A Scalable Algorithm for Radiative Heat Transfer Using Reverse Monte Carlo Ray Tracing

Alan Humphrey[1], Todd Harman[1], Martin Berzins[1], and Phillip Smith[1]

University of Utah, Salt Lake City, USA

Abstract. Radiative heat transfer is an important mechanism in a class of challenging engineering and research problems. A direct all-to-all treatment of these problems is prohibitively expensive on large core counts due to pervasive all-to-all MPI communication. The massive heat transfer problem arising from the next generation of clean coal boilers being modeled by the Uintah framework has radiation as a dominant heat transfer mode. Reverse Monte Carlo ray tracing (RMCRT) can be used to solve for the radiative-flux divergence while accounting for the effects of participating media. The ray tracing approach used here replicates the geometry of the boiler on a multi-core node and then uses an all-to-all communication phase to distribute the results globally. The cost of this all-to-all is reduced by using an adaptive mesh approach in which a fine mesh is only used locally, and a coarse mesh is used elsewhere. A model for communication and computation complexity is used to predict performance of this new method. We show this model is consistent with observed results and demonstrate excellent strong scaling to 262K cores on the DOE Titan system on problem sizes that were previously computationally intractable.

Keywords: Uintah, radiation modeling, parallel, scalability, adaptive mesh refinement, simulation science, Titan

1 Introduction

Our study is motivated primarily by the target problem of the University of Utah Carbon Capture Multi-Disciplinary Simulation Center (CCMSC). This project aims to eventually simulate a 350MWe clean coal boiler being developed by Alstom Power during the next five years, by using large parallel computers in a scalable manner for reacting, large eddy simulations (LES)-based codes within the Uintah open source framework, and to use accelerators at large scale.

Within the boiler, the hot combustion gases radiate energy to the boiler walls and to tubes carrying water and steam that is superheated to a supercritical fluid. This steam acts as the working fluid to drive the turbine for power generation. The residual energy in the mixture passes through a convective heat exchange system to extract as much of the remaining energy as possible into the working fluid. This radiative flux depends on the radiative properties of the participating media and temperature. The mixture of particles and gases emits, absorbs and scatters radiation, the modeling of which is a key computational element in these simulations. The radiation calculation, in which the radiative-flux divergence at each cell of the discretized domain is calculated,

can take up to 50% of the overall CPU time per timestep using the discrete ordinates method (DOM), one of the standard approaches to computing radiative heat transfer. This method, which Uintah currently uses, is computationally expensive, involves multiple global, sparse linear solves and presents challenges both with the incorporation of radiation physics such as scattering and to the use of parallel computers at very large scales. Reverse Monte Carlo ray tracing (RMCRT), the focus of this work, is one of the few numerical techniques that can accurately solve for the radiative-flux divergence while accounting for the effects of participating media, naturally incorporates scattering physics, and lends itself to scalable parallelism. The principal challenges with our initial, single fine mesh (single-level) RMCRT approach are the all-to-all communication requirements and on-node memory constraints. To address these challenges, our study explores a multi-level, adaptive mesh refinement (AMR) approach in which a fine mesh is only used close to each grid point and a successively coarser mesh is used further away. The central question of our study will be to determine if our AMR approach can scale to large core counts on modern supercomputers, and if our communication and computation models can accurately predict how this approach to radiation scales on current, emerging and future architectures.

In what follows, Section 2 provides an overview of the Uintah software, while Section 3 describes our RMCRT model in detail and provides an overview of the key RMCRT approaches considered and used within Uintah. Section 4 details our model of communication and computation for our multi-level AMR approach. Section 5 provides strong scaling results over a wide range of core counts (up to 262K cores) for this approach, and an overview of related work is given in Section 6. The paper concludes in Section 7 with future work in this area.

2 The Uintah Code

The Uintah open-source (MIT License) software has been widely ported and used for many different types of problems involving fluids, solids and fluid-structure interaction problems. The present status of Uintah, including applications, is described by [4]. The first documented full release of Uintah was in July 2009 and the latest in January 2015 [37]. Uintah consists of a set of parallel software components and libraries that facilitate the solution of partial differential equations on structured adaptive mesh refinement (AMR) grids. Uintah presently contains four main simulation components: 1.) the multi-material ICE [20] code for both low and high-speed compressible flows; 2.) the multi-material, particle-based code MPM for structural mechanics; 3.) the combined fluid-structure interaction (FSI) algorithm MPM-ICE [12] and 4.) the ARCHES turbulent reacting CFD component [19] that was designed for simulating turbulent reacting flows with participating media radiation. Uintah is highly scalable [24], [6], runs on many National Science Foundation (NSF), Department of Energy (DOE) and Department of Defense (DOD) parallel computers (Stampede, Mira, Titan, Vulcan, Vesta, Garnet, Kilraine, etc) and is also used by many NSF, DOE and DOD projects in areas such as angiogenesis, tissue engineering, green urban modeling, blast-wave simulation, semi-conductor design and multi-scale materials research [4].

Uintah is unique in its combination of the MPM-ICE fluid-structure-interaction solver, ARCHES heat transfer solver, AMR methods and directed acyclic graph (DAG)-based runtime system. Uintah is one of the few codes that uses a DAG approach as part of a production strength code in a way that is coupled to a runtime system. Uintah also provides automated, large-scale parallelism through a design that maintains a clear partition between applications code and its parallel infrastructure, making it possible to achieve great increases in scalability through changes to the runtime system that executes the taskgraph, *without changes to the taskgraph specifications themselves*. The combination of the broad applications class and separation of the applications problems from a highly scalable runtime system has enabled engineers and computer scientists to focus on what each does best, significantly lowering the entry barriers to those who want to compute a parallel solution to an engineering problem. Uintah is open source, freely available and is the only widely available MPM code. The broad international user-base and rigorous testing ensure that the code may be used on a broad class of applications.

Particular advances made in Uintah are scalable adaptive mesh refinement [25] coupled to challenging multiphysics problems [5]. A key factor in improving performance has been the reduction in MPI wait time through the dynamic and even out-of-order execution of task-graphs [29]. The need to reduce memory use in Uintah led to the adoption of a nodal shared memory model in which there is only one MPI process per multicore node, and execution on individual cores is through Pthreads [27]. This has made it possible to reduce memory use by a factor of 10 and to increase the scalability of Uintah to 768K cores on complex fluid-structure interactions with adaptive mesh refinement. Uintah's thread-based runtime system [27], [30] uses: decentralized execution [29] of the task-graph, implemented by each CPU core requesting work itself and performing its own MPI. A shared memory abstraction through Uintah's data warehouse hides message passing from the user but at the cost of multiple cores accessing the warehouse originally. A shared memory approach that is lock-free [30] was implemented by making use of atomic operations (supported by modern CPUs) and thus allows efficient access by all cores to the shared data on a node. Finally, the nodal architecture of Uintah has been extended to run tasks on one or more on-node accelerators [15]. This unified, heterogeneous runtime system [28] makes use of a multi-stage queue architecture (two sets of task queues) to organize work for CPU cores and accelerators in a dynamic way, and is the focus of current development.

2.1 The ARCHES Combustion Simulation Component

The radiation models in Uintah have previously been a part of the ARCHES component, which was designed for the simulation of turbulent reacting flows with participating media. ARCHES is a three-dimensional, large eddy simulation (LES) code that uses a low-Mach number variable density formulation to simulate heat, mass, and momentum transport in reacting flows. The LES algorithm solves the filtered, density-weighted, time-dependent coupled conservation equations for mass, momentum, energy, and particle moment equations in a Cartesian coordinate system [19]. This set of filtered equations is discretized in space and time and solved on a staggered, finite volume mesh. The staggering scheme consists of four offset grids, one for storing scalar quantities

and three for each component of the velocity vector. Stability preserving, second order explicit time-stepping schemes and flux limiting schemes are used to ensure that scalar values remain bounded. ARCHES is second-order accurate in space and time and is highly scalable through Uintah and its coupled solvers like hypre [10] to 256K cores [36]. Research using ARCHES has been done on radiative heat transfer using the parallel discrete ordinates method and the P1 approximation to the radiative transport equation [22]. Recent work has shown that RMCRT methods are potentially more efficient [17], [39].

3 RMCRT Model

Scalable modeling of radiation is currently one of the most challenging problems in large-scale simulations, due to the global, *all-to-all* nature of radiation [31]. To simulate thermal transport, two fundamental approaches exist: random walk simulations, and finite element/finite volume simulations, e.g., discrete ordinates method (DOM) [3], which involves solving many large systems of equations. Accurate radiative-heat transfer algorithms that handle complex physics are inherently computationally expensive [16], particularly when high-accuracy is desired in cases where spectral or geometric complexity is involved. They also have limitations with respect to scalability, bias and accuracy.

The Uintah ARCHES component is designed to solve the mass, momentum, mixture fraction, and thermal energy governing equations inherent to coupled turbulent reacting flows. ARCHES has relied primarily on a legacy DOM solver to compute the radiative source term in the energy equation [19]. Monte Carlo ray tracing (MCRT) methods for solving the radiative transport equation offer higher accuracy in two key areas where DOM suffers: geometric fidelity and spectral resolution. In applications where such high accuracy is important, MCRT can become more efficient than DOM approaches. In particular, MCRT can potentially reduce the cost significantly by taking advantage of modern hardware on large distributed shared memory machines [14], and now on distributed memory systems with on-node graphics processing unit (GPU) accelerators, using a prototype GPU implementation of the single-level RMCRT [15], written using NVidia CUDA.

3.1 Radiation and Ray Tracing Overview

The heat transfer problems arising from the clean coal boilers being modeled by the Uintah framework has thermal radiation as a dominant heat transfer mode and involves solving the conservation of energy equation and radiative heat transfer equation (RTE) simultaneously. Thermal radiation in the target boiler simulations is loosely coupled to the computational fluid dynamics (CFD) due to time-scale separation and is the rightmost source term in the conservation of energy equation shown by:

$$c_v \frac{DT}{Dt} = -\nabla \cdot (\kappa \nabla T) - p\nabla \cdot v + \Phi + Q''' - \nabla \cdot q_r \tag{1}$$

where c_v is the specific heat, T is the temperature field, p is the pressure, k is the thermal conductivity, c is the velocity vector, Φ is the dissipation function, Q''' is the heat generated within the medium, e.g. chemical reaction, and $\nabla \cdot q_r$ is the net radiative source. The energy equation is then conventionally solved by ARCHES (finite volume) and the temperature field, T is used to compute net radiative source term. This net radiative source term is then fed back into energy equation (for the ongoing CFD calculation) which is solved to update the temperature field, T.

A radiatively participating medium can emit, absorb and scatter thermal radiation. The RTE (2) as shown in [41], is the equation describing the interaction of absorption, emission and scattering for radiative heat transfer and is an integro-differential equation with three spatial variables and two angles that determine the direction of $\hat{s}$ [41].

$$\begin{aligned} \frac{dI_\eta(\hat{s})}{ds} &= \hat{s}\nabla I_{\eta(\hat{s})} \\ &= k_\eta I_\eta - \beta_\eta I_\eta(\hat{s}) \\ &+ \frac{\sigma_{s\eta}}{4\pi}\int_{4\pi} I_\eta(\hat{s})\Phi_\eta(\hat{s_i},\hat{s})d\Omega_i, \end{aligned} \tag{2}$$

In (2), k_η is the absorption coefficient, $\sigma_{s\eta}$ is the scattering coefficient (dependent on the incoming direction s and wave number η), β_η is the extinction coefficient that describes total loss in radiative intensity, I_η is the change in intensity of incoming radiation from point s to point $s + ds$ and is determined by summing the contributions from emission, absorption and scattering from direction $\hat{s}$ and scattering into the same direction $\hat{s}$ at wave number η. $\Phi_\eta(\hat{s_i},\hat{s})$ is the phase function that describes the probability that a ray coming from direction s_i will scatter into direction $\hat{s}$ and integration is performed over the entire solid angle Ω_i [32], [41].

DOM, MCRT and RMCRT all aim to approximate the radiative transfer equation. In the case of RMCRT, a statistically significant number of rays (photon bundles) are traced from a computational cell to the point of extinction. This method is then able to calculate energy gains and losses for every element in the computational domain. The process is considered "reverse" through the Helmholtz Reciprocity Principle, e.g. incoming and outgoing intensity can be considered as reversals of each other [13]. Through this process, the divergence of the heat flux for every sub-volume in the domain (and radiative heat flux for surfaces, e.g., boiler walls) is computed by Equation 3, as rays accumulate and attenuate intensity (measured in watts per square meter, SI units based on the StefanBoltzmann constant) according to the RTE for an absorbing, emitting and scattering medium.

$$\nabla \cdot q = \kappa(4\pi I_{emmited} - \int_{4\pi} I_{absorbed}\mathrm{d}\Omega), \tag{3}$$

where the rightmost term, $\int_{4\pi} I_{absorbed}\mathrm{d}\Omega$ is represented by the sum $\sum_{r=1}^{N} I_r \frac{4\pi}{N}$ for each ray r up to N rays. The integration is performed over the entire solid angle Ω. Ray origins are randomly distributed throughout a given computational cell. In our implementation, the Mersenne Twister random number generator [26] is used to generate ray origins. The ray marching algorithm proceeds in a similar fashion to that shown by [1].

Within the Uintah RMCRT module, rays are traced backwards from the detector, thus eliminating the need to track ray bundles that never reach the detector [32]. Rather than integrating the energy lost as a ray traverses the domain, RMCRT integrates the incoming intensity absorbed at the origin, where the ray was emitted. RMCRT is more amenable to domain decomposition, and thus Uintah's parallelization scheme due to the backward nature of the process [38], and the mutual exclusivity of the rays themselves. Figure 1 shows the back path of a ray from **S** to the emitter **E**, on a nine cell structured mesh patch. Each i^{th} cell has its own temperature T_i, absorption coefficient κ_i, scattering coefficient σ_i and appropriate pathlengths $l_{i,j}$ [15]. In each case the incoming intensity is calculated in each cell and then traced back through the other cells. The Uintah RMCRT module computes how much of the outgoing intensity has been attenuated along the path. When a ray hits a boundary, as on surface 17 in Figure 1, the incoming intensities will be partially absorbed by the surface. When a ray hits a hot boundary surface, its emitted surface intensity contributes back to point **S**. Rays are terminated when their intensity is sufficiently small [15].

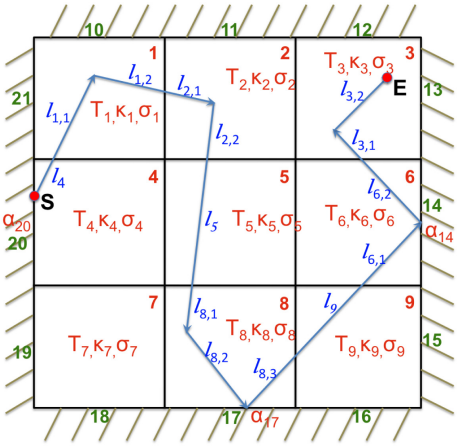

Fig. 1. 2D Outline of reverse Monte Carlo ray tracing [15]

RMCRT uses rays more efficiently than forward MCRT, but it is still an *all-to-all* method, for which all of the geometric information and radiative properties (temperature T, absorption coefficient κ, and cellType (boundary or flow cell)) for the entire computational domain must be accessible by every ray [38]. When using a ray tracing approach (forward or backward), two approaches for parallelizing the computation are considered when using structured grids, 1.) parallelize by patch-based domain decomposition with local information only and pass ray information at patch boundaries via MPI, and 2.) parallelize by patch-based domain decomposition with global information and reconstruct the domain for the quantities of interest on each node by passing domain information via MPI. The first approach becomes untenable due to potentially billions of rays whose information would need to be communicated as they traverse the domain. In the second approach the primary difficulty is efficiently constructing the global information for millions of cells in a spatially decomposed (patch-based) domain. The

second approach is the one taken used in this work. While reconstruction of all geometry on each node has shown to limit the size of the problem that can be computed [17], we will show that the multi-level mechanisms in Uintah allows representing a portion of the domain at a coarser resolution, thus lowering the memory usage and message volume, ultimately scaling to over 256K CPU cores. The hybrid memory approach of Uintah also helps as only one copy of geometry and radiative properties is needed per multi-core node [30]. RMCRT will be invoked largely on coarser mesh levels and the CFD calculation will be performed on the highest resolved mesh.

3.2 Uintah RMCRT Approaches

Within the Uintah RMCRT module there are numerous approaches, each designed for a specific use case, and range from a single-level method to a full adaptive mesh refinement using an arbitrary number of grid levels with varying refinement ratios. Our study focuses on the multi-level mesh refinement approach and its scalability to large core counts. CPU Scaling results for this approach are shown in Section 5.

Single-Level RMCRT: The single level RMCRT approach was initially implemented as a proof-of-concept to begin comparisons against the legacy DOM solver within the Uintah ARCHES component. This approach focused on the benchmark problem described by Burns and Christen in [9]. In this approach, the quantity of interest, the divergence of the heat flux, ∇q is calculated for every cell in the computational domain. The entire domain is replicated on every node (with all-to-all communication) for the following quantities; κ, the absorption coefficient, a property of the medium the ray is traveling through, σT^4, a physical constant $\sigma \cdot$ temperature field, T^4 and, $cellType$, a property of each computational cell in the domain to determine if along a given path, a ray will reflect or stop on a given computational cell. These three properties are represented by 1 double, 1 double and 1 integer value respectively.

For N_{total} mesh cells, the amount of data communicated is $\mathcal{O}(N^2)$. While accurate and effective at lower core counts, the volume of communication in this case overwhelms the system for large problems in our experience. Calculations on domains up to 512^3 cells are possible on machines with at least 2GB RAM per core and only when using Uintah's multi-threaded runtime system, described in Section 2. Strong scaling breakdown for the single-level approach occurs around 8-10K CPU cores for a 384^3 domain. Currently, Uintah has a production-grade GPU implementation of this single-level approach that delivers a 4-6X speedup [1] in mean time per timestep for this benchmark with a domain size of 128^3 cells. The work done to achieve accelerator task scheduling and execution is detailed in [15]. Initial scalability and accuracy studies of the single-level RMCRT algorithm are also shown in [17] which examines the accuracy of the computed divergence of the heat flux as compared to published data and reveals expected Monte-Carlo convergence.

Multi-level Adaptive Mesh Refinement: In this adaptive meshing approach, a fine mesh is used locally and only coarser representations of the entire domain are replicated on every node (with all-to-all communication) for the radiative properties, $T,\ \kappa\ cellType$. The fine level consists of a collection of patches where each patch is

[1] 1-NVIDIA K20 GPU vs. 16-Intel Xeon E5-2660 CPU cores @2.20GHz

considered a region of interest and individually processed using a local fine mesh and underlying global coarse mesh data. Figure 2 illustrates a three-level mesh coarsening scheme and how a ray might traverse this multi-level domain. Surrounding a patch is a halo region which effectively increases the size (at the finest resolution) of each patch in each direction, x, y and z. This distance is user specified. An arbitrary number of successively coarser levels (received by each node during the all-to-all communication phase) reside beneath the fine level for the rays to travel across once they have left the fine level. Each ray first traverses a fine level patch until it moves beyond the boundary and surrounding halo of this fine-level patch. At this point, the ray moves to a coarser level. Once outside this coarse level, the ray moves again to a coarser level. The rays move from level to level, similar to stair stepping, until the coarsest level is reached. Once on the coarsest level, a ray cannot move to a finer level. The key goal of this approach is to achieve a reduction in both communication and computation costs as well as memory usage. This approach is fundamental to our target problem, the 350MWe boiler predictive case where the entire computational domain needs to be resolved to adequately model the radiative heat flux.

Initial scalability results on a two-level methane jet problem are shown in [31]. This problem was run on the the DOE Titan, Mira and NSF Stampede systems with 10 rays per cell, two grid levels, a refinement ratio of four and a problem size of 256^3 cells on the highest resolved mesh. These results provided an excellent starting point by showing scaling to 16K CPU cores. Scaling results beyond 16K cores at the time was not possible due to algorithmic issues, which were ultimately resolved in this work and are detailed in Section 5.

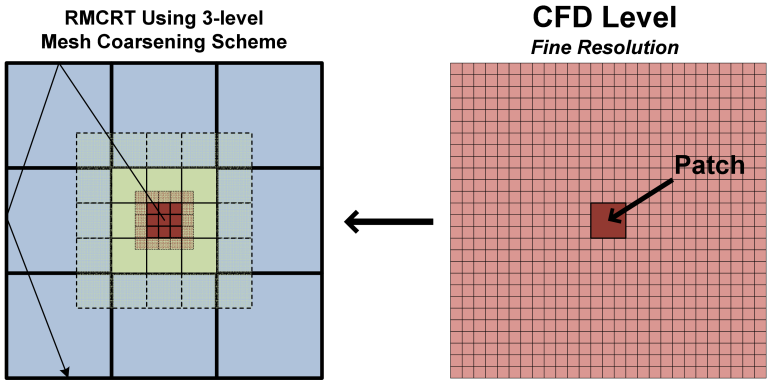

Fig. 2. RMCRT - 2D diagram of three-level mesh refinement scheme, illustrating how a ray from a fine-level patch (right) might be traced across a coarsened domain (left).

4 Complexity Model

In this section we generalize the somewhat simplistic analysis given in [31] (as suggested there) of the two-level scheme of [17] to a detailed discussion of both the computational and communications costs of a multiple mesh level approach. Initially our approach involves replicating the geometry of the target problem and constructing an adaptive mesh for the radiation calculations. The adaptive mesh used by the radiation calculation may be constructed directly from the efficient mesh data structure used to describe the whole mesh. This is a one-time procedure and so is not analyzed further here.

We suppose that on N_{nodes}^3 compute nodes there is a global fine mesh of n_{mesh}^3 cells in n_{patch}^3 mesh patches. Define $n_{local} = n_{mesh}/N_{nodes}$, and defines $n_{plocal} = n_{patch}/N_{nodes}$ so that each node has n_{local}^3 fine mesh cells in n_{plocal}^3 patches. In the ray tracing algorithm, each compute node then has to compute the heat fluxes, ∇q on its local mesh by ray tracing **and** to export the temperatures T and and the absorption coefficient κ on the original mesh to neighbouring "halo" nodes, or in a coarsened form (possibly at multiple levels) to other nodes. Finally the coarsest mesh representations are distributed to all the other nodes. The amount of information per cell transmitted is two doubles $\kappa, \sigma T^4$ and one integer, $cell\_type$.

To assess the complexity of RMCRT on a fixed fine mesh, computational experiments to measure the per cell and per ray cost of the RMCRT:CPU implementation were conducted on a single CPU with a single-level grid. Ray scattering and reflections were not included in these experiments. In both experiments the absorption coefficient was initialized according to the benchmark of Burns & Christen [9] with a uniform temperature field. A grid with a single patch and 1 MPI process was used, thus eliminated any communication costs. The grid resolution varied from $16^3, 32^3, 64^3$ to 128^3 cells with each cell using emitting 25 rays. The mean time per timestep (MTPTS) was computed using 7 timesteps. The code was instrumented to sum the number of cells traversed during the computation and it was shown that $MTPTS = (n_{mesh}^3)^{1.4}$. In the second experiment the number of grid cells in the domain was fixed at 41^3 and the number of rays, n_{ray} per cell varied. The MTPTS was computed over 47 timesteps. Here it was shown that the MTPTS varies linearly with the number of rays per cell. Based on these experiments, the cost for a single patch, without any communication, is approximately given by

$$T_{rmcrt}^{global} = C^* n_{rays} {n_{mesh}^3}^{4/3}, \tag{4}$$

where C^* is a constant. This result may be interpreted as saying that the rays from each of the n_{mesh}^3 cells travel a distance of n_{mesh} cells on average. In the case of a fine mesh on a node and a coarse representation of the rest of the mesh.

$$T_{rmcrt}^{local} = C^* n_{rays} \left[{n_{local}^3}^{4/3} + (n_{mesh} 2^{-m})^3)^{4/3} \right] \tag{5}$$

Where 2^m is the refinement ratio used to obtain the coarse mesh. It is possible to extend this analysis to more mesh refinement levels.

Communications Costs: The main step with regard to communication is to update the temperatures T and and the absorption coefficients κ every timestep. On a uniform fine

mesh this is done by each node sending out the values of these quantities to all the other compute nodes, and in a multiple mesh level approach this is done by each node sending out the values of these quantities on the coarse mesh and fine mesh values locally.

Fine Mesh Global Communications: Each node has to transmit $(N_{nodes}^3 - 1)$ messages of size $(n_{local})^3 \cdot 3$. This is currently done by a series of asynchronous sends but could be done with an MPI_Allgather. This has a complexity of $\alpha 3log(N_{nodes}) + \beta \frac{N_{nodes}^3 - 1}{N_{nodes}^3}(n_{local})^3$ for N_{nodes}^3 nodes with n_{local}^3 elements per mesh patch, where α is the latency and β is the transmission cost per element [40]. This result applies for both the recursive doubling and Bruck algorithms [40]. Other recursive doubling algorithms result in a complexity of $\alpha 3log(N_{nodes}) + \beta(N_{nodes}^3 - 1)(n_{local})^3$, so the cost may be dependent on the MPI implementation used.

Coarse Mesh All-to-All: In the case of using a coarse mesh in which the mesh is refined by a factor of 2^m in each dimension, each node has to transmit $(N_{nodes}^3 - 1)$ messages of size $(n_{local}2^{-m})^3 * 3$. Thus reducing the communications volume, but not the number of messages, by a factor of 2^{3m} overall

Multi-level Adaptive Mesh Refinement: This approach considers each fine level patch (individually) in the domain as a region of interest (ROI) and for each fine level patch, the highest resolved CFD mesh is used. Figure 2 illustrates one patch being such a region of interest. In the case of a region of interest consisting of P_{int} patches, the compute node must transmit the fine mesh information to all the local nodes close to the ROI. In this context let L_i be the nodes that are i levels of nodes removed from node containing the region of interest. There will then be 26 level-1 nodes and 98 level-2 nodes. Of course at the edges of a spatial simulation domain or in the case of a small domain of interest each node will only have to communicate fine mesh values of $\kappa, \sigma T^4$ to a fraction of the nodes. In this case let $L_{active}^{i,j}$ be the number of active nodes (halo-level nodes) at level j, where $j < N_{levels}$, active for the ith level of interest, where active nodes are the local halos from the fine mesh. Furthermore let the refinement factor be $\frac{1}{2^{m(i,j)}}$ active at this level. Then the fine mesh communication associated with this region of interest is given by

$$Com_{fhalo} = \sum_{j=1}^{N_{levels}} L_{active}^{i,j}(\alpha + \beta(n_{local}2^{-m(i,j)})^3 * 3) \tag{6}$$

This means that the ratio of communications to computations R_{atio} is now be given as:

$$R_{atio} = \frac{((N_{nodes}^3 - 1))(\alpha + \beta(n_{local}2^{-m})^3 * 3) + Com_{fhalo}}{T_{rmcrt}^{local}}, \tag{7}$$

where α and β are defined above and scaled by the cost of a FLOP. Overall this expression allows us to analyze the relationship between computation and communications.

Strong scaling of RMCRT does not change the overall volume of data communicated. Increasing the number of N_{nodes} by a factor of two simply reduces n_{local} by two. This does mean that the number of messages increases even with the total communications value being constant. Moving to MPI_Allgather also has the same issue but the factor of $3logN_{nodes}$ also increased by adding 3. Thus for enough rays n_{rays} with

enough refinement by a factor of 2^m on the coarse radiation mesh, the computation will likely dominate. A key challenge is that storage of $O(n_{mesh}2^{-m})^3)$ is required on a multicore node and that an AMR mesh representation is needed at very large core counts. Some aspects of this analysis are not dissimilar to earlier work by one of us on PDE solvers with global coarse mesh operations [11] using algorithms related to those of [8]. The results of this analysis will make it possible to prioritize subsequent serial and parallel performance tuning and and also perhaps to make projections regarding performance on forthcoming petascale and exascale architectures.

5 Scaling Studies

In this section, we show strong scalability results on the DOE Titan XK7 [2] system for the Burns and Christen [9] benchmark problem using the multi-level mesh refinement approach. We define strong scaling as a decrease in execution time when a fixed size problem is solved on more cores. This work focuses on using all CPU cores available on Titan. Subsequent work will focus on additionally using all of Titan's GPUs in addition to its CPUs, following our prototype work on a single mesh in [15], [31].

The scaling challenges faced in this work have only become apparent by running this challenging problem at such high core counts, stressing areas of infrastructure code in ways never before seen, specifically Uintah's task-graph compilation phase. With Uintah's directed acyclic graph (DAG)-based design [31], during an initial simulation timestep, the initial timestep of a restart, or when the grid layout or its partition changes, a new task graph needs be to created and compiled. Task-graph compilation is a complex operation with multiple phases, including creation and scheduling of tasks themselves on local and neighboring patches (for halo exchange), keeping a history of what these tasks require and compute, setting up connections between tasks (edges in the DAG), and finally assigning MPI message tags to dependencies.

As the RMCRT ray trace task requests ghost cells across the entire domain (a global halo) for ray marching, Uintah's task-graph compilation algorithm was overcompensating when constructing lists of neighboring patches for local halo exchange. The cost of this operation grew despite the number of patches per node remaining constant, resulting in task-graph compilation times of over four hours at 32K cores with 32,000 total patches. This necessitated extensive algorithmic improvements to the task-graph compilation algorithm. The original complexity of this operation was $\mathcal{O}(n_1 \cdot log(n_1) + n_2 \cdot log(n_2))$, and after optimization became $\mathcal{O}(n_1 \cdot log(n_1)) + \mathcal{O}\left(\frac{n_2}{p} \cdot log(n_2)\right)$, where n_1 is the number of patches on coarse level, n_2 is the number of patches on fine level, and p is the number of processor cores. This reduced the four hour task-graph compilation time to under one minute at 32K cores, thus making possible the results presented here.

[2] Titan is a Cray KX7 system located at Oak Ridge National Laboratory, where each node hosts a 16-core AMD Opteron 6274 processor running at 2.2 GHz, 32 GB DDR3 memory and 1 NVIDIA Tesla K20x GPU with 6 GB GDDR5 ECC memory. The entire machine offers 299,008 CPU cores and 18,688 GPUs (1 per node) and over 710 TB of RAM. Titan uses a Cray Gemini 3D Torus network, 1.4 μs latency, 20 GB/s peak injection bandwidth, and 52 GB/s peak memory bandwidth per node.

5.1 CPU Strong Scaling of Multi-level Adaptive Mesh Refinement, RMCRT

Our scaling study focuses on a two-level AMR problem based on benchmark described in [9], which exercises all of the main features of the AMR support within Uintah in addition to the radiation physics required by our target problem. A fine level halo region of four cells in each direction, x, y, z was used. The AMR grid consisted of two levels with a refinement ratio of four, the CFD mesh being four times more resolved than the radiation mesh. For three separate cases, the total number of cells on the highest resolved level was 128^3, 256^3 and 512^3 (green, red and blue lines respectively in Figure 3), with 100 rays per cell in each case. The total number of cells on the coarse level was 32^3, 64^3 and 128^3. In all cases, each compute core was assigned at least 1 fine mesh patch from the CFD level. Figure 3 shows excellent strong scaling characteristics for our prototype, two-level benchmark problem [9]. The eventual breakdown in scaling in each problem size is due to diminishing work, when a patch's MPI messages begins to exceed the cost of its computation, and hence the runtime system cannot overlap computation with communication. Figure 4 additionally shows the MPI wait associated with the global and local communications for this calculation along side the execution times for each of the three cases above.

Though the actual communication patterns for this problem are perhaps more complicated than our predictive model, due to MPI message combining and packing done by Uintah, both Table 1 and Figure 4 illustrate points made in Section 4, that global communications dominate and that the local communications do not have a significant impact, and for enough rays and enough refinement on the coarse radiation mesh, the computation does in fact dominate (Equation 7 of our predictive model). These results also show how the number of MPI messages grows with the number of cores. A key point to note, as is evidenced by the dominating global communications, is that the refinement ratio of four reduces the global communication phase by a factor of 64 (ignoring communications latency for large messages) over a fine mesh all-to-all. If this communications phase took 8-64 times as long it would destroy scalability.

Table 1. Total number of MPI messages and average number of messages per MPI rank for each problem size, 128^3, 256^3 and 512^3 (fine mesh)

Cores	256	1k	4k	8K	16K	32K	64K	128K	256k
128^3 total msgs	1001	5860	36304						
avg msgs/node	62.5	91.6	141.8						
256^3 total msgs		9843	52.1K	105.2K	212.1K	437.7K			
avg msgs/node		153.8	203.3	205.6	207.0	213.7			
512^3 total msgs				338.2K	673.8K	1.36M	2.71M	5.42M	10.88M
avg msgs/node				660.5	658.0	663.65	662.6	661.36	662.83

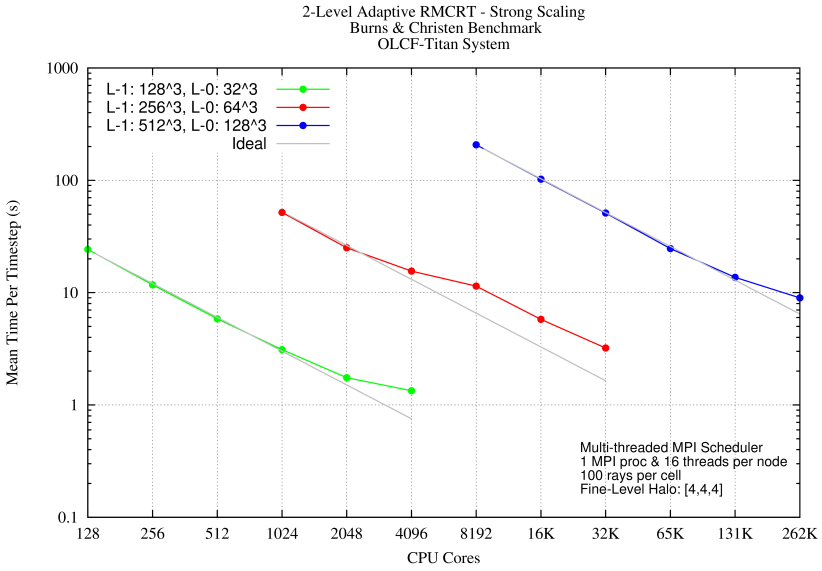

Fig. 3. Strong scaling of the two-level benchmark RMCRT problem on the DOE Titan system. L-1 (Level-1) is the fine, CFD mesh and L-0 (Level-0) is the coarse, radiation mesh.

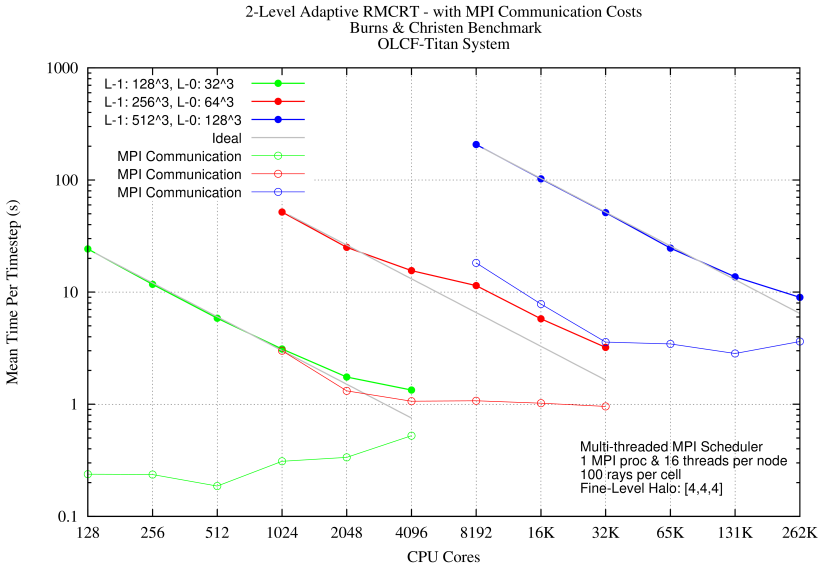

Fig. 4. Strong scaling with communication costs of the two-level benchmark. L-1 (Level-1) is the fine, CFD mesh and L-0 (Level-0) is the coarse, radiation mesh

5.2 Multi-Level Accuracy Considerations

To quantify the error associated with coarsening the radiative properties (temperature T, absorption coefficient κ, and cellType (boundary or flow cell)), an error analysis was performed using a simplified version of the adaptive-meshing approach described in Section 3. The grid consisted of a fine and coarse mesh and during a radiation timestep the quantities necessary to compute ∇q were interpolated to the coarser grid level. The radiation calculation was performed on the coarse level including all ray tracing. The ∇q was then compared using the computed solution of the Burns and Christen [9] benchmark problem at the prescribed 41 locations. 100 rays per cell were used in the computation and the refinement ratio between the coarse and fine grids was varied from 1 to 8. Figure 5 shows the L2 norm error of ∇q versus refinement ratio. This represents a worse case scenario, as only coarsened quantities are used in the computation.

In addressing the issue of accuracy, our approach will be to continue sending the coarse mesh in the *all-to-all* communication phase of each simulation timestep, but to recover the fine mesh values of the radiative properties through interpolation. This approach is well suited for GPU accelerators such as those on the Titan system, where FLOPS are inexpensive relative to the cost of data movement. Further compression of the coarse mesh information will also be investigated.

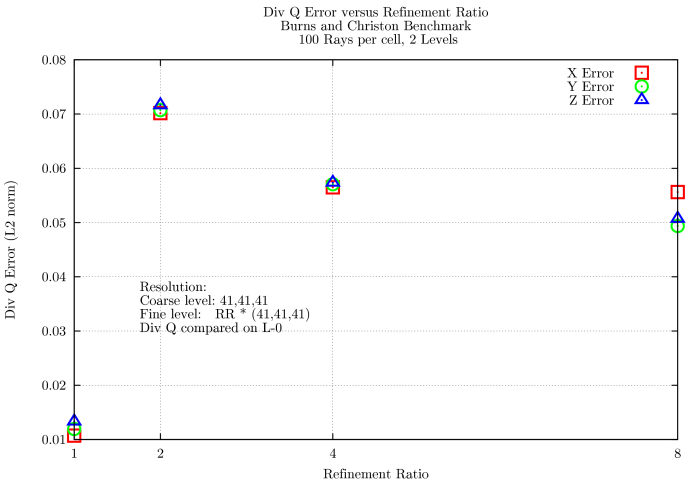

Fig. 5. L2 norm error of ∇q vs refinement ratio, The error in each direction (x,y,z) is shown.

6 Related Work

Industrial codes, such as Fluent, incorporate straightforward radiation models such as the discrete ordinates method in Fluent and Airpack [2], but scale to relatively small numbers of cores. At the national labs, many cutting edge codes are developed,

such as Fuego, CFDLIB, Kiva, and radiation codes ATTILA, DANTE, WEDGEHOG, PARTISN [23], [21], but many of these are not generally available, are unsupported, or are targeted at other problems such as neutron transport. There are also radiation transport problems that use CFD codes and AMR techniques [18], [34], however, a broad range of problems exist that require the concept of tracing rays or particles, such as the simulation of light transport and electromagnetic waves. In the case of adaptive mesh codes there are many such solvers. Specific examples of these codes are the Flash code [7], [35] based on adaptive oct-tree meshes and the physics AMR code Enzo, [33], [42]. These examples are perhaps closest as these combine AMR and radiation, but for very different problem classes. Most of these codes do not target the problems that Uintah has been designed for, with large deformations, complex geometries, high degrees of parallelism and now radiation.

7 Conclusions and Future Work

We have demonstrated that through leveraging the multi-level AMR infrastructure provided by the Uintah framework, we have developed a scalable approach to radiative heat transfer using reverse Monte Carlo ray tracing. The scaling and communication cost results shown in Section 5 provide a promising alternative to approaches to radiation modeling such as discrete ordinates. Using our cost model for communication and computation, we can predict how our approach to radiation modeling may scale and perform on current, emerging and future architectures.

The addition of a scalable, hierarchical radiation solver within Uintah will also benefit the general computational science engineering community in applications areas such as turbulent combustion simulation and other energy-related problem. The broader impact of our work may ultimately include algorithmic developments for related problems with pervasive all-to-all type communications in general, such as long-range electrostatics in molecular dynamics, and will be of importance to a broad class of users, developers, scientists and students for whom such problems are presently a bottleneck.

Our primary focus in moving beyond this study will be continued development of RMCRT capabilities explored here, to provide support for several additional energy-related problems within the scope of the Utah CCMSC. The relationship between accuracy, number of rays cast, refinement ratios between grid levels and extent of the fine-level halo region is being explored as part of the ongoing research goals. The calculations demonstrated in this work are ideal candidates for large-scale accelerator use, employing large numbers of rays for every cell in the computational domain. As such, implementation of a multi-level GPU:RMCRT module is now underway with the aim of using the whole of machines like Titan with accelerators.

Acknowledgments. This material is based upon work supported by the Department of Energy, National Nuclear Security Administration, under Award Number(s) DE-NA0002375, and by DOE ALCC award CMB109, "Large Scale Turbulent Clean Coal Combustion", for time on Titan. This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725. We would also like to thank all those involved with Uintah past and present, Isaac Hunsaker and Qingyu Meng in particular.

References

1. Amanatides, J., Woo, A.: A fast voxel traversal algorithm for ray tracing. In: Eurographics 87. pp. 3–10 (1987)
2. ANSYS, I.: Fluent Web Page (2014), http://www.ansys.com/Products/
3. Balsara, D.: Fast and accurate discrete ordinates methods for multidimensional radiative transfer. part i, basic methods. Journal of Quantitative Spectroscopy and Radiative Transfer 69(6), 671 – 707 (2001), http://www.sciencedirect.com/science/article/pii/S002240730000114X
4. Berzins, M.: Status of Release of the Uintah Computational Framework. Tech. Rep. UUSCI-2012-001, Scientific Computing and Imaging Institute (2012), http://www.sci.utah.edu/publications/SCITechReports/UUSCI-2012-001.pdf
5. Berzins, M., Luitjens, J., Meng, Q., Harman, T., Wight, C., Peterson, J.: Uintah - a scalable framework for hazard analysis. In: TG '10: Proc. of 2010 TeraGrid Conference. ACM, New York, NY, USA (2010)
6. Berzins, M., Meng, Q., Schmidt, J., Sutherland, J.: Dag-based software frameworks for pdes. In: Proceedings of HPSS 2011 (Europar, Bordeaux August, 2011) (2012)
7. B.Fryxell, K.Olson, P.Ricker, F.X.Timmes, M.Zingale, D.Q.Lamb, P.Macneice, R.Rosner, Rosner, J., Truran, J., H.Tufo: FLASH an adaptive mesh hydrodynamics code for modeling astrophysical thermonuclear flashes. The Astrophysical Journal Supplement Series 131, 273–334 (November 2000)
8. Brandt, A., Lubrecht, A.: Multilevel matrix multiplication and fast solution of integral equations. Journal of Computational Physics 90(2), 348–370 (1990), http://www.sciencedirect.com/science/article/pii/002199919090171V
9. Burns, S.P., Christen, M.A.: Spatial domain-based parallelism in large-scale, participating-media, radiative transport applications. Numerical Heat Transfer, Part B: Fundamentals 31(4), 401–421 (1997)
10. Falgout, R., Jones, J., Yang, U.: Numerical Solution of Partial Differential Equations on Parallel Computers, vol. UCRL-JRNL-205459, chap. The Design and Implementation of Hypre, a Library of Parallel High Performance Preconditioners, pp. 267–294. Springer-Verlag, 51 (2006)
11. Goodyer, C.E., Berzins, M.: Parallelization and scalability issues of a multilevel elastohydrodynamic lubrication solver. Concurrency and Computation: Practice and Experience 19(4), 369–396 (2007), http://dx.doi.org/10.1002/cpe.1103
12. Guilkey, J.E., Harman, T.B., Xia, A., Kashiwa, B.A., McMurtry, P.A.: An Eulerian-Lagrangian approach for large deformation fluid-structure interaction problems, part 1: Algorithm development. In: Fluid Structure Interaction II. WIT Press, Cadiz, Spain (2003)
13. Hapke, B.: Theory of Reflectance and Emittance Spectroscopy. Cambridge University Press (1993), http://dx.doi.org/10.1017/CBO9780511524998, cambridge Books Online
14. Howell, J.R.: The monte carlo in radiative heat transfer. Journal of Heat Transfer 120(3), 547–560 (1998)
15. Humphrey, A., Meng, Q., Berzins, M., Harman, T.: Radiation Modeling Using the Uintah Heterogeneous CPU/GPU Runtime System. In: Proceedings of the 1st Conference of the Extreme Science and Engineering Discovery Environment (XSEDE 2012). ACM (2012)
16. Hunsaker, I.: Parallel-distributed, Reverse Monte-Carlo Radiation in Coupled, Large Eddy Combustion Simulations. Ph.D. thesis, Dept. of Chemical Engineering, University of Utah (2013)
17. Hunsaker, I., Harman, T., Thornock, J., Smith, P.: Efficient Parallelization of RMCRT for Large Scale LES Combustion Simulations, paper AIAA-2011-3770. 41st AIAA Fluid Dynamics Conference and Exhibit, 2011

18. Jessee, J.P., Fiveland, W.A., Howell, L.H., Colella, P., Pember, R.B.: An adaptive mesh refinement algorithm for the radiative transport equation. Journal of Computational Physics 139(2), 380–398 (1998)
19. J.Spinti, Thornock, J., Eddings, E., Smith, P., Sarofim, A.: Heat transfer to objects in pool fires. In: Transport Phenomena in Fires. WIT Press, Southampton, U.K. (2008)
20. Kashiwa, B., Gaffney., E.: Design basis for cfdlib. Tech. Rep. LA-UR-03-1295, Los Alamos National Laboratory (2003)
21. Kerbyson, D.: A look at application performance sensitivity to the bandwidth and latency of infiniband networks. In: Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International. pp. 7 pp.– (April 2006)
22. Krishnamoorthy, G., Rawat, R., Smith, P.: Parallelization of the P-1 Radiation Model, numerical Heat Transfer, Part B: Fundamentals, 49 (1), 1-17, 2006.
23. Los Alamos National Security, L.: Los Alamos National Laboratory Transport Packages (2014), http://www.ccs.lanl.gov/CCS/CCS-4/codes.shtml
24. Luitjens, J., Berzins, M.: Improving the performance of Uintah: A large-scale adaptive meshing computational framework. In: Proc. of the 24th IEEE Int. Parallel and Distributed Processing Symposium (IPDPS10) (2010), http://www.sci.utah.edu/publications/luitjens10/Luitjens_ipdps2010.pdf
25. Luitjens, J., Berzins, M.: Scalable parallel regridding algorithms for block-structured adaptive mesh refinement. Concurrency and Computation: Practice and Experience 23(13), 1522–1537 (2011), http://dx.doi.org/10.1002/cpe.1719
26. Matsumoto, M., Nishimura, T.: Mersenne twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. ACM Trans. Model. Comput. Simul. 8(1), 3–30 (Jan 1998), http://doi.acm.org/10.1145/272991.272995
27. Meng, Q., Berzins, M., Schmidt, J.: Using Hybrid Parallelism to Improve Memory Use in the Uintah Framework. In: Proc. of the 2011 TeraGrid Conference (TG11). Salt Lake City, Utah (2011)
28. Meng, Q., Humphrey, A., Berzins, M.: The Uintah Framework: A Unified Heterogeneous Task Scheduling and Runtime System. In: Digital Proceedings of Supercomputing 12 - WOLFHPC Workshop. IEEE (2012)
29. Meng, Q., Luitjens, J., Berzins, M.: Dynamic task scheduling for the uintah framework. In: Proceedings of the 3rd IEEE Workshop on Many-Task Computing on Grids and Supercomputers (MTAGS10) (2010), http://www.sci.utah.edu/publications/meng10/Meng_TaskSchedulingUintah2010.pdf
30. Meng, Q., Berzins, M.: Scalable large-scale fluid-structure interaction solvers in the Uintah framework via hybrid task-based parallelism algorithms. Concurrency and Computation: Practice and Experience (2013), http://dx.doi.org/10.1002/cpe.3099
31. Meng, Q., Humphrey, A., Schmidt, J., Berzins, M.: Investigating applications portability with the uintah dag-based runtime system on petascale supercomputers. In: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis. pp. 96:1–96:12. SC '13, ACM, New York, NY, USA (2013), http://doi.acm.org/10.1145/2503210.2503250
32. Modest, M.F.: Backward Monte Carlo Simulations in Radiative Heat Transfer. Journal of Heat Transfer 125(1), 57–62 (2003), http://link.aip.org/link/JHTRAO/v125/i1/p57/s1&Agg=doi
33. O'Shea, B., Bryan, G., Bordner, J., Norman, M., Abel, T., Harkness, R., Kritsuk, A.: Introducing Enzo, an amr cosmology application. In: Adaptive Mesh Refinement - Theory and Applications. Lecture Notes in Computational Science and Engineering, vol. 41, pp. 341–350. Springer-Verlag, Berlin, Heidelberg (2005)
34. Pernice, M., Philip, B.: Solution of equilibrium radiation diffusion problems using implicit adaptive mesh refinement. SIAM J. Sci. Comput. 27(5), 1709–1726 (2005)

35. Rijkhorst, E.J., Plewa, T., Dubey, A., Mellema, G.: Hybrid characteristics: 3d radiative transfer for parallel adaptive mesh refinement hydrodynamics. Astronomy and Astrophysics 452(3), 907–920 (2006)
36. Schmidt, J., Berzins, M., Thornock, J., Saad, T., Sutherland, J.: Large Scale Parallel Solution of Incompressible Flow Problems using Uintah and hypre. In: Proceedings of CCGrid 2013. IEEE/ACM (2013)
37. Scientific Computing and Imaging Institute: Uintah Web Page (2015), http://www.uintah.utah.edu/
38. Sun, X.: Reverse Monte Carlo ray-tracing for radiative heat transfer in combustion systems. Ph.D. thesis, Dept. of Chemical Engineering, University of Utah (2009)
39. Sun, X., Smith, P.J.: A parametric case study in radiative heat transfer using the reverse monte-carlo ray-tracing with full-spectrum k-distribution method. Journal of Heat Transfer 132(2) (2010)
40. Thakur, R., Rabenseifner, R., Gropp, W.D.: Optimization of collective communication operations in mpich. International Journal of High Performance Computing Applications 19(1), 49–66 (2005)
41. Viswanath, K., Veljkovic, I., Plassmann, P.E.: Parallel load balancing heuristics for radiative heat transfer calculations. In: CSC. pp. 151–157 (2006)
42. Wise, J.H., Abel, T.: enzo+moray: radiation hydrodynamics adaptive mesh refinement simulations with adaptive ray tracing. Monthly Notices of the Royal Astronomical Society 414(4), 3458–3491 (2011), http://dx.doi.org/10.1111/j.1365-2966.2011.18646.x