UNIVERSITY OF CAMBRIDGE

Accelerating surface tracking via distance caching

Mikolaj Adam Kowalski

Paul Cosgrove

Engineering - Energy, Fluid dynamics and Turbo-machinery

SCONE

All results obtained with SCONE

Goals:

- Education
- Prototyping new MC methods

Complementary to established codes (Serpent)

Stochastic
Calculator
Of
Neutron Transport
Equation

Optimise speed of:
Idea → Prototype Implementation; not *Input → Result*

- Academic Focus: Target audience →Master's and PhD Students
- Designed for modification: Object-Orientation, well-defined abstractions
- **Prioritise modifiability over performance**

Language: Fortran 2008

- Easy to learn; Informative Compiler Errors; Easy to read standard
- Offers good performance
- Well-established (generally supported, OpenMP, OpenACC etc.)

SCONE

Modification should be easier

- Small scope → Brand new tally in 1 file
- Unit tests → Confidence after change
- In-source documentation → Function/class specification (with edge cases)

Still a work in progress

- No S(α,β) and unresolved resonances
- Needs further improvements in architecture
- No parallelisation
- Patchy documentation
- No clear contributing guide

```
!!
!! Returns number of particles produced on average by the reaction
!!
!! Args:
!!   E [in] -> Incident particle energy [MeV]
!!
!! Result:
!!   Average number of particles for an incident energy E.
!!
!! Errors:
!!   If E is invalid (e.g. -ve) or outside of bounds of data
!!   table N = 0.0 is returned.
!!
function release(self, E) result(N)
  import :: defReal, uncorrelatedReactionCE
  class(uncorrelatedReactionCE), intent(in) :: self
  real(defReal), intent(in)                 :: E
  real(defReal)                             :: N
end function release
```

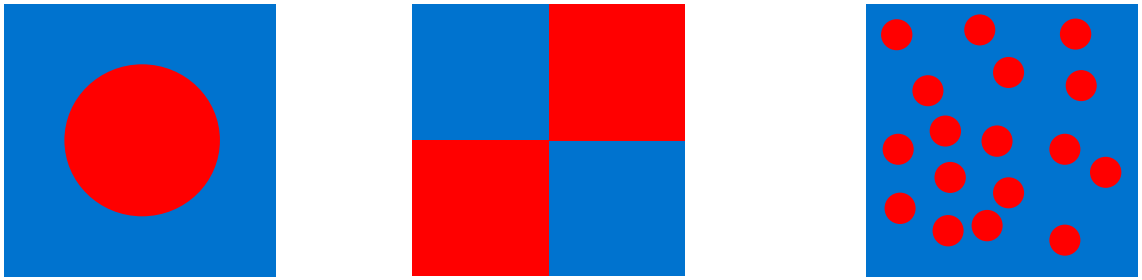
Documentation Comment

Open Source (MIT licence):
bitbucket.org/Mikolaj_Adam_Kowalski/scone

We don't want it to stay "Cambridge Code" → Feel free to contact us or start an ***Issue***

Multi-level Geometries

Nuclear Reactors are repeated structures → Concept of a **Universe**



Universe can be defined as:

- Complete subdivision of space into disjoint cells
- Cells can be filled with a material or ***portals*** to other universes
- ***Unspecified*** regions are still defined (in a sense)

Two orthogonal problems:

- Representation of spatial decomposition within universe → Universe object
- Representation of nesting structure → Directed Acyclic Graph

Specialisation of universes can be accommodated by ***universe polymorphism***

- In SCONE by using OO features
- In Serpent explicitly via “switch statements”

Multi-level Geometries

Directed Acyclic Graph (DAG)

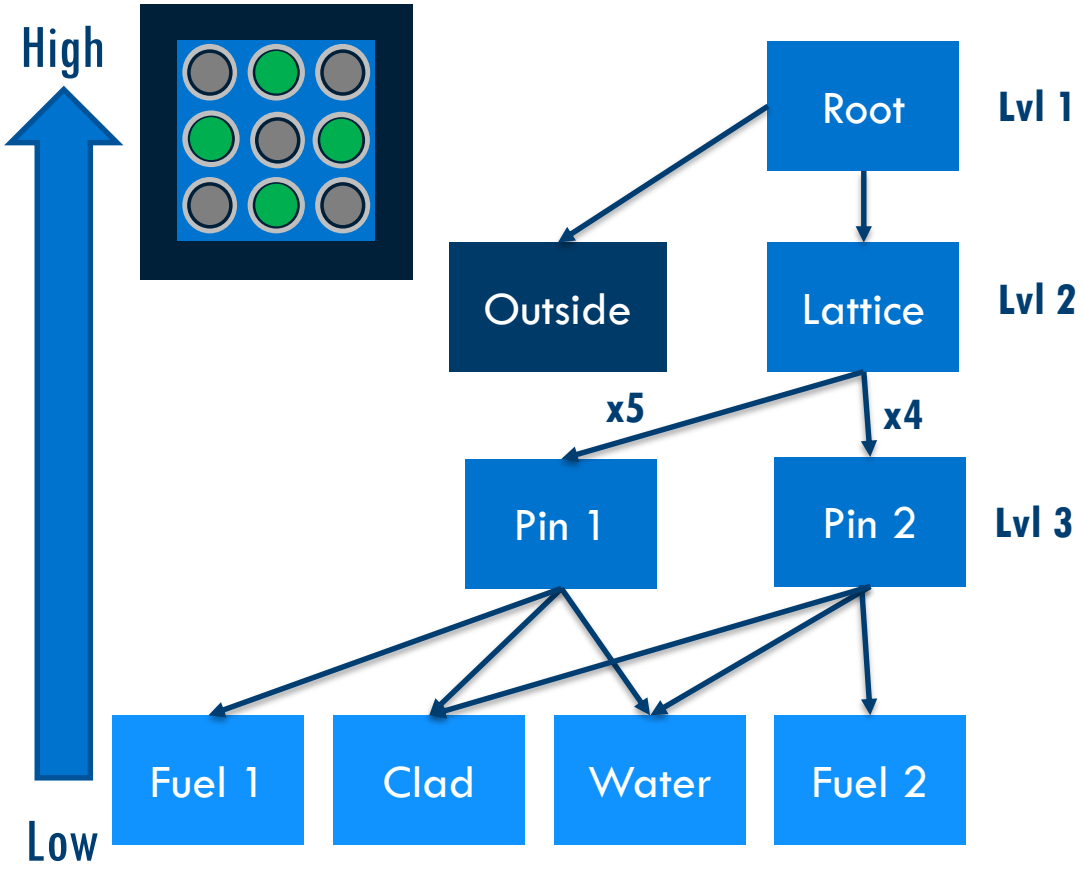
- Each Node is a universe
- Each Transition is a cell
- Single Source is the root universe
- Each Sink is a material

Terminology

- High level → Closer to root universe
- Low level → Further to root

Cell transitions can be associated with translations/rotations

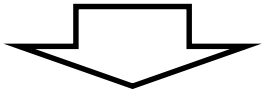
Particle may have different co-ordinates on each level



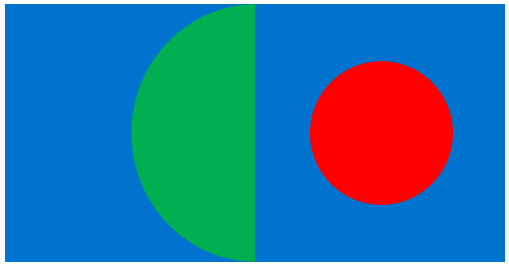
Surface Tracking in Multi-level Geometries

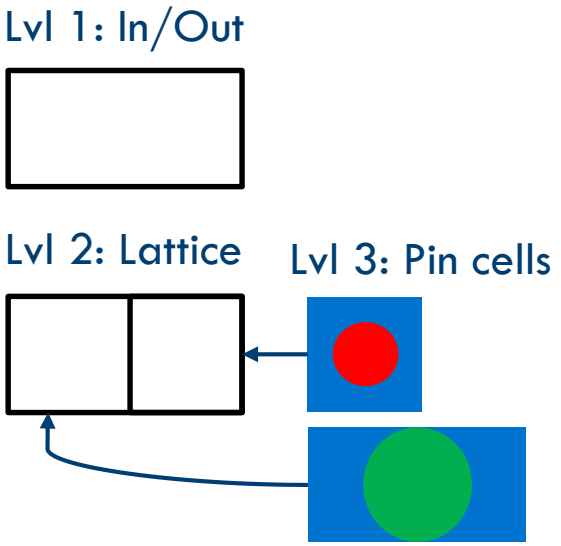
Ability to "cut" cells in lower universes by portals is incredibly useful

- Allows to superimpose object over a "background" definition
- e.g. Project partially inserted Y-shaped CR over hexagonal fuel lattice
- It is problematic for Surface Tracking



- Crossing distance needs to calculated at every level
- Smallest is the true closest boundary
- In case of ambiguity (same distance) → Choose higher level





Some older codes require portals and universes to "fit" → One evaluation is sufficient

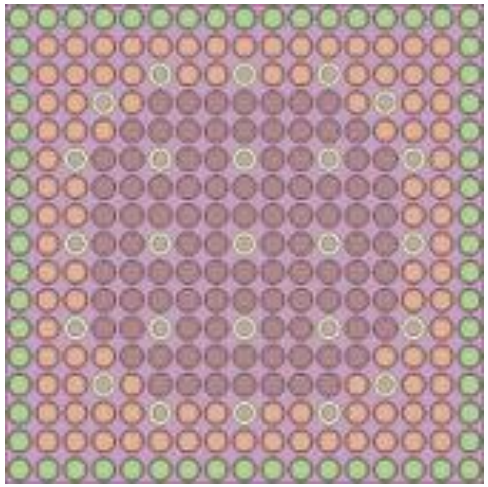
Profiling with Perf

- Reflected fuel assembly
- Fresh fuel (limited number of isotopes)
- 3-level geometry

SCONE requires root level to be composed of Inside/Outside only

Profile results:

- ~30% → interpolating cross-sections (expected)
- ~18% → distance calculations
- ~7% → distance to boundary (top level)



MOX PWR 17×17 Assembly

Perf

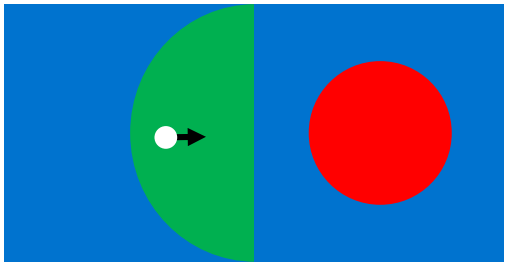
- Linux tool for performance analysis
- Profiling & access to CPU events (e.g. branch misses)

CPU time in different procedures

```
23.56%  __aceneutronnuclide_class_MOD_search
 6.76%  __aceneutrondatabase_class_MOD_updatemicroxss
 6.63%  __squarecylinder_class_MOD_distance
 5.74%  __ieee754_log_fma
 5.65%  __latuniverse_class_MOD_distance
 5.27%  __cylinder_class_MOD_distance
 2.91%  __particle_class_MOD_particlestate_fromparticle
```

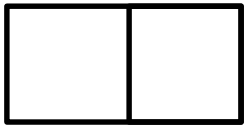
Boundary is axis-aligned box. Distance calculation is simple. Why so much time?

Distance Caching

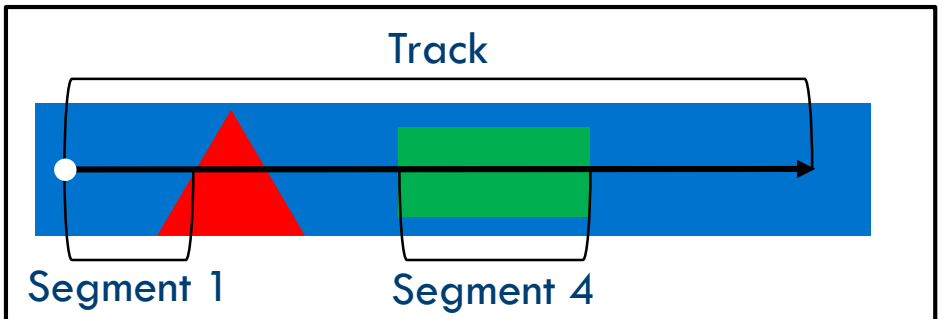


Lvl 1: In/Out

Lvl 2: Lattice



Lvl 3: Pin cells



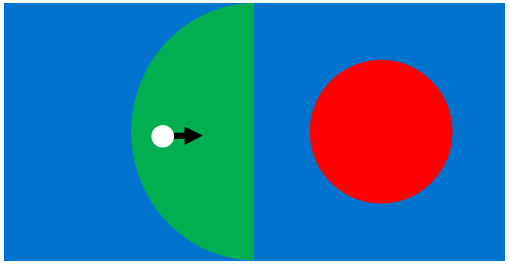
Segment: Section of track in a single cell
Track: Line between collisions/boundary hits

- Each Track can be composed of multiple segments
- Direction does not change after surface crossing
  - Distance for lvl 1&2 is still valid if crossing at lvl 3

Many extraneous distance calculations at high levels (e.g. lvl 1)

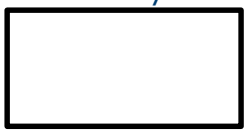
Why don't save & reuse valid distance values?

Distance Caching

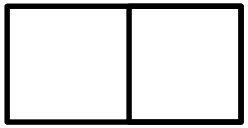


- Save distances between segments in a "cache"
- Need to decrement distance after segment → FP error accumulation

Lvl 1: In/Out

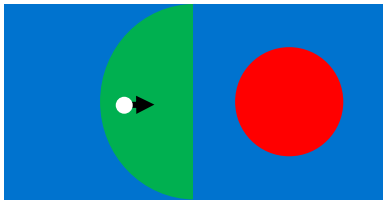


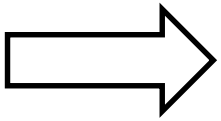
Lvl 2: Lattice

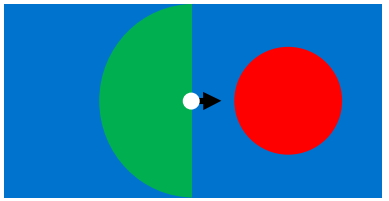


Lvl 3: Pin cells

Segment 1	Cross surface	Segment 2
Cache 1: $\leftarrow d_1$ 2: $\leftarrow d_2$ 3: $\leftarrow d_3$	**Cache** 1: $d_1' \leftarrow d_1 - d_2$ 2: ~~d_2~~ 3: ~~d_3~~	**Cache** 1: d_1' 2: $\leftarrow d_2'$ 3: $\leftarrow d_3'$





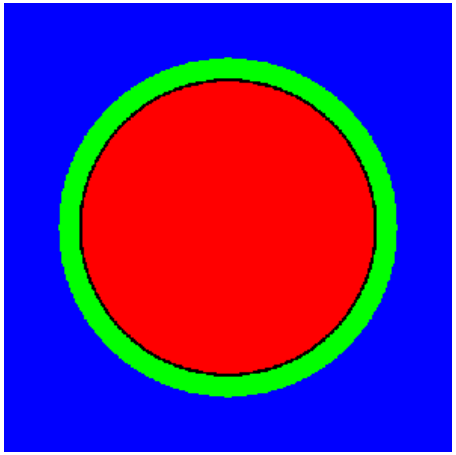


Test Cases

Pin cells are shown,
but geometry was full assembly

Single assembly, 3-level Test Problems

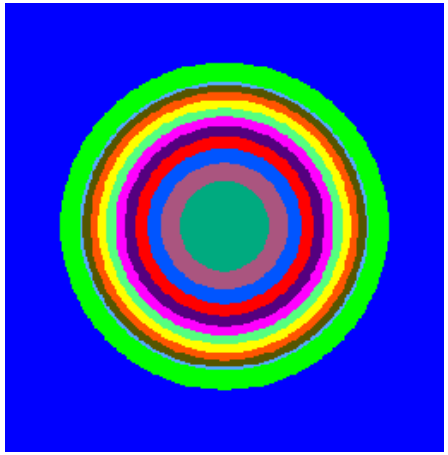
Assembly



- 17x17 PWR
- UOX Fuel

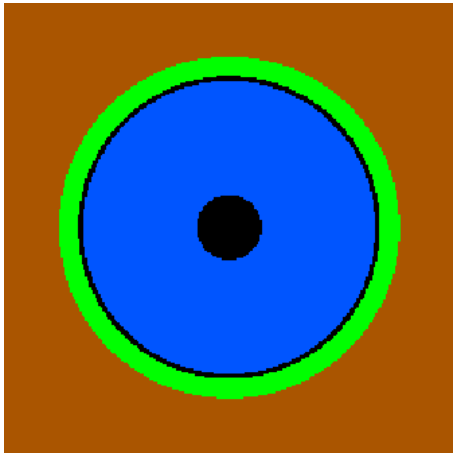
All calculations:
- Only k-eff tally
- Collision estimator

Fine Resolution



- 17x17 UOX PWR
- 10 radial regions
- 200 axial regions

LFR

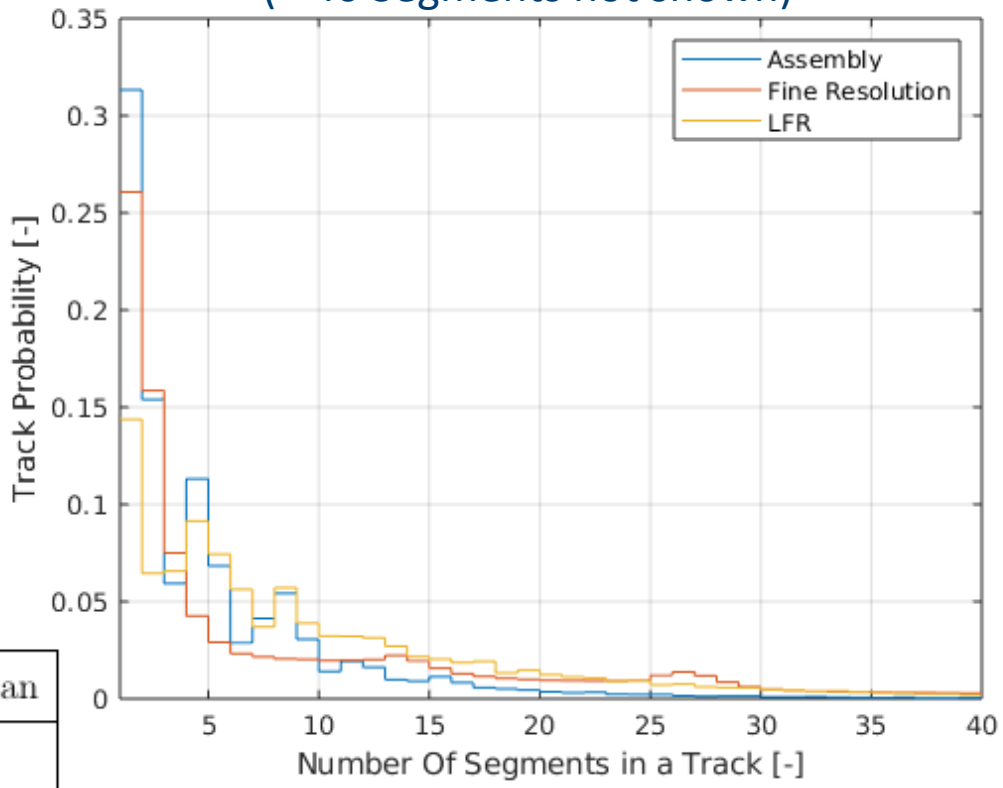


- 21x21 Assembly
- Annular Pin
- Based on ELSY
- Simplified Material compositions

Segments Distribution

- Majority of tracks composed of multiple segments
- Peaks in probability → Fuel-coolant transfers through thin cladding and gap

Segments per Track Distribution
(>40 Segments not shown)



Segments per track statistics

Case	Depth	Max	Mean	Median
Assembly	3	131	5.1304	3
Fine Resolution	3	431	10.8360	4
LFR	3	189	10.0795	7

Based on a sample of 60 million tracks

Results & Observations

Runtime

Test Case	Tracking Type	Avg. Time [min:sec]	Speedup
Assembly	DT	10:09	1.286
	ST Cached	11:45	1.110
	ST Standard	13:03	N/A
Fine Resolution	DT	11:40	1.846
	ST Cached	17:50	1.207
	ST Standard	21:32	N/A
LFR	DT	25:21	2.026
	ST Cached	43:36	1.178
	ST Standard	51:20	N/A

- 6 independent runs
- 30k neutrons
- 300 active, 100 inactive cycles

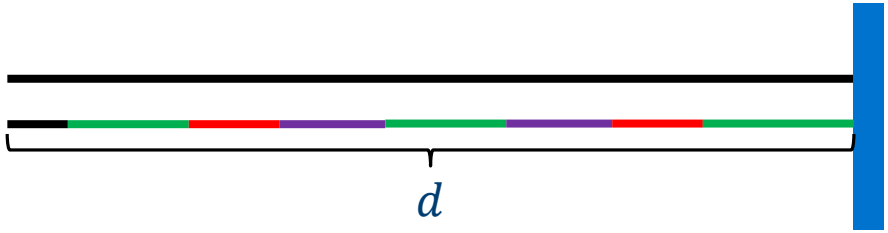
Segments per track statistics

Case	Depth	Max	Mean	Median
Assembly	3	131	5.1304	3
Fine Resolution	3	431	10.8360	4
LFR	3	189	10.0795	7

- Speedup roughly proportional to mean number of segments
- Moderate improvement ~10-20%
- LFR slower then Fine Resolution case
  - Cell crossings at higher level
- DT Tracking is superior

Potential Danger – FP error accumulation

Floating Point error may accumulate → Basically a summation problem



- 1st Large +ve term
- N small –ve terms

E_N - Absolute error in last segment

Double precision: $\epsilon \approx 2.22 \times 10^{-16}$

Naïve summation ⟶ Current implementation

$$|E_N| \leq 2d(N-1)\epsilon + O(\epsilon^2)$$

- Worst case error $\propto N$
- Cache timeout → Recalculate distance periodically

Kahan Summation

$$|E_N| \leq 4d\epsilon + O(N\epsilon^2)$$

- Extra memory → compensator
- Slightly more complicated caching logic
- Should have no effect on performance

No FP-related problems were observed in test calculations
Implementation will be changed to Kahan in near future
Drift in co-ordinates (?) → Use Kahan as well (?)

Conclusions

Conclusions:

- Modest improvement ~10-20%
- Provided geometry handling is significant to runtime
- Not superior to DT
  - May change with strong localised absorbers

May enables to use highly-complex surfaces in higher universes (Reduce # of iterative solutions)

Future work:

- Implement Kahan summation
- Test in larger problems
- Verify conclusions with other codes (OpenMC?)
- Track-length estimator & FoM

Without Cache

```
23.56%  __aceneutronnuclide_class_MOD_search
 6.76%  __aceneutrondatabase_class_MOD_updatemicroxss
 6.63%  __squarecylinder_class_MOD_distance
 5.74%  __ieee754_log_fma
 5.65%  __latuniverse_class_MOD_distance
 5.27%  __cylinder_class_MOD_distance
 2.91%  __particle_class_MOD_particlestate_fromparticle
```

With Cache

```
26.21%  __aceneutronnuclide_class_MOD_search
 7.08%  __aceneutrondatabase_class_MOD_updatemicroxss
 6.78%  __ieee754_log_fma
 5.88%  __cylinder_class_MOD_distance
 3.32%  __neutroncestd_class_MOD_scatterfrommoving
 3.12%  __particle_class_MOD_particlestate_fromparticle
 2.89%  __aceneutrondatabase_class_MOD_updatetotalmatxs
```

Only cylinder remains in the profile

<*)))))><

Thank you for your attention!

mak60@cam.ac.uk