Nuclear Mathematical and Computational Sciences: A Century in Review, A Century Anew
Gatlinburg, Tennessee, April 6-11, 2003, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2003)

ANALYSIS OF AND REFINEMENTS TO THE SABRINA VOLUME FRACTION ALGORITHM

Kenneth A. Van Riper
White Rock Science
P. O. Box 4729, Los Alamos, NM 87544 USA
kvr@rt66.com

ABSTRACT

The Sabrina code contains a volume fraction algorithm for calculating the amount of materials in each cell of a rectangular mesh overlying a combinatorial geometry description, such as used in Monte Carlo codes. The algorithm uses a spatial subdivision process combined with ray tracing. Using a test problem with known fractions, we explore the efficacy of both methods and the consequences of various choices and parameters used in the algorithms. The spatial subdivision method results in high accuracy but at the price of large calculation times. With several changes, the ray tracing method alone can achieve desired accuracies without the need for the expensive subdivision process. The ray tracing algorithm employs an iterative process where successively more rays are used until the fractions converge to the same values. Using a fixed large number of rays rather than the iterations is more efficient in the presence of curved material boundaries but not necessarily when only planar boundaries are present. Bad choices for the convergence criterion can lead to inaccurate results. We describe the changes made to the Sabrina volume fraction algorithms as a result of these studies, and plans for implementing volume fraction calculations on rectangular and cylindrical meshes in the Moritz code.

Key Words: Fractions, Materials, Volumes, Sabrina, Moritz

1. INTRODUCTION

Most radiation transport programs describe the problem geometry by one of two basic methods. Combinatorial descriptions, in terms of surfaces and/or solid bodies, are typically used by Monte Carlo codes. Mesh based descriptions are employed in discrete ordinates and other methods. Conversion from one type of geometry description the other is often necessary for analyzing the same model with different transport methods. Initial geometry definition with a combinatorial scheme may be more convenient for some users even if the intended use is in a mesh based transport code. Translation from combinatorial to mesh geometry requires an adequate mesh and the assignment of materials to each mesh cell. When a mesh cell encompasses regions of different materials, the translation algorithm must compute the proper fraction of each material.

The Sabrina code [1, 2, 3] contains a complicated algorithm for calculating volume fractions in a rectangular mesh. The algorithm involves a spatial subdivision of mesh cells followed by a ray tracing method in subcells containing multiple materials once a subdivision limit is reached. We intend to extend the volume fraction algorithm to cylindrical and spherical meshes and to implement a volume fraction capability in our new Moritz [4, 5] geometry editing program. Before proceeding, we used the existing algorithm to address several issues. The subdivision process leads to increased accuracy but with a large penalty in computation time. Would a ray

tracing method alone achieve adequate accuracies in comparable or less time? The ray tracing algorithm increases the number of rays traced until a convergence criterion is satisfied (or until a limit on the number of rays is reached). How do the parameters used in this criterion affect the accuracy?
In this paper, we describe the original Sabrina volume fraction algorithm, the accuracies and computation times obtained by applying the algorithm to a geometry model, and modifications to the algorithm we will use in new implementations.

2. ORIGINAL SABRINA VOLUME FRACTION ALGORITHM

To our knowledge, the original Sabrina volume fraction algorithm has not been described elsewhere. Jim West of the LANL Radiation Transport group developed it in the late 1980's. It was originally intended for use in conjunction with the group's FRAC-IN-THE-BOX code [6]; Sabrina would calculate fractions in mesh cells that could not be handled by FRAC's analytical algorithms.

2.1. Single Material Test

A mesh cell or any of its subdivisions is deemed to be a single material if the geometry cells at the 8 corners and center of the mesh cell all contain the same material. Fig. 1 shows the points used in the test.

2.2. Spatial Subdivision

If a mesh cell does not contain a single material according to the corners and center test, it is subdivided into 8 equally sized subcells. If a subcell does not pass the single material test, it is

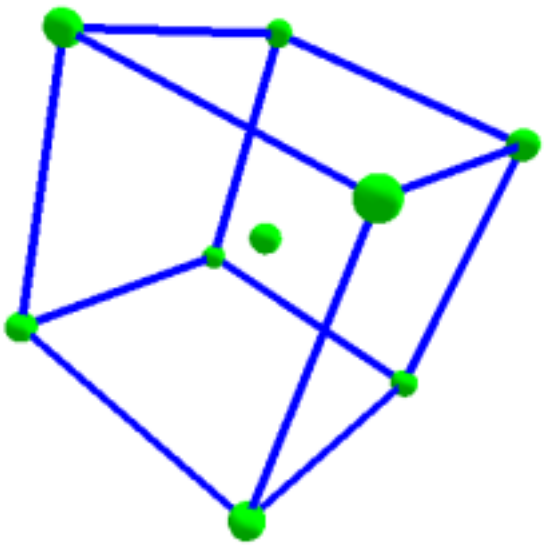

Figure 1. Points (green) used in single material test.

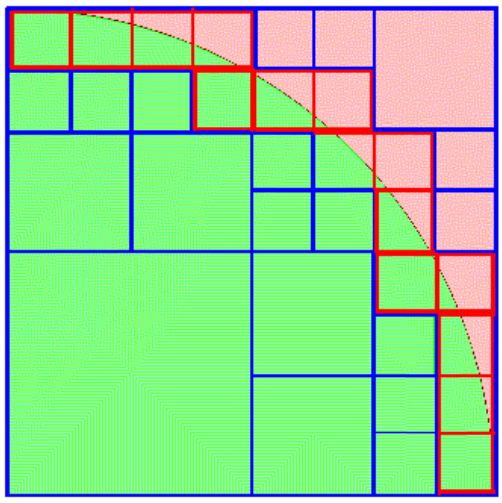

Figure 2. Subdivision of a mesh cell.

subdivided again. The subdivision process continues up to a maximum of L levels of subdivision. Material fractions in multiple material subcells are calculated by the ray tracing algorithm. The sum of the material fractions over the subcells, weighted by the subcell volume, gives the fractions for the mesh cell.

The limit L is an input parameter to the algorithm. Fig. 2 shows a 2 dimensional representation of the subdivision of a cell containing two materials (green and pink) with a limit $L = 4$ ($L = 1$ yields no subdivision). Subcells outlined in blue contain a single material; those outlined in red contain two materials, the fractions of which are found by ray tracing.

2.3. Volume Fractions by Ray Tracing

Ray tracing consists of advancing a straight line, or *ray*, across the mesh cell (or subcell) and keeping track of the path length in each material along the line. The fraction of material I in the cell is

$$f_I = \sum_j L_{Ij} \Big/ (DK), \tag{1}$$

where the sum is over all rays, L_{Ij} is the path length of material I for ray j, K is the number of rays, and the ray length D is the same for each ray.

All rays are parallel and perpendicular to a mesh cell face. (In the Sabrina implementation, the rays are parallel to the Z axis.) In the plane perpendicular to the ray direction, the ray positions form an evenly spaced $n \times n$ grid. The number of rays in Eq. (1) is then $K = n \times n$. The distance between cell edge and the ray closest to the edge is ½ of the distance between the rays. For example, the X coordinates of 5 rays between cell boundaries at $X = 0$ and $X = 10$ are 1, 3, 5, 7, and 9. Each ray is at the center of one of $n \times n$ equal sized rectangles that fill the cell.

Multiple ray tracings are made in iterations of increasing n, starting with $n^1 = 1$. For each iteration m, $n^m = n^{m-1} + 1$. For $m > 1$, the current set of volume fractions is an average of the fractions calculated by equation (1) and those from previous iterations:

$$f_I^m = \left[(n^m - 1) f_I^{m-1} + f_I \right] / n^m \tag{2}$$

for each material I encountered. The iterations are terminated if

$$\left| \left(f_I^m - f_I^{m-1} \right) / f_I^{m-1} \right| \le F \tag{3}$$

for each nonzero f_I^{m-1} or if $n^m = N$ where the limit of the grid size N is an input parameter. The convergence parameter in the original algorithm is $F = 0.05$.

3. VOLUME FRACTION TEST PROBLEM

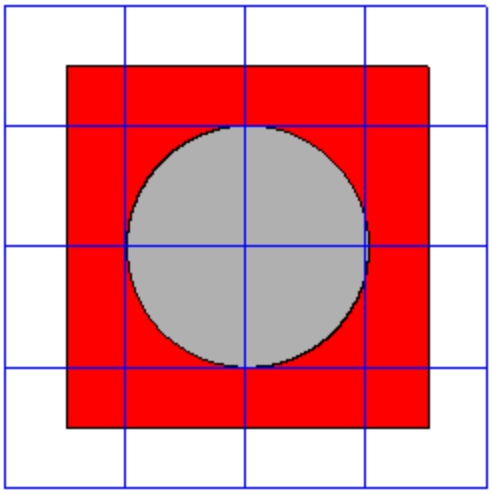

Figure 3. Volume fraction test problem.

To test the volume fraction algorithm, we applied it to problems in which the material volumes can be calculated analytically. The accuracy of the algorithm is judged by how closely it could reproduce the known material volumes. In models with only planar boundaries, such as a Right Parallelepiped (RPP) enclosed within another RPP, the algorithm yields the correct result quickly with low values of L and N. Models containing a curved surface provide a more exacting test. For the results presented below, we used a sphere of radius 1 centered in an RPP of length 3 on all sides. Both bodies are centered on the coordinate origin. A void region lies exterior to the RPP. An evenly spaced rectangular grid with distance 1 between grid lines extends from –2 to 2 in each direction. Fig. 3 shows a slice through the center of the model with the grid lines superimposed. Except for the lowest values of the limits L and N, the algorithm gives the volume of the void region within the grid exactly and the total volume within the grid to 9 digit precision or better. Equally accurate were the sum of the sphere and the surrounding cell (red in Fig. 3). We judged the accuracy of the calculation by the total volume within the sphere V_s. The accuracy shown below

$$A = 100\left|\frac{4\pi/3 - V_s}{4\pi/3}\right| \tag{4}$$

is the percent difference to the actual sphere volume.

3.1. Algorithm Modifications

To achieve the best accuracy from the ray tracing component, we replaced the default convergence parameter [Eq. (3)] by 0. (We explore below the influence of this parameter.) A number of initial runs showed that the averaging of Eq. (2) gave too much weight to the relatively inaccurate iterations with a small number of rays, resulting in convergence to the wrong result. Instead of averaging with previous iterations, we used only the result of the current iteration

$$f_I^m = f_I\,. \tag{5}$$

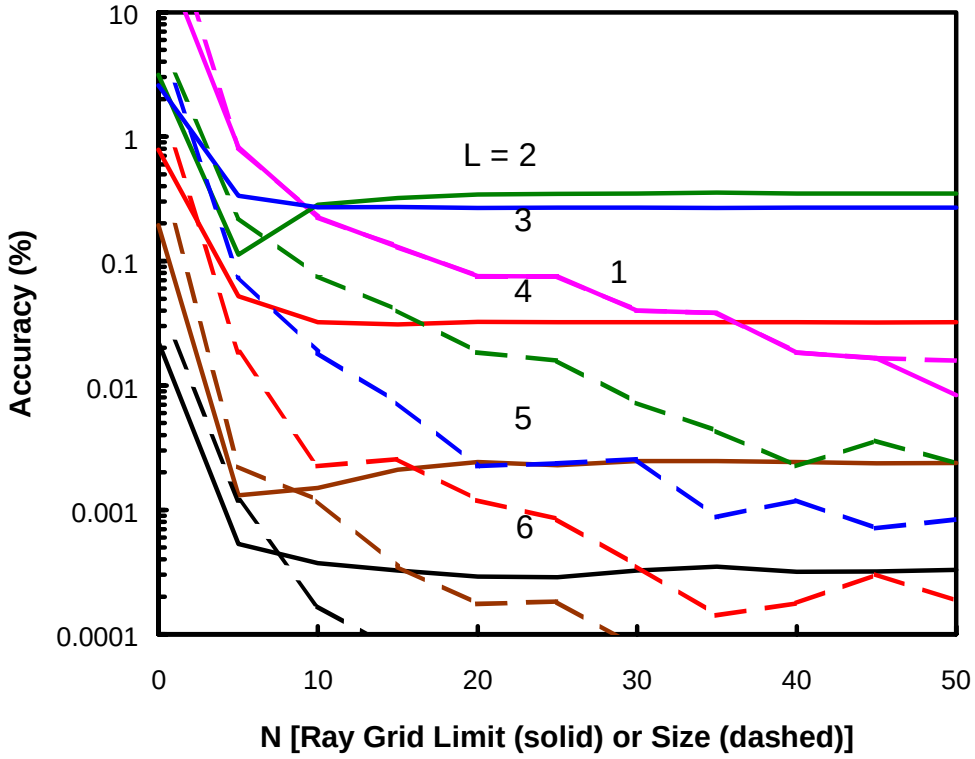

Figure 4. Accuracy as a function of *N*.

3.2. Test Problem Results

3.2.1. Dependence on *N* and *L*

Fig. 4 shows the accuracies of the volume fractions in the test problem as a function of the ray tracing grid size limit *N* for values of the subdivision limit *L* between 0 and 6. Fig. 5 shows the corresponding calculation times on a 733 MHz personal computer running Windows. Figs. 6 and 7 show the accuracies and calculation times as functions of *L* for various values of *N*. The solid lines are the results of the revised algorithm described above.

For $L = 1$, the accuracy improves (i. e., becomes smaller) with increasing values of *N*. For the same *N*, the accuracy increases from $L = 1$ to $L = 2$ and 3, and then decreases rapidly with increasing *L*. For $L \geq 2$, there is little improvement in the accuracy for $N > 10$. The improvement in accuracy with *L* comes at a steep price in computation time. The time increases by a factor of 4 for each increment of *L*.

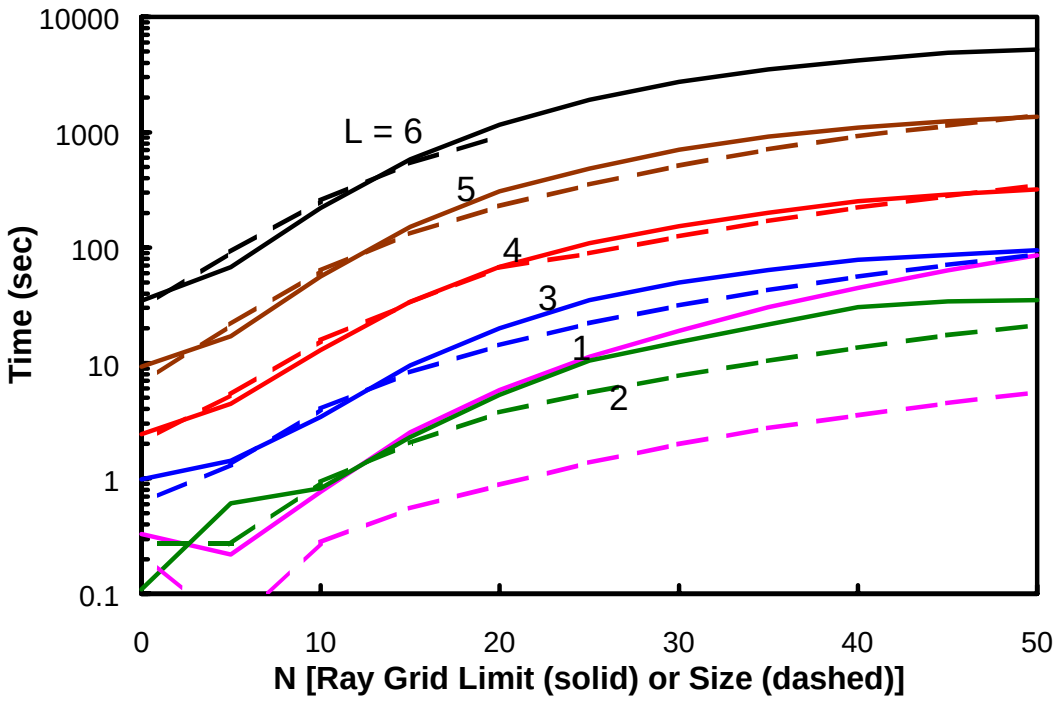

Figure 5. Calculation time as a function of *N*.

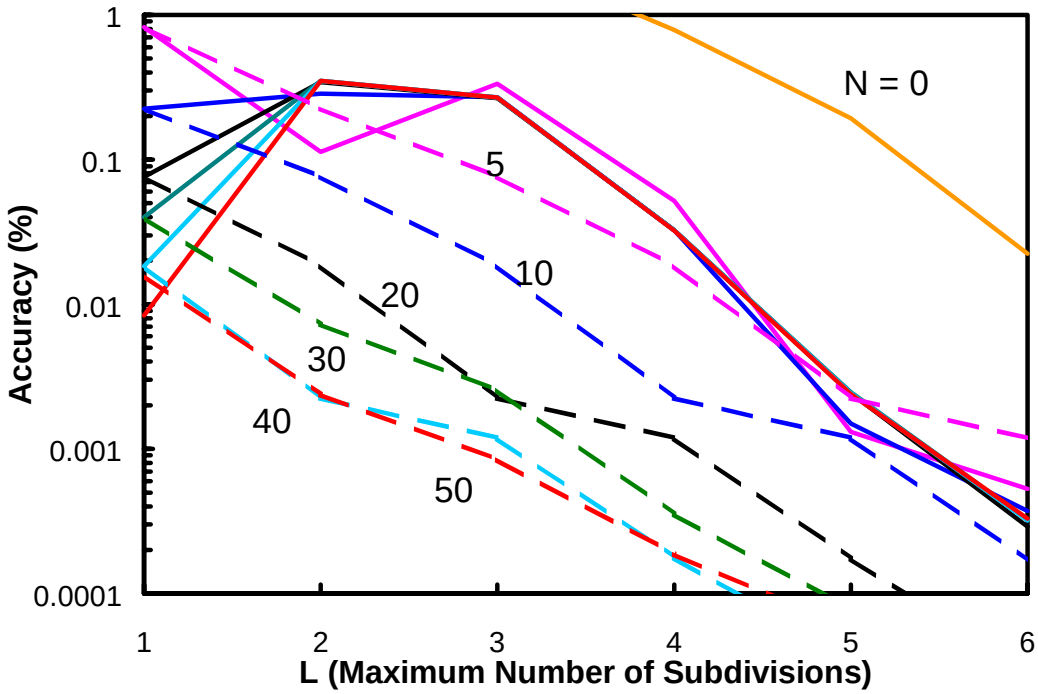

Figure 6. Accuracy as a function of *L*.

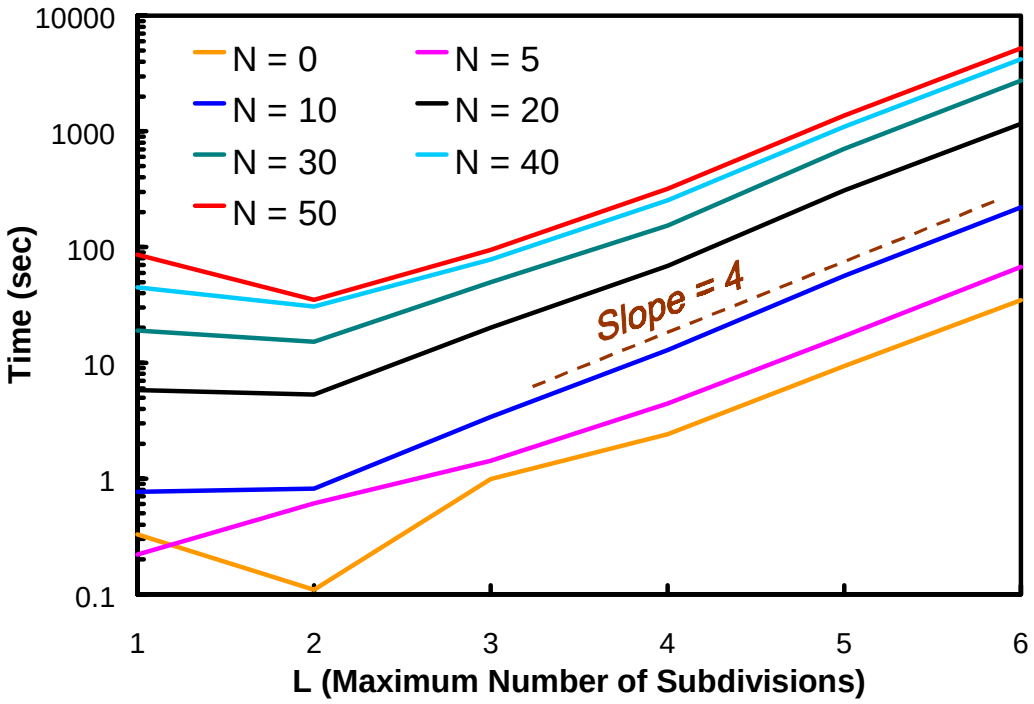

Figure 7. Calculation time as a function of *L*.

3.2.2. Skipping the ray tracing iterations

With the convergence factor F set to 0, it is likely that the ray tracing iterations are terminated by reaching the limit rather than because of convergence. Bypassing the iteration process and tracing $N \times N$ rays directly would offer a savings in time. The dashed lines in Figs. 4 through 6 show the results of doing so. For no subdivision ($L = 1$), the accuracy remains the same for each N, and the reduction in time is significant—a factor of 10 or more. With subdivision, the accuracy improves, but with little or no reduction in calculation time. The accuracy improves with increasing N, rather than remaining flat for $N > 10$ as in the previous set of calculations. The increase in A from $L = 1$ to $L = 2$ and 3 is also gone; A now decreases with increasing L.

3.2.3. Dependence on convergence factor

To explore the dependence on the convergence parameter F, we made a series of calculations with different values of F without subdivision ($L = 1$). The iteration process was activated for these calculations. Fig. 8 shows the accuracy as a function of ray tracing grid size limit N. Fig. 9 shows the corresponding computation times.

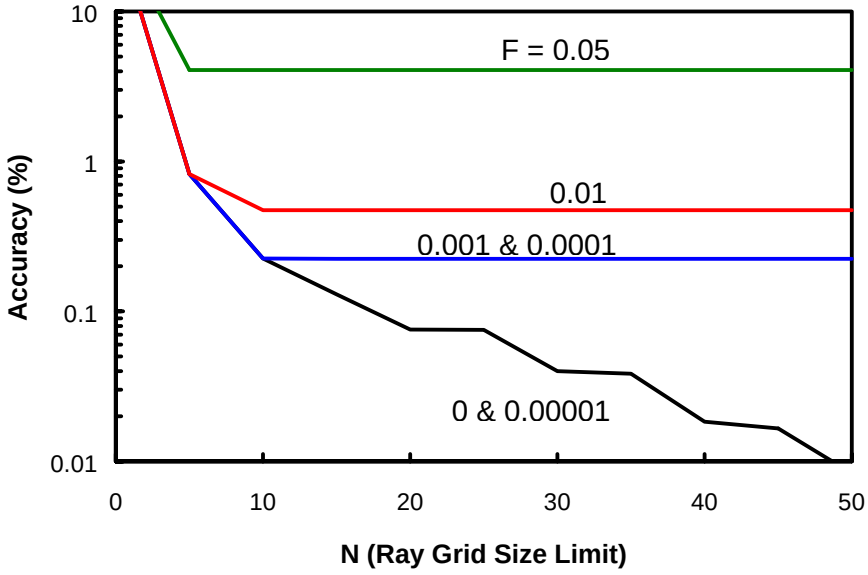

Figure 8. Accuracy for several values of *F*.

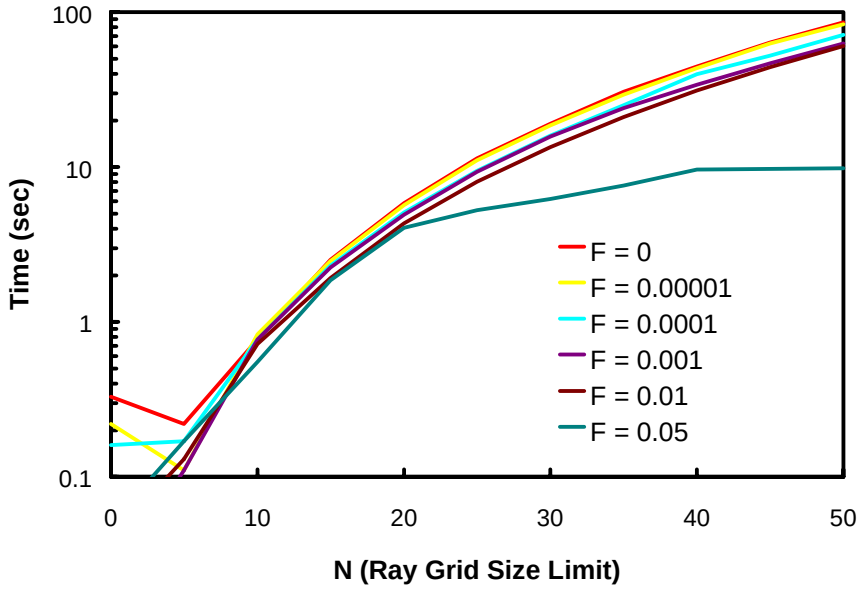

Figure 9. Calculation time for several values of *F*.

For $F < 0.00001$, the accuracy reaches a limit and fails to improve with larger N. The convergence criterion of eq. (3) is reached at a relatively small value of N. The original default value $F = 0.05$ is much too large and allows the iterations to terminate at very inaccurate values. Except for $F = 0.05$, little reduction in computation time results from using a larger F.

The flattening of the low F curves at large N is the same behavior seen in the solid curves (with the iteration process active) in Fig. 4 for $L > 1$. This flattening and the lack of it when the

iterations are bypassed shows that the convergence criterion is being satisfied for some number of rays $n < N$ even though $F = 0$.

4. CRITIQUE OF ORIGINAL ALGORITHM

4.1. Single Material Test

Deciding that a mesh cell consists of a single material by testing the corners and centers can lead to false positives when a geometry cell of another material intrudes through the side or edges. Fig. 10 shows two examples of a sphere penetrating a mesh cell that would nevertheless pass the single material test. Fig. 11 shows a more extreme example of an elliptical cylinder passing through a mesh cell that also passes the single material test. Such a case, however, reflects an inadequate mesh that is too coarse for the model.

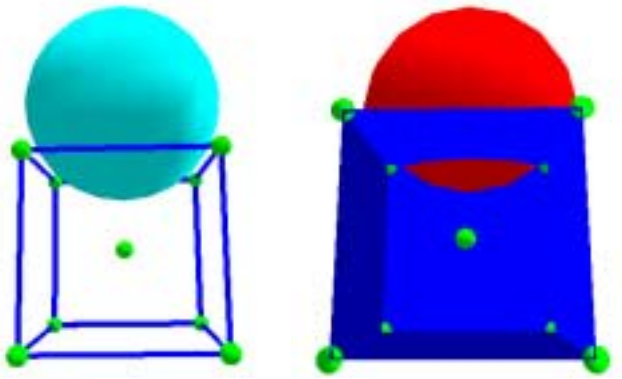

Figure 10. Intrusions of spheres into a mesh cell.

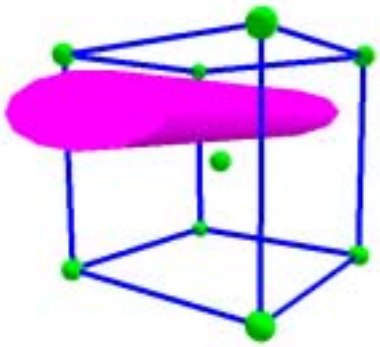

Figure 11. Cylinder passing through a mesh cell.

4.2. Spatial Subdivision

Although the process of spatial subdivision can lead to accurate fractions for large levels of division, the method is undesirable because of the increased calculation time. Desired accuracies can be achieved with ray tracing alone in less time.

4.3. Ray Tracing

When material boundaries are curved surfaces, it is best to skip the iterative ray tracing process and use the maximum number of rays directly. When using the iterations, very small values of the convergence parameter—$F \le 10^{-5}$—are necessary to prevent convergence to false fractions.

When the boundaries are planar, few, and not parallel to the ray direction, the iteration process will quickly converge to accurate fractions. In the simplest case of a plane boundary perpendicular to the ray, only a single ray is required. Fig. 12 shows two nested RPPs with a mesh grid superimposed. With the modifications of §3.1 in place, the calculated volume fractions are accurate to 10 digits with $N = L = 1$. Fig. 13 shows a cube bisected by a slanted plane. The ray direction is from top to bottom. The modified algorithm gives accurate fractions for $N \ge 2$ and $L \ge 1$.

When the cell contains boundaries that are parallel to the ray direction, the fractions will change slightly with the number of rays as the distribution of rays changes the number of rays on either side of the boundary. To handle such cases, the ideal algorithm would monitor the change in the fractions for the last few iterations, if used, or repeat the ray tracing with a slightly different number of rays if the iteration process is not used. The change would give an estimate of the accuracy of the fractions and, if large enough, cause the process to be repeated with more rays. Another solution would be to change the ray direction to minimize the number of surfaces parallel to the rays. Such a change is relatively straightforward in rectangular grids if the direction is restricted to be parallel to X, Y, or Z, but more complicated in cylindrical meshes where the optimal direction is parallel to the axis.

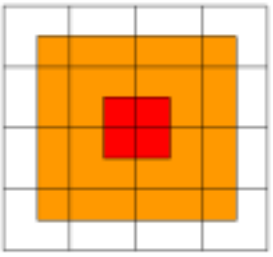

Figure 12. Nested RPPs.

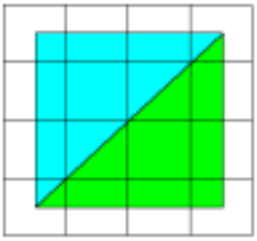

Figure 13. Two Wedges

The number of rays used, or the limit N when using the iterative algorithm, must be sufficiently large to achieve the desired accuracy. One may use the results from our test problem as a guide. Similar test problems more representative of the model in question, or cells from the model for which the fractions can be calculated analytically, could be used. The fractions should converge as the number of rays increases; the user should check that the convergence is not to a false value.

The grid of starting points in the plane perpendicular to the ray is always $n \times n$—the same number of points is used in each direction. Different values in the two directions would offer more uniform sampling of cells with large aspect ratios.

5. CHANGES AND IMPLEMENTATION PLANS

5.1. Sabrina

In Sabrina, we have changed the default convergence parameter to $F = 10^{-5}$ and replaced the averaging of equation (2) by equation (5). New commands permit the user to set F, bypass the iteration process, and to use all features of the original algorithm.

5.2 Moritz

In Moritz, we will implement volume fraction algorithms on both rectangular and cylindrical meshes. Implementation on a spherical mesh could be added if there is strong user demand. We will use ray tracing only without spatial subdivision. The single material test as described in §2.1 will not be used. Instead, we will use an algorithm based on the bounding boxes (the maximum extent in X, Y, and Z) of the geometry cells. If only a single bounding box intersects a mesh cell, the mesh cell is a single material. The ray tracing algorithm will be based on the observations noted in §4.3. We will provide user input for the number of rays to use or for N and F when using

the iteration algorithm, whether or not to do the iterations, and other algorithm choices and parameters that may become apparent as the features are implemented.

6. CONCLUSIONS

The volume fraction algorithm as originally implemented in Sabrina gives accurate results if large levels of spatial subdivision are used. Using a large degree of subdivision, however, comes at the price of significantly increased computer time. Several unfortunate choices in the original ray tracing algorithm prevent accurate results when relying on that method alone. With some changes, acceptable accuracies can be achieved with ray tracing without relying on spatial subdivision. To achieve optimal performance, the algorithm should adapt to the type and direction of material boundaries within each mesh cell.

REFERENCES

1. James T. West III, *SABRINA: An Interactive Three-Dimensional Geometry-Modeling Program for MCNP*, Los Alamos National Laboratory Report LA–10688–M (1986).
2. Kenneth. A. Van Riper, *Sabrina User's Guide*, Los Alamos National Laboratory Report LAUR–93–3696 (1993).
3. Kenneth. A. Van Riper, "New Features in Sabrina," *Proceedings of the Topical Meeting on Radiation Protection for our National Priorities*, Spokane, WA, Sept. 17–21, 2000, pp. 316–323 (2000).
4. Kenneth. A. Van Riper, "Interactive 3D Display of MCNP Geometry Models," *Proceedings of the ANS International Meeting on Mathematical Methods for Nuclear Applications*, Salt Lake, UT, Sept. 10–13, 2001.
5. Kenneth. A. Van Riper, "Creation and Viewing of MCNP/MCNPX Meshes and Grid Tally Data in Moritz," *Proceedings of the Topical Meeting on Radiation Serving Society*, Santa Fe, NM, April 14–18, 2002.
6. Dave G. Collins and James T. West III, *FRAC-IN-THE-BOX Utilization*, Los Alamos National Laboratory Report LA–11606–MS, Rev. 1, (1990).