

Massively Parallel, Three-Dimensional Transport Solutions for the k -Eigenvalue Problem

Gregory G. Davidson*, Thomas M. Evans, and Joshua J. Jarrell

Oak Ridge National Laboratory
P.O. Box 2008 Oak Ridge, TN 37831
davidsongg@ornl.gov; evanstm@ornl.gov; jarrelljj@ornl.gov

Rachel N. Slaybaugh

Department of Engineering Physics
University of Wisconsin-Madison
147 Engineering Research Bldg., 1500 Engineering Drive, Madison, WI 53706
slaybaugh@wisc.edu

ABSTRACT

We have implemented a new multilevel parallel decomposition in the Denovo discrete ordinates radiation transport code. In concert with Krylov subspace iterative solvers, the multilevel decomposition allows concurrency over energy in addition to space-angle, enabling scalability beyond the limits imposed by the traditional KBA space-angle partitioning. Furthermore, a new Arnoldi-based k -eigenvalue solver has been implemented. The added phase-space concurrency combined with the high-performance Krylov and Arnoldi solvers has enabled weak scaling to $O(100K)$ cores on the Jaguar XT5 supercomputer. Furthermore, the multilevel decomposition provides sufficient concurrency to scale to exascale computing and beyond.

Key Words: Arnoldi, Krylov, k -eigenvalue, multilevel decomposition

1 INTRODUCTION

Predictive nuclear energy simulations will involve the coupled modeling of many physical regimes, including Boltzmann transport, and will require tremendous computational resources. Experience in the Department of Energy (DOE) Advanced Simulation and Computing program, astrophysics, and nuclear energy have demonstrated that, in coupled-physics calculations, a 3-D Boltzmann transport solver will generally require the vast majority of computational resources, in terms of both memory and operations, because of the seven-dimensional phase-space (location, velocity [energy + angle], and time). For a nuclear reactor simulation, the scale of the problem is large: 5 orders of magnitude in space and 10 in neutron energy. A transport solver that incorporates a resolved discretization for all scales using current discrete models would require 10^{17-21} degrees of freedom (DOF) for a single timestep, which is beyond even exascale computational resources. This problem size precludes, for the time being, a single integrated ab initio computational approach. Instead, variations of current multilevel techniques will continue,

*Corresponding Author

with the new objective being to make the process more consistent and, subsequently, more predictive.

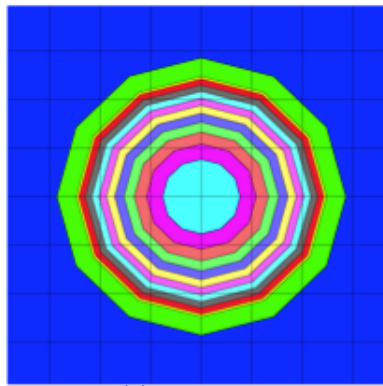
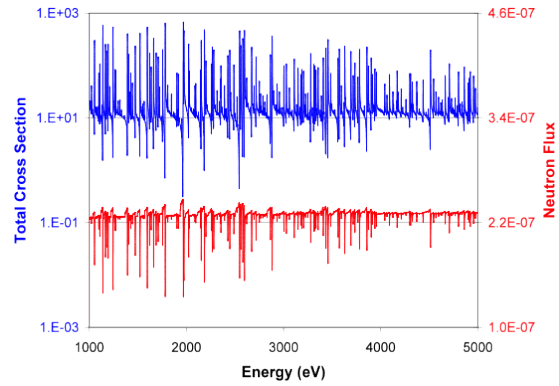
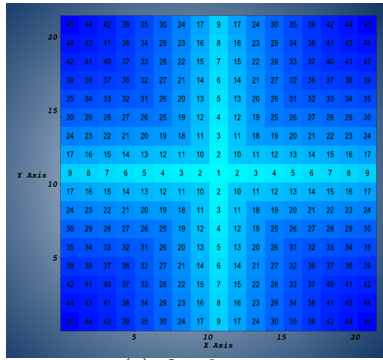
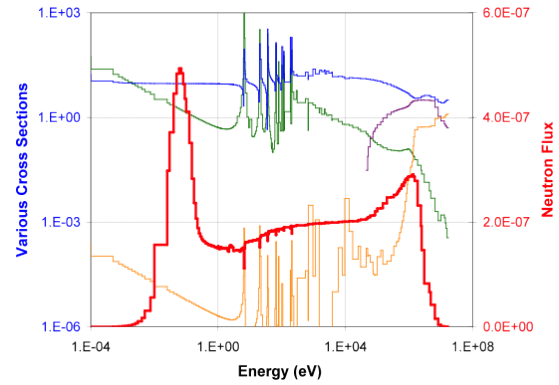
Present reactor transport methods use an *inconsistent* three-level homogenization approach, often utilizing distinct simulation codes, in modeling radiation transport in the core of a nuclear reactor. Figure 1 shows the spatial and energy domains, respectively, of this multiscale challenge: α , use of a fine mesh in 1-D cylindrical geometry of an approximate small subset (pincell) of the reactor core with a first-principles representation of the energy spectrum; β , use of a coarser mesh with a 2-D transport solution in a larger subset (lattice) of the core with grouped representation of the energy spectrum provided by the previous step; and γ , use of a very coarse mesh in a 3-D diffusive transport of neutrons in the full homogenized core of the reactor with a very coarse representation of the energy spectrum provided by the previous step. The first two steps (α, β) require 10^7 – 10^8 DOF with 10^2 – 10^3 independent calculations, each on single-processor machines. Recent work at Oak Ridge National Laboratory (ORNL) has demonstrated that, with moderate computational resources, this can be reduced to an *inconsistent* two-step approach, where step (A) utilizes the energy fidelity of α with the spatial domain of β and step (B) uses the energy fidelity of β and the spatial domain of γ [1, 2]. In this approach, each step would require 10^{11} – 10^{13} DOF per step with 10^2 independent high-order (A) calculations for every timestep.

The solution of the first-order form of the Boltzmann transport equation requires multiple wavefront solutions over each discrete angle. The wavefront solver imposes a fundamental limitation on the scalability of the algorithm because of the upstream dependencies of downstream regions. Various approaches to this problem over the last decade have yielded mixed results. The Koch-Baker-Alcouffe (KBA) [3] algorithm for structured 3-D grids is very efficient in that the transport operator can be inverted in a single solve. However, the scalability is limited by communication latency such that this algorithm is limited by problem size and generally will not scale to $O(100K)$ cores. Parallel-Block-Jacobi methods have been applied to both structured and unstructured grids yielding excellent weak-scaling results [4]. But, these methods are less efficient in terms of memory and iteration count when compared with direct wavefront algorithms. Graph-based direct methods have been investigated on unstructured grids, but these algorithms have significant scaling limitations [5].

To reduce today’s three-level approach to consistent single- or two-level schemes, the orders-of-magnitude increase in fidelity of each of the steps will require substantially more computational resources than present algorithms utilize. Therefore, novel approaches to parallelize the transport equation must be developed that can overcome the wavefront limitation and take full advantage of the complete computational resources.

A parallel decomposition over energy groups and an Arnoldi-based k -eigenvalue solver have been developed for the Denovo code. Denovo is a 3-D discrete ordinates (S_N) multigroup radiation transport code for radiation shielding and reactor physics applications under active development at ORNL [6, 7]. The new energy decomposition is in addition to the KBA-based spatial domain decomposition already present in Denovo. This multilevel decomposition allows parallel scaling up to hundreds of thousands of cores. Additionally, the new Arnoldi solver can solve k -eigenvalue problems with many fewer mesh sweeps than the traditional power iteration method.

In this paper we will describe Denovo’s multilevel parallel decomposition and the solver technologies that complement it. Solvers for the multigroup S_N equation are described in § 2. In § 3 we show the limitations of KBA space-angle parallelism on leadership class platforms. Finally,


 (a) α = pincell

 (b) α energy spectrum

 (c) β = lattice

 (d) β energy spectrum

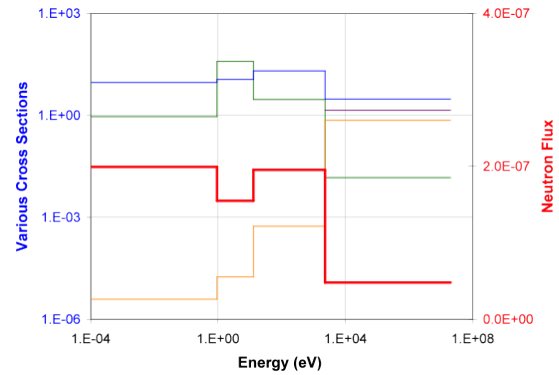
 (e) γ = core

 (f) γ energy spectrum

Figure 1: Three levels of reactor geometries and energy structures.

we describe a multilevel parallel decomposition that overcomes these shortcomings in § 4, and in § 5 we summarize the complete set of solver options available in Denovo. Results that show the efficiency of these methods are shown in § 6.

2 MULTIGROUP SOLVERS

To see how a multilevel energy decomposition can be beneficial and how it will be implemented, we briefly review the fundamental solver strategies in Denovo; Ref. [6] can be consulted for full details. The multigroup S_N equations can be written in operator form as

$$\mathbf{L}\psi = \mathbf{M}\mathbf{S}\phi + q_e, \quad (\text{fixed source}) \quad (1)$$

$$\mathbf{L}\psi = \mathbf{M}\mathbf{S}\phi + \frac{1}{k}\mathbf{M}\chi\mathbf{f}^T\phi. \quad (\text{eigenvalue}) \quad (2)$$

The state of these equations is defined in angular flux moments, ϕ , that are related to the discrete angular flux through

$$\phi = \mathbf{D}\psi, \quad (3)$$

where \mathbf{D} is the discrete-to-moment operator that integrates discrete angles into angular flux moments using quadrature rules. \mathbf{L} is the first-order linear differential transport operator, \mathbf{M} is the moment-to-discrete operator that projects angular flux moments into discrete angle space, and \mathbf{S} is the group-to-group scattering matrix. In the eigenvalue form of the equation, \mathbf{f}^T is the row vector of fission cross sections, χ is the fission spectrum vector, and k is the largest eigenvalue.

Operating by $\mathbf{D}\mathbf{L}^{-1}$, defining $\mathbf{T} = \mathbf{D}\mathbf{L}^{-1}$, and rearranging terms, the fixed-source problem is

$$(\mathbf{I} - \mathbf{TMS})\phi = q, \quad (4)$$

where $q = \mathbf{T}q_e$. Similarly, the eigenvalue problem becomes

$$(\mathbf{I} - \mathbf{TMS})\phi = \frac{1}{k}\mathbf{TMF}\phi, \quad (5)$$

where $\mathbf{F} = \chi\mathbf{f}^T$ is the full-rank fission matrix. The operator \mathbf{L}^{-1} can be formed into a lower-triangular system if one sweeps the space-angle grid in the direction of neutron travel. The resulting transport *sweep* is the operation that is parallelized using the KBA algorithm. This matrix is never formed in practice; only the action of the operator on a vector, $y = \mathbf{L}^{-1}v$, is required.

The traditional method for solving Eq. (4) is Gauss-Seidel iteration,

$$(\mathbf{I} - \mathbf{TMS}_{gg})\phi_g^{(l+1)} = q_g + \mathbf{TM} \left(\sum_{g'=0}^{g-1} \mathbf{S}_{gg'}\phi_{g'}^{(l+1)} + \sum_{g'=g+1}^G \mathbf{S}_{gg'}\phi_{g'}^{(l)} \right), \quad (6)$$

which is the same as solving G one-group space-angle problems per Gauss-Seidel iteration. We refer to these one-group problems as *within-group* equations, and they have the following form:

$$(\mathbf{I} - \mathbf{TMS}_{gg})\phi_g = \bar{q}_g, \quad (7)$$

where \bar{q}_g is an effective group source that includes all in-scattering, fission, and/or external sources for the group. When using Gauss-Seidel iteration over energy, the within-group solves represent a set of inner iterations. Denovo provides several solvers (Krylov, source iteration) for the within-group equations, and Diffusion Synthetic Acceleration (DSA) is available as a preconditioner for the Krylov options. Obviously, when \mathbf{S} is lower triangular (indicating down-scattering only) the solution converges in one Gauss-Seidel iteration. However, when \mathbf{S} is energy dense, the solution can converge very slowly. This structure occurs in the low-energy region of the spectrum where neutrons undergo Maxwellian upscattering. A typical \mathbf{S} for a 27-group set of common reactor materials is illustrated in Fig. 2.

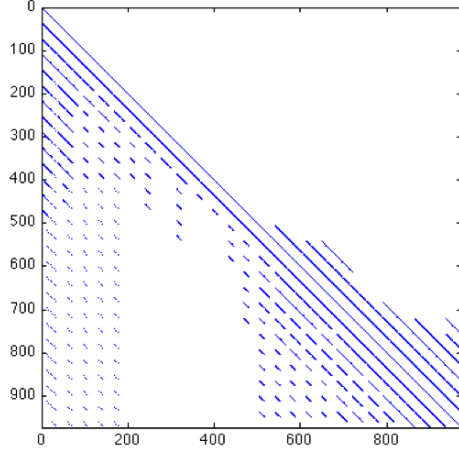


Figure 2: Sparsity plot of the scattering matrix for a problem containing iron, graphite, and heavy water. The data has 27 energy groups, and the matrix is lower-triangular through group 14. The full matrix is dimensioned over energy-space-angle with angle represented by Legendre moment expansion.

The standard way to solve Eq. (5) is power iteration. We start by defining an energy-independent eigenvalue,

$$\Gamma = \mathbf{f}^T \phi. \quad (8)$$

Then, Eq. (5) can be written

$$\mathbf{A}\Gamma = k\Gamma, \quad (9)$$

with

$$\mathbf{A} = \mathbf{f}^T (\mathbf{I} - \mathbf{TMS})^{-1} \mathbf{TM}\chi. \quad (10)$$

Solving by power iteration proceeds as follows

$$\Gamma^{(l+1)} = \frac{1}{k} \mathbf{A}\Gamma^{(l)}. \quad (11)$$

The operation $\mathbf{A}\Gamma^k$ necessarily involves solving multigroup problems with the same form as Eq. (4),

$$(\mathbf{I} - \mathbf{TMS})y^{(l)} = \mathbf{TM}\chi\Gamma^{(l)}. \quad (12)$$

Both Gauss-Seidel iteration for fixed-source problems and power iteration (with Gauss-Seidel inners) are provided in the current version of Denovo. While the implementation in Denovo is enhanced by applying Transport Two-Grid (TTG) acceleration to the Gauss-Seidel iterations [6] and using GMRES(m) on the inner one-group solves, a fundamental limitation of these solvers is that they are parallelizable only over space-angle variables. The recursive nature of Gauss-Seidel prevents effective parallelization over energy. One could use parallel block-Jacobi iteration, but this option results in solvers that are too inefficient to be of practical use on most problems.

Instead, the Krylov solver framework in Denovo that is currently used in the inner one-group space-angle solves has been expanded to include energy. Including energy in the Krylov vectors enables the following benefits:

- The energy variable is decoupled allowing groups to be solved independently.
- Krylov subspace iteration is more efficient and robust than Gauss-Seidel iteration.
- Preconditioning a Krylov iteration is generally more robust and stable than Gauss-Seidel acceleration.

Furthermore, including energy in the Krylov vector does not invalidate any of the existing sweep mechanics that are already implemented in Denovo.

For multigroup fixed-source problems in the form of Eq. (4), application of a Krylov method requires the following two steps:

1. A full energy-space-angle sweep of the right-hand side source,

$$q = \mathbf{T}\hat{q}, \quad (13)$$

where \hat{q} is an effective source that could be an external source (q_e) in the case of true fixed-source problems, or it could be a fission source iterate when nested inside power iteration.

2. A full energy-space-angle sweep each Krylov iteration to calculate the action of the operator on the latest iterate,

$$y^{(l)} = (\mathbf{I} - \mathbf{TMS})v^{(l)}, \quad (14)$$

where $v^{(l)}$ is the Krylov vector in iteration l . We note that this vector is dimensioned $v^{(l)} \equiv \{v_{g,c,n,l,m}^{(l)}\}$ where g is the energy group, c is the cell index, n is the spatial unknown index in the cell, and (l, m) are the spherical harmonic moment indices.

For eigenvalue problems we have implemented an Arnoldi Krylov subspace solver using the Trilinos Anasazi package [8] that can (1) take full advantage of the energy parallelism and (2) be more efficient than power iteration. Arnoldi iteration requires the eigenproblem to be written in standard form,

$$\mathbf{A}x = \lambda x. \quad (15)$$

Arnoldi iteration can be implemented with either an energy-dependent or energy-independent eigenvector as follows:

$$\mathbf{A}\phi = k\phi, \quad \mathbf{A} = (\mathbf{I} - \mathbf{TMS})^{-1}\mathbf{TMF}, \quad (\text{energy dependent}) \quad (16)$$

$$\mathbf{A}\Gamma = k\Gamma, \quad \mathbf{A} = \mathbf{f}^T(\mathbf{I} - \mathbf{TMS})^{-1}\mathbf{TM}\chi. \quad (\text{energy independent}) \quad (17)$$

In either case, the implementation of an Arnoldi iteration requires a matrix-vector multiply at each Krylov iteration of the form

$$y^{(l)} = \mathbf{A}v^{(l)}. \quad (18)$$

For the energy-dependent case we have

$$z^{(l)} = \mathbf{TMF}v^{(l)}, \quad (\text{matrix-vector multiply and sweep}) \quad (19)$$

$$(\mathbf{I} - \mathbf{TMS})y^{(l)} = z^{(l)}. \quad (\text{fixed-source solve}) \quad (20)$$

Similarly, for the energy-independent eigenvector the steps are

$$z^{(l)} = \mathbf{TM}\chi v^{(l)}, \quad (\text{matrix-vector multiply and sweep}) \quad (21)$$

$$(\mathbf{I} - \mathbf{TMS})y^{(l)} = z^{(l)}, \quad (\text{fixed-source solve}) \quad (22)$$

$$y^{(l)} \leftarrow \mathbf{f}^T y^{(l)}. \quad (\text{dot-product}) \quad (23)$$

Both methods require a fixed-source solve each iteration. We consider both the energy-dependent and independent approaches because we are uncertain a priori which method will be optimal for a given problem. The energy-dependent approach allows parallelization of the eigenvalue solve across energy at the expense of a much larger eigenvector. The energy-independent approach allows only energy-domain parallelization over the fixed-source solve, and the eigenvalue solve is parallel only over space-angle. However, this decomposition may be more efficient because the eigenvector is smaller, especially when work is dominated by the inner multigroup fixed-source solve.

3 SWEEP-BASED PARALLELISM

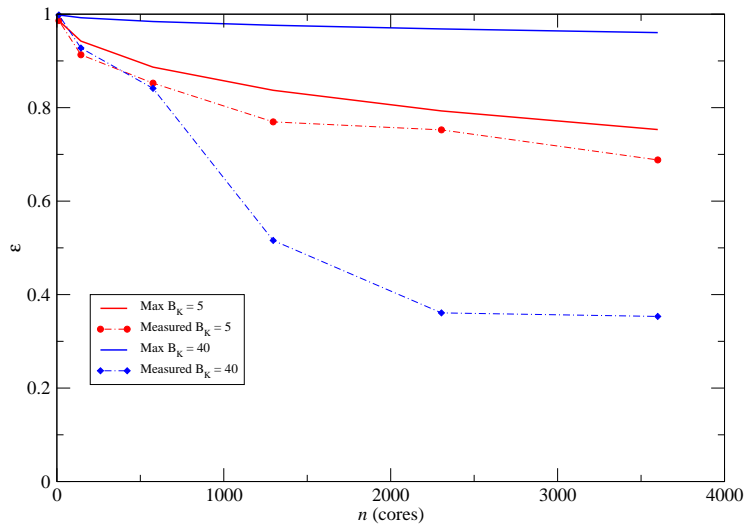
Denovo [6] uses the KBA wavefront algorithm to parallelize the space-angle transport sweeps shown in Eqs. (4) and (5). Unfortunately, KBA alone provides insufficient parallelism on very large systems. The following analysis will demonstrate its shortcomings. The theoretical efficiency of KBA, ignoring machine latency, is

$$\varepsilon_{\max} = \frac{2MK}{2MK + K_b(I/I_b + J/J_b - 2)} = \frac{2MB_K}{2MB_K + P_I + P_J - 2}, \quad (24)$$

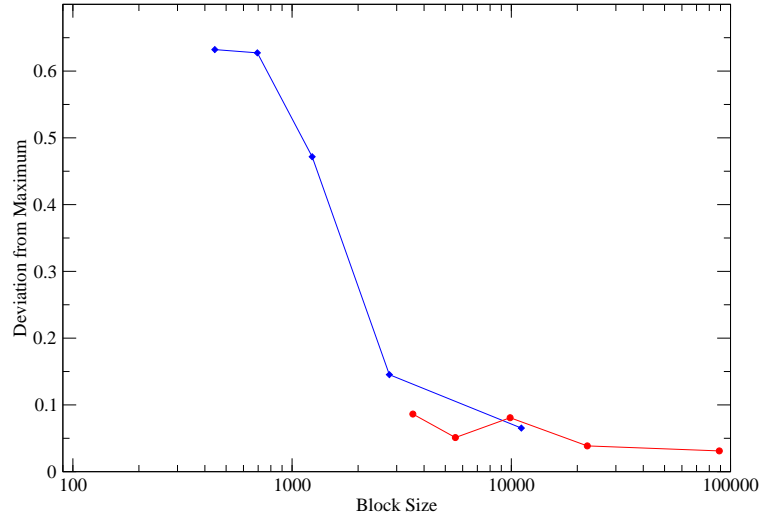
where (I, J, K) are the number of cells in (x, y, z) , respectively. The number of cells per domain in (x, y) is given by (I_b, J_b) , and K_b is the number of cells per on-processor block in the z dimension. Then, P_I and P_J are the number of processors in the x and y directions, respectively; B_K is the number of blocks in the z direction on each domain; and M is the number of angles per octant.

We have done strong scaling studies on Jaguar¹ with a $400 \times 400 \times 400$ cell mesh. The results are shown in Fig. 3. Figure 3a shows that while a very high ($> 90\%$) maximum theoretical parallel

¹Jaguar XT5 supercomputer at the Oak Ridge Leadership Computing Facility (OLCF).



(a)



(b)

Figure 3: Denovo strong scaling results on Jaguar; (a) strong scaling with $B_K = 40$ blocks (blue) and $B_K = 5$ blocks (red) and (b) the deviation from the theoretical maximum as a function of number of cells per block.

efficiency is estimated, this value is impossible to achieve in practice. To get high theoretical efficiencies, the number of cells per block ($I_b \times J_b \times K_b$) becomes very small. This makes sense in the abstract because work is passed to subsequent blocks more rapidly. However, in reality the latency per message quickly overwhelms the theoretical prediction. When the number of cells per block becomes very small, data spend more time waiting in MPI message queues than working to solve the block. This effect is illustrated in Fig. 3b. The deviation between the theoretical prediction and the measured efficiency becomes small as the block size grows. In summary, high theoretical efficiencies requiring small numbers of cells per block cannot be achieved, but theoretical efficiencies that result from larger block sizes can be realized.

The repercussions from this analysis are obvious; namely, the full extent of computational resources available on Jaguar cannot be effectively utilized. The best efficiencies are obtained when the block size can be set greater than ~ 1500 . Thus, for any given problem, the maximum number of cores is predetermined by the minimum block size. Even a 500M cell problem will be limited to 15,000–20,000 cores under these restrictions. To utilize the full resources of Jaguar we must find additional variables to parallelize. Using the advanced solvers in Denovo, a multilevel decomposition over energy will provide the necessary parallelism.

4 MULTILEVEL PARALLEL DECOMPOSITIONS

Having described the multigroup solvers in § 2 and discussed the limitations of parallelizing only the space-angle sweep, we now explain the parallel implementation of Denovo’s energy-space-angle decomposition. The multilevel energy-space decomposition is illustrated in Fig. 4. In this decomposition, space is partitioned into *blocks*. Energy is partitioned into *sets*. Each set contains the full mesh (all of the blocks) such that KBA sweeps never cross set boundaries. Every (block, set) combination is termed a *domain*. The total number of domains is currently the same as the number of MPI processes in a parallel job. The old Denovo space-angle decomposition can be thought of as a single-set energy decomposition over $P_I \times P_J$ blocks. A benefit of this energy-space partitioning is that all of the solvers described in § 2 can be implemented in the new decomposition using Denovo’s existing space-angle sweep machinery.

To solve the multigroup equations, Eq. (14) implies a matrix-vector multiply of the form

$$s_g = \mathbf{S}_{g0}v_0 + \mathbf{S}_{g1}v_1 + \dots + \mathbf{S}_{gG}v_G. \quad (25)$$

Instead of communicating all of the groups to each set so that the matrix-vector multiply can be completed locally, we replicate the source vector s . Then, each set performs its part of the matrix-vector multiply followed by a global reduction. The global reduction is performed using a communicator that relates all blocks with the same index across sets. After the global reduction, each set has the complete source vector s , even though it will utilize only the components of s that are local to the set.

After calculating the sweep source s , the sweeps for each local group on a set can be performed without any intersets communication. The sweeps require only communication between blocks within a set. The only exception to this rule is when the energy-independent version of Arnoldi is applied. In this case, the eigenvector must be summed across all sets in a manner analogous to that described above for calculating the sweep source.

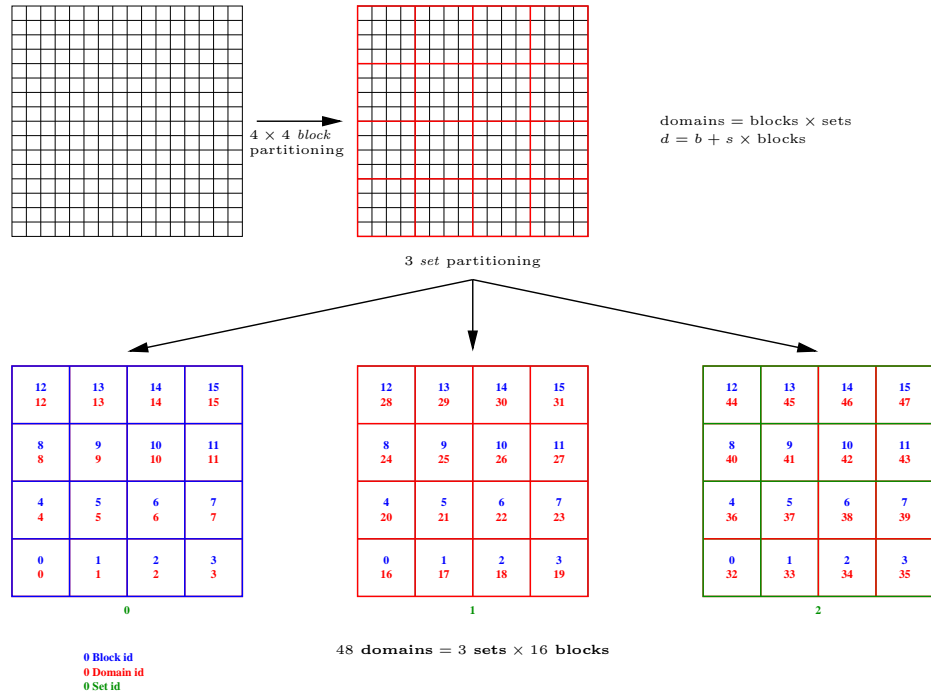
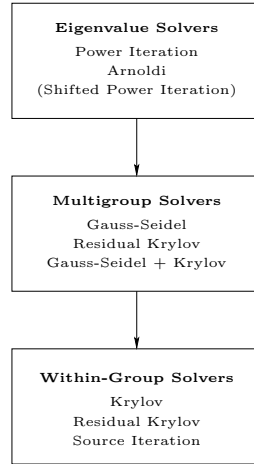


Figure 4: Multilevel energy-space decomposition in Denovo. This example has 3 *sets*, each containing 16 *blocks*, resulting in 48 total *domains*. The block (blue), set (green), and domain (red) IDs are indicated by b , s , and d , respectively.

5 DENOVO SOLVER TAXONOMY

Having reviewed the new solvers and parallel decompositions in Denovo, we will now summarize the complete set of solver options that are available in the code. We have separated the taxonomy into *Within-Group Solvers*, *Multigroup Solvers*, and *Eigenvalue Solvers*. These are arranged in levels according to the following:



Fixed-source problems are solved using multigroup solvers.

5.1 Within-Group Solvers

The within-group solvers are used to solve Eq. (7). All of the within-group solvers are parallelized over space because, by definition, they require no coupling between energy groups. Thus, they operate only within a set, not across sets.

Solver	Preconditioning	Parallelization
Direct Krylov	DSA	interblock KBA
Residual Krylov	DSA	interblock KBA
Source Iteration		interblock KBA

5.2 Multigroup Solvers

The multigroup solvers are used to solve Eq. (4). They can be used independently to solve fixed-source problems, or they can be used in the inner iterations of eigenvalue problems. The multigroup solvers are parallelized over space-angle (blocks) and energy (sets), although not all solvers support energy parallelization. For example, the Gauss-Seidel solver does not support parallelization over energy.

Solver	Preconditioning	Parallelization
Gauss-Seidel	TTG	single-set energy partitioning
Gauss-Seidel/Krylov		Gauss-Seidel over downscatter groups replicated on each set, Krylov iteration over upscatter groups using multiset energy partitioning
Krylov	Multigrid Energy, LU [†]	multiset energy partitioning
[†] in development		

5.3 Eigenvalue Solvers

The eigenvalue solvers solve Eq. (5). The parallelization is largely determined by the choice of multigroup solver. Some eigenvalue solvers can solve both energy-dependent and energy-independent eigenvectors, and this choice dictates the parallelization strategy.

Solver	Eigenvector	Multigroup Solvers
Power Iteration	energy independent	Gauss-Seidel, Krylov, and Gauss-Seidel/Krylov
Arnoldi	energy independent/ energy dependent	Krylov and Gauss-Seidel/Krylov
Rayleigh Quotient Iteration [†]	energy dependent	Krylov
[†] in development		

6 RESULTS

Each of the solvers and parallel algorithms has been independently verified through the Denovo test suite. To test the performance on leadership-class computing hardware, we have chosen two large-scale test problems. The first is a c5g7 benchmark proposed by the Organisation for Economic Cooperation and Development (OECD) [11]. These results are discussed in § 6.1. The second test problem is a whole-core, pin-homogenized PWR test problem adapted from an Électricité de France (EDF) benchmark. This is presented in § 6.2. All calculations were performed on the Jaguar XT5 computer at the OLCF.

6.1 C5G7 Benchmark

The c5g7 benchmark problem, presented in Ref. [11], is a three-dimensional, seven group MOX fuel assembly surrounded by a water reflector. We have used the unrodded configuration in this test. Each homogenized fuel pin was divided into $11 \times 11 \times 90$ cells, which gives (together with the addition of a water reflector as described in the benchmark geometry) a total of 14,689,440 cells. The geometry is shown in Figure 5. This benchmark problem contains seven energy groups, with four upscatter groups, and was executed using a Gauss-Legendre product quadrature set with six azimuthal and eight polar angles per octant, for a total of 384 angles. P_0 scattering and a linear discontinuous spatial discretization was used, which results in a total of 157,940,858,880 DOF. Each of the solvers converged to within 30–35 pcm of the benchmark k_{eff} of 1.14308.

In Table I, we show the run time performance of the power iteration method with Gauss-Seidel, power iteration Gauss-Seidel/Krylov, and power iteration with Krylov. We also examine the

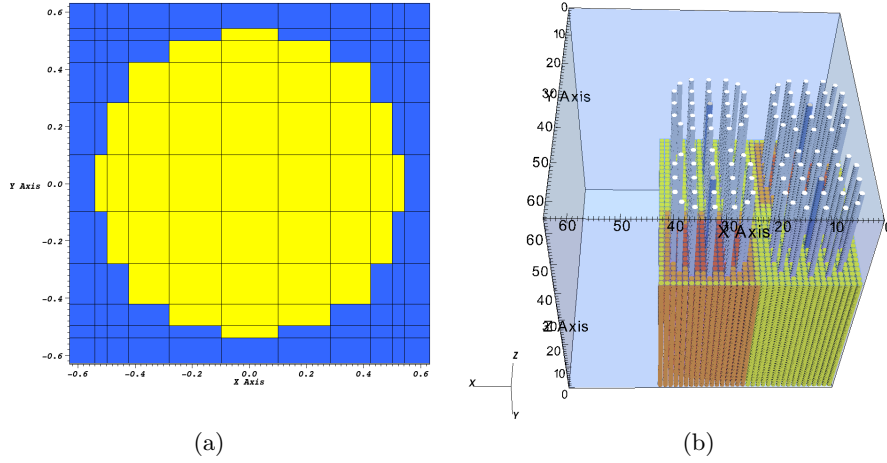


Figure 5: Geometry Specification for the c5g7 benchmark: (a) mesh over a pincell, (b) full assembly.

Arnoldi solver, both with energy-dependent (EDE) and energy-independent (EIE) eigenvectors, and with both Gauss-Seidel/Krylov and Krylov multigroup solvers. The performance of each of

Table I: Solver comparisons for the c5g7 benchmark problem. In each case the eigenvalue was converged to 1×10^{-5} . Each problem had 157,940,858,880 DOF.

Solver	Blocks	Sets	Domains	Solver Time (min)
PI/GS + TTG	11,664	1	11,664	392.22
PI/GS+GMRES	11,664	4	46,656	33.51
PI/GMRES	11,664	7	81,648	21.94
Arnoldi (EDE)/GS+GMRES	11,664	4	46,656	8.25
Arnoldi (EDE)/GMRES	11,664	7	81,648	5.39
Arnoldi (EIE)/GS+GMRES	11,664	4	46,656	8.21
Arnoldi (EIE)/GMRES	11,664	7	81,648	4.69

the solvers is shown in Fig. 6.

From Table I and Fig. 6, we see that a significant improvement is gained, both in wall-clock time and in CPU-hours when moving from the power iteration with a Gauss-Seidel inner solve to power iteration with a multigroup Krylov solve. The Arnoldi-based k -eigenvalue solvers are even more efficient.

6.2 EDF Benchmark

The second test is a generic whole-core pressurized water reactor (PWR) model. This model was originally developed as a whole-core, pin-homogenized, two-group benchmark problem by

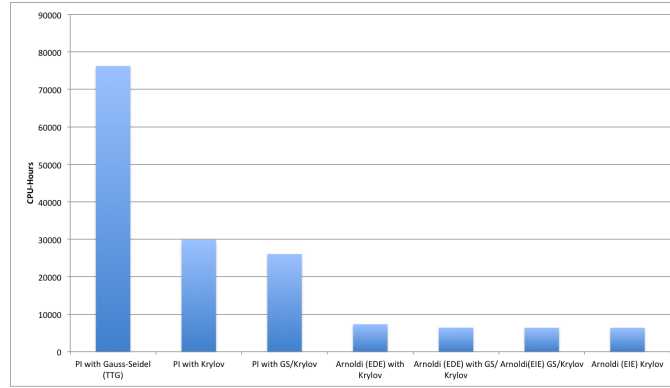


Figure 6: Time to solution in CPU-Hours for each of the solvers. Both the inner and outer tolerance is 10^{-5} . The k_{eff} for all of the solvers converged to within 30–35 pcm of the benchmark.

Électricité de France (EDF) working with the University of Florida [9]. Although the ultimate objective is to perform whole-core, pin-resolved, 3-D transport simulations, a pin-homogenized problem serves as a useful and realistic benchmark problem. Denovo is used to solve Eq. (5) in order to calculate the k -eigenvalue and the scalar flux throughout the core. Using the scalar flux, the pin power distribution, fission source, and group-wise power distributions can be analyzed. The ability to solve pin-homogenized, whole-core problems with transport, as opposed to diffusion or other low-order approximations, is the first step towards fully predictive reactor core modeling and simulation. To achieve first-principles predictive capability, each pin would be fully resolved in the whole-core 3-D model. This objective cannot be approached until we have demonstrated the ability to solve pin-homogenized 3-D reactor problems with full transport.

The model is a generalization of a Westinghouse PWR-900 core that has a core height of 4.2 m, an assembly height of 3.6 m, and a lattice pitch of 1.26 cm. The core features 289 assemblies, of which 157 are fuel and 132 are in the reflector. Each assembly contains a 17×17 array of homogenized fuel pins that are arranged with 1/4 lattice symmetry as shown in Fig. 7a. Three different fuel enrichments ranging from 1.5% to 3.25% are used in the assemblies. We implemented this model in the TRITON sequence of SCALE [10] to generate 44-group pin-homogenized cross sections to supplement the two-group set defined in the benchmark. Each set of pin-homogenized cross sections contains 135 unique materials (45 pins at 3 levels of enrichment). The 2-D radial view of the core is shown in Fig. 7b.

For Denovo, the model was discretized into $2 \times 2 \times 700$ spatial cells per homogenized fuel pin, resulting in a total mesh size of 233,858,800 cells. An angular quadrature containing 168 angles, P_0 scattering (one angular moment), and step-characteristics spatial differencing was used for all calculations. These parameters yielded 39,288,278,400 DOF per energy group.

Table II compares the different solvers for the two-group version of the PWR-900 benchmark. The 17,424 spatial domains used for the standard Gauss-Seidel (GS) solver represent the maximum number of cores that could be effectively used for this problem. To use more computing resource the multilevel decomposition is required. These results show that the multigroup Krylov solvers are very effective, and they allow an efficient multilevel decomposition. However, because this is a coarse energy benchmark, we cannot make dramatic conclusions about the performance

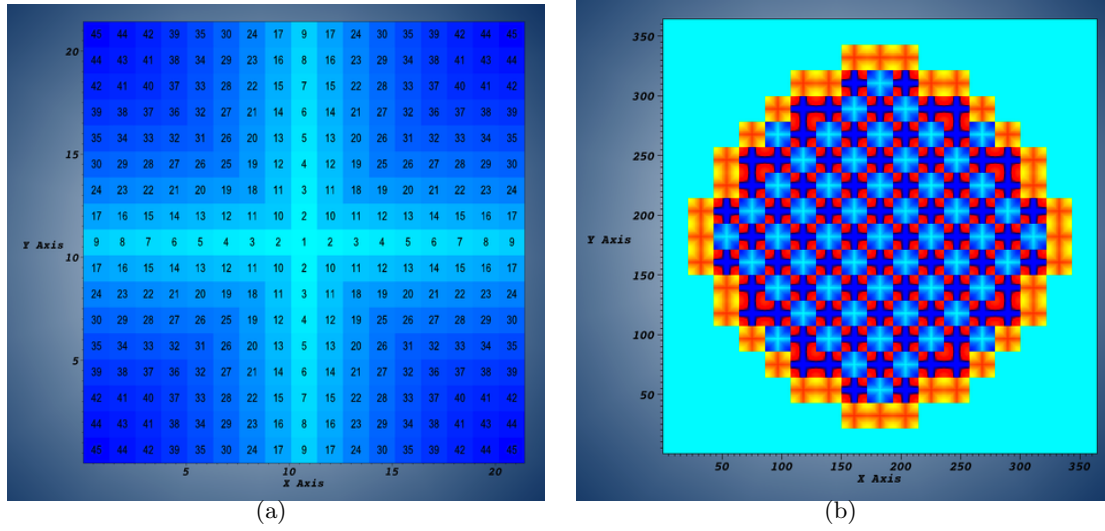


Figure 7: (a) PWR-900 17×17 pin fuel assembly. The pins have been homogenized into 45 unique materials in each assembly. All assembly enrichments have the same $1/4$ symmetry pattern. (b) 2-D radial cut of the reactor core. The low-enrichment assemblies are light blue, the medium enrichment assemblies are red/blue, and the high-enrichment assemblies are yellow/orange.

of the solvers on real problems of interest.

Table II: Solver comparisons for 2 group version of the PWR-900 benchmark problem. In each case the eigenvalue was converged to 1×10^{-3} . Each problem had 78,576,556,800 DOF.

Solver	Blocks	Sets	Domains	Solver Time (min)
PI/GS + TTG	17,424	1	17,424	11.00
PI/GMRES	10,200	2	20,400	3.03
Arnoldi (EDE)/GMRES	10,200	2	20,400	2.05

To show the validity of this approach for fully resolved reactor problems, we must investigate the performance of the parallel algorithm and solvers on problems with more groups. We have run the same PWR-900 problem with 44 groups, resulting in 1,728,684,249,600 DOF. Four versions of the problem are executed: power iteration (PI) with GS plus GMRES, PI with GMRES, Arnoldi with GS plus GMRES, and Arnoldi with GMRES. With the multigroup GS plus GMRES option, multilevel partitioning is performed only over the upscattering region of the scattering matrix. For this option, the lower-triangular downscatter region is replicated on each set. With the multigroup GMRES option, multilevel partitioning is performed over the whole scattering matrix, regardless of its structure. Each problem was run with 10,200 blocks and 11 sets, resulting in 112,200 domains.

The results of these runs are given in terms of the weak-scaling parallel efficiency that is defined

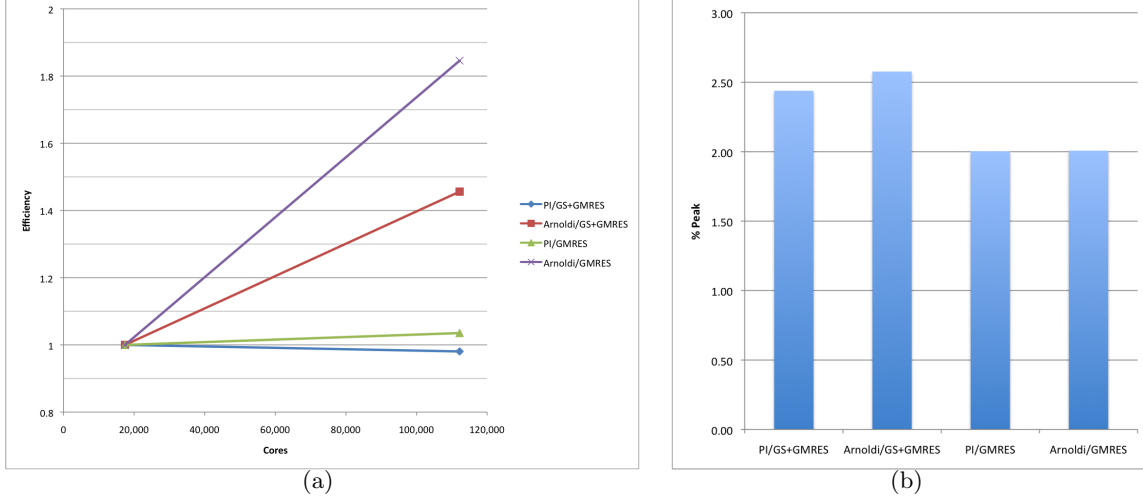


Figure 8: (a) Weak scaling efficiencies for the 44-group version of the PWR-900 benchmark, and (b) peak machine efficiencies on Jaguar XT5.

as

$$\epsilon = \frac{\tau_{\text{ref}}}{\tau_{\text{P}}} \left(\frac{\text{DOF}_{\text{P}}}{\text{DOF}_{\text{ref}}} \right), \quad \tau = t \times N_p. \quad (26)$$

Here, t is the wall-clock time and N_p is the number of processors (cores). The “ref” subscript denotes a reference problem run, and the “P” subscript refers to the target problem. Weak-scaling curves for the four multilevel solvers are shown using the PI/GS plus TTG problem as the reference in Fig. 8. These results show that (1) the multilevel parallel decomposition allows scaling up to $O(100K)$ cores and (2) the solvers combined with the multilevel parallel decomposition are generally more efficient than standard PI/GS-based schemes (even with acceleration). In particular, the Arnoldi eigenvalue solver is significantly more efficient than PI, and we expect this efficiency difference to become larger as tighter eigenvalue tolerances are required. In general, reactor design calculations require eigenvalue tolerances of approximately 1×10^{-5} , 2 orders of magnitude tighter than our current 1×10^{-3} tolerance. Early test problems indicate that the efficiency of the Arnoldi solver approaches 6 times the efficiency of PI at these tighter tolerances.

7 CONCLUSIONS

We have implemented a new suite of eigenvalue and multigroup solvers in the Denovo radiation transport code that utilize a multilevel energy decomposition. Using these solvers in concert with the multilevel parallel decomposition allows Denovo to scale to $O(100K)$ processors on leadership-class hardware. Original space-angle parallel methods were fundamentally limited in how much machine resource could be used for a given problem size. The multilevel decomposition overcomes this barrier and should allow scaling to exascale-level platforms and beyond.

ACKNOWLEDGEMENTS

The authors wish to thank Kevin Clarno and Brenden Mervin, ORNL, for help in generating the multigroup cross-section sets. Work for this paper was supported by Oak Ridge National Laboratory, which is managed and operated by UT-Batelle, LLC, for the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

REFERENCES

1. K.T. Clarno, ORNL LDRD Report Tech. Rep. D06-087, Oak Ridge National Laboratory (2007).
2. Z. Zhong et al., Continuous-Energy Multidimensional S_N Transport for Problem-Dependent Resonance Self-Shielding Calculations, *Nuclear Science and Engineering*, **154** pp. 190-201 (2006).
3. R.S. Baker, An S_N Algorithm for the Massively Parallel CM-200 Computer, *Nuclear Science and Engineering*, **128**, pp. 312-320 (1998).
4. K.T. Clarno, Implementation of Generalized Coarse-Mesh Rebalance in NEWTRNX for Acceleration of Parallel Block-Jacobi Transport, *Transactions of the American Nuclear Society*, **97**, (2007).
5. S.D. Pautz, An Algorithm for Parallel S_N Sweeps on Unstructured Meshes, *Nuclear Science and Engineering*, **140**, pp. 111-136 (2002).
6. T.M. Evans et al., DENOVO: A New Three-Dimensional Parallel Discrete Ordinates Code in SCALE, *Nuclear Technology*, **171**, pp. 171-2002 (2010).
7. T.M. Evans, K.T. Clarno, and J.E. Morel, A Transport Acceleration Scheme for Multigroup Discrete Ordinates with Upscattering, *Nuclear Science and Engineering*, **165**, pp. 292-304 (2010).
8. C.G. Baker et al., Anasazi Software for the Numerical Solution of Large-Scale Eigenvalue Problems, *ACM Transactions on Mathematical Software*, **36**, (2009).
9. T. Courau, Specifications of a 3D PWR Core Benchmark for Neutron Transport, Technical Note CR-128/2009/014 EDF-SA.
10. Oak Ridge National Laboratory, SCALE: A Modular Code System for Performing Standardized Computer Analyses for Licensing Evaluations, Version 6.
11. Organisation for Economic Cooperation and Development, Benchmark on Deterministic Transport Calculations Without Spatial Homogenisation: MOX Fule Assembly 3-D Extension Case, NEA/NSC/DOC(2005)16 (2005).