AN EXPERT SYSTEM FOR AUTOMATIC MESH GENERATION FOR S_N PARTICLE TRANSPORT SIMULATION IN PARALLEL ENVIRONMENT

Apisit Patchimpattapong[1], Alireza Haghighat, Daniel Shedlock
Department of Nuclear and Radiological Engineering
University of Florida
202 Nuclear Sciences Building, Gainesville, FL 32611, USA
axp227@ufl.edu, haghighat@ufl.edu, shedlock@ufl.edu

Abstract

An expert system for generating an effective mesh distribution for the S_N particle transport simulation has been developed. This expert system consists of two main parts: 1) an algorithm for generating an effective mesh distribution in a serial environment, and 2) an algorithm for inference of an effective domain decomposition strategy for parallel computing. For the first part, the algorithm prepares an effective mesh distribution considering problem physics and the spatial differencing scheme. For the second part, the algorithm determines a parallel-performance-index (PPI), which is defined as the ratio of the granularity to the degree-of-coupling. The parallel-performance-index provides expected performance of an algorithm depending on computing environment and resources. A large index indicates a high granularity algorithm with relatively low coupling among processors. This expert system has been successfully tested within the PENTRAN (Parallel Environment Neutral-Particle Transport) code system for simulating real-life shielding problems.

[1] Current contact information: Electricity Generating Authority of Thailand, Office of Corporate Planning, Bangkruai, Nonthaburi 11130, Thailand, apisitpc@egat.or.th

Introduction

The discrete ordinates (S_N) method has been widely used to obtain numerical solutions of the transport equation. The method calls for discretization of independent variables: angle, energy, and space. To generate an "effective" spatial mesh distribution, one has to consider various factors including particle mean-free-path, material and source discontinuities, problem objectives and numerics. For large and complex physical systems, parallel computing has become necessary to obtain accurate results in a reasonable time. To perform a parallel S_N calculation, one may decompose the three independent variables and distributes them among processors. Depending on how they are distributed, memory requirement and parallel performance may vary from one scheme to the other. To overcome these difficulties, we have developed an expert system for preparing inputs for an effective S_N particle transport simulation. This expert system comprises two major components: 1) an algorithm for generating an effective mesh distribution, common for both serial and parallel calculations; and 2) an algorithm for selecting an effective domain-decomposition-strategy (DDS) for parallel computing. The first component has been previously reported in Refs. 1 and 2, and the second component has been tested in Refs. 2 and 3 for the 4-processor cases. In this paper, we present additional tests of this second component on 8 and 16 processors.

The following sections discuss methodologies developed for our expert system, and examine its capability using the PENTRAN (Parallel Environment Neutral-Particle Transport) code system [4] to simulate the VENUS-3 benchmark experimental facility [5].

Algorithm for generating an effective mesh distribution in a serial environment

This algorithm can be divided into four steps: 1) creation of a 3-D geometric model and coarse meshes, 2) calculation of uncollided fluxes, 3) selection of differencing schemes, and 4) generation of a fine mesh distribution. Following subsections describe methodologies and formulations used in each step.

1. Creation of a 3-D geometric model and coarse meshes

A 3-D physical model can be partitioned into x-y-z coarse meshes based on material and source boundaries and problem objectives. These coarse meshes are continuous, i.e. coarse mesh boundaries along each of the coordinate axes are applied throughout the entire model. To obtain a 3-D geometric model, we first partition a physical system into z-levels along the z-axis. For each z-level, we utilize AutoCAD [6] to create a drawing that is used through its corresponding *Drawing Interchange File* (DXF) which contains dimensions, material identifications and coarse mesh boundaries.

2. Calculation of uncollided fluxes

To select an appropriate differencing scheme, one needs to know the behavior of the particle flux. We obtain an approximate flux shape based on an uncollided flux distribution from a parallel code, PENFC (Parallel Environment Neutral-Particle First Collision) [1]. PENFC is capable of calculating uncollided and first collision fluxes in a 3-D Cartesian geometry in a parallel environment. It is worth noting that the uncollided flux at one location can be calculated independently and PENFC includes an MPI-based

algorithm for parallel processing. As a result, the amount of computation time is reduced significantly with parallel processing.

3. Selection of differencing schemes

A differencing scheme is a fitting formulation that represents the behavior of angular flux within a spatial mesh. In the S_N method, considering the problem physics, we allow the use of different spatial differencing schemes throughout a model. We fit a function, representing the behavior of differencing scheme, to an uncollided flux distribution obtained from PENFC. Then, we apply the least squares method to determine which differencing scheme best suits the flux distribution.

4. Generation of fine mesh distribution

We have developed a serial code PENXMSH [3], which generates a fine mesh distribution based on preserving material boundaries, conserving material masses (volumes), and considering particle interactions (mean-free-path). PENXMSH utilizes a geometric model and coarse mesh layout from the AutoCAD DXF file provided from Step 1 (Creation of a 3-D geometric model and coarse meshes). A coarse mesh is partitioned into fine meshes based on user-specified mean-free-path. To assign material to a fine mesh, PENXMSH checks the center of the fine mesh against the layers of geometric shapes described in Step 1. To achieve certain accuracy in preserving material volumes, an iterative procedure is used to examine different mesh sizes.

Algorithm for selecting an effective DDS for parallel computing

To perform a parallel S_N calculation, one may decompose the three independent domains (variables): angle, energy and space. Depending on how sub-domains are distributed among processors, each decomposition strategy may require different amount of memory, and results in different parallel performance. There are four main factors that affect parallel performance: 1) number of processors and memory available per processor, 2) load balance, 3) granularity, and 4) degree-of-coupling. We now discuss the nature of each factor, our approach to estimate them, and their use for selection of an effective DDS.

Factors affecting parallel computing

1. Number of processors and memory available per processor

This factor indicates which DDS are feasible within available computing resources. The majority of memory in PENTRAN is used for storing angular fluxes and flux moments. These arrays are allocated based on the number of local coarse meshes, local energy groups, and local angular sweep octants on each processor. Therefore, different decomposition strategies may require different amounts of memory. For this, we utilize a mapping algorithm used in PENTRAN to estimate the memory requirement.

2. Load balance

This factor indicates how workload is distributed among processors. For any decomposition strategy, the number of octants, groups and coarse meshes is always divided equally among processors. However, each coarse mesh may contain different numbers of fine meshes and may use different numerics (differencing scheme). These differences result in a load imbalance among processors. In our algorithm this load imbalance is accounted for by estimating the idle time of each processor.

3. Granularity

Granularity is defined as the number of operations performed per number of communications. Each decomposition strategy has its own unique computation and communication structure, which results in different granularity.

4. Degree-of-coupling

A general parallel S_N code such as PENTRAN allows hybrid domain decompositions including any combination of angular, energy and spatial decompositions. Each sub-domain (octant, group and coarse mesh), is processed by one processor only. Degree-of-coupling (DCP) is defined as the contribution to the total source in a sub-domain from other processors. This quantity indicates how one processor depends on data from others. We define DCP as:

$$DCP^{P}_{V,g,k} = \frac{\left(\sum_{surface=1}^{6} \left(J^{In}_{g,k} A \right)_{surface} + \sum_{g'=1}^{g} \sigma_{g' \to g} \phi_{g'} V \right)_{P}}{\sum_{i=1}^{Nproc} \left(\sum_{surface=1}^{6} \left(J^{In}_{g,k} A \right)_{surface} + \sum_{g'=1}^{g} \sigma_{g' \to g} \phi_{g'} V \right)_{i} + S^{Fixed}_{g} V} \qquad (1)$$

where $DCP^{P}_{V,g,k}$ = degree-of-coupling of a processor P to the total source of volume V, group g and octant k; $J^{In}_{g,k}$ = incoming current; A = surface area; $\sigma_{g'®g}$ = scattering cross section from group $g'\gamma g$; $\phi_{g'}$ = scalar flux of the energy group g' ; S^{Fixed}_{g} = fixed source strength

Estimation of parallel performance index

First, we estimate the performance based on the granularity and the load imbalance. For this, we define a parameter CPCM as:

$$CPCM = \frac{\text{Computation Time}}{\text{Communication time} + \text{Idle time}} \qquad (2)$$

This parameter is determined based on a transport sweep, and does not account for the impact of the decomposition on the convergence (numerical) behavior of the parallel algorithm. For this purpose, we use the DCP parameter and define a parallel-performance-index (PPI) as:

$$\mathrm{PPI} = \frac{\mathrm{CPCM}}{\mathrm{DCP}} \tag{3}$$

A large value of the PPI indicates that the parallel algorithm can perform a large number of computations relative to the number of communications, while it does not diminish the rate of convergence.

Estimation of CPCM

To estimate CPCM for a transport sweep without performing the actual transport calculation, we need the following parameters:

- Number of fine meshes per coarse mesh
- Differencing scheme of each coarse mesh
- Computation time of a transport sweep for each differencing scheme
- Computation time of a scattering source
- Communication time of a transport sweep

Fine mesh distribution and differencing schemes can be obtained from Steps 4 (Generation of fine mesh distribution) and 3 (Selection of differencing scheme) of the serial algorithm, respectively. To estimate the computation and communication times, we utilize the PENTRAN code and its basic algorithms. To estimate the transport sweep computation time for each differencing scheme, we perform serial calculations using a simple model (e.g. 1 coarse mesh, 1,000 fine meshes, S6, P3, 1 group). For the computation time of the scattering source, we utilize the scattering source algorithm available in PENTRAN to estimate its computation time. Note that the computation times of each differencing scheme and the scattering source are estimated only once for each computing platform to account for processor characteristics. To estimate the communication time of the transport sweep, we need to obtain a parallel message passing time and a waiting time caused by load imbalance as described in Factor 2 (Load balance). We utilize the PENTRAN communication to: 1) determine sending and receiving processors; and 2) simulate parallel message passing. The first part is carried out in a serial environment and the second part is performed in the parallel environment of interest to measure the time spent for message passing. By using the preceding information, we can simulate the transport sweep and the scattering source calculation to estimate the waiting time. This waiting time is added to the parallel message passing time to obtain the communication time of the transport sweep. Consequently, the CPCM can be obtained from a ratio of the total computation time to the communication time of the transport sweep.

Estimation of DCP

To estimate DCP using Eq. 1, we obtain an angular flux distribution, partial currents, and a source distribution from a “small model” serial PENTRAN calculation. This small model must represent the problem physics yet should require a relatively short computation time on a single processor. We create

this model by coarsening meshes and reducing a quadrature order. We find the maximum DCP of each energy group. To account for different group convergence behaviors, we weight the DCP of each group by its corresponding c-ratio ($C_g = (\sigma_s/\sigma_t)_g$, which is related to the spectral radius, i.e. a measure of the rate of convergence). We define the DCP of the whole problem by:

$$DCP = \sum_{g=1}^{G} C_g DCP_g^{Max} \tag{4}$$

where DCP^{Max}_g = maximum degree-of-coupling of the energy group g; C_g = c-ratio of the energy group g; and G = number of energy groups.

Test problem description

The VENUS-3 facility is the LWR-PVS benchmark experimental facility with partial length shielded assemblies located at SCK•CEN, Mol (Belgium). Its overall dimensions are 65.573x65.573 cm^2 and 70 cm high. In the following sections, we examine different aspects of our expert system by simulating the VENUS-3 benchmark facility. All calculations of the first component (Algorithm for generating an effective mesh distribution in a serial environment) were performed on the PCPEN cluster[2] while those of the second component (Algorithm for selecting an effective DDS for parallel computing) were performed on the PCPEN-II cluster[2].

Evaluation of the algorithm for generating an effective mesh distribution in a serial environment

1. Evaluation of the algorithm for creating a 3-D geometric model and coarse meshes

We partition a 3-D geometric model of the VENUS-3 facility into four z-levels. Fig. 1 illustrates the process of creating a 2-D model for z-level 2. For this z-level, there are ten layers of shapes and a layer of coarse mesh layout. We utilize AutoCAD to create these layers and to project them onto a plane to create a 2-D model. Repeat the same process for other z-levels to complete the 3-D model.

[2] PCPEN and PCPEN-II clusters are owned by the Department of Nuclear and Radiological Engineering, University of Florida. PCPEN has 8 nodes: 7 nodes of 1 GHz Pentium III processors and a head node of 1.7 GHz AMD Athlon processor, each of which has 2 GB RAM. PCPEN-II has 8 nodes (16 processors) of Dual Intel Xeon processors at 2.4 GHz with 4 GB RAM each node.

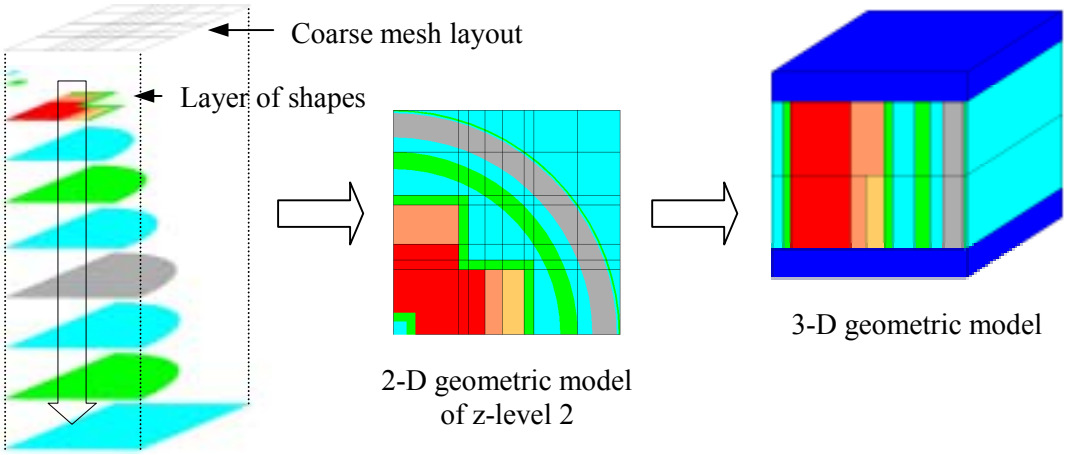


Figure 1. Illustration of the process to create a 2-D geometric model for z-level 2 of the VENUS-3 facility.

2. Evaluation of the algorithm for calculating uncollided fluxes

The VENUS-3 problem model is partitioned into 6x5x4 x-y-z coarse meshes. For simplicity, we homogenize materials in each coarse mesh, and use the first energy group (15.76 MeV) of the BUGLE-96 library [7]. For the reference PENTRAN model, we utilize a uniform fine mesh distribution of 5x5x5 per coarse mesh, and S8 level-symmetric quadrature set. For PENFC, we calculate the flux at seven locations per coarse mesh, i.e., three locations along each axis through the center of the mesh. We use 10x10 angles (polar x azimuthal angles) for the numerical integration. Fig. 2 shows a comparison of the normalized uncollided fluxes for the mid-point energy of 15.76 MeV along x-axis at y = 3.15 cm and z = 22.5 cm.

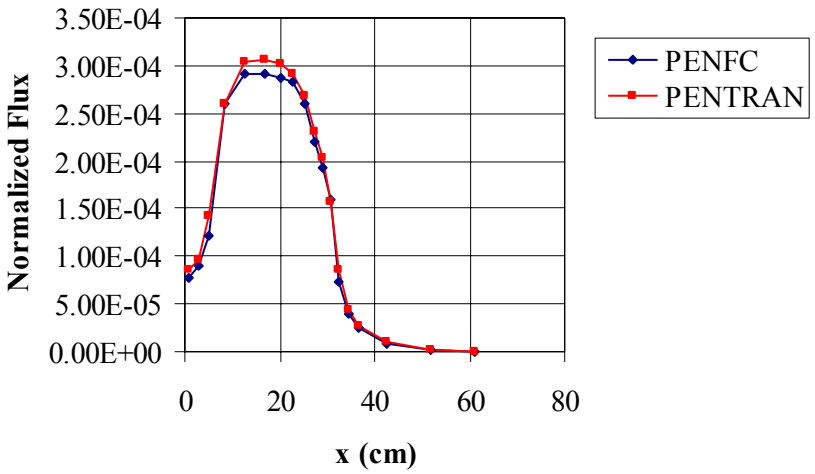


Figure 2. Comparison of the normalized uncollided fluxes along x-axis at y = 3.15 cm and z = 22.5 cm, at 15.76 MeV.

We observe a good agreement between PENFC and PENTRAN flux distributions. The computation times of PENFC are 147.99 sec for a serial calculation and 38.51 sec for a 4-processor parallel calculation while PENTRAN requires 33.3 sec on one processor. We observe a high parallel efficiency of 96%. This high efficiency is expected because PENFC formulation is highly parallelizable, i.e. each position can be processed independently. For this test problem, PENTRAN requires less computation time than PENFC. However, PENFC solves for uncollided fluxes semi-analytically; therefore, it is not suffered by the ray-effect, i.e. the unphysical oscillation of scalar fluxes due to insufficient number of directions in problems with very low scattering and localized sources.

3. Evaluation of the algorithm for selecting differencing schemes

We examine the capability of our algorithm in predicting an appropriate differencing scheme based on the uncollided flux shape obtained from PENFC in the previous step. We select a differencing scheme within the PENTRAN adaptive differencing strategy [8]. For this test problem, our predictions agree with the PENTRAN predictions for 84% of the coarse meshes. This demonstrates that the uncollided flux distribution is adequate for determination of an appropriate differencing scheme. The computation time of this step is a few seconds.

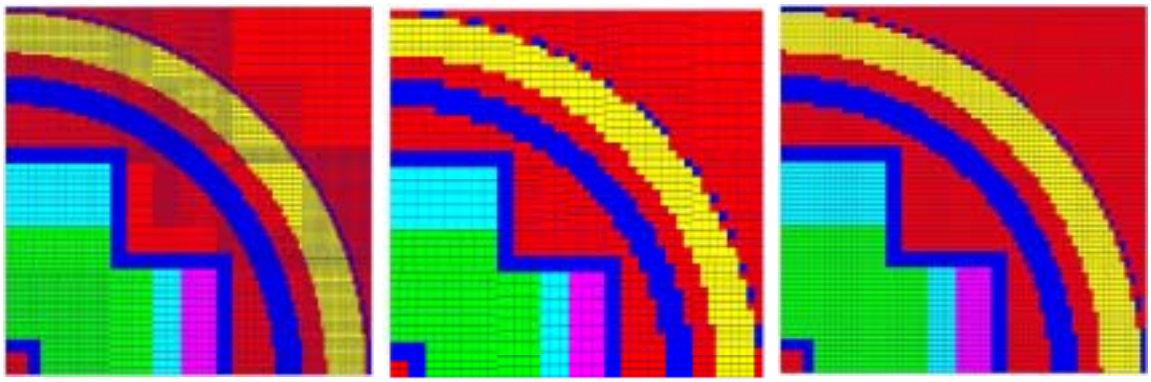
4. Evaluation of the algorithm for generating a fine mesh distribution

We investigate the effectiveness of PENXMSH for generating a fine mesh distribution. For demonstration, we compare flux values of the following three fine mesh distributions (generated by PENXMSH).

Reference-mesh: Max. material loss of 2 %, 104,920 fine meshes (0.1-0.2 mean-free-paths)
Variable-mesh: Max. material loss of 13 %, 20,560 fine meshes (0.3-0.35 mean-free-paths)
Uniform-mesh: Max. material loss of 8 %, 57,928 fine meshes (1-cm uniform mesh)

Fig. 3 shows the fine mesh distributions for z-level 2 of the VENUS-3 model. Numbers of coarse meshes along x, y and z axes are 8x7x4, respectively. We use the first energy group of the BUGLE-96 library, S8 level-symmetric quadrature set and P3 Legendre scattering order. The computation time of PENXMSH is less than 10 sec for each mesh distribution.

The flux distributions obtained from all three mesh distributions are in good agreement. We observe the maximum differences of ~10% and ~16% for the variable-mesh and the uniform-mesh compared to the reference-mesh, respectively. The computation times of PENTRAN are: 1,233.9 sec for the reference-mesh (8-processor parallel calculation), 284.2 sec for the variable-mesh (serial calculation), and 875.5 sec for the uniform-mesh (serial calculation). The variable-mesh yields more accurate results than the uniform-mesh, while requiring significantly less computation time. This is caused by the fact that the variable-mesh is generated based on expert's knowledge of problem physics, i.e. preserving material boundaries and considering particle interactions (mean-free-path). As a result the mesh density varies depending on problem geometry and material properties. In contrast, the uniform-mesh leads to over-meshing and large computing time. Note that the reference-mesh calculation was performed on 8 processors, because it requires more memory than what is available per processor.



Reference-mesh Variable-mesh Uniform-mesh

Figure 3. Fine mesh distributions for z-level 2 of the simplified VENUS-3 model.

Evaluation of the algorithm for selecting an effective DDS for parallel computing

Thus far, we have examined the effectiveness of our expert system in a serial environment. In this section, we evaluate our algorithm for selecting an effective DDS for parallel processing.

For reference calculations, we utilize the variable-mesh distribution as previously described with S8 level-symmetric quadrature set, P3 Legendre scattering order, and the first 26 energy groups of the BUGLE-96 library. We perform 8- and 16-processor parallel calculations for four different domain-decomposition-strategies (DDS) (one spatial only, two angular-spatial, and angular only decomposition strategies), and examine the CPCM and PPI parameters for each strategy. To measure the accuracy of the CPCM and PPI predictions, we determine the "actual" CPCM, and define the "actual" parallel-performance-index (APPI) as:

$$\mathrm{APPI} = \frac{1}{\mathrm{Transport\,sweep\,time}} \tag{5}$$

Before estimating the performance parameters, we determine if different DDS can fit the memory requirements of the problem on the available memory of processor cases. Table I gives the required memory per processor for each DDS for 8- and 16-processor calculations.

Table I. Memory required per processor for different DDS using different number of processors

Number of processors	DDS	A	G	S	Memory (MB/Proc)
8	1	1	1	8	157.7
	2	2	1	4	218.7
	3	4	1	2	343.9
	4	8	1	1	595.7
16	1	1	1	16	91.6
	2	2	1	8	121.1
	3	4	1	4	183.1
	4	8	1	2	308.8

A / G / S = Number of processors devoted to angular/group/spatial domain

Since each processor has 2 GB of memory, it is apparent that all the DDS cases can be examined because they require less memory than the available memory. Also, for all processor cases, the spatial DDS results in the lowest amount of memory per processor while the angular DDS requires the highest amount. This is because the angular DDS requires all of the spatial arrays for angular fluxes and moments, while the spatial decomposition partitions them among processors.

Evaluation of the algorithm for estimating the CPCM

Figs. 4 and 5 compare the actual and the predicted CPCM for 8- and 16-processor calculations, respectively.

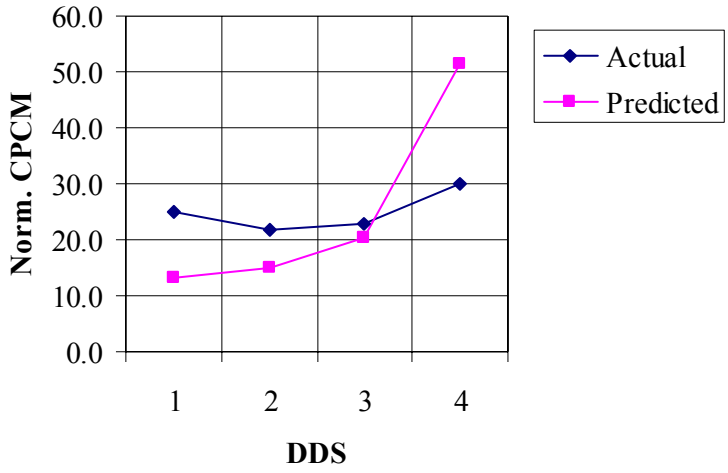


Figure 4. Comparison of the actual and the predicted CPCM for 8-processor calculation.

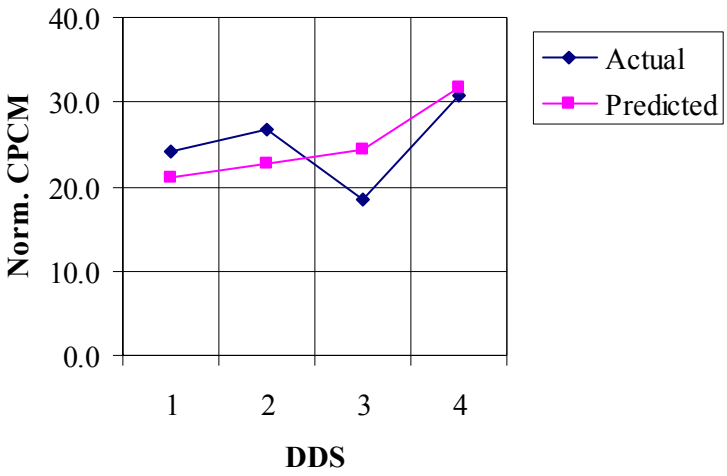


Figure 5. Comparison of the actual and the predicted CPCM
for 16-processor calculation.

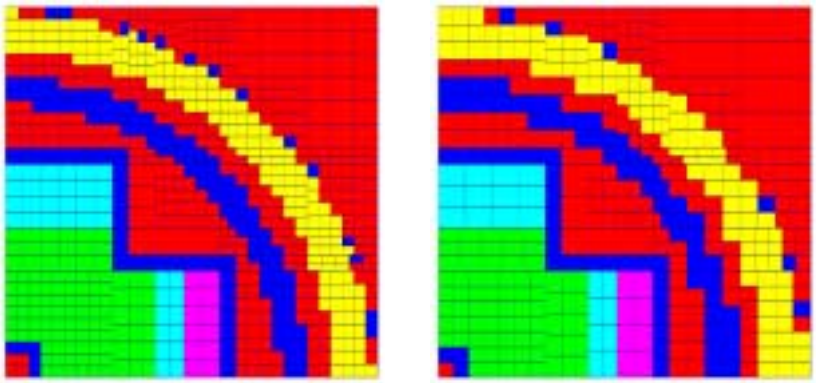
We observe that our predictions follow similar trends as the estimated actual CPCM.

Evaluation of the algorithm for estimating the DCP

To examine the DCP algorithm, using PENXMSH, we prepare three mesh distributions as follows:

Max. material loss of 13 % (reference), S8 level-symmetric quadrature set
Max. material loss of 40 %, S6 level-symmetric quadrature set
Max. material loss of 50 %, S6 level-symmetric quadrature set

For demonstration, in Fig. 6, we show the last two fine mesh distributions for z-level 2 of the VENUS-3 model.



Max. material loss of 40 % Max. material loss of 50 %

Figure 6. Fine mesh distributions for z-level 2 of the simplified VENUS-3 model.

We perform PENTRAN serial calculations using P0 and 26 energy groups, with the inner flux iteration tolerance of 10.0%. Note that for the reference calculation, the inner flux tolerance is 0.1%. Figs. 7 and 8 show a comparison of the inverses of the DCP of the three mesh distributions for 8- and 16-processor calculations, respectively.

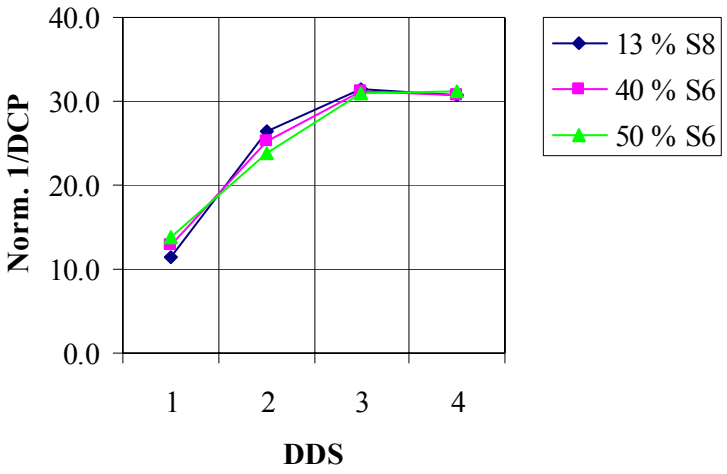


Figure 7. Comparison of the inverses of the DCP of different mesh distributions for 8-processor calculation.

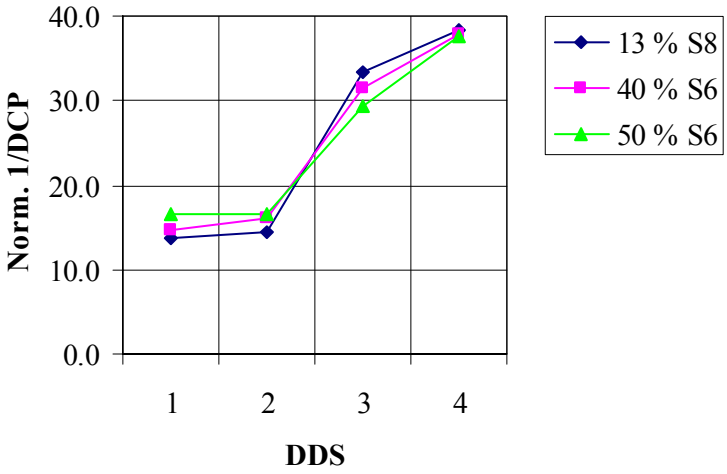


Figure 8. Comparison of the inverses of the DCP of different mesh distributions for 16-processor calculation.

We observe a similar behavior of DCP among the three mesh distributions for both 8 and 16 processors. It is clear that the DCP prediction is highly insensitive to the material conservation, and numerical parameters, and very approximate model with 50% loss of material can be used for this application. Table II gives the computing resources required for the PENTRAN and the DCP calculations.

Table II. Comparison of computing resources required for estimating DCP.

Max. material loss (%)	Quadrature set	PENTRAN serial calculation		Number of processors	DCP calculation (sec)
		Memory (MB/Proc)	CPU time (sec)		
13	S8	633.2	510.7	8 16	611.8 643.7
40	S6	330.2	174.1	8 16	188.0 206.0
50	S6	202.4	113.9	8 16	109.9 121.4

The 50%-material-loss mesh distribution has fewer meshes than the 40%-material-loss mesh distribution; therefore, it requires less computing resources.

Evaluation of the algorithm for estimating the PPI

Figs. 9 and 10 show a comparison of the APPI and the PPI (using 50%-material-loss mesh) for 8- and 16- processor calculations, respectively.

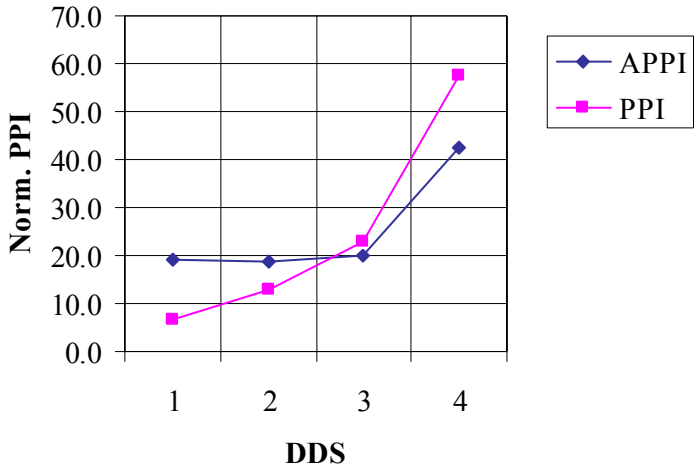


Figure 9. Comparison of the APPI and the PPI for 8-processor calculation.

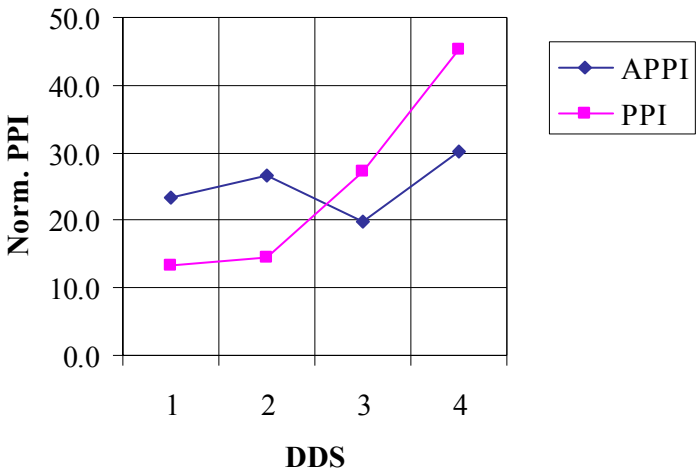


Figure 10. Comparison of the APPI and the PPI for 16-processor calculation.

We observe that the PPI follow similar trends as the APPI. They both indicate that the angular decomposition strategy is the most effective DDS. However, it is interesting to note that for this problem the CPCM by itself provides a better parallel performance prediction than the PPI, as shown in Figs. 11 and 12.

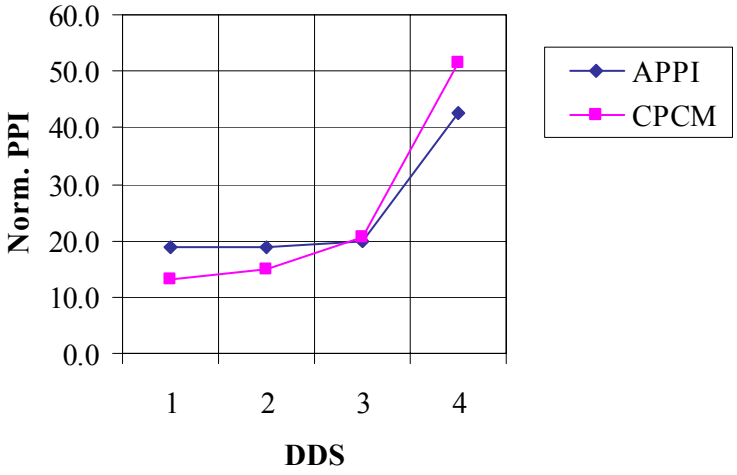


Figure 11. Comparison of the APPI and the CPCM
for 8-processor calculation.

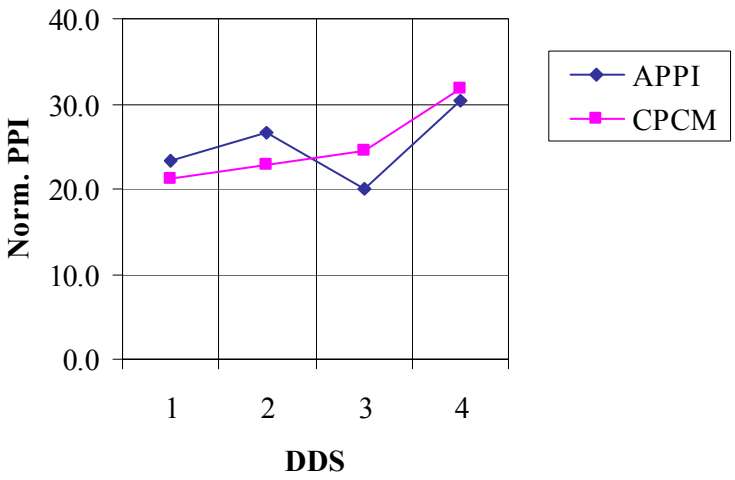


Figure 12. Comparison of the APPI and the CPCM for 16-processor calculation.

We observe a good agreement between the two indexes. This demonstrates that for this problem the CPCM is sufficient for obtaining the PPI, which agrees with the previous result of 4-processor calculation in Ref. 3. Further study and testing is necessary to verify the CPCM and the PPI parallel performance predictions and to derive an alternative DCP formulation.

Computation time of the algorithm for selecting an effective DDS

Our algorithm required total computation times of 244.2 and 261.5 sec for 8- and 16-processor calculations, respectively. About 90% of these computation times are spent on estimation of the DCP. Table III gives wall-clock times of each DDS for 8 and 16 processors.

Table III. Comparison of wall-clock times.

Number of processors	DDS	A	G	S	Wall-clock time (sec)
8	1	1	1	8	7809.3
	2	2	1	4	8102.4
	3	4	1	2	8980.5
	4	8	1	1	6483.2
16	1	1	1	16	5694.7
	2	2	1	8	5063.7
	3	4	1	4	7049.0
	4	8	1	2	6104.3

A / G / S = Number of processors devoted to angular/group/spatial domain

These results indicate that our predictive model is very effective. It accurately predicts the most effective DDS in a very short amount of time, i.e. 5% and 1% of the actual computation time for estimating PPI and CPCM, respectively.

Conclusions

An expert system for preparing an effective mesh distribution for the S_N method has been developed and verified. This system consists of two main parts: 1) generation of an effective mesh distribution in a serial environment, common for both serial and parallel calculations, and 2) selection of an effective DDS for parallel computing. The system has been successfully tested within the PENTRAN code system for simulating the VENUS-3 experimental facility. Our analysis has demonstrated that the expert system can reduce user's time and effort for preparing an effective particle transport simulation in both serial and parallel environments.

References

1. A. Patchimpattapong, A. Haghighat, "Developing an Expert System for Preparing an Effective Mesh Distribution for the S_N Method in the Parallel Environment," *Proceedings of the 12th ANS Radiation Protection and Shielding Division Topical Meeting (RPSD 2002)*, Santa Fe, NM, April 14-18, CD (2002).
2. A. Patchimpattapong, A. Haghighat, "An Expert System for Automatic Mesh Generation for S_N Particle Transport Method in Parallel Environment," *the 11th International Symposium on REACTOR DOSIMETRY (ISRD 2002)*, Brussels, Belgium, August 18-23 (2002).
3. A. Patchimpattapong, A. Haghighat, "Testing an Expert System for Selection of Mesh and Domain Decomposition of Parallel S_N Method," *Proceedings of the 2003 ANS Topical Meeting in Mathematics and Computations (M&C 2003)*, Gatlinburg, TN, April 6-11, CD (2003).
4. G. E. Sjoden, A. Haghighat, "PENTRAN – A 3-D Cartesian Parallel S_N Code with Angular, Energy, and Spatial Decomposition," *Proc. Joint Int. Conf. Mathematical Methods and Supercomputing for Nuclear Applications*, Saratoga Springs, New York, October 5-9, Vol. 1, pp. 553-562 (1997).
5. L. Leenders, *LWR-PVS BENCHMARK EXPERIMENT VENUS-3 (with Partial Length Shielded Assemblies)*, SCK.CEN, Nuclear Research Department, Mol, Belgium (1988).
6. AutoDesk, *AutoCAD 2002 User's Guide* (2002).
7. J. E. White et al., "BUGLE-96: A Revised Multigroup Cross Section Library for LWR Applications Based on ENDF/B-VI Release 3," *Proc. Tpl. Mtg. Radiation Protection and Shielding*, No. Falmouth, Massachusetts, Vol. 2, pp. 1071 (1996).
8. A. Haghighat, G. E. Sjoden, V. N. Kucukboyaci, "Effectiveness of PENTRAN's Unique Numerics for Simulation of the Kobayashi Benchmarks," *Progress in Nuclear Energy*, **39**, n 2, pp. 191-206 (2001).