the P-URAN wall clock time towards the PLP time also shows that the stage 2 in P-URAN significantly reduces the overall wall clock time relative to using only PLP for the entire simulation. The P-URAN strategy with more time spent in the PLP stage is expected to show better relative performance for longer simulations due to increased chances of local retrieves. In these tests, out of the 2000 time steps, the number of steps completed in the PLP stage with $\tau = 0.5$, 1, 3 and 5 h is 112, 246, 977 and 1665 respectively. In general when performing 24 h simulations using the P-URAN strategy, the data suggests using a value of τ between 0.5 and 1 h, which corresponds to performing about 100–200 time steps (i.e., resolving $\mathcal{O}(10^7)$ particles per core) in the PLP stage.

Next, we fix the time spent in the PLP stage, $\tau = 0.2$ h and change the partition size, κ, to 16, 32 and 64. The results are shown in Fig. 8. Here again, the P-URAN strategy shows very little sensitivity to changes in the partition size. This again is because of a balance achieved between the communication cost and load imbalance. As seen in the breakdown of the wall clock time for the P-URAN tests presented in Fig. 8, smaller partitions reduce the communication cost, but increase the load imbalance between partitions and thus increase the Waiting time. On the other hand, bigger partitions achieve better load balance and reduce the Waiting time, but result in more communication cost and thus increase the Reaction time. For this reason, we suggest using a partition size close to $\sqrt{N_c}$ to strike a balance between the communication cost and load imbalance.

Sensitivity studies with the 38-species mechanism yielded similar results, and hence are not presented here for conciseness.

5.3. Parallel scalability

To assess the parallel scalability of our combined LES/HPDF solver to large numbers of cores, in the next two subsections we study the *weak* and *strong* scaling of our solver.

Table 1
Summary of relative overall wall clock times required for simulating the Sandia Flame D (based on results presented in Figs. 5 and 6) using different methods for representing chemistry.

Method	Chemistry (species, n_s)	Wall clock time	
		(Rel. to LES/flamelet)	(Rel. to LES/HPDF/flamelet)
LES/flamelet	1	1	0.3
LES/HPDF/flamelet	1	3.2	1
LES/HPDF/PLP	16	11.9	3.7
LES/HPDF/URAN	16	12.6	3.9
LES/HPDF/P-URAN	16	8.6	2.7
LES/HPDF/PLP	38	33.9	10.6
LES/HPDF/URAN	38	19.5	6.1
LES/HPDF/P-URAN	38	17.3	5.4

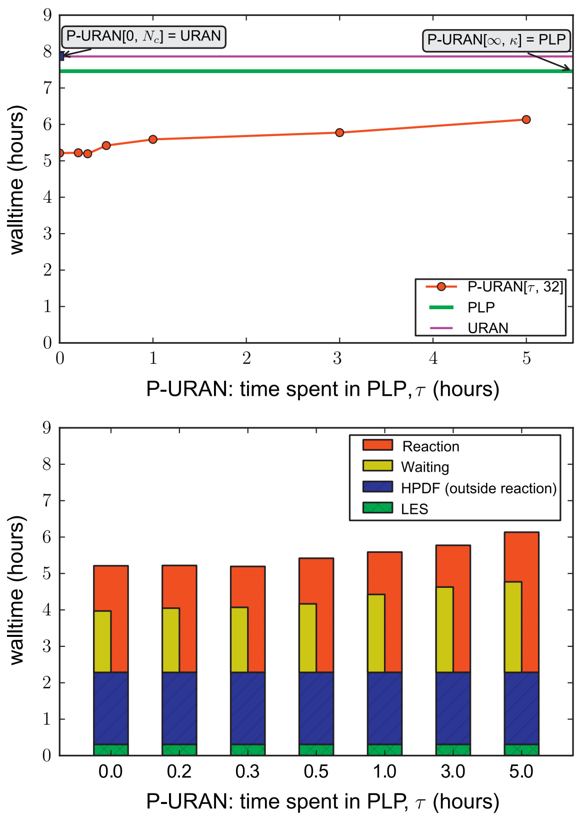


Fig. 7. P-URAN sensitivity tests with the 16-species mechanism. Top: wall clock time for 2000 time steps with (i) PLP; (ii) URAN; and (iii) P-URAN[τ,32] with time τ spent in PLP varied from 0 to 5 hours. Bottom: breakdown of wall clock time spent in LES, HPDF (outside reaction), Reaction (including *x2f_mpi* communication) and Waiting (average idle time) in the P-URAN tests.

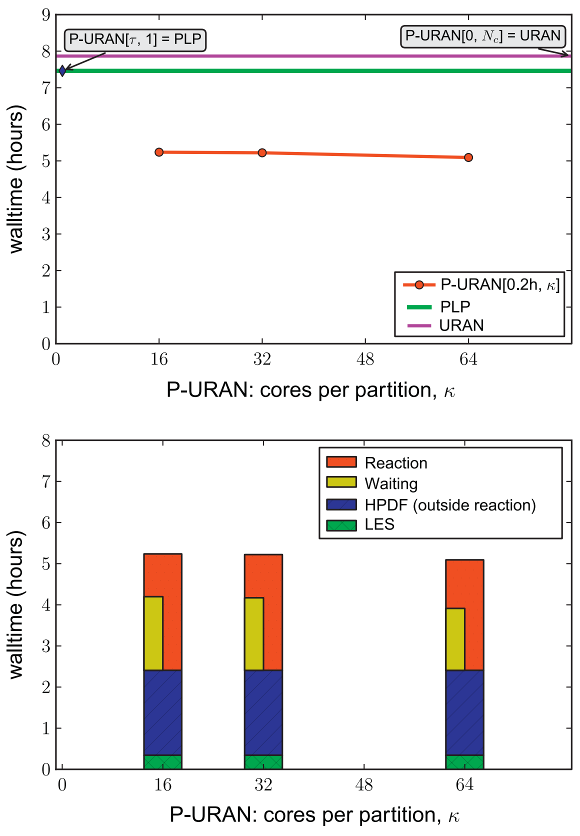


Fig. 8. P-URAN sensitivity tests with the 16-species mechanism. Top: wall clock time for 2000 time steps with (i) PLP; (ii) URAN; and (iii) P-URAN[0.2 h,κ], with the partition size, κ = 16, 32, 64. Bottom: breakdown of wall clock time spent in LES, HPDF (outside reaction), Reaction (including *x2f_mpi* communication) and Waiting (average idle time) in the P-URAN tests.

5.3.1. Weak scaling

The weak scaling study determines how the computational time varies with number of cores for a *fixed workload per core*. Accordingly, the overall problem size is increased in proportion to the number of cores to measure the scalability of the solver. Here the weak scaling tests consist of increasing the number of particles per cell (N_{pc}) in proportion to the number of cores.

We perform weak scaling tests with the URAN and P-URAN strategies. Typically we use 30–50 particles per cell (N_{pc}) in LES/PDF computations. We start the weak scaling tests from 2304 cores with N_{pc} = 30 and go up to 9216 cores with N_{pc} = 120. (We do not start the weak scaling tests from around 1000 cores to avoid an unrealistically large value of N_{pc} at 9216 cores.) The details of the tests performed for the 16-species and 38-species mechanisms are provided in Table 2. In each case, we perform an LES/PDF simulation for N_t = 1000 time steps. The wall clock time per time step (averaged over 1000 time steps) along with breakdown of time spent in LES, HPDF (outside reaction), Reaction and Waiting for the 16-species and 38-species mechanisms is shown in Figs. 9 and 10, respectively. We see that the P-URAN strategy consistently performs better than URAN on all four of the core counts for both the mechanisms, and also shows better weak-scaling up to 9216 cores than the URAN strategy. For the 16-species mechanism, compared to results presented in Fig. 5, in the weak scaling results (Fig. 9) the P-URAN strategy does not perform significantly better than URAN because these are relatively smaller runs (1000 time steps) and we expect the wall clock time with P-URAN to improve for longer runs due to increased probability of local retrieves and reduced communication cost. In Fig. 10, we notice that the Reaction time with the P-URAN strategy is not monotonic, and the wall clock time slightly reduces when moving from 4608 to 6144 cores. This could simply be due to load variations on the compute cluster. As mentioned in Appendix A, we typically observe a 5% variation in the computed wall clock times on repeated runs of our solver.

In order to quantify the weak scaling behavior more accurately, we define the weak scaling efficiency at N_c cores relative to a base case at N_b cores, denoted by $E_W(N_c|N_b)$, as

$$E_W(N_c|N_b) = \frac{T(N_b)}{T(N_c)}, \tag{2}$$

where $T(N_b)$ and $T(N_c)$ denote the wall clock time using N_b and N_c cores, respectively. Here we take the base case to be N_b = 2304 cores. The relative weak scaling efficiency overall and of reaction for the 16-species and 38-species mechanisms using the URAN and P-URAN strategies is shown in Fig. 11. We see that

1. at higher core counts the relative weak scaling efficiency with P-URAN is better than URAN by 5–10%;
2. with P-URAN the relative weak scaling efficiency of reaction at 9216 cores is close to 85% for both the mechanisms, and over 90% overall.

5.3.2. Strong scaling

The strong scaling study determines how the computational time varies with number of cores for a *fixed overall problem size*. For ideal strong scalability, the time to solution for the LES/PDF solver would decrease in inverse proportion to the number of cores employed (so-called linear speedup).

Table 2
Computational details of the weak-scaling tests performed for the 16-species and 38-species mechanisms using the URAN and P-URAN strategies.

N_c	$D_x \times D_y$	N_{pc}	Strategies (16-species mech.)	Strategies (38-species mech.)
2304	48 × 48	30	URAN, P-URAN[0.2 h,48]	URAN, P-URAN[0.1 h,48]
4608	96 × 48	60	URAN, P-URAN[0.2 h,48]	URAN, P-URAN[0.1 h,48]
6144	96 × 64	80	URAN, P-URAN[0.2 h,64]	URAN, P-URAN[0.1 h,64]
9216	96 × 96	120	URAN, P-URAN[0.2 h,96]	URAN, P-URAN[0.1 h,96]

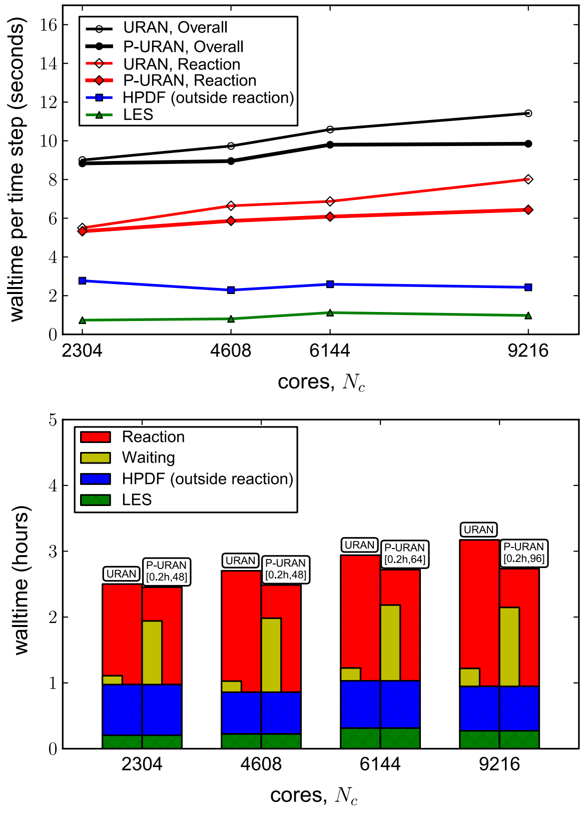


Fig. 9. Weak scaling tests using the URAN and P-URAN strategies with the 16-species mechanism (test details provided in Table 2) for $N_t = 1000$ time steps. Top: weak scaling – wall clock time per time step spent in LES, HPDF (outside reaction), Reaction and Overall. Bottom: breakdown of wall clock time spent in LES, HPDF (outside reaction), Reaction (including *x2f_mpi* communication) and Waiting (average idle time).

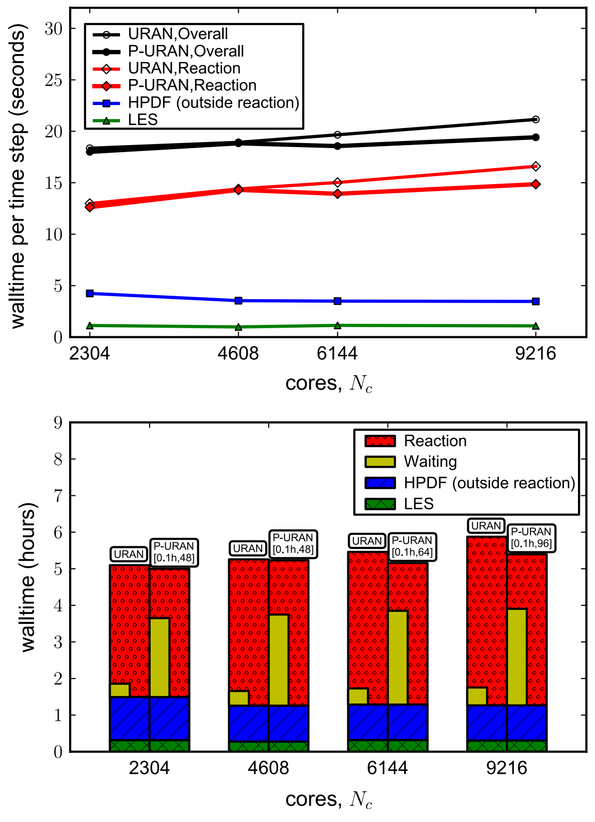


Fig. 10. Weak scaling tests using the URAN and P-URAN strategies with the 38-species mechanism (test details provided in Table 2) for $N_t = 1000$ time steps. Top: weak scaling – wall clock time per time step spent in LES, HPDF (outside reaction), Reaction and Overall. Bottom: breakdown of wall clock time spent in LES, HPDF (outside reaction), Reaction (including *x2f_mpi* communication) and Waiting (average idle time).

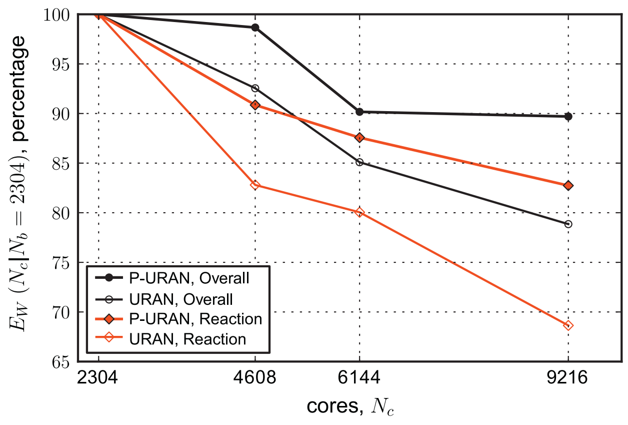
We perform strong scaling studies with just the P-URAN strategy on 1152–9216 cores. In these strong scaling tests, we use a fixed number of particles per cell, $N_{pc} = 40$, and increase the number of cores to see how different parts of the code scale. We perform $N_t = 2000$ and $N_t = 1000$ time steps with the 16-species and 38-species mechanisms, respectively. The computational details of the tests performed are listed in Table 3. Here to determine the strong scaling, we estimate the wall clock time per time step for each core count using two methods:

1. by computing the average wall clock time per time step over the first $N_t \times N_c/9216$ time steps; this corresponds to the same average number of particles evaluated per core for each core count;
2. by computing the average wall clock time per time step over the complete N_t time steps; this corresponds to same overall workload independent of core count.

Among these, the first estimate gives a more accurate measure of strong scaling for the reaction due to approximately same number of particles evaluated per core. The wall clock time per time step using the above two estimates (along with breakdown of time spent in LES, HPDF (outside reaction), Reaction and Waiting) for the 16-species and 38-species mechanisms is shown in Figs. 12 and 13, respectively.

In these results, we observe the following:

1. The LES solver does not show much, if any, parallel speedup in this range of cores. (The Stanford LES code is found not to scale well beyond 500–1000 cores.) However, even on 9216 cores, the LES time represents just 10% of the total time, and so this is not a critical issue in these tests.
2. The HPDF solver shows some parallel speedup on up to 4608 cores, but flattens out beyond that.
3. The reaction part shows a monotonically decaying wall clock time on up to 9216 cores for both the mechanisms. Superficially, the data appear to fit a power law, but the behavior could equally well be explained by a model such as Amdahl's Law.
4. Beyond 4608 cores, the wall clock time of the parts of the code that do not scale well starts becoming comparable to the reaction time, and hence the overall speedup begins to deteriorate.



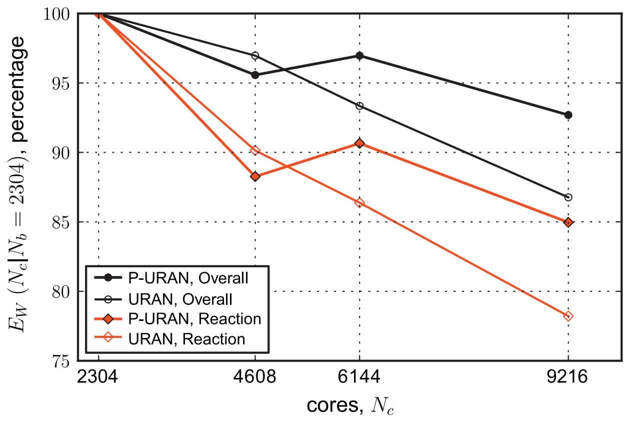


Fig. 11. Relative weak scaling efficiency of Reaction and Overall using the URAN and P-URAN strategies. Top: with the 16-species mechanism for a simulation of N_t = 1000 time steps. Bottom: with the 38-species mechanism for a simulation of N_t = 1000 time steps.

Similar to the relative weak scaling efficiency (Eq. (2)), in order to quantify the strong scaling behavior more accurately, we define the strong scaling efficiency at N_c cores relative to a base case at N_b cores, denoted by $E_S(N_c|N_b)$, as

$$E_S(N_c|N_b) = \frac{N_b}{N_c} \times \left(\frac{T(N_b)}{T(N_c)}\right), \quad (3)$$

where in this case we take the base case to be N_b = 1152 cores. We measure the relative strong scaling efficiency up to 9216 cores, individually for LES, HPDF (outside reaction), Reaction and Overall. The relative strong scaling efficiency (with respect to the base case at N_b = 1152 cores) for the 16-species and 38-species mechanisms is shown in Fig. 14.

In these strong scaling results we notice:

1. at N_c = 9216 cores, the relative strong scaling efficiency of reaction for the 16-species mechanism is about 50% and for the 38-species mechanism is close to 60%. However, up to N_c = 6144 cores, the relative strong scaling efficiency of reaction stays above 60% for both the mechanisms;
2. the LES and HPDF parts show poor scaling, with respective relative strong scaling efficiencies dropping below 30% and 40% beyond N_c = 6144 cores for both the mechanisms;
3. the overall relative strong scaling efficiency drops below 50% and 60% beyond N_c = 6144 cores for the 16-species and 38-species mechanisms, respectively.

In short, these results show that the overall scaling still needs significant improvement, especially in the LES and HPDF parts. However, the reaction part alone with the P-URAN strategy shows acceptable strong scaling up to 6144 cores.

Below we list a couple of possible reasons for not being able to achieve better strong scaling for reaction with the current implementation of the P-URAN strategy, and some ideas for improvement:

1. One possible reason for not achieving perfect scaling could be the increase in the partition size (κ) used in the P-URAN strategy with the increase in cores, N_c. As the partition size increases, the communication cost increases, thereby worsening the scaling. Under the current partitioning scheme (described in Fig. 2), we choose the partition size, κ, to be an exact multiple of the domain decomposition in the radial direction, D_y, to achieve good load balance. So, at larger number of cores like N_c = 9216 with a domain decomposition of 96 × 96, we are forced to use a minimum partition size of κ = 96. For example, under the current partitioning scheme, if we use $\kappa = D_y/2 = 48$ for the 96 × 96 domain decomposition, then this leads to a significant load imbalance between the partitions involving the first 48 cores (handling the chemically reactive sub-domains), and those involving the last 48 cores (handling the coflow/air sub-domains) in the radial direction. One possible way to avoid this load imbalance is to use κ = 48, but group every alternating ranked core in the radial direction into one partition, i.e., for every D_y cores in the radial direction, group all the even ranked cores in one partition, and the remaining odd ranked cores in another partition. This way we achieve good load balancing with smaller partition size and reduced communication cost. This new partitioning scheme can similarly be extended to any partition size $\kappa = D_y/m$, where m is some positive integer, by grouping every mth ranked core into one partition. However, both the current and the new suggested partitioning schemes use the *a priori* knowledge about the direction of load imbalance (which in the simulation of turbulent reacting jet flows is the radial direction) to form the partitions. Ideally, one would like to have an adaptive partitioning strategy to form the partitions "on the fly" without using any *a priori* knowledge about the computational problem being studied, and we are currently working on developing such an adaptive strategy which might help improve the scaling.
2. Another possible reason for not achieving perfect scaling could be the use of one ISAT table per core and MPI alone for communication and parallelization. As the number of cores increase, the use of one ISAT table per core leads to significant duplication of reaction mapping evaluation and addition of similar particle compositions to ISAT tables on multiple cores. One

Table 3
Computational details of the strong-scaling tests performed for the 16-species and 38-species mechanisms using the P-URAN strategy.

N_c	$D_x \times D_y$	N_{pc}	Strategies (16-species mech.)	Strategies (38-species mech.)
1152	96 × 12	40	P-URAN[0.2 h, 36]	P-URAN[0.1 h, 36]
2304	96 × 24	40	P-URAN[0.2 h, 48]	P-URAN[0.1 h, 48]
4608	96 × 48	40	P-URAN[0.2 h, 48]	P-URAN[0.1 h, 48]
6144	96 × 64	40	P-URAN[0.2 h, 64]	P-URAN[0.1 h, 64]
9216	96 × 96	40	P-URAN[0.2 h, 96]	P-URAN[0.1 h, 96]

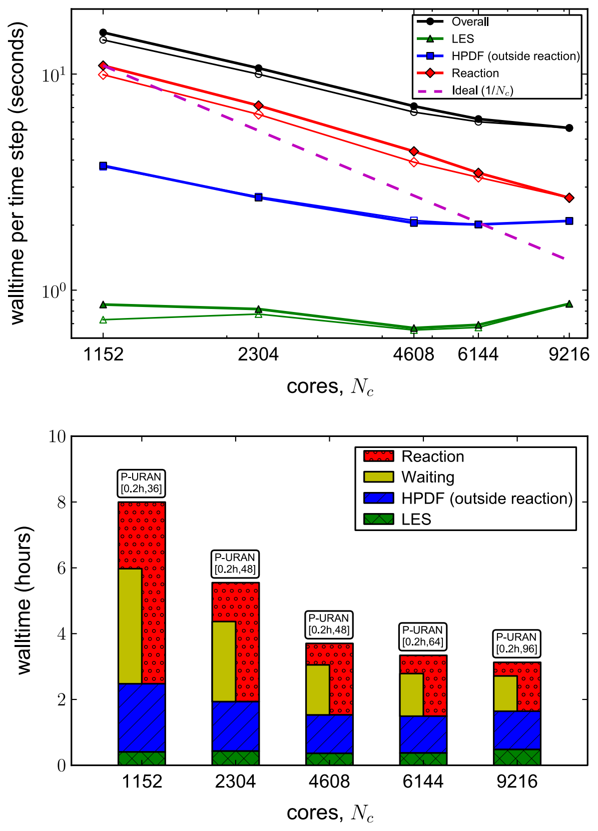


Fig. 12. Strong scaling tests using the P-URAN strategy with the 16-species mechanism (test details provided in Table 3) for $N_t = 2000$ time steps. Top: strong scaling – wall clock time per time step spent in LES, HPDF (outside reaction), Reaction and Overall. solid symbols, average over $N_t \times N_c/9216$ time steps; hollow symbols, average over N_t time steps. Bottom: breakdown of wall clock time spent in LES, HPDF (outside reaction), Reaction (including *x2f_mpi* communication) and Waiting (average idle time) for N_t time steps.

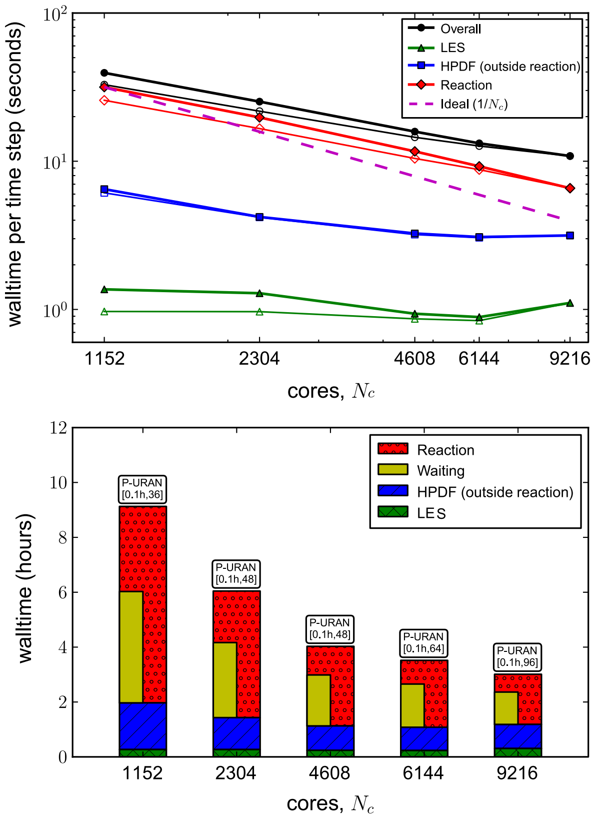


Fig. 13. Strong scaling tests using the P-URAN strategy with the 38-species mechanism (test details provided in Table 3) for $N_t = 1000$ time steps. Top: strong scaling – wall clock time per time step spent in LES, HPDF (outside reaction), Reaction and Overall. solid symbols, average over $N_t \times N_c/9216$ time steps; hollow symbols, average over N_t time steps. Bottom: breakdown of wall clock time spent in LES, HPDF (outside reaction), Reaction (including *x2f_mpi* communication) and Waiting (average idle time) for N_t time steps.

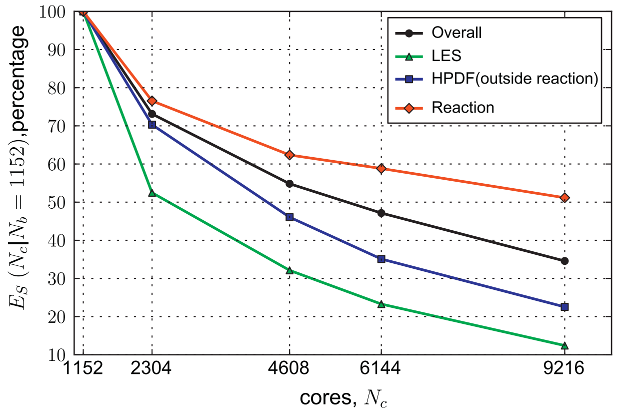
possible way to reduce the duplication of work and data is to use one ISAT table per processor shared by all the cores (4 to 16) on that processor. In addition, hybrid MPI/OpenMP or Graphics Processing Unit (GPU) implementation can be used to achieve better scaling. It would involve enormous work for us to incorporate these ideas in our current implementation of the LES/PDF solver, but it could be considered in future applications of ISAT for other problems.

5.4. Generality of the results

To conclude this section, we discuss the generality of the conclusions drawn from this work. In particular, we consider the sensitivity of our results to changes in the combustion problem being simulated and to changes in the compute cluster architecture.

In this study the results are based on the simulation of a relatively simple turbulent jet flame, the Sandia Flame D. However, we expect the conclusions drawn from this work to be valid over a wider range of combustion problems of interest. For instance, for the simulation of Sandia Flames E and F, which exhibit significantly more local extinction than Flame D and are computationally more expensive, we expect the P-URAN strategy to again yield the lowest wall clock time. Relative to Flame D, the Flames E and F are expected to exhibit greater load-imbalance between regions of flame front and co-flow due to stronger turbulent chemistry interactions, and as a result the PLP strategy should perform poorly. The P-URAN strategy (with a partitioning scheme similar to the one used in this study) should perform better than the PLP and URAN strategies, especially at large numbers of cores. Even for the simulation of turbulent flames with different geometry, the P-URAN strategy with an appropriate partitioning scheme is expected to perform better than the PLP and URAN strategies as it is able to strike a good balance between the communication cost and load imbalance.

In this study, all the simulations are performed on the TACC Ranger cluster. Due to the generality of the MPI implementation in our solver, we expect the results presented here to be relatively insensitive to changes in the cluster architecture. However, we do realize that by taking into account the cluster architecture and core-connectivity we may be able to come up with a superior mapping of the LES/PDF sub-domains onto nodes or cores. In the present study, we have attempted to do this by assigning the MPI ranks in radial order, then axial (as shown in Fig. 2). As a consequence, each block of 16 sub-domains in the radial direction is assigned to a single node (16 cores) on Ranger. Since in the P-URAN strategy the communication is restricted to sub-domains in the same partition, and partitions are set according to axial location, the current MPI rank-assignment and partitioning schemes ensure that most of the MPI communication happens intra-node. To reduce inter-node communication still further, we try to use a partition size κ which is an exact multiple of the number of cores per node, which is 16 on Ranger. We thereby minimize the relatively slow inter-node communication and take the best advantage of the faster intra-node connectivity. Similar considerations are easily extended to other clusters, given the ubiquity of architectures that feature multiple cores per node.



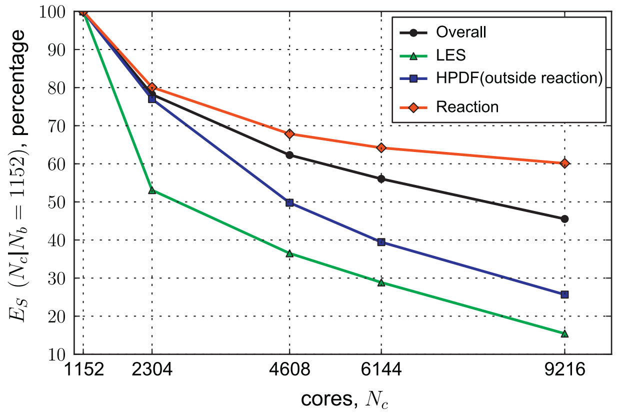


Fig. 14. Relative strong scaling efficiency of LES, HPDF (outside reaction), Reaction and Overall using the P-URAN strategy. Top: with the 16-species mechanism for a simulation of $N_t = 2000$ time steps. Bottom: with the 38-species mechanism for a simulation of $N_t = 1000$ time steps.

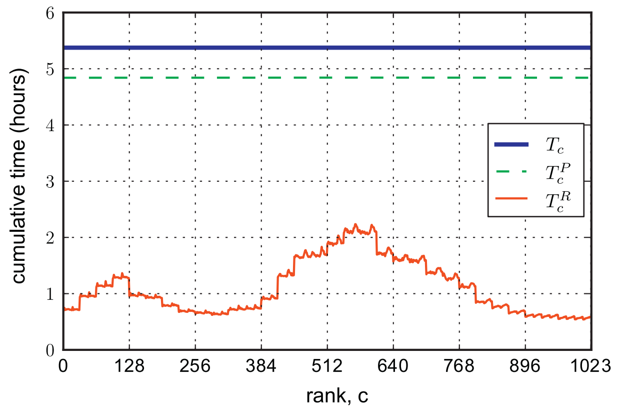


Fig. 15. LES/PDF simulation of the Sandia Flame D for $N_t = 2000$ time steps on $N_c = 1024$ cores with chemistry represented using the 16-species mechanism with the P-URAN[0.2 h, 32] parallel strategy. On each core ranked, $c = 0–1023$, plotted are the cumulative wall clock time spent on the computations, T_c; the part of the time spent within HPDF, T_c^P; and within HPDF the part of the time spent on Reaction (including *x2f_mpi* communication), T_c^R.

6. Conclusions

We have successfully developed an integrated LES/HPDF/x2f_mpi/ISAT solver for performing turbulent combustion calculations with realistic combustion chemistry. We have demonstrated a new parallel strategy, P-URAN, implemented using the *x2f_mpi* library for performing chemistry calculations efficiently. In particular, we have shown that for performing LES/PDF calculations, the P-URAN strategy:

- yields the lowest wall clock time among all the strategies tested;
- yields a wall clock time within a factor of 1.5 and 1.7 of estimates for the lowest theoretically achievable wall clock time for the 16-species and 38-species mechanisms, respectively;
- compared to the single scalar mixture-fraction based flamelet implementation, is more expensive by only a factor of 2.7 and 5.4 for the 16-species and 38-species mechanisms, respectively;
- achieves a relative weak scaling efficiency for reaction of about 85% when scaling from 2304 to 9216 cores; and
- achieves a relative strong scaling efficiency for reaction of over 60% when scaling from 1152 to 6144 cores.

Acknowledgments

V.H.'s work on the methane ARM1 mechanism is supported by Office of Energy Research, Office of Basic Energy Sciences, Chemical Sciences, Geosciences and Biosciences Division of the US Department of Energy (DOE) under Grant No. DE-FG02-90ER. The work of V.H. and S.R.L. on the ethylene mechanism is supported by Grant No. FA9550-09-1-0611 funded by the National Center for Hypersonic Combined Cycle Propulsion, sponsored by the AFOSR and NASA ARMD. S.R.L.'s initial work on this project was supported by NASA Grant No. NNX08AB36A. This research was also supported in part by the National Science Foundation through TeraGrid resources provided by the Texas Advanced Computing Center under Grant No. TG-CTS090020. S.B.P. has a financial interest in Ithaca Combustion Enterprise, LLC., which has licensed the software ISAT-CK used in this work.

Appendix A. Wall clock time statistics and estimates

In our combined LES/HPDF solver, we collect various wall clock time statistics. Here we describe the method used to estimate the breakdown of the wall clock time spent in LES, HPDF (outside reaction), Reaction and Waiting, as presented in some of the results (figures) in this work.

In our LES/HPDF solver, at each time step $n = 1$ to N_t of a simulation performed on N_c cores ranked from $c = 0$ to $N_c - 1$, we compute: the overall wall clock time spent in that time step, $t_{n,c}$; the part of the time spent within HPDF, $t_{n,c}^{P}$; and, within HPDF, the part of the time spent on Reaction (including *x2f_mpi* communication), $t_{n,c}^{R}$. Cumulative wall clock time statistics can then be collected on each core by summing over all the time steps as follows:

$$T_c = \sum_{n=1}^{N_t} t_{n,c}, \tag{4}$$

and similarly for T_c^P and T_c^R. For instance, Fig. 15 shows the cumulative wall clock time statistics collected from a Sandia Flame D simulation performed for $N_t = 2000$ time steps on $N_c = 1024$ cores with the chemistry represented using the 16-species mechanism with the P-URAN parallel strategy.

Now using these time statistics collected on each core, we estimate the global wall clock time statistics for the entire run as follows. The LES and HPDF solvers are synchronized (among all the cores) at the end of each time step, and so the wall clock time spent in each time step, $t_{n,c}$, and the wall clock time spent in HPDF, $t_{n,c}^{P}$, are approximately the same on all the cores, and so are the cumulative times T and T^P (as seen in Fig. 15). Hence, we take the time statistics from the core ranked $c = 0$ and estimate the overall wall clock time spent for the computations as

$$T = T_0 = \sum_{n=1}^{N_t} t_{n,0}, \tag{5}$$

and the wall clock time spent within HPDF as

$$T^{\mathrm{P}} = T_0^{\mathrm{P}} = \sum_{n=1}^{N_t} t_{n,0}^{\mathrm{P}}, \tag{6}$$

which gives the wall clock time spent within LES as

$$T^{\mathrm{L}} = T - T^{\mathrm{P}}. \tag{7}$$

However, the wall clock time spent in Reaction, $t_{n,c}^{\mathrm{R}}$, is found to vary significantly across the cores (as seen in Fig. 15), depending on the strategy used for implementing chemistry. So we estimate the overall wall clock time spent in Reaction (including *x2f_mpi* communication) to be the cumulative sum of the maximum reaction time (at each time step, over all the cores) as follows:

$$T^{\mathrm{R}} = \sum_{n=1}^{N_t} \max_c \left(t_{n,c}^{\mathrm{R}} \right). \tag{8}$$

The time spent in HPDF (outside reaction) is then given as

$$T^{\mathrm{H}} = T^{\mathrm{P}} - T^{\mathrm{R}}, \tag{9}$$

and consequently we have

$$T = T^{\mathrm{L}} + T^{\mathrm{H}} + T^{\mathrm{R}}. \tag{10}$$

In addition, we also estimate the average (idle) Waiting time as follows. On a given time step, the "slowest" core is defined to be that which takes the greatest time for reaction. The Waiting time is the average idle time of the other cores spent waiting for the slowest core to complete, and is computed as follows

$$T^{\mathrm{W}} = \sum_{n=1}^{N_t} \frac{1}{(N_c - 1)} \sum_{c=0}^{N_c-1} \left(\max_{c'} \left(t_{n,c'}^{\mathrm{R}} \right) - t_{n,c}^{\mathrm{R}} \right). \tag{11}$$

Note that the Waiting time is in parallel with the Reaction time, and is indicative of the extent of reaction load imbalance. The Waiting time has a lower bound of zero indicating perfect reaction load balancing, and an upper bound equal to the Reaction time for the extreme case where the complete reaction load is concentrated on a single core at each time step.

In summary, in the figures, we plot the overall wall clock time, T; the LES time, T^{L}; the HPDF (outside reaction) time, T^{H}; the Reaction (including *x2f_mpi* communication) time, T^{R}; and the Waiting time, T^{W}.

It should be noted that typically we observe about 5% variation in the computed wall clock times (on repeated runs of our solver with identical initial conditions) due to load variations on the TACC Ranger cluster.

Appendix B. Best performance estimates

In the results presented in Figs. 5 and 6, we make estimates for the lowest theoretically achievable wall clock time. These estimates are made using the following method.

For both the mechanisms, we consider the LES/PDF simulation performed on 1024 cores using the URAN strategy (which achieves near-ideal load balancing), and find the core rank, c, with the maximum cumulative reaction time, T_c^{R}. On this core, we compute the total number of ISAT queries performed (i.e., particles resolved), N_q; the average ISAT query time, t_q, (i.e., the average time taken to resolve a particle using ISAT); and the average ISAT retrieve time, t_r, (i.e., the average time taken to retrieve a particle's reaction mapping using the ISAT table). For the 16-species mechanism, we find t_q = 9 μs and t_r = 4 μs; and for the 38-species mechanism, we find t_q = 32 μs and t_r = 12 μs.

Now using these data, we make two estimates for the best wall clock time for reaction:

1. Estimate (only retrieves) – estimate based on performing only local retrieves using pre-built ISAT tables. In this we estimate the reaction wall clock time on all the cores to be the same, $T^{\mathrm{R}} = T_c^{\mathrm{R}} = N_q \times t_r$.
2. Estimate (no commun.) – estimate based on perfect load balancing with no *x2f_mpi* communication cost, while allowing for a typical fraction of direct evaluations to be performed in addition to retrieves. In this we estimate the reaction wall clock time on all the cores to be the same, $T^{\mathrm{R}} = T_c^{\mathrm{R}} = N_q \times t_q$.

References

[1] W.J. Pitz, N.P. Cernansky, F.L. Dryer, F.N. Egolfopoulos, J.T. Farrell, D.G. Friend, H. Pitsch, Society of Automotive Engineers SAE-2007-01-0175, 2007.
[2] C.K. Westbrook, W.J. Pitz, O. Herbinet, H.J. Curran, E.J. Silke, Combust. Flame 156 (2009) 181–199.
[3] T. Lu, C.K. Law, Proc. Combust. Inst. 30 (2005) 1333–1341.
[4] P. Pepiot-Desjardins, H. Pitsch, Combust. Flame 154 (2008) 67–81.
[5] T. Nagy, T. Turanyi, Combust. Flame 156 (2009) 417–428.
[6] J.C. Keck, D. Gillespie, Combust. Flame 17 (1971) 237–241.
[7] S.H. Lam, D.A. Goussis, Int. J. Chem. Kinet. 26 (1994) 461–486.
[8] U. Maas, S.B. Pope, Combust. Flame 88 (1992) 239–264.
[9] Z. Ren, S.B. Pope, A. Vladimirsky, J.M. Guckenheimer, J. Chem. Phys. 124 (2006). Art. No. 114111.
[10] S.B. Pope, Combust. Theory Model. 1 (1997) 41–63.
[11] L. Lu, S.B. Pope, J. Comput. Phys. 228 (2) (2009) 361–386.
[12] S.R. Tonse, N.W. Moriarity, N.J. Brown, M. Frenklach, Israel J. Chem. 39 (1999) 97–106.
[13] T. Turanyi, Comput. Chem. 18 (1994) 45–54.
[14] V. Hiremath, Z. Ren, S.B. Pope, Combust. Theory Model. 14 (5) (2010) 619–652.
[15] V. Hiremath, Z. Ren, S.B. Pope, Combust. Flame 158 (11) (2011) 2113–2127.
[16] Z. Ren, G.M. Goldin, V. Hiremath, S.B. Pope, Combust. Theory Model. 15 (6) (2011) 827–848.
[17] Z. Ren, G.M. Goldin, An efficient time scale model with tabulation of chemical equilibrium, Combust. Flame 158 (10) (2011) 1977–1979.
[18] L. Lu, Z. Ren, V. Raman, S.B. Pope, H. Pitsch, in: Proceedings of the CTR Summer Program, 2004, pp. 283–294.
[19] L. Lu, Z. Ren, S.R. Lantz, V. Raman, S.B. Pope, H. Pitsch, in: Proceedings of the 4th Joint Meeting of US Sections of the Combustion Institute, Philadelphia, PA, 2005.
[20] L. Lu, S.R. Lantz, Z. Ren, S.B. Pope, J. Comput. Phys. 228 (15) (2009) 5490–5525.
[21] Y. Shi, W.H. Green, H.-W. Wong, O.O. Oluwole, Combust. Flame 158 (5) (2011) 836–847.
[22] R.S. Barlow, J.H. Frank, Proc. Combust. Inst. 27 (1998) 1087–1095. <http://www.sandia.gov/TNF/DataArch/FlameD.html>.
[23] A. Kumar, S. Mazumder, Comput. Chem. Eng. 35 (7) (2011) 1317–1327.
[24] J.D. Hedengren, T.F. Edgar, Comput. Chem. Eng. 32 (2008) 706–714.
[25] A. Arsenlis, N. Barton, R. Becker, R. Rudd, Comput. Methods Appl. Mech. Eng. 196 (2006) 1–13.
[26] C.D. Pierce, Progress-variable approach for Large-Eddy Simulation of turbulent combustion, Ph.D. thesis, Stanford University, 2001.
[27] C.D. Pierce, P. Moin, J. Fluid Mech. 504 (2004) 73–97.
[28] H. Wang, S.B. Pope, Proc. Combust. Inst. 33 (2011) 1319–1330.
[29] H. Wang, P.P. Popov, S.B. Pope, J. Comput. Phys. 229 (2010) 1852–1878.
[30] P. Kloeden, E. Platen, Numerical Solution of Stochastic Differential Equations, Springer Verlag, Berlin, 1992.
[31] J. Janicka, W. Kolbe, W. Kollmann, J. Non-Equil. Thermodynam. 4 (1970) 47–66.
[32] M. Caracotsios, W.E. Stewart, Comput. Chem. Eng. 9 (1985) 359–365.
[33] R. Rabenseifner, G. Hager, G. Jost, in: Proceedings of the 17th Euromicro International Conference on Parallel, Distributed and Network-Based Processing, PDP 2009, Weimar, Germany, 18–20 Febuary 2009, IEEE Computer Society, 2009.
[34] C.J. Sung, C.K. Law, J.-Y. Chen, Symp. (Int.) Combust. 27 (1998) 295–304. <http://www.princeton.edu/~cklaw/kinetics/12-step/index.html>.
[35] G.E. Esposito, H.K. Chelliah, Combust. Flame 158 (3) (2011) 477–489.
[36] V. Hiremath, S.R. Lantz, H. Wang, S.B. Pope, Proc. Combust. Inst. 34 (2012) submitted for publication.
[37] J. Xu, S.B. Pope, Combust. Flame 123 (2000) 281–307.
[38] V. Raman, R.O. Fox, A.D. Harvey, Combust. Flame 136 (2004) 327–350.
[39] M. Sheikhi, T. Drozda, P. Givi, F. Jaberi, S. Pope, Proc. Combust. Inst. 30 (2005) 549–556.
[40] R. Mustata, L. Vali, C. Jimnez, W. Jones, S. Bondi, Combust. Flame 145 (2006) 88–104.
[41] M.B. Nik, S.L. Yilmaz, P. Givi, M.R.H. Sheikhi, S.B. Pope, Am. Inst. Aeronaut. Astronaut. 48 (2010) 1513–1522.