JOURNAL OF THERMOPHYSICS AND HEAT TRANSFER
Vol. 17, No. 4, October–December 2003

Parallel Computing of an Integral Formulation of Transient Radiation Transport

X. Lu* and P.-f. Hsu†
Florida Institute of Technology, Melbourne, Florida 32901

Parallel computing of the transient radiative transfer process in participating media is studied with an integral equation model. Two numerical quadratures are used: the discrete rectangular volume (DRV) method and YIX method. The parallel versions of both methods are developed for one-dimensional and three-dimensional geometries, respectively. Both quadratures achieve good speedup in parallel performance. Because the integral equation model uses very small amount of memory, the parallel computing can take advantage of having each processor store the full spatial domain information without using the typical domain decomposition parallelism, which will be necessary in other solution methods, for example, discrete ordinates and finite volume methods, for large-scale simulations. The parallel computation is conducted by assigning a different portion of the quadrature to different compute node. In DRV method a variation of the spatial domain decomposition is used. In the case of YIX scheme, the angular quadrature is divided up according to the number of compute nodes. These parallel schemes minimize the communications overhead. Two new discrete ordinate sets are used in the YIX angular quadrature, and their parallel performances are discussed. One of the discrete ordinates sets, called a spherical ring set, is also suitable for use in the conventional discrete ordinates method.

Nomenclature

A	=	area
a	=	absorption coefficient
c	=	propagation speed of radiation transport in the medium
F	=	unknown functions
G	=	incident radiation, integrated intensity, or fluence rate
I	=	radiation intensity
I_0	=	radiation intensity at boundary $z = 0$
K	=	kernel functions
$\boldsymbol{k}$	=	unit vector in the z direction
M	=	number of volume or surface element for integration; number of angular directions in a hemisphere
m	=	index of angular direction
N	=	number of angular quadrature point or angular direction
$\boldsymbol{n}$	=	inward unit normal vector of boundary surface
q	=	radiative flux
r	=	radial coordinate
$\boldsymbol{r}$	=	position vector of a location (x, y, z) in space
s	=	geometric path length
$\boldsymbol{s}$	=	unit vector along a given direction
t	=	time
V	=	volume of the medium
W	=	weight of discrete ordinate or angular quadrature
x, y, z	=	rectangular coordinates
γ	=	unit vector along the line of sight
θ	=	polar angle
κ	=	extinction coefficient, $\kappa = a + \sigma$
ξ, η, ζ	=	direction cosines
σ	=	scattering coefficient
τ	=	optical thickness of the medium, κz_o
Φ	=	scattering phase function
φ	=	azimuthal or circumferential angle
Ω	=	solid angle
ω	=	scattering albedo

Subscripts

i, j	=	angular direction index
o	=	side length in coordinate direction
s	=	spherical ring surface
v	=	quantity at the volume element
w	=	quantity at the surface; wall clock time or total execution time

Superscript

$'$	=	dummy variables

Received 18 February 2003; revision received 22 May 2003; accepted for publication 22 May 2003.

*Graduate Student, Mechanical and Aerospace Engineering Department.
†Associate Professor, Mechanical and Aerospace Engineering Department; phsu@fit.edu. Senior Member AIAA.

Introduction

TRANSIENT radiative transfer within the participating medium has gained attention because of its applications in emerging new technologies, for example, optical tomography and remote sensing. In such situations the variation of radiative intensity within the length scale of the radiative transport per unit time is comparable to the variation over the length scale of the medium, or the timescale of the internal or external disturbance to the radiative field is comparable to the time scale of the radiation transport within the medium. Such situation occurs when an ultrashort light pulse, with temporal pulse width as short as several femtoseconds or less, propagates in a scattering and absorbing medium. Therefore, the transient effect of radiative transfer must be considered. The transient term of the radiative transfer equation (RTE) has to be retained to correctly simulate such processes.

Many practical problems have multidimensional geometry with nonhomogeneous radiative property distribution and possibly the radiation transport being coupled with other physical processes. It is essential to develop transient radiation models that can treat these situations realistically. Such models require significant computational resources. The parallel computing based on the Beowulf concept[1] has advanced to the stage that allows the high-performance simulations to be carried out at relatively low cost in comparison with the conventional supercomputer or proprietary massive parallel systems. This is possible because of the high-speed network, open source software, and the commodity nature of personal-computer components that are used in the Beowulf cluster systems. The cluster

can scale up to large number of processors without users changing their codes. The utilization of Beowulf cluster in solving transient radiation problems is very advantageous, as demonstrated in this study. However, the selection of computational algorithm that is suitable for parallel computing needs careful consideration. Some algorithms can lend to good speedup at a few dozen processors but quickly reach the speedup limit when processor number is further increased. It is therefore important to examine algorithms that can achieve good speedup not only at small number of processors, but also at several hundreds, even thousands of processors.

So far, several numerical models for transient radiation process have been developed. Each model provides a different degree of success and computational requirement.[2] As compared with the time-independent radiation transport process, the computational requirement of a transient model is far greater in terms of execution time and memory usage. The computational algorithm is also a bit more complicated because of the hyperbolic wave equation in conjunction with the scattering term. In the case of the integral formulations of the transport equation, the appropriate domain of influence (DOI) has to be considered.[2,3] For integro-differential formulations the radiation wave front has to be preserved and resolved with high-order difference schemes to avoid dispersion and diffusion errors.[4,5]

Our recent analytical studies produced accurate solutions with three different models: discrete ordinates, Monte Carlo, and integral equation methods.[2,3,6,7] The results indicated that the scattering phase function has a much stronger influence in the transient radiative transport than that in the steady-state process, even at a very large optical thickness (on the order of 100). The anisotropic scattering effect is most evident in the short time reflectance and transmittance signals. The long time signals of different phase function are less distinguishable from one another if the medium optical thickness is large and all other conditions remain the same. Therefore, models that work well in steady-state optically thick media might not do so in the transient analysis. This will require different and careful consideration in the modeling effort.

Parallelization of Monte Carlo radiative transfer codes is straightforward.[7] Each compute node can perform independently random sampling, and the results from all nodes are then tallied at a designated node. Nearly linear speedup has been achieved.[7,8] On the other hand, the parallelization strategy for deterministic models of the transport equation is far more complex. In a distributed memory system, for example, the Beowulf cluster used in this study, two different decomposition approaches or parallelization strategies have been used: spatial domain decomposition (SDD) and angular quadrature decomposition (AQD). As the radiative transport is a volumetric process, that is, the effect of one point will propagate to all of the points within the physical boundary, it is found that in the spatial domain decomposition the communication overhead between compute nodes is significant.[9–11] These results were based on the steady-state analysis. In fact, because the communications are pairwise (all computing nodes to all computing nodes) the communication time can scale proportional to the square of the number of processors. High communication overhead corresponds to a decrease in parallel efficiency. The total iteration number of the SDD scheme increases with the number of processors, and parallel efficiency is drastically reduced. This was observed in the earlier studies.[9,11–13] However, spatial decomposition is necessary if the radiation transport is coupled with a diffusive or convective process and for the reasons discussed next when AQD cannot be used alone.

Angular quadrature decomposition can achieve good parallel efficiency in steady-state analysis because less communication occurs among computing nodes. This has been reported on different parallel system architectures and with the integro-differential models of discrete ordinates, finite volume, and discrete transfer methods.[9–14] This type of parallelization will be possible only if the individual computing node memory can accommodate the whole spatial grids. In many large-scale problems grid size can reach $\mathcal{O}(10^8)$, and the number of variables on each grid (depends on the Sn discrete ordinates set or angular direction number) can be in $\mathcal{O}(10^2)$. In addition, the number of angular ordinates direction has to be greater than the number of processors. This might not be the case for a massive parallel system, where processor count can reach $\mathcal{O}(10^4)$ or higher and the conventional Sn directions can at most be in $\mathcal{O}(10^2)$. For these two reasons the AQD scheme is not practical in large-scale radiation transport simulations with the existing integro-differential models. A parallelization scheme of combining AQD and SDD reported to have good parallel efficiency was developed by Burns.[9] Although scattering was not considered, the scheme appears to be very promising for integral-differential models.

Saltier and Naraghi[15] used a proprietary massive parallel system [Connection Machine, CM-2 with 16384 ($=2^{14}$) processors] to study the parallel performance of the discrete exchange factor method. The method is directly applied using the Fortran-90 array processing statement. Each computer chip has very small amount of memory and relatively slow and simple 1-bit CPU. Sixteen processors were grouped into a chip. The communication between chips was linked by the hyper-cube structure. The details of the parallelization were not given, and the speedup and parallel efficiency were not available. The computational time was about linearly dependent on the spatial grid size. This indicates good parallel performance. Nevertheless, it is difficult to compare with the parallelizations already discussed, and it is also unclear whether the approach can be applied to the Beowulf cluster because of the architecture and software differences from CM-2. The unified matrix approach requires a large amount of memory, much larger than the integral formulation, but still smaller than those of the integro-differential formulations.

Most prior parallelization efforts are based on integro-differential treatment of the steady-state radiative transport equation. Little is known about the parallel performance of the integral equation model and of the transient RTE. To the authors' knowledge, none exists for the integral formulation of transient radiation transport. It is therefore unclear that integral formulations of radiation transport are viable option in the high-performance, parallel computing environment. An integral equation formulation for obtaining the higher moments of radiative intensity, for example, incident radiation and radiative heat flux, that can greatly reduce the memory requirement and allow the use of AQD parallelization in large-scale simulations will be very desirable. This study examined the parallel performance of an integral formulation developed by the authors using two different numerical quadratures: the discrete rectangular volume (DRV) method and the YIX method. In the latter method the performance of AQD on two different angular quadrature sets, that is, the symmetric slice (SS) and asymmetric spherical ring (ASR) sets, were compared. Both angular quadrature sets can generate arbitrarily large number of directions, which is needed when ray effects exist and allow AQD parallelism in massive parallel systems. The asymmetric angular quadrature set can also be used in the conventional discrete ordinate method.

As the medium optical thickness increases, the radiative transport becomes diffusive in the steady-state process. In such a situation the long-range pairwise communication is less important to achieve solution convergence. It was reported that at large optical thickness the parallel efficiency improved for SDD in the steady-state simulation.[10,11] The reason is that the local emission term becomes dominant as optical thickness increase. The cross grid boundary terms, on the other hand, become less important. The result is that less iteration across the spatial domain is needed to achieve convergence. This indicates SDD parallelism can perform well in the steady-state, optically thick media problem. However, in the case of transient transport simulation, the SDD will not perform well even with optically thick media. It is well known that the behavior of the light pulse propagation within a highly scattering, optically dense medium is not diffusive in the initial transient.[16,17] Only at a large time, long after the pulse leaves the medium, the incoherent, multiply scattered photons remain, and their behavior becomes diffusive. It is therefore important to recognize the physical differences between the steady and transient processes. As pointed out earlier, not only the consideration of numerical model for treat transient process is different, but also the parallelization strategy is different from the steady-state process.

Integral Form of the Transient Radiative Transfer Equation

The transient radiative transfer equation in an absorbing, nonemitting, and scattering medium in direction $\boldsymbol{s}$ (Fig. 1) can be written as[18]

$$\frac{\mathrm{d}I(\boldsymbol{r},\boldsymbol{s},t)}{\mathrm{d}s}=\frac{\mathrm{D}I(\boldsymbol{r},\boldsymbol{s},t)}{c\,\mathrm{D}t}=-\kappa(\boldsymbol{r})I(\boldsymbol{r},\boldsymbol{s},t)+\frac{\sigma(\boldsymbol{r})}{4\pi}\int_{\Omega'=4\pi}I(\boldsymbol{r},\boldsymbol{s}',t)\Phi(\boldsymbol{s}',\boldsymbol{s})\,\mathrm{d}\Omega' \tag{1}$$

where the D symbol represents the substantial derivative. Equation (1) is based on the Lagrangian viewpoint. It is found that the Lagrangian viewpoint can simplify the analysis of the time of flight of photon and allow the deduction of the domain of influence.[2] $\Phi(\boldsymbol{s}',\boldsymbol{s})$ is the scattering phase function. For isotropic scattering $\Phi(\boldsymbol{s}',\boldsymbol{s})=1$. Equation (1) reduces to

$$\frac{\mathrm{d}I(x,y,z,\boldsymbol{s},t)}{\mathrm{d}s}=-\kappa(x,y,z)I(x,y,z,\boldsymbol{s},t)+\frac{\sigma(x,y,z)}{4\pi}G(x,y,z,t) \tag{2}$$

where $G(x,y,z,t)$ is the incident radiation or integrated intensity defined as

$$G(x,y,z,t)=\int_{4\pi}I(x,y,z,\boldsymbol{s},t)\,\mathrm{d}\Omega \tag{3}$$

Following the procedure given in Ref. 2, Eq. (3) can be obtained as an integral equation of the form

$$G(x,y,z,t)=\int_{4\pi}I(x_w,y_w,0,\boldsymbol{s},t-s/c)e^{-\kappa s}\,\mathrm{d}\Omega+\frac{1}{4\pi}\int_{4\pi}\left[\int_0^s\sigma(x',y',z')G(x',y',z',t')e^{-\kappa(s-s')}\,\mathrm{d}s'\right]\mathrm{d}\Omega=\iint_{A(t)}\frac{I(x_w,y_w,0,\boldsymbol{s},t-s/c)e^{-\kappa s}}{s^2}\cos\theta\,\mathrm{d}A+\frac{1}{4\pi}\iiint_{V(t)}\frac{\sigma(x',y',z')G(x',y',z',t')e^{-\kappa(s-s')}}{(s-s')^2}\,\mathrm{d}V \tag{4}$$

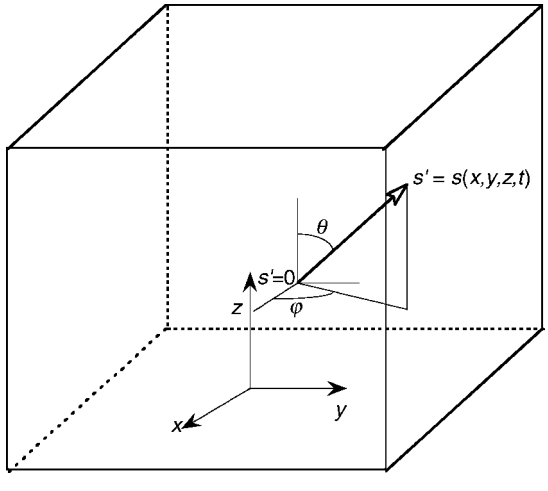

Fig. 1 Geometry and coordinate system.

Equation (4) is a Volterra integral equation of the second kind. For anisotropic scattering phase function additional volume integral terms that contain the higher moments of intensity will appear on the right-hand side. The integrated intensity $G(x,y,z,t)$ at location (x,y,z) and instant t depends on the entire time history from $t'=0$ to t during which the radiation field in the medium is established. The solution requires the knowledge of the integrated intensity in different location (x',y',z') and at different time t', $G(x',y',z',t')$, within the region of $z'>0$ and $z'<ct$-$(s-s')$ if the irradiation is along the z axis. The domain of influence or the domain of integration is a time- and position-dependent paraboloid, and the detailed derivation is given in Tan and Hsu.[2] If the external irradiation, that is, boundary condition of a temporal step function or a Gaussian pulse, is specified, then numerical quadrature can be carried out to find the solution of G. The numerical quadratures developed for solving the steady-state radiation transfer equation, that is, the Fredholm integral equation of the second kind, can be used to solve the Volterra integral equation, although some revisions are needed to consider the domain of integration.

Numerical Quadratures

Discrete Rectangular Volume Quadrature

For the computation of $G(x,y,z,t)$ in Eq. (4) by the DRV method, because $t'[=t-(s-s')/c]$ in the volume integration region is always smaller than t except the current location (x,y,z) and the $G(x',y',z',t')$ values at $t'<t$ are known from the preceding time steps, the $G(x,y,z,t)$ value can be obtained by discretizing the equation and the summation of the finite discretized terms. In the DRV method the volume and surface integrations on the right-hand side of Eq. (4) are constructed as

$$\iiint K(s,s')F(s')\,\mathrm{d}V(s')\approx\sum_{i=1}^{M_V}K(s,s_i')F(s_i')\Delta V(s_i') \tag{5a}$$

$$\iint K(s,s')F(s')\,\mathrm{d}A(s')\approx\sum_{i=1}^{M_W}K(s,s_i')F(s_i')\Delta A(s_i') \tag{5b}$$

M_v is the number of volume elements enclosed within the domain of influence. M_w is the number of surface elements enclosed within the circle, which is the intersection of the domain of influence and $z=0$ plane. Both numbers depend on the current time and position of interest because the domain of influence changes with time and position. To avoid singularity in the kernel function K, each volume is divided into subvolumes. In this one-dimensional geometry two subvolumes are used. All subvolumes in a given volume element have identical piecewise-constant radiative properties and function value of F. The DRV is a variation of the general quadrature method.[19]

In this study DRV method is applied to one-dimensional slab geometry that has a 100-volume mesh.

YIX Quadrature

For the YIX method the volume and surface integrals in Eq. (4) are first written in distance-angular integration forms using piecewise-constant interpolation in the volume and surface elements.[20]

$$\iiint K(s,s')F(s')\,\mathrm{d}V(s')=\int_{4\pi}\mathrm{d}\omega\times\int_0^{R(s,\gamma)}\exp\left[-\int_0^{s'}\kappa(s+\gamma t')\,\mathrm{d}t'\right]F(s+\gamma t)\,\mathrm{d}t\approx\sum_{i=1}^{N_w}W_i\int_0^{R(s,\gamma)}\exp\left[-\int_0^{s'}\kappa(s+\gamma_i t')\,\mathrm{d}t'\right]F(s+\gamma_i t)\,\mathrm{d}t \tag{6a}$$

$$\iint K(s,s')F(s')\,\mathrm{d}A(s') = \int_{4\pi} \times \exp\left[-\int_0^t \kappa(s+\gamma t')\,\mathrm{d}t'\right]F(s+\gamma t)\,\mathrm{d}\omega$$

$$\approx \sum_{i=1}^{N_w} W_i \exp\left[-\int_0^t \kappa(s+\gamma_i t')\,\mathrm{d}t'\right]F(s+\gamma_i t) \qquad (6b)$$

N_W is the number of angular ordinates or directions from position s. $R(s,\gamma)$ is the line of sight distance from position s, along the γ direction to the nearest boundary point of the domain of influence. The determination of the angular directions and the corresponding weight is given in the next subsection. The distance integrations in Eqs. (6a) and (6b) use unevenly spaced integration points that are distributed along the angular directions. The distance integration accuracy is controlled by the given first integration point and is unaffected by the radiation property distribution. The computation steps are as follows: 1) provide an initial guess for $G(x', y', z', t')$ at the location (x, y, z) and instant time t on the right-hand side of Eq. (4); 2) calculate integrals on the right-hand side by the YIX quadrature because all of the G values in the volume integration region are known except the value at the location (x, y, z) and time instant t; 3) check to see whether the difference between the newly calculated G value on the left-hand side of Eq. (4) and the preceding value falls within the given convergence criterion, and if not, repeat the step 2 with the updated G values; 4) with converged G functions in current time step, move forward to the next time step and repeat step 1) until all time steps are calculated. Unlike the DRV method the YIX method requires iteration at each time step and takes longer computational time. The iteration is needed as the nearby integration points are most likely fall within the same volume element of (x, y, z), and therefore, the G functions at these points are unknown at the current time. By revising the code to identify all of these points, much faster convergence can be achieved. It is not implemented in this study, however.

YIX method is applied to a three-dimensional cubic medium. The cube is divided into a $17 \times 17 \times 17$ mesh, that is, $N_{xo} = N_{yo} = N_{zo} = 17$, equal volume elements ($\Delta x = x_o/N_{xo}$, $\Delta y = y_o/N_{yo}$, and $\Delta z = z_o/N_{zo}$), and each volume element has $\Delta V = \Delta x \Delta y \Delta z$, where Δx, Δy, and Δz are element sizes along x, y, and z coordinates, respectively. The time interval Δt, which the radiation takes to travel through the element thickness Δz, is $\Delta z/c$, where c stands for the propagation speed of the radiation in the medium. At one time interval Δt the radiation moves the distance of $\Delta z (= c\Delta t)$. The total time step used in all calculations is $N_{to} = 72$. Hence, the z coordinate of the element is $z = (N_z - 0.5)\Delta z$ with $N_z = 1, 2, \ldots, N_{zo}$ and the local time is $t = N_t \Delta t$ with N_t being the number of time step.

Angular Quadrature or Discrete Ordinate Sets

Two discrete ordinates sets were developed in the angular quadrature of the YIX method. These are the asymmetric spherical ring and the symmetric slice sets. It should be emphasized that the main purpose of these sets is to generate arbitrarily large number of angular directions. Although the reflection and rotation symmetry might not be preserved,[21] for a very large number $\mathcal{O}(10^4)$ of angular directions the symmetry requirement can be approximately satisfied. The details of generating the discrete ordinates are given here:

Asymmetric spherical ring angular quadrature generation (Fig. 2). 1) Generate an elementary basis area A_0 on a unit sphere. From the given input N, set an initial value for $A_0 = (\pi/2N)^2$, where N is the number of polar angle division of $\pi/2$ of the upper hemisphere. Let this area be the small spherical cap area, and find the corresponding polar angle spans the cap, that is, using the cap surface area, formula: $A_0 = 2\pi(1 - \cos\theta)$.

2) At $\theta = 0$, set $m = 1$. In this direction, that is, the z axis, $\zeta(1) = 1$, $\xi(1) = \eta(1) = 0$, and $W(1) = A_0$.

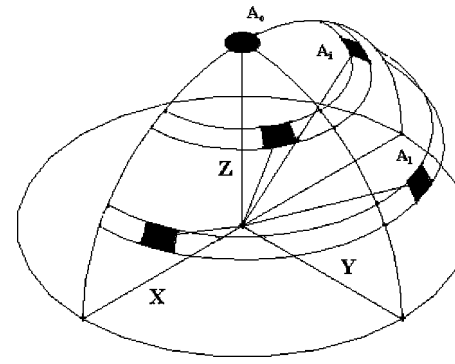

Fig. 2 Asymmetrical-spherical-ring angular quadrature. On the unit sphere the associated solid angle for each direction is approximately equal, that is, $A_o \approx A_i \approx A_l$. On each spherical ring the solid angle or area is exactly the same.

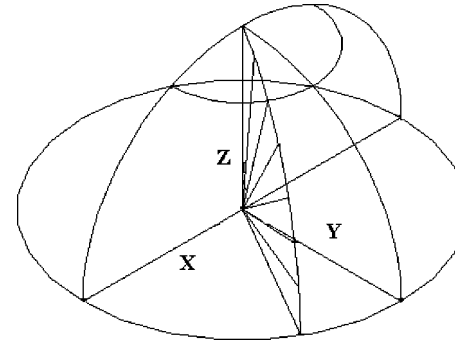

Fig. 3 Symmetrical-slice angular quadrature. Each slice of angular directions is assigned to one compute node.

3) Divide the polar angle from θ to $\pi/2$ by N to obtain the new polar angles $\theta_i = \theta + (i - \frac{1}{2})(\pi/2 - \theta)/N$. For i from 1 to N, find an integer number of A_0 that can be fit in the spherical ring strip, that is, $n_i = \mathrm{INT}(A_{si}/A_0)$, with A_s is the surface area of the spherical ring that spans a finite polar angle. Obtain the new basis area (or weight) for each direction in the same ring by using $A_i = A_{si}/n_i$.

4) In each spherical ring of A_{si}, divide the azimuthal angle from 0 to 2π by n_i to obtain successive azimuthal angles $\phi_{i,j}$. From the first angular direction the direction cosines $\xi(m)$, $\eta(m)$, $\zeta(m)$ and weights $W(m)$ are generated while incrementing $m = m + 1$.

5) After the upper-hemisphere angular directions are generated, the lower-hemisphere directions can be mirror mapped by setting $\zeta(M + m) = \zeta(m)$ while keeping x- and y-component direction cosines and the weight the same. The total number of angular directions is $2M$ [$\sim 4\pi/(\pi/2N)^2 = 16N^2/\pi$].

Symmetric slice angular quadrature generation (Fig. 3). 1) On a unit sphere divide the 2π azimuthal angle by p (an input, which is same as the processor number p). At each azimuthal angle ϕ_j imagine a knife cuts through the sphere to form a slice, as shown in Fig. 3.

2) At $\theta = 0$, set $m = 1$. The first direction will be along the $+z$ axis, $\zeta(1) = 1$, $\xi(1) = \eta(1) = 0$, and $W(1) = A_0$. The last direction will be the opposite of the first one.

3) At each azimuthal angle position ϕ_j or along the cut, divide the polar angle from θ to $\pi/2$ by N (an input) to obtain new polar angles $\theta_i = \theta + (i - \frac{1}{2})(\pi/2 - \theta)/N$.

4) The direction cosines and weight (the corresponding spherical surface patch area) can be determined from (θ_i, ϕ_j). The lower-hemisphere directions are again the mirror images of the upper-hemisphere ones. The total number of directions is $2M = 2pN + 2$.

In this angular quadrature set the axis of symmetry of all angular directions aligns with the direction of external irradiation (z axis). In comparison, ASR set has nearly uniform weight in all directions, but the SS set weight will gradually increase from a very small

value near the axis of symmetry direction to a larger value at the orthogonal direction. Therefore, the directions concentrate along the z axis. Obviously, the closely spaced directions at smaller θ_i will help little to gain solution accuracy, unless there is ray effect along the z axis direction. However, the approach greatly simplifies the quadrature set generation and allows very good load balancing, which is discussed in the next section. The SS set is identical to the polar/azimuthal angular discretization used in the finite volume method.

Parallel Computing Algorithms

The parallel computer architecture used in this study is simply a collection of computers with commodity processors and parts working together to solve a problem efficiently and in less time and less cost. The coding is based on the single-program multiple-data model and using message-passing-interface (MPI) library. MPI is one of the two commonly used standard libraries to achieve parallelization. The system consists of one root or head node and many slave or compute nodes. The communication among nodes, in this case, is through a private, channel-bonded FastEthernet. A 48-node IBM PC-based Linux cluster was installed and used in this study. This system can be extended to 96 processors with dual processors in each node. Currently, except the head node (a dual-processor IBM Netfinity 4500R), only one processor is installed in each of the compute nodes. Each compute node is an IBMX330X-Series Pentium III 866-MHz processor with 512 MB SDRAM. The total system memory space is 24 GB. The interconnect between nodes is channel-bonded FastEthernet, that is, double bandwidth of 200 MB/s. The job queuing is provided by the portable batch system OpenPBS, which is built on top of the Maui job scheduler. The serial code was modified to run on the cluster by implementing MPI. Detailed hardware and software configuration information is given in the website (http://olin.fit.edu/beowulf/). All parallel results are identical to the serial solutions.

Another cluster that the authors used, but with limited access, is a Compaq AlphaServer SC45, which is configured with 128 nodes connected by a Quadrics high-speed interconnect switch (see data online at http://www.quadrics.com/). Each node contains four 1-GHz Alpha EV 68 processors and 4 GB of SDRAM. Other than the processor, the notable difference from the Pentium cluster is the Quadrics network. The network is rated at 340 MB/s (or 2.72 Gb/s) bandwidth and 5-μs latency for MPI messages. In comparison, the FastEthernet used in the Pentium cluster has about 90-μs latency. It is found in this study that high-speed network has a significant impact on the parallel performance of the numerical quadratures. The Alpha cluster was mainly used to determine the communication performance of the algorithms. Most calculations were carried out on the Pentium cluster.

Efficiency and speedup for parallel algorithms can be measured in several ways.[22] In the following are standard definitions for parallel computing performance measurement: speedup S_p, efficiency E_p, and communication penalty C_p:

$$S_p = \frac{\text{wall (or execution) time using a single processor}}{\text{wall time using } p \text{ processors}} \tag{7}$$

$$E_p = \frac{S_p}{p} \tag{8}$$

$$C_p = \frac{\text{wall time using } p \text{ processors}}{\text{CPU time using } p \text{ processors}} \tag{9}$$

In the preceding equations the wall time equals the sum of CPU time, communication time, and idle time. With a well-balanced computation load among all compute nodes, the idle time in each node is negligible. In such a case one can determine the difference between wall time and CPU time to be the communication time and calculate C_p. This is shown in the parallelization of DRV method that follows. If the compute load is not very balanced, then the C_p measurement will be meaningless unless the idle time of each compute node can be determined.

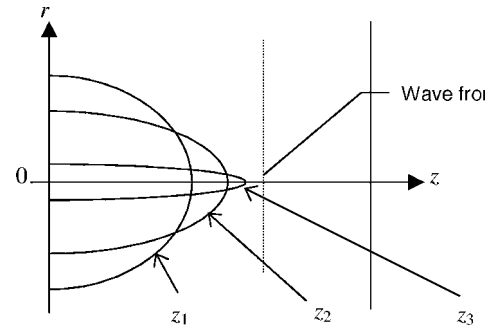

Fig. 4 One-dimensional geometry and three domains of influence shown as parabolic curves that are corresponding to the positions: $z_1 < z_2 < z_3$. The three domains are shown at the same time instant.

In some literature CPU time is used in Eq. (7). However, in order to have a precise representation of how long to complete a computing task with a given algorithm, the execution time should be used. An inefficient parallel algorithm might have elongated the communication time and thereby slows down the computing task, even though the S_p based on CPU time can still indicate good speedup.

Parallelization of DRV Method

There are a couple of ways to parallelize the right-hand sides of Eqs. (5a) and (5b). One can always use spatial domain decomposition of the integrations. However, in this case it is not an efficient or practical scheme because of the time- and position-dependent DOI. Figure 4 shows, at a certain time step t, the integration domains at three different positions: z_1, z_2, and z_3. Each curve represents a paraboloid. Apparently, it is not efficient if one chooses to decompose the z-domain, as this would have led to using either uneven z-domain decomposition to achieve even workload or uniform z-direction decomposition to have uneven workload. Neither strategy is favorable.

Considering the nature of DOI, the parallelization strategy used in the DRV method is then developed as the following: each processor has exactly the same information needed for the integration of the right-hand side of Eq. (4). During the integration, the time-marching step is at the outer loop. Because the integration of the current time step depends on the solutions at the preceding time steps, therefore time-loop parallelization is impossible. The z-direction and r-direction integrations are nested inside the time loop, with r direction as the innermost loop. It was determined the efficient way for parallelization is to decompose the r domain. As each processor contains all of the independent variables information, that is, $G(z, t)$, there is no need for message passing in the integration of Eq. (5) at each time step. At the end of each time step, the segmented r-integration results are shared among all compute nodes. Each node then contains the updated $G(z, t)$, and the same integration scheme was repeated for the next time step. This can result in redundant accuracy when DOI is small, for example, the one related to position z_3 in Fig. 4. However, the advantage is that identical compute load for each node can be achieved, and only minimum amount of communication is needed. This allows very good parallel performance. The parallelization scheme is not an SDD, as the physical domain in z direction is not decomposed. The r-direction decomposition can be considered a special variation of SDD.

Parallelization of YIX Method

Spatial domain decomposition is neither appropriate nor necessary for YIX quadrature as this method uses very small amount of system memory. Therefore, $G(x, y, z, t)$ information is stored in all compute nodes. For the numerical quadrature of the right-hand sides of Eqs. (6a) and (6b), the logical treatment is to divide up the N_W angular directions by the number of compute nodes. The angular quadrature decomposition scheme was used on two different angular quadrature sets: the ASR set (Fig. 2) and the SS set (Fig. 3). Three-dimensional geometry was used to illustrate the parallelization strategy and performance. In the case of collimated laser pulse

irradiation into the medium, the pulse direction is along the z-axis direction as shown in Fig. 1.

Asymmetric Spherical Ring Angular Quadrature Decomposition

The first angular direction points toward the positive z-axis direction. The angular directions in the next spherical rings are numbered sequentially. The direction numbering proceeds to the next ring, so on and so forth. The last angular direction points toward the negative z axis. Based on the number of compute nodes to be used, the angular directions are allocated accordingly. For examples, if 222 angular directions are decomposed into 8 compute nodes then in the first 7 nodes each will have 28 directions, and the last compute node has the last 26 directions.

Symmetric Slice Angular Quadrature Decomposition

In this case because the number of "slices" (Fig. 3) is divisible by the number of compute nodes, the AQD can be easily achieved by allocating one or more slices of angular directions to each node. Each compute node has exactly the same number of angular directions. This ensures balanced load with symmetric geometries.

In the cases that arbitrary physical geometry is of interest, then ASR decomposition is preferred for its tendency to reach more balanced computational load. This will be discussed in the next section.

Results and Discussion

The coordinate system is shown in Fig. 1. The external irradiation is applied at $z = 0$, the bottom surface. In the case of one-dimensional slab geometry (Fig. 4), the left boundary is at $z = 0$, and the right boundary is at $z = z_o$. The problems to be solved are the one-dimensional case A3 given in Hsu[23] and three-dimensional nonhomogeneous case 3 in Tan and Hsu.[3] The detailed problem descriptions and serial code solutions can be found therein. Because this study is about parallel computing performance, the solutions are not repeated here. Unless noted, all reported results were obtained from the Pentium cluster.

DRV Method in One-Dimensional Geometry

The parallel performance of DRV method is given in Table 1. The CPU time reduces nearly by half when the processor number doubles. Besides, the CPU time reported from each processor is nearly identical, which indicates balanced computing load. As the processor number increases from 16 to 32, the wall time does not decrease as does the CPU time; on the contrary, the wall time increases slightly. This can be explained by the large increase in the communication time (=wall time – CPU time) as given in Table 1 and shown in Fig. 5. The corresponding effect is also observed in S_p and E_p. C_p data indicate a large communication penalty when p increases to 32. Apparently, for the DRV method a communication bottleneck exists when more than 16 processors are used in the Pentium cluster, as indicated by the sudden increase of C_p from 16 to 32 processors. Beyond 16 processors, the speedup does not increase, and parallel efficiency drops to a very low value.

Figure 5 clearly depicts the increase of communication time when $p > 16$. The bottleneck is produced by the simultaneous message passing at the end of each time step when every compute node is sending the partial integrated result to be shared with all other compute nodes. Because of the even computing load, the initiation of message passing almost occurs at the same time. The situation results in the communication contention.[22] The problem can be remedied by using very low latency, high-speed inter-node network. The same parallel code was also run on the Alpha cluster with Quadrics network. The results on the Alpha cluster show negligible difference between wall time and CPU time, that is, the communication time is very small. Therefore, for the same parallel algorithm the parallel performance measurements are far superior to those based on the Pentium cluster.

Table 1 Parallel performance of one-dimensional DRV method

p	CPU time[a]	Wall time	Communication time	S_p	E_p	C_p
1	282.02	282.59	N/A	1	1	1
2	135.50	140.60	5.10	2.01	1.00	1.04
4	69.22	77.90	8.68	3.63	0.91	1.13
8	36.35	51.05	14.70	5.54	0.69	1.40
16	20.17	39.91	19.74	7.08	0.44	1.98
32	13.14	40.27	27.13	7.02	0.22	3.06

[a]All reported times are given in seconds.

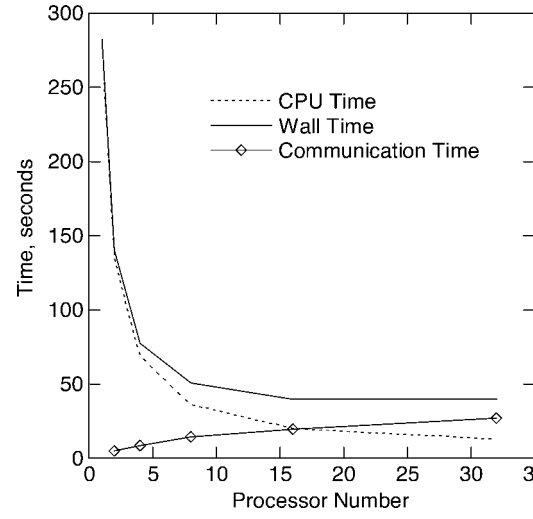

Fig. 5 Parallel performance of DRV quadrature decomposition.

The results demonstrate that DRV with the special integration domain decomposition, not the typical SDD, can led to very good parallel performance if a low latency network interface is used in the cluster. It was determined that parallelization of an existing three-dimensional DRV code[3] would have led to similar parallel performance as in the one-dimensional case. Therefore, the parallelization of DRV method in three-dimensional geometry was not conducted.

YIX Method in Three-Dimensional Geometry

Besides the full-domain calculation, because of the geometrical symmetry in the cubical medium, one-eighth of the physical domain was also used in the computation to save the computation time. The computational times for full- and partial-domain computation were recorded. Although partial-domain computation did save time, the parallel performance, however, was slightly inferior to or worse than that of the full-domain computation, depending on the type of angular quadrature used.

Figure 6 shows the full- and partial-domain wall and averaged CPU times with different processor number. The asymmetric spherical ring angular quadrature decomposition is used. The CPU time is the averaged time from all processors. The corresponding parallel efficiencies are also plotted. As the processor number is greater than 16, the E_p drops below 50%. The time difference between wall time and CPU time increases very quickly from two to eight processors and then stays relatively constant up to $p = 32$. The time difference includes the communication time and idle times from compute nodes. Although the E_p based on the wall time is not ideal, it is noted that the averaged CPU time does reduce nearly by half as the processor number increases by a factor of 2.

In Fig. 7 the same problem was solved with symmetric-slice angular quadrature decomposition. The difference between the wall time and averaged CPU time is smaller than that in the asymmetric-spherical-ring angular quadrature case. Most notably, although the E_p also decreases with increasing processor number the rate of decrease is slower. For example, at $p = 16$ the E_p is about 61% for both full and partial domain SS decomposition. But for the ASR case, the E_p is only about 53%.

In the same SS case the $\frac{1}{8}$-domain computation E_p decreases at a slightly higher rate than that of the full-domain computation. The

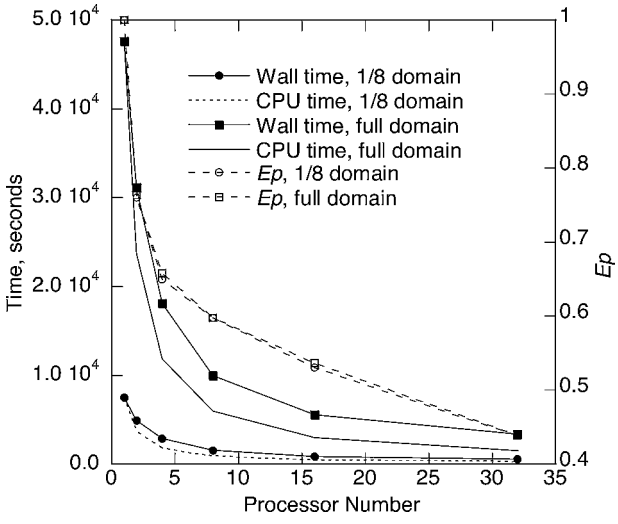

Fig. 6 Parallel performance of asymmetric-spherical-ring angular quadrature decomposition.

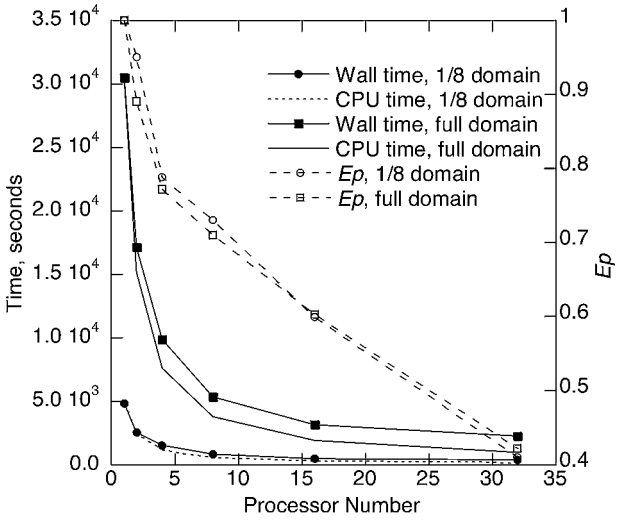

Fig. 7 Parallel performance of symmetric-slice angular quadrature decomposition.

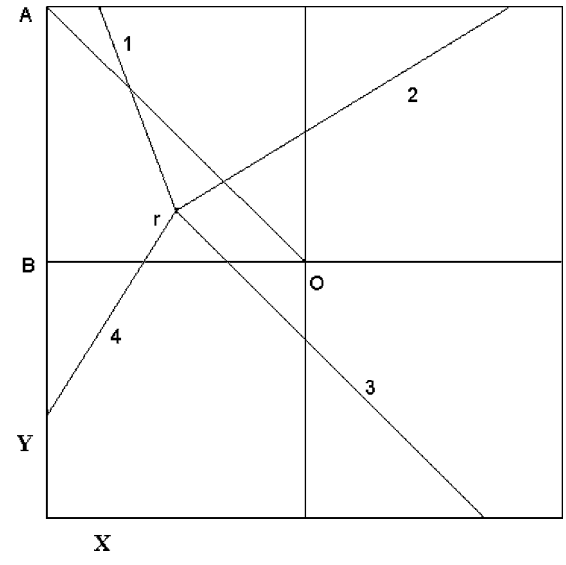

Fig. 8 Distribution of symmetric-slice angular directions in $\frac{1}{8}$-partial domain computation.

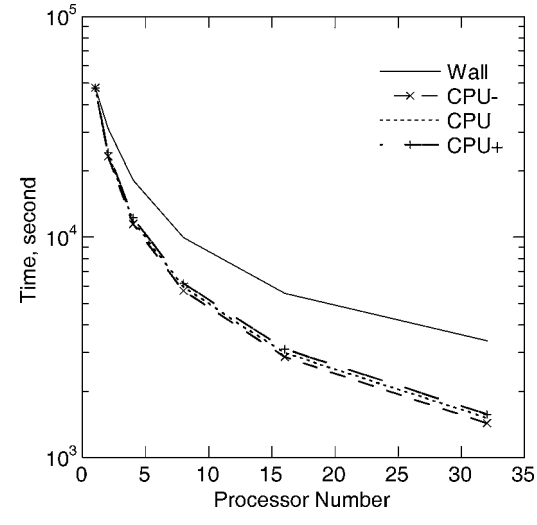

Fig. 9 Asymmetric spherical-ring angular quadrature decomposition with CPU time distribution.

difference can be explained with the angular direction distribution and the $R(s, \gamma)$ in each direction as shown in Fig. 8, which is a view in the x–y plane. Consider position r inside the $\frac{1}{8}$ domain, shown as ΔABO; the symmetric slices are uniformly distributed in four quadrants. Four lines represent the slices: 1, 2, 3, and 4. The angular directions in slice 3 in the lower-right quadrant always have the longest $R(s, \gamma)$, and angular directions in slice 1 in the upper-left quadrant have the shortest $R(s, \gamma)$. The line integration along the angular directions in slice 3 will always take a longer time than the slices in the other three quadrants. This leads to unbalanced computing load. In fact, at a large processor number, the standard deviation of the CPU time is about 20% of the averaged CPU time. This indicates uneven computing load. In comparison, for the same SS decomposition with full-domain computation the standard deviation is about 2.5% of the averaged CPU time. Therefore, although $\frac{1}{8}$-domain computation does save overall CPU time the parallel efficiency is worse than that of the full-domain computation.

In the case of ASR decomposition, the partial-domain computation maintains similar parallel efficiency as the full-domain computation. Again, this can be explained by the way the angular directions are being divided up among compute nodes in the ASR decomposition: each compute node takes the same number of angular directions, and these directions have no dominant orientation as in the case of SS quadrature. The standard deviation of CPU time is 3.5% of the averaged CPU time for full-domain computation and 4.3% for $\frac{1}{8}$-domain computation. Because of the same reason, the parallel efficiencies of ASR full- and partial-domain computation are about the same (Fig. 6). It is thus concluded that for full-domain computation SS decomposition achieves better load balancing. For partial-domain computation ASR decomposition maintains a better load balancing. The latter conclusion is also applied to solution involves with arbitrary geometry.

To understand the parallel efficiency difference between ASR and SS decompositions, the CPU time distributions of full-domain computation are plotted in Figs. 9 and 10. The CPU+ curve is the average CPU time adding one standard deviation of the CPU time distribution at the given processor number. Similarly, the CPU-curve is the average CPU time subtracting one standard deviation. For all processor numbers ASR decomposition clearly leads to wider variation of CPU time, that is, not very balanced computing load, compared with the SS decomposition. Thus, the parallel efficiency of ASR decomposition is not as good as the SS decomposition.

Unlike the one-dimensional DRV performance shown in Fig. 5, the message transmission contention is unlikely to occur as each node completes its calculation at a different time and therefore initiates transmission at different time accordingly. It was suspected that the large time difference between wall time and averaged CPU time

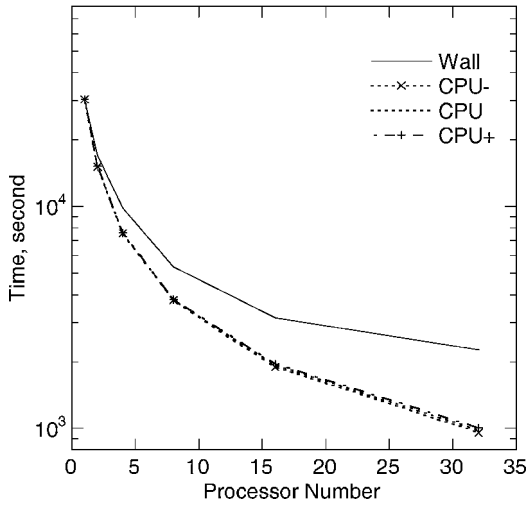

Fig. 10 Symmetric-slice angular quadrature decomposition with CPU time distribution.

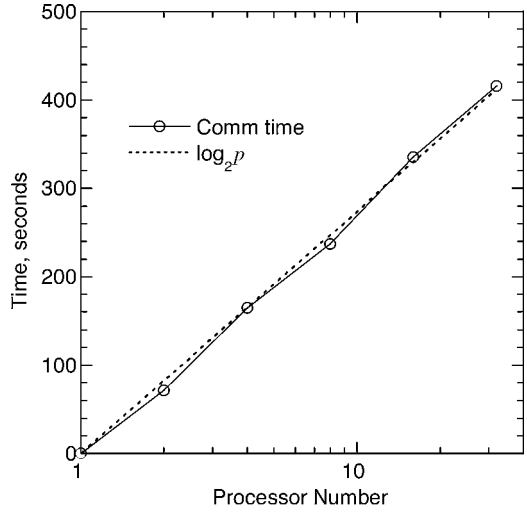

Fig. 11 Measured communication time follows the message propagation down a binary tree of compute nodes; p is the number of compute nodes or processors.

was caused by the network transmission. However, a closer examination of the actual network communication time (up to 416 s at 32 processors) reveals that the communication time alone will not account for the larger time difference (about 1280 s at 32 processors). Figure 11 shows that the accumulated communication time follows exactly the MPI ALLREDUCE call's transmission pattern, which is a binary tree-structured process.[22] Such a process has communication time proportional to the tree nodes in the form of $\mathrm{INT}(\log_2 P)$. The time difference was mainly caused by the idling compute nodes.

The average CPU time in both ASR and SS decompositions scales up very well with the processor number. Similar to the one-dimensional computation, the three-dimensional AQD algorithms were also run on the Alpha cluster, and the time difference between the wall time and CPU time was much smaller. The parallel efficiency is much better than that in the Pentium cluster. Both AQD algorithms are considered to be suitable for parallel computing.

Conclusions

Two parallel strategies on simulating transient radiative transport process were developed. One is based on the discrete rectangular volume method and uses integral domain decomposition. The other is based on the YIX method and uses angular quadrature decomposition. Their respective parallel performances are demonstrated and discussed. The decomposition of integration domain, not physical domain, results excellent load balancing in DRV method. The speedup increases up to 16 processors and then decreases because of the increasing communication contention. However, the DRV parallel algorithm achieved excellent speedup with a low latency internode network. For YIX method two discrete ordinate angular quadrature sets were developed: asymmetric spherical ring and symmetric slice. These sets can generate arbitrarily large number of angular directions. The use of large-order angular quadrature sets in conjunction with the integral equation model avoids the difficulties of applying angular quadrature decomposition parallelization with the conventional discrete ordinate sets in the large scale simulations. In a symmetric geometry asymmetric-spherical-ring angular quadrature decomposition can maintain consistent parallel efficiency and balanced load for both partial- and full-domain computations. On the other hand, the symmetric-slice angular quadrature decomposition achieves a higher parallel efficiency in full-domain computation because of better load balancing. However, the partial-domain computation does not have balanced load because of the distribution of the angular directions. In all calculations it is found that the low latency, high-speed interconnect within the cluster has a great impact on the parallel performance of the integral equation model used in this study.

Acknowledgments

This work is supported by Sandia National Laboratories with Contract AW-9963, and Shawn P. Burns is the Program Manager of this project. The parallel system is provided by a grant from National Science Foundation MRI Program Grant EIA-0079710, and Rita V. Rodriguez is the Program Director. The help from our colleague Gary Howell to run the codes on the Alpha cluster is much appreciated.

References

[1] Sterling, T., Becker, D. J., Savarese, D., Dorband, J. E., Ranawak, U. A., and Packer, C. V., "Beowulf: A Parallel Workstation for Scientific Computation," *Sagamore Computer Conference on Parallel Processing*, Pennsylvania State Univ. Press, University Park, PA, 1995.

[2] Tan, Z.-M., and Hsu, P.-f., "An Integral Formulation of Transient Radiative Transfer," *Journal of Heat Transfer*, Vol. 123, No. 3, 2001, pp. 466–475.

[3] Tan, Z.-M., and Hsu, P.-f., "Transient Radiative Transfer in Three-Dimensional Homogeneous and Nonhomogeneous Participating Media," *Journal of Quantative Spectroscopy and Radiative Transfer*, Vol. 73, No. 2–5, 2002, pp. 181–194.

[4] Sakami, M., Mitra, K., and Hsu, P.-f., "Transient Radiative Transfer in Anisotropically Scattering Media Using Monotonicity-Preserving Schemes," *Proceedings of the ASME 2000 International Mechanical Engineering Congress and Exposition*, HTD-Vol. 366-1, American Society of Mechanical Engineers, New York, 2000, pp. 135–143.

[5] Balasar, D. S., "Exact Jacobians of Roe-Type Flux Difference Splitting of the Equation of Radiation Hydrodynamics (and Euler Equations) for Use in Time-Implicit Higher-Order Godunov Schemes," *Journal of Quantitative Spectroscopy and Radiative Transfer*, Vol. 62, No. 3, 1999, pp. 255–278.

[6] Sakami, M., Mitra, K., and Hsu, P.-f., "Analysis of Light-Pulse Transport Through Two-Dimensional Scattering and Absorbing Media," *Journal of Quantitative Spectroscopy and Radiative Transfer*, Vol. 73, No. 2–5, 2002, pp. 169–179.

[7] Sawetprawichkul, A., Hsu, P.-f., and Mitra, K., "Parallel Computing of Three-Dimensional Monte Carlo Simulation of Transient Radiative Transfer in Participating Media," *Proceedings of the 8th AIAA/ASME Joint Thermophysics and Heat Transfer Conference*, AIAA, Reston, VA, 2002.

[8] Siegel, R., and Howell, J. R., *Thermal Radiation Heat Transfer*, 4th ed., Taylor and Francis, New York, 2002, Chap. 17.

[9] Burns, S. P., "Applications of Spatial and Angular Domain Based Parallelism to a Discrete Ordinates Formulation with Unstructured Spatial Discretizaion," *Proceedings of the International Symposium on Radiation Transfer*, edited by M. P. Menguc, Begell House, New York, 1997, pp. 173–194.

[10] Goncalves, J., and Coelho, P. J., "Parallelization of the Discrete Ordinates Method," *Numerical Heat Transfer*, Vol. 32, Part B, 1997, pp. 151–173.

[11] Goncalves, J., and Coelho, P. J., "Parallelization of the Finite Volume Method," *Proceedings of the International Symposium Radiation Transfer*, edited by M. P. Menguc, Begell House, New York, 1997, pp. 209–219.

[12] Liu, J., and Chen, Y. S., "Simulation of Rapid Thermal Processing in a Distributed Computing Environment," *Numerical Heat Transfer, Part A*, Vol. 38, 2000, pp. 129–152.

[13]Novo, P. J., Coelho, P. J., and Carvalho, M. G., "Parallelization of the Discrete Transfer Method: Two Different Approaches," HTD-Vol. 325, American Society of Mechanical Engineers, New York, 1996, pp. 45–54.

[14]Novo, P. J., Coelho, P. J., and Carvalho, M. G., "Parallelization of the Discrete Transfer Method," *Numerical Heat Transfer, Part B*, Vol. 35, 1999, pp. 135–161.

[15]Saltier, C., and Naraghi, M. H. N., "Parallel Processing Approach for Radiative Heat Transfer Prediction in Participating Media," *Journal of Thermophysics and Heat Transfer*, Vol. 7, No. 4, 1993, pp. 739–742.

[16]Ishimaru, A., *Wave Propagation and Scattering in Random Media*, Academic Press, New York, 1978, Chap. 15.

[17]Hsu, P.-f., "Optical Diagnostics Using Temporal Reflectance from a Ultra-Short Pulsed Laser," AIAA Paper 2002-3106, June 2002.

[18]Ozisik, M. N., *Radiative Transfer and Interaction with Conduction and Convection*, Wiley, New York, 1973, Chap. 8.

[19]Wu, S.-H., Wu, C.-Y., and Hsu, P.-f., "Solutions of Radiative Transfer in Inhomogeneous Participating Media Using the Quadrature Method," *Proceedings of the ASME 1996 International Mechanical Engineering Congress and Exposition*, HTD-Vol. 332, American Society of Mechanical Engineers, New York, 1996, pp. 101–108.

[20]Hsu, P.-F., Tan, Z., and Howell, J. R., "Radiative Transfer by the YIX Method in Nonhomogeneous, Scattering and Non-Gray Medium," *Journal of Thermophysics and Heat Transfer*, Vol. 7, No. 3, 1993, pp. 487–495.

[21]Li, B.-W., Chen, H.-G., Zhou, J.-H., Cao, X.-Y., and Cen, K.-F., "The Spherical Surface Symmetrical Equal Dividing Angular Quadrature Scheme for Discrete Ordinates Method," *Journal of Heat Transfer*, Vol. 124, No. 3, 2002, pp. 482–490.

[22]Pacheco, P. S., *Parallel Programming with MPI*, Morgan Kaufmann, San Francisco, 1997, Chap. 5.

[23]Hsu, P.-f., "Effects of Multiple Scattering and Reflective Boundary on the Transient Radiative Transfer Process," *International Journal of Thermal Sciences*, Vol. 40, No. 6, 2001, pp. 539–545.