M&C 2017 - International Conference on Mathematics & Computational Methods Applied to Nuclear Science & Engineering, Jeju, Korea, April 16-20, 2017, on USB (2017)

CACTUSOT: A 3D Method of Characteristics Solver in WIMS

J. G. Hosking, B. A. Lindley and P. J. Smith

ANSWERS Software Service, Amec Foster Wheeler, Dorchester, Dorset, UK
glynn.hosking@amecfw.com

Abstract - *Modern 3D core analysis is increasingly looking to detailed transport theory solutions to improve accuracy. The Method of Characteristics (MoC) has been applied to such problems by several authors. However, the use of MoC for 3D core-scale calculations can have very high computational costs. This paper describes the CACTUSOT 3D MoC solver in the WIMS reactor physics code. CACTUSOT includes specialised methods to optimise track configuration and minimise track storage requirements for core calculations. It uses a once-through tracking method which has been shown to improve spatial coverage. The slice-geometry scheme can be used to represent a problem in terms of a series of repeated axial slice types. Track data are generated for individual slice types rather than for the entire problem, meaning that the amount of data to be stored can be significantly reduced. The application of CACTUSOT to core-scale Sodium Fast Reactor and PWR benchmark problems is presented.*

I. INTRODUCTION

The CACTUS Method of Characteristics (MoC) solver was first implemented in the WIMS reactor physics code in the 1970s [1]. Since this time, the solver has been extensively used for reactor physics analyses worldwide. The more-recent development of the CACTUS3D solver has extended the capability to facilitate the modelling of 3D lattice geometries [2].

To meet the demands for ever-increasing modelling accuracy it is now commonly desirable to perform transport theory solutions for whole reactors. One example of where this may be desirable is the modelling of Small Modular Reactors (SMRs). To help meet these goals new capabilities have been added to WIMS, which is developed and marketed by the ANSWERS Software Service of Amec Foster Wheeler. The CACTUSOT solver has been developed for the purpose of performing whole-reactor MoC calculations. CACTUSOT is included in WIMS11, which will be the next quality-assured release of the code.

CACTUSOT contains features to help address the specific difficulties in applying MoC to whole-reactor problems. This paper focuses on the tracking algorithm used within CACTUSOT, as well as a specialized slice-based geometry treatment for reducing RAM requirements. Example CACTUSOT calculations for a Sodium Fast Reactor (SFR) benchmark and a PWR mini-core benchmark are presented.

II. TRACKING ALGORITHM

A fundamental requirement of MoC solvers is the generation of a set of characteristic tracks that provide suitable spatial coverage of the model being considered.

CACTUS is a 2D solver with reflective or translational boundary conditions and uses a cyclic tracking algorithm wherein any track is followed, in two dimensions, until its intercept with a point that is geometrically equivalent to its starting point. Analogous cyclic tracking methods are used by many codes with MoC solution schemes (e.g. [3] and [4]). When using this scheme it is usually possible for tracks to provide suitable spatial coverage. CACTUS3D uses an analogous cyclic tracking algorithm in three dimensions. The CACTUS3D algorithm has been shown to provide suitable track coverage, and hence accurate MoC solutions, for 3D lattice models [2].

However, analysis has indicated that the CACTUS3D algorithm can become unreliable when considering large 3D models. Significant changes in track distributions have been observed following modest changes in tracking parameters, which can lead to poor convergence behaviour of track coverage as the tracking parameters are refined. Unless specialised methods are adopted, similar problems could be exhibited by any 3D MoC solver using cyclic tracking.

To circumvent the problems observed when applying CACTUS3D to large 3D models, CACTUSOT instead uses a once-through tracking algorithm. Here, a set of parallel tracks is defined at each tracking angle. The starting points of the tracks at a given angle are uniformly distributed on the external surfaces of a model. Each track is followed from its starting point until the point at which it next intercepts an external surface; the track is then terminated.

1. Calculation of Track Starting Points

This section describes the calculation of starting positions for CACTUSOT tracks in a full-plan cube model.

For each combination of azimuthal/polar angle, there is a set of parallel tracks that will be incident on the model. The method used for determining the starting points of these tracks depends on the particular azimuthal and polar angles that define their direction of travel.

A. Starting Positions for Tracks Travelling with $0 < \theta < \pi/2$ *and* $0 < \phi < \pi/2$

Initially, a square array of points is considered on the y-z plane at $x = 0$. The y and z separation between adjacent points is taken to be the track separation (s) entered by the user.

Assume that any given point in the array is identified by integer values n and m. The co-ordinate of a point in the array

is therefore

$$p(n,m) = (0, y_0, z_0) = (0, \frac{s}{2} + ns, \frac{s}{2} + ms). \quad (1)$$

An offset of $s/2$ is used to avoid situations where tracks could start at the corner of the model when n and/or m equals zero.

Constraints on n and m are chosen such that

$$-\alpha L_y \leq ns \leq L_y \quad (2)$$

$$-\alpha L_z \leq ms \leq L_z \quad (3)$$

where L_y and L_z are respectively the lengths of the model in the y and z directions. α is taken to be 5.

To determine the starting locations of each track in the set travelling at the given azimuthal angle θ and polar angle ϕ, the following operations are performed on each point in the array:

- The azimuthal angle θ is defined as the angle between the positive x axis and the direction of travel. The polar angle is defined as the angle between the azimuthal plane and the direction of travel. The initial co-ordinates of the array point are rotated by θ and ϕ to obtain intermediate co-ordinates given by:

$$x' = -y_0 sin\theta - z_0 sin\phi cos\theta \quad (4)$$

$$y' = y_0 cos\theta - z_0 sin\phi sin\theta \quad (5)$$

$$z' = z_0 cos\phi \quad (6)$$

- A line is extrapolated from this intermediate point towards the model. It is determined whether this line is incident on the model. If it is, the point of incidence of the line on the model is calculated. This point forms the starting location of one of the tracks of the set at the given θ and ϕ.

Figure 1 illustrates the determination of track starting points. A potential starting point of a track, travelling at a given azimuthal angle θ and polar angle ϕ, initially lies on the y-z plane of the model. The co-ordinates of this point are rotated from the y-z plane as described above; the point now lies on the shaded plane in the figure. A line is extrapolated from the point on the shaded plane towards the model. The point of incidence of the line on the model is calculated and forms the starting location of one of the tracks of the set at the given θ and ϕ.

B. Starting Positions for Tracks Travelling at all Other Angles

Calculations that use the full range of azimuthal angle space, $0 < \theta < 2\pi$, must use a number of azimuthal angles that is a factor of four. This enables the azimuthal angles to be split equally between each of the four quadrants in the azimuthal plane. Quarter-core calculations can achieve equivalent track resolution to their full-core equivalents by using four times fewer azimuthal angles.

Any calculations that use the full range of polar angle space, $-\pi/2 < \phi < \pi/2$, must use a number of polar angles that is a factor of two. This enables the polar angles to be split equally into those above and below the azimuthal plane. The uniform distribution of tracking angles is intended to assist in providing uniform coverage of models.

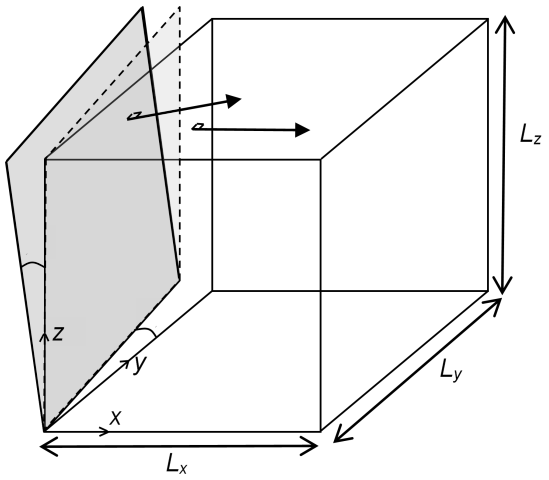

Fig. 1: Illustration of determination of track starting points

Symmetry is used to generate the starting locations for tracks travelling at angles outside of the range $0 < \theta < \pi/2$ and $0 < \phi < \pi/2$. The starting locations for these tracks are found by rotating the starting positions of the equivalent tracks travelling in the $0 < \theta < \pi/2, 0 < \phi < \pi/2$ octant.

2. Assessment Calculation

Due to the uniformity of the track distribution, it has been shown that uniformity of spatial coverage for 3D reactor-scale models can be assured. This has been demonstrated by considering a homogeneous cube model which is uniformly split into 10^6 mesh elements. Spatial coverage of the model has been compared when using the CACTUS3D and CACTUSOT tracking algorithms. Results are shown in Table I and Table II.

The results in Table I indicate that it is necessary to use a large number of track directions in the CACTUS3D tracking algorithm to assure complete spatial coverage of the model (i.e. zero untracked mesh elements). Even when complete coverage is achieved, the standard deviation on the tracked mesh element volumes is large. In comparison, the results in Table II indicate that complete spatial coverage of a model (and a relatively low standard deviation on the tracked mesh element volumes) can be achieved with just a small number of track directions. These results indicate the improvements in spatial coverage offered by the CACTUSOT tracking algorithm. For the CACTUSOT method it is noted that, for a given number of track directions (i.e. angular resolution), the total number of track segments is greater than that when using the CACTUS3D algorithm. This is at-least partially attributed to the improved mesh element coverage achieved with CACTUSOT (assuming the number of track segments increases as the number of tracked mesh elements increases and the standard deviation on the tracked mesh element volumes decreases).

No. Track Directions	No. Track Segments	Standard Deviation on Tracked Mesh Element Volume (%)	No. Untracked Mesh Elements
16	1796794	123.0	165363
36	4401599	79.7	11608
64	8254791	60.6	384
100	13469194	50.6	4
144	20053953	41.7	0

TABLE I: Cube Model Spatial Coverage with CACTUS3D Tracking Algorithm

No. Track Directions	No. Track Segments	Standard Deviation on Tracked Mesh Element Volume (%)	No. Untracked Mesh Elements
4	5125800	5.8	0
6	7661600	4.8	0
8	10206600	4.2	0
10	12749800	3.7	0
12	15300000	3.4	0

TABLE II: Cube Model Spatial Coverage with CACTUSOT Tracking Algorithm

3. Other Aspects of CACTUSOT Tracking

The above discussion relates to the calculation of track starting positions in cube models. Starting positions in cuboid models can be generated by first considering an interim cube model that has side equal to the largest side of the cube model. Track starting positions are first calculated on the surface of the interim cube model, using the methodology outlined above. The tracks are then tracked forward from the surface of the interim cube to the surface of the cuboid. The intercepts on the surface of the cuboid form the track starting positions for the CACTUSOT calculation.

It is not currently possible for CACTUSOT to treat models that do not have a cube or cuboid external boundary.

The method of angular discretization can be chosen by the user. It is possible to distribute azimuthal and polar angles uniformly within θ/ϕ space, or to specify the discretization through quadrature sets.

The CACTUSOT tracking algorithm can only be applied to models with a vacuum external boundary condition (i.e. whole-reactor calculations). Here, the inward angular flux at the start of each track is set to zero. It is also possible to treat quadrant geometries with reflective or rotational boundary conditions on their internal surfaces. Future work could consider the application of the CACTUSOT tracking algorithm to models with reflective and/or translational boundary conditions on their external surfaces, thus allowing the method to be applied to 3D lattice models.

III. SLICE-BASED GEOMETRY TREATMENT

The storage requirements for track data in detailed whole-reactor calculations can become extremely large.

The *compound trajectories* and *chord classification* schemes have been developed in the TDT solver in APOLLO3 to address this problem [5]. Both of these methods aim to reduce the amount of track information that needs to be stored. The *compound trajectory* scheme considers symmetries to allow data to be stored for smaller sections of trajectories, which are subsequently combined to form complete trajectories. The *chord classification* scheme makes use of regularities in 3D axial geometries to define a set of representative chord classes; data are thus stored for individual classes rather than individual chords. It has been indicated that the *chord classification* scheme can reduce track storage by factors between ~3 and ~15, with associated reductions in runtime of between ~30% and ~40%.

Methods have also been developed in the OpenMOC code to address this problem [6]. Track segments are calculated and stored in a 2D plane. Then, during the transport sweeps, appropriate 3D segments are reconstructed on-the-fly. This method has been shown to have the potential to reduce track storage memory requirements by more than a factor ~10, whilst typically having minimal computational overhead.

To help overcome similar track storage problems in CACTUSOT, a method is included wherein track data are generated for sub-regions within a model as opposed to the model in its entirety.

In comparison with radial heterogeneity, the axial heterogeneity of most typical reactors is limited. Hence, most reactors can be represented as a stack of repeated axial slice types. For example, one slice type could represent an axial reflector whilst another type could represent a section of the active core.

CACTUSOT allows the definition of a series of slice types alongside a description of how these types are to be stacked in order to form the composite reactor model. Track data are then generated for each of the slice types. An illustration of a core model formed from two slice types is shown in Figure 2.

Fig. 2: Illustration of a simple core model formed from two slice types. One slice type would represent the top/bottom reflectors (blue) and another slice type would represent the active core (red) plus radial reflector (blue). The composite model is formed as a stack of the slice types.

A fundamental aspect of this method is the manner in which the slices are coupled within the flux solution. Consider the coupling between the bottom slice (Slice 1) and the second-to-bottom to slice (Slice 2) in a model. A regular, XY mesh is overlaid onto the inter-slice surface (ISS) between Slice 1 and Slice 2.

Where the end points of tracks in Slice 1 are located on the ISS, they will be located in one of the ISS mesh cells (as shown in Figure 3). The projection of the track's associated cross-sectional area onto the ISS may overlap one or more ISS cells. The angular flux at the end of the track is used to form a contribution to the total leakage associated with each of the ISS cells that are overlapped. The contribution for a given ISS cell is proportional to the fraction of the track's projected cross-sectional area that overlaps the ISS cell.

After calculation of the contributions from all tracks to the leakage associated with the ISS mesh cells, the same leakages can be used to form the inward angular fluxes for tracks entering Slice 2.

Where tracks in Slice 2 start on the ISS, their starting points will be in one of the ISS cells. Again, the cross-sectional area of a track is projected onto the ISS and the cells that are overlapped are determined. The inward angular flux for a track in Slice 2 is calculated as contributions from all overlapped cells. The contribution from a given cell is proportional to the fraction of the track's projected cross-sectional area that overlaps the ISS cell.

This method is intended to preserve the total leakage between axial planes. Calculation accuracy depends on the resolution of the regular mesh on the ISSs. This resolution determines how well the spatial variation in angular flux across each ISS is represented.

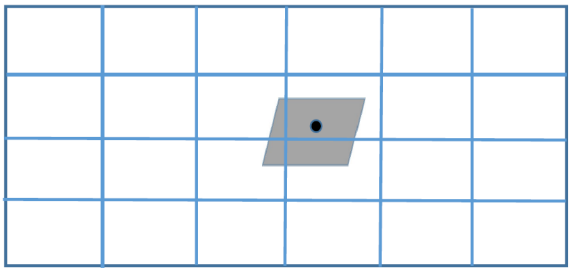

Fig. 3: Mapping of track end points onto ISS cells. The boundaries of the ISS cells are shown by the blue lines. A track end point (on the ISS) is shown by the black circle. The projection of the track's associated cross-sectional area onto the ISS is shown in grey. The total leakage for each ISS cell is calculated from all similar "grey" projections that overlap the ISS cell.

For each track direction the solution scheme proceeds as a sweep from the bottom slice to the top slice (or the top slice to the bottom slice for tracks travelling in the negative Z direction), with the ISS outward angular flux from one slice forming the ISS inward angular flux for the next slice.

For a model formed from three-slices (Slice 1 at the bottom, Slice 3 at the top), the transport sweep includes the following steps for tracks travelling in the positive Z direction:

1. All tracks in Slice 1 will have a zero inward angular flux (due to the assumed vacuum boundary condition).
2. Perform transport sweep along all tracks in Slice 1.
3. Calculate contributions to the outward leakage profile for the ISS between Slice 1 and Slice 2.
4. Calculate inward angular fluxes for tracks in Slice 2, based on the leakage profile calculated in the previous step.
5. Perform transport sweep along all tracks in Slice 2.
6. Calculate contributions to the outward leakage profile for the ISS between Slice 2 and Slice 3.
7. Calculate inward angular fluxes for tracks in Slice 3, based on the leakage profile calculated in the previous step.
8. Perform transport sweep along all tracks in Slice 3.

1. Assessment Calculation

Table III gives some performance statistics for a calculation of a homogenised cube model when using the standard and slice-based solution schemes. The model was of side 100cm and was split into 10^6 uniform meshes. Angular discretization was defined using a S6 quadrature set and the track separation was set at 0.5cm. In the slice-based solution, one

slice type of height 20cm was used (meaning that the composite model was formed by five stacked instances of this slice type).

By using the slice-based solution, the size of the track data is reduced by a factor ~4 (i.e. a factor comparable to some of the *chord classification* results given in [5]). It is noted that, when using the slice-based solution, the total amount of tracking data includes information on the mappings between the tracks and the ISSs; this data is not required for the standard solution scheme. Therefore, the factor reduction in the size of the track segment data alone will be greater than ~4. This reduction in track data is beneficial in making the execution of reactor-scale calculations feasible on computing systems where only limited amounts of RAM are available.

A disadvantage of the slice-based scheme is the ~10% increase in solver runtime, which is attributed to the additional cost of having to calculate terms associated with the ISS mappings. However, given that the memory savings offered by the scheme would make the execution of reactor-scale calculations feasible on computing systems where only limited amounts of RAM are available, such modest increases in runtime are judged to be acceptable.

IV. OTHER FEATURES OF CACTUSOT

As well as the specialized tracking algorithm and slice-based geometry treatment, CACTUSOT includes a large range of features to assist in performing reactor-scale calculations. These include:

- Convergence acceleration methods based on Chebychev, CMR, CMFD and GMRES schemes. The CMR and CMFD schemes are currently restricted to models with regular XYZ geometry; developments to allow the methods to be applied to hexagonal core models or cases with resolved pins will be considered in the future.
- The ability to derive an initial scalar flux guess from a diffusion theory solution (and hence improve on the use of a flat flux guess). This method is also currently restricted to models with regular XYZ geometry, and developments to generalise this will be considered in the future.
- Treatment of both transport-corrected P0 and anisotropic P1 scatter.
- Diamond-difference representation of the variation of neutron source along track segments. This has the potential benefit of enabling solution accuracy to be preserved whilst increasing the size of mesh elements and hence improving calculation efficiency.
- MPI parallelization of the track generation process and the flux solution algorithm. The parallelization framework is structured such that each process handles calculations for all tracks at a subset of the track directions.
- Angular discretization definable via symmetric quadrature sets.
- Geometry specification using the Fractal Geometry package, which can be used to simplify the preparation of complex geometry models.
- Full integration with the GEOM calculation route in WIMS, which simplifies the setup and execution of core-scale problems and facilitates operations such as rod movements.

V. SFR BENCHMARK CALCULATIONS

CACTUSOT has been used to perform calculations for the Sodium Fast Reactor benchmark defined by the OECD/NEA Working Party Reactor and System Expert Group on Reactor Physics and Advanced Nuclear Systems. Cross-section data for the analyses were prepared using the ECCO module within WIMS.

The medium oxide core of the benchmark was analysed with CACTUSOT. A detailed description of this analysis can be found in [7]. Figure 4 shows the model geometry.

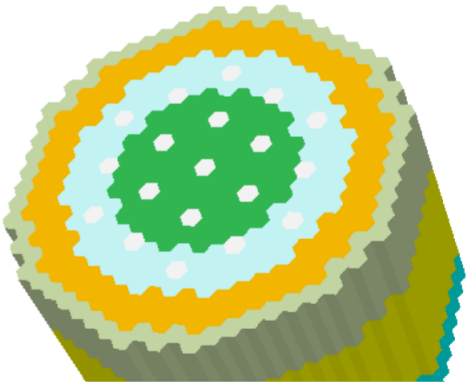

Fig. 4: Illustration of SFR Benchmark Model Geometry

The slice-based geometry scheme was used for the CACTUSOT calculations, so that it was possible to reduce the memory requirements and complete the analysis with the available computer hardware. For these calculations the assembly geometry was homogenised. However, in principle, a pin-by-pin geometric representation is feasible using CACTUSOT.

To confirm the accuracy of CACTUSOT for these calculations, its solutions were compared with those from the MONK Monte Carlo code and also with those from other benchmark participants. The MONK calculations used identical 33-group cross-sections to the CACTUSOT calculations; the comparisons were therefore solely a check of the accuracy of the flux solver. Values of start-of-life k-effective from the calculations are given in Table IV. The CACTUSOT and MONK calculations both used cross-sections derived from the JEFF3.1.2 nuclear data evaluation.

A total of 72 tracking directions were used in the CACTUSOT calculations; the track separation was set at 1.0cm. The geometry was split into approximately 250 axial slices, which were represented by 10 different slice types. To calculate the inter-slice couplings the ISSs were divided into 200x200 mesh cells.

Excellent agreement is observed between the CACTU-

Solution Scheme	Storage for Track Data (MB)	Tracking CPU Runtime (mm:ss)	Solver CPU Runtime (mm:ss)
Standard	~2200	~00:48	~09:07
Slice-based	~560	~01:00	~09:40

TABLE III: Performance Statistics when using the Standard and Slice-Based Solution Schemes

	k-effective	$\Delta\rho$ Sodium Void (pcm)	$\Delta\rho$ Doppler (pcm)	$\Delta\rho$ Control Rods (pcm)
WIMS/MONK	1.0346	2070	-385	18457
WIMS/CACTUSOT	1.0370	2218	-390	19469
Benchmark	1.0354 ± 0.0078	2024 ± 407	-346 ± 44	19697 ± 2087

TABLE IV: Comparison of Results for SFR Benchmark

SOT and MONK predictions. This provides confidence in the accuracy of the CACTUSOT methodology when applied to such models. Furthermore, the CACTUSOT and MONK k-effective predictions are in close agreement with the benchmark average values.

The calculations were performed using 8, 16, 32 and 64 MPI processes. In each case the time required to generate the tracking data was less than 10% of the total CACTUSOT runtime. The solver runtime in each case is shown in Table V.

When doubling the number of processes the ideal strong-scaling reduction in runtime would be a factor 2. The reduction in runtime when moving from 8 to 16 processes is a factor of ~1.47; the equivalent runtime reduction factors when moving from 16 to 32 and 32 to 64 processes are ~1.53 and ~1.07 respectively. The parallel scaling is close to saturation when using 64 processes.

VI. PWR MINI-CORE BENCHMARK

The performance of CACTUSOT when applied to a PWR mini-core benchmark has been assessed. The calculations were based on the OECD Nuclear Energy Agency (NEA) Benchmark on Deterministic Transport Calculations Without Spatial Homogenisation: MOX Fuel Assembly 3-D Extension Case [8], a variant of the C5G7 benchmarks. This consists of a 3D mini core of UO2 and MOX assemblies surrounded by a water reflector in all directions. The mini core has octant in-plane symmetry and reflective symmetry in the axial direction. There are three cases, with control rods progressively inserted into some of the fuel assemblies. Both fuel pins and guide tubes (rodded and unrodded) are treated as a single cylinder inside of a square pincell. For example, for the fuel pins the clad, gap and fuel are smeared together. Seven-group cross sections are supplied as part of the benchmark specification.

Detailed comparisons between CACTUSOT results for the benchmark and reference results from MCNP are reported in [9]. It is found that in virtually all of the benchmark cases the agreement between CACTUSOT and MCNP for distributed parameters is within 1%. Furthermore, the discrepancies between CACTUSOT and MCNP are found to be consistent with those between deterministic and Monte Carlo codes for this benchmark.

The performance of the CACTUSOT slice-geometry scheme and parallelization has been assessed for these mini-core calculations. In order to assess performance, the solution parameters (e.g. mesh resolution) were slightly coarsened with respect to those used to obtain the best-estimate results presented in [9]. This allowed reference calculations to be performed when using 3D tracking across the whole model, instead of using the slice-geometry scheme. The coarsened calculations all used 64 track directions and a track spacing of 0.18cm.

When using the slice-geometry scheme it was found that the amount of RAM required to run the calculation was reduced by a factor ~ 3.

An assessment of the strong-scaling parallel performance of the solver was also made for this benchmark problem. Results are shown in Figure 5. Very good scaling is observed up to 64 MPI processes, which is the maximum number that can be used by CACTUSOT in this case since 64 track directions are used.

The parallel scaling for this case is better than that for the SFR benchmark calculation described above. This is likely to be a consequence of running the PWR mini-core calculations on a single HPC node, whereas the SFR calculations were spread across multiple nodes (and hence the scaling performance is reduced by the inter-node communication overheads). This indicates that the parallel scaling performance for the SFR calculations could potentially be improved by running on higher-specification hardware.

Future improvement of CACTUSOT could allow the number of permitted MPI processes to exceed the number of track directions. The current parallelization framework allows each MPI process to handle all tracks at a subset of the track directions. This framework could be extended so that an MPI process can treat a subset of the tracks travelling in a given direction.

No. MPI Processes	Solver CPU Runtime (minutes)
8	~4843
16	~3285
32	~2143
64	~2010

TABLE V: CACTUSOT Parallel Performance for SFR Calculation

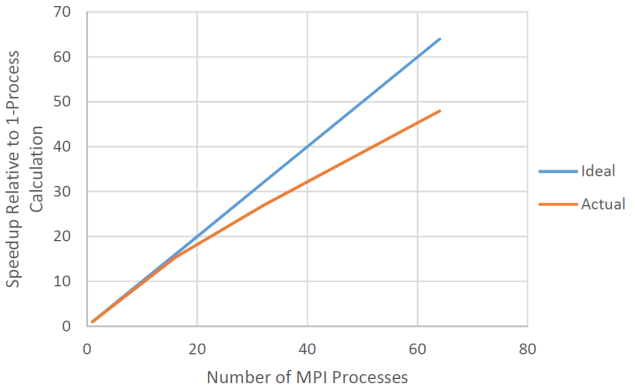

Fig. 5: Parallel Scaling Performance for PWR Mini-Core Calculations

VII. CONCLUSIONS

The CACTUSOT MoC solver in WIMS has been applied to a selection of core-scale benchmark problems. Use of the specialised methods in CACTUSOT has made these calculations feasible when running on a modest HPC cluster (12 nodes, 16 multicore Intel Xeon processors per node, 64GB of RAM per node). Results obtained using CACTUSOT are found to be in close agreement with reference Monte Carlo solutions, and also solutions from other benchmark participants' codes.

By using the once-through tracking scheme it has been shown to be possible to achieve significant improvements in spatial coverage, relative to the CACTUS3D tracking scheme which has been successfully used for 3D lattice problems. The slice-geometry method can provide factor 3 to 4 reductions in the amount of RAM required to store track data. Parallel scaling of the CACTUSOT solver is good.

REFERENCES

1. J. ASKEW, "A Characteristics Formulation of the Neutron Transport Equation in Complicated Geometries," Tech. Rep. AAEW-M 1108, UK Atomic Energy Establishment (1972).
2. T. NEWTON, G. HOSKING, L. HUTTON, D. POWNEY, B. TURLAND, and T. SHUTTLEWORTH, "Developments within WIMS10," in "Proceedings of PHYSOR 2008," (September 2008).
3. A. MARLEAU and R. R., "A User Guide for DRAGON. Version DRAGON 000331 Release 3.04," Tech. Rep. IGE-174 Rev. 5, Institut de génie nucléaire, École Polytechnique de Montréal (2000).
4. W. BOYD, S. SHANER, L. LI, B. FORGET, and K. SMITH, "The OpenMOC Method of Charactheristics Neutral Particle Transport Code," *Annals of Nuclear Energy*, **68**, 43–52 (2014).
5. D. SCIANNANDRONE, S. SANTANDREA, and R. SANCHEZ, "Optimized Tracking Strategies for Step MoC Calculations in Extruded 3D Axial Geometries," *Annals of Nuclear Energy*, **87**, 49–60 (2016).
6. G. GUNOW, S. SHANER, B. FORGET, and K. SMITH, "Reducing 3D MOC Storage Requirements with Axial On-the-fly Ray Tracing," in "Proceedings of PHYSOR 2016," [10].
7. B. STRAY, M. DELANEY, B. LINDLEY, M. SHEPHERD, S. RICHARDS, G. HOSKING, and P. SMITH, "Solution of the OECD/NEA SFR Neutronic Benchmark using WIMS and MONK," in "Proceedings of PHYSOR 2016," [10].
8. "Benchmark on Deterministic Transport Calculations Without Spatial Homogenisation: MOX Fuel Assembly 3-D Extension Case," Tech. rep., OECD Nuclear Energy Agency (2011).
9. B. A. LINDLEY and J. G. HOSKING, "Developments within the WIMS Reactor Physics Code for Whole Core Calculations," in "Proceedings of M&C 2017," (April 2017).
10. *Proceedings of PHYSOR 2016* (May 2016).