New features for pebble bed reactors simulations with Cerberus/Serpent

Yves Robert
Ville Vartavirta

31/08/2022 Serpent Users Group Meeting 2022, Garching

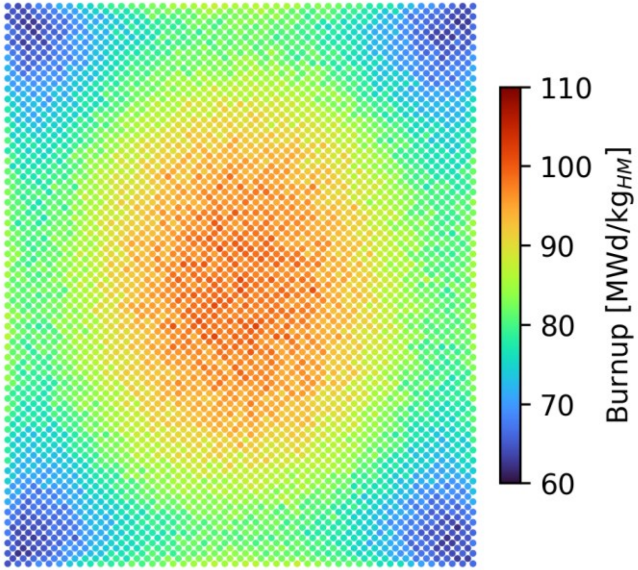
Content of the presentation

- Current Serpent capabilities and limitations for pebble bed reactors
- Recent Serpent/Cerberus developments and new features
- Application to pebble bed reactors
- Conclusions

Current Serpent capabilities and limitations for pebble bed reactors

Serpent already includes some key features allowing for large pebble beds depletion

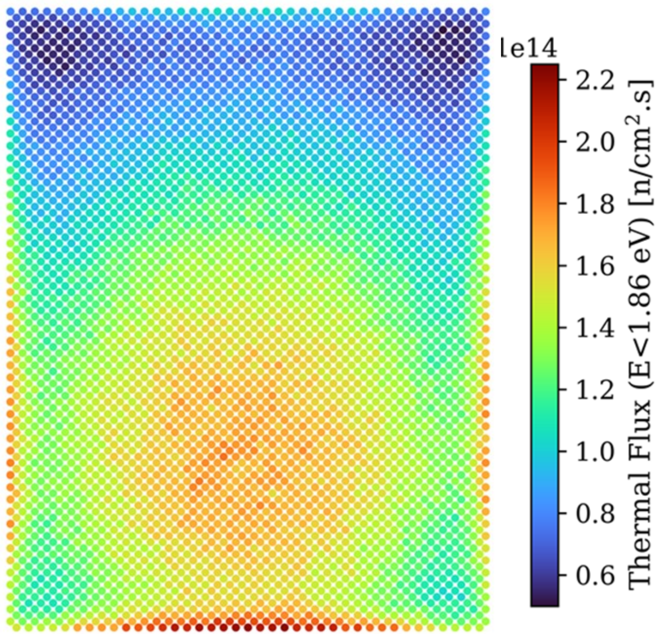
- Modeling of pebble bed and TRISO particles: explicit stochastic geometry
- Fast transport: delta-tracking, cartesian search mesh overlaid on top of pebble bed geometry
- Division of pebbles into separate depletion zones: recursive division
- Parallel computing and memory savings: MPI and domain decomposition



Burnup distribution for static FHR core with individual pebbles depletion at core average burnup of 80 MWd/kg

Serpent suffers from some feature/ performance limitations

- ❑ Static geometry: cartesian search mesh is fixed
- ❑ Low control over values and simulation: input with branches
- ❑ Large post-processing requirements: bumat files/restart/depletion files not always adapted
- ❑ Long initialization times to create depletion zones
- ❑ Possible to overcome, to the price of approximations or extensive development

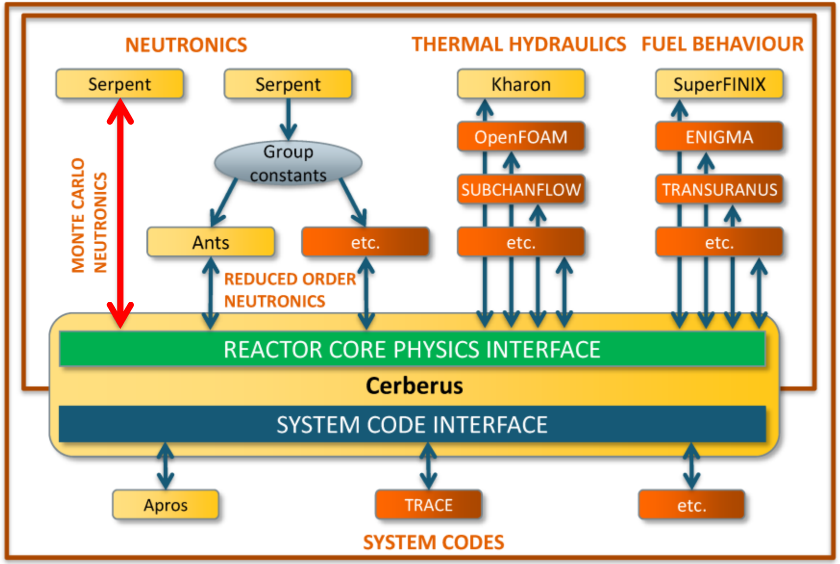


Thermal flux distribution for gFHR core at equilibrium obtain with discrete-motion

Recent Serpent/Cerberus developments and new features

The Cerberus driver is Python-based and can interface with Serpent

- Python-based Multi-physics driver of Kraken
- Able to communicate with Serpent during calculation and drive the Serpent simulation
- Two types of variables used:
  - Output variables (*sss_ov_*...): used to send from Serpent → Python
  - Input variables (*sss_iv_*...): used to send from Python → Serpent
- Easy to add variables in Serpent source code



Kraken computational framework (Leppänen, Energies 2022)

Cerberus gives access to relevant information for Serpent calculations

- ❑ Works with domain decomposition
- ❑ Output for divided materials as an array or individually

Output parameters		
Simulation	Transport parameters (keff, fission rate, etc.)	Memory usage
	Detectors and uncertainties	Simulation BU/time
	Materials list	
Materials-wise information	Atomic/Mass density	Burnup/FIMA
	Nuclides names, ZAI, densities	Domain
	Inventory	Initial composition
	Burnable nature	Volume
pbed specific (pebble bed or TRISO)	Coordinates	Radii
	Universes	

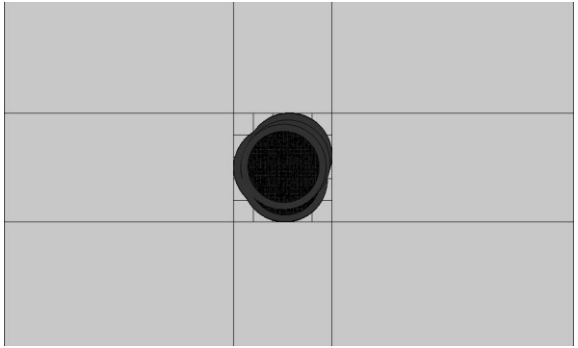
Cerberus gives more control over the Serpent 2 simulations

- ❑ Works with domain decomposition
- ❑ Synchronizes when predictor and corrector are run

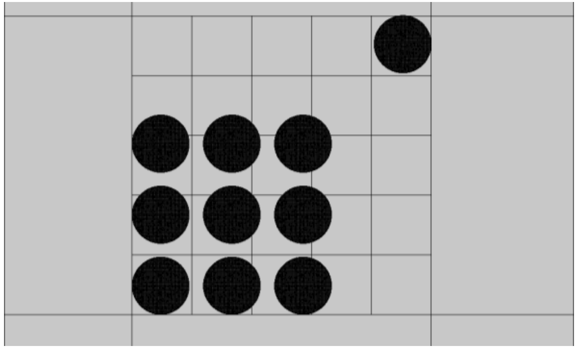
Input parameters		
Simulation	Normalization parameters (power, power density, flux, fission rate, …)	Depletion mode (burnup, decay, activation)
	Neutron population and cycles	Simulation BU/time
Materials-wise information	Nuclides densities array	Burnup (FIMA through nuclides)
	Burnable nature	Volume
	Reset to initial composition	Burn
	Domain	Decay
pbed specific (pebble bed or TRISO)	Coordinates	Plot geometry (motion)
	Radii	

Motion is now possible with pbed geometries

- Possibility to change positions and radii with Cerberus
- Cartesian search mesh updated
- Pebbles with 0 radius are set as “ghosts”:
  - Removed from the mesh
  - Not taken into account in burnup, only decay
- Can plot the geometry every time



Moving pebbles and cartesian search mesh



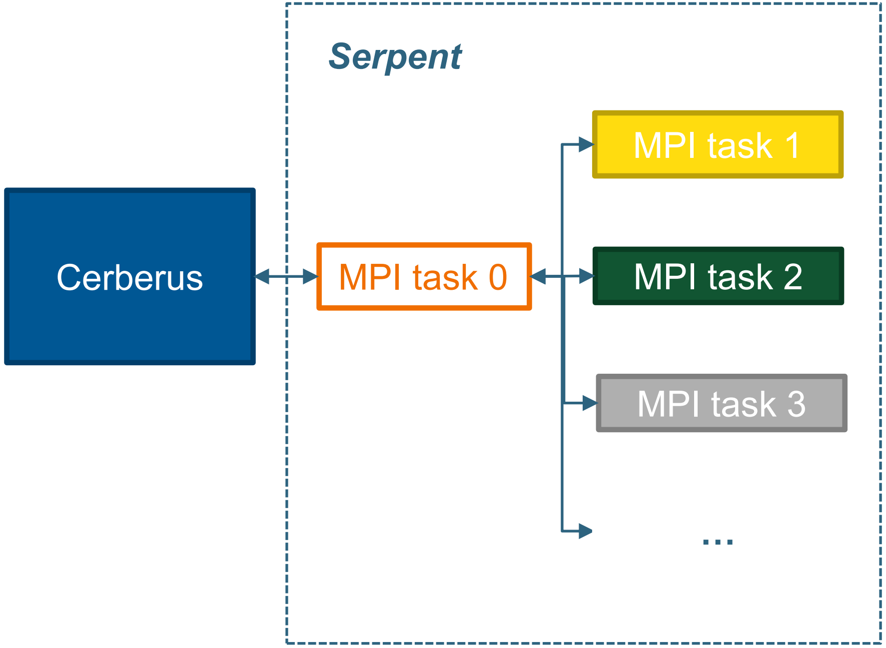
Effect on mesh of varying radius pebble

Depletion can be better controlled with recent additions

- New depletions features
  1. Convert simulation time/burnup to burnup/time
  2. Control simulation execution:
     - Set a simulation time manually
     - Change burnup mode on-the-fly: burnup, decay, activation
  3. Control material-wise depletion:
     - Turn burnable materials into non-burnable
     - Independent materials burn/decay
     - On-the-fly modification of parameters: volume, burnup, concentrations
     - Reset to original composition

Domain decomposition reduces materials memory/BU time and is handled in Cerberus

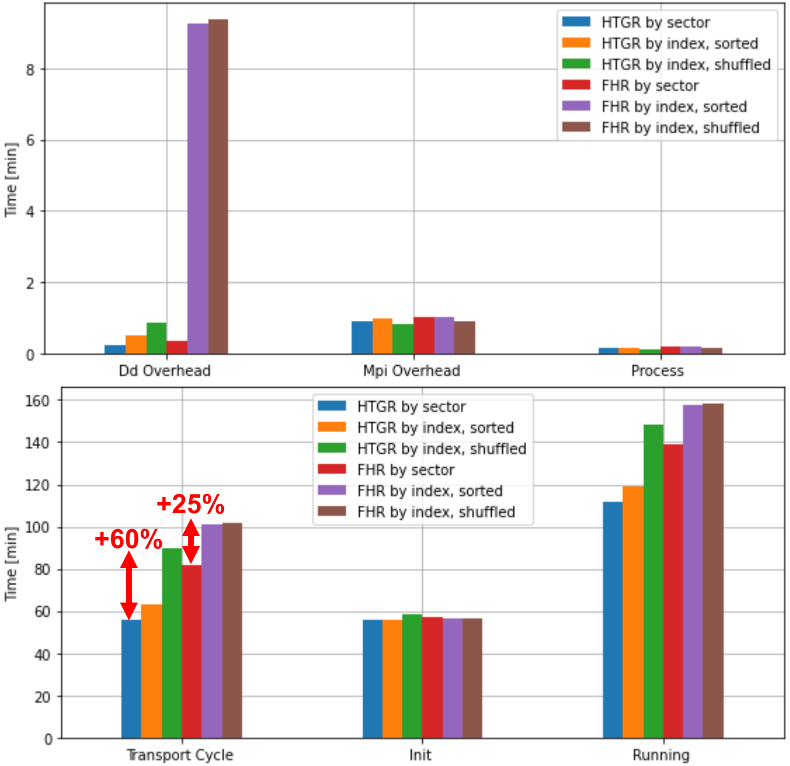
- Domains interface with Cerberus through the MPI task 0 (master)
- Possibility to send input/output variables over domains
- That includes sending new domain ID on-the-fly
  - Can tailor DD to the needs (axial, radial, sectors, spherical, hexagonal, …)



Cerberus/Serpent communication with domain decomposition

Well defined domain decomposition allows for better performances

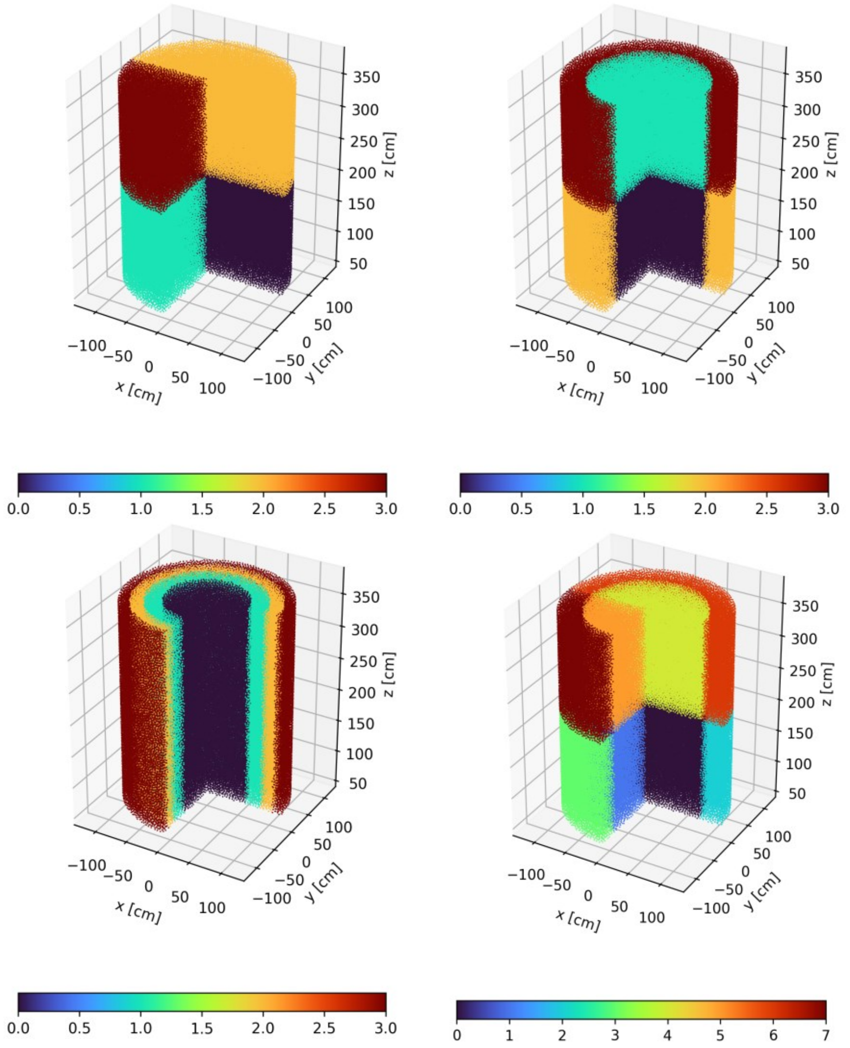
- 400,000 pebbles in 4 domains:
  - In sectors
  - By index, created as axial layers
  - By index, randomly shuffled
- Non-negligible impact of how pebbles are decomposed, both for HTGR and FHR
- Specific decompositions are missing from Serpent
- Pebbles changing position keep their domains



Domain decomposition time breakdown for HTGR and FHR cores

User-defined domains can now be used

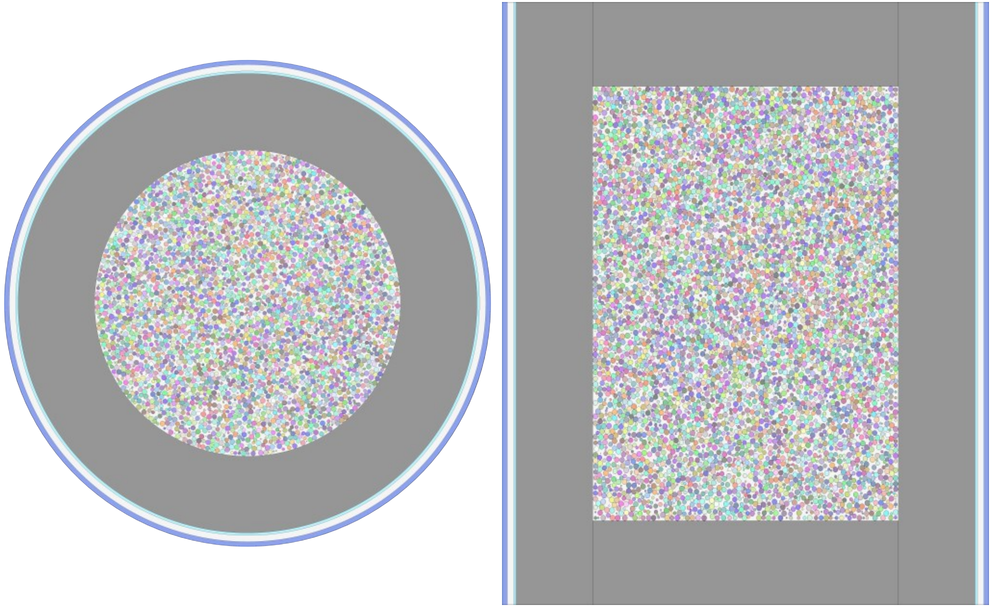
- Any routine can be written in Python language
- Possible benefits:
  - Optimal performance for static geometries
  - Updated domains for moving geometries



Examples of custom domain decompositions (each color corresponds to a domain ID)

Pebble beds division performance has been enhanced

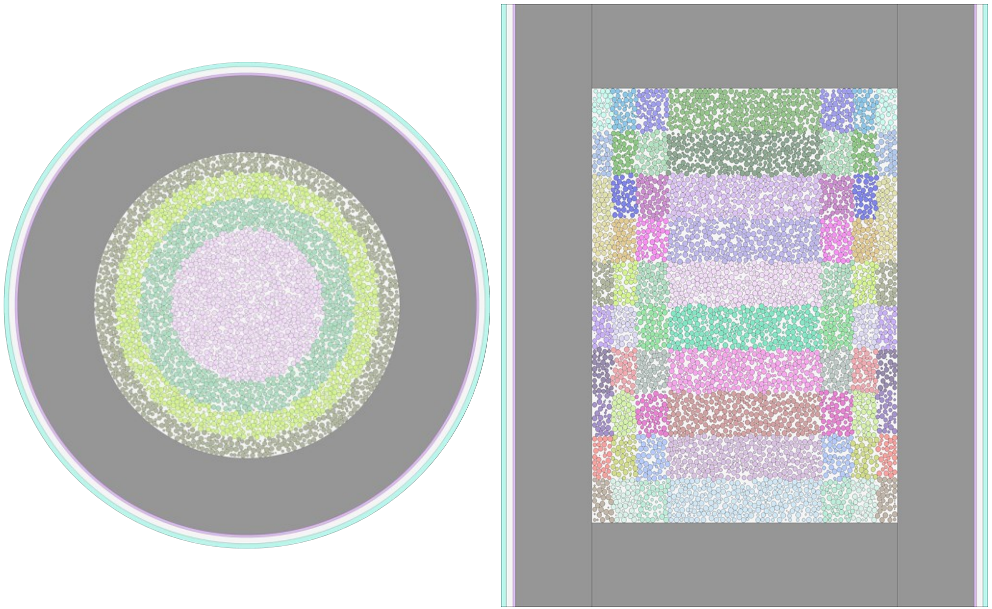
- Initialization, 500,000 divided pebbles: **>1 hour**
- Special treatment for pebble bed geometries:
  - No recursive routine: div peb <upbed>
  - New initialization time: **<5 minutes**
- Divide pbed object universe-wise. Example:
  - 3 universes in pbed object: u1, u2, u3
  - div fuel peb <upbed> 0 divides u1, u2, u3 individually
  - div fuel peb <upbed> 2 u1 u2 divides u1 and u2 individually



Generic FHR (gFHR) model, pebbles divided individually (each color corresponds to a different material)

Pebble beds can now be divided in spatial zones

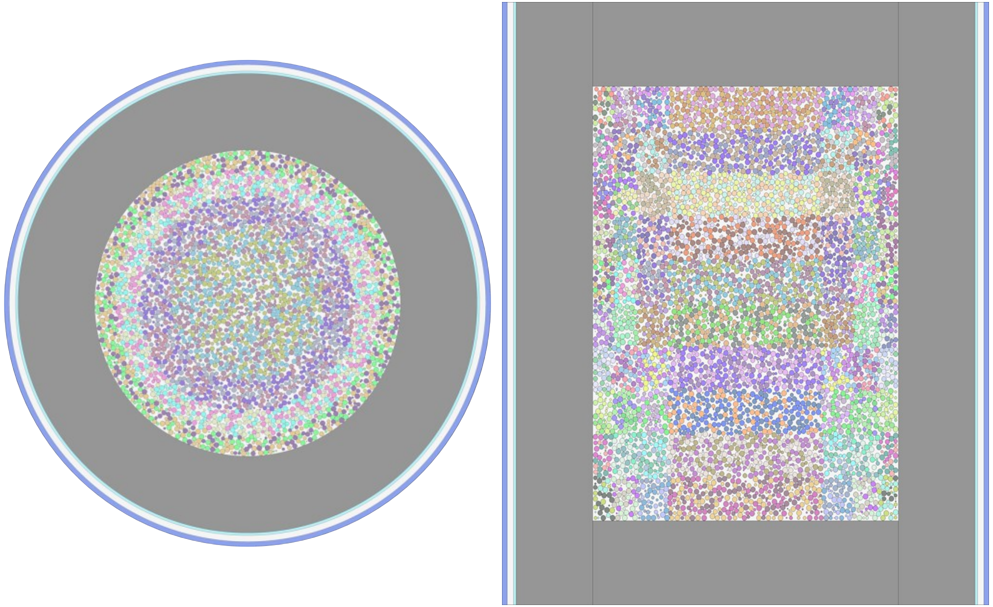
- Before, spatial division on the pebble level
- Now, can be on the pebble bed universe
- Example:
  - div peb <upbed> 0 subz 10 <zmin> <zmax> subr 4 <rmin> <rmax>
  - Divides the pebble bed into 4 radial zones 10 axial zones
  - No individual division



Generic FHR (gFHR) model, pebbles divided into 10 axial and 4 radial zones (each color corresponds to a different material)

Pebbles can be divided based on their indices (modulo division)

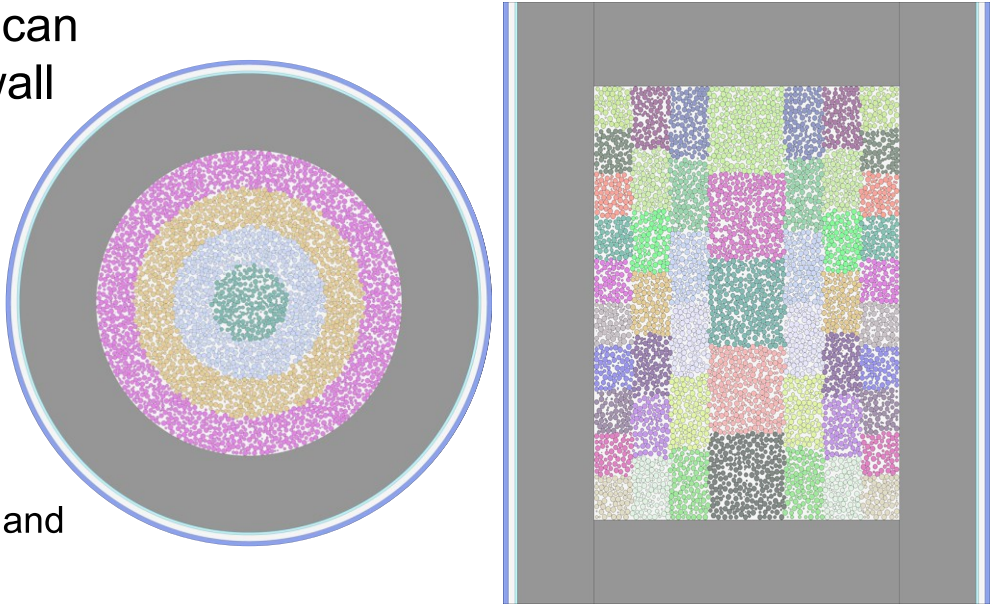
- It was difficult to divide pebbles to replicate random pass numbers
- Modulo division: adds an indexing level based on pebble index.
- Example:
  - div peb <upbed> 0 submod 3
  - Divides the pebble bed into 3 modulo zones.
  - Pebbles 0, 3, 6, … have the same material, same for 1, 4, 7, … and 2, 4, 8
  - No individual division
- Can be associated with spatial division



Generic FHR (gFHR) model, pebbles divided into 10 axial, 4 radial, and 3 passes (modulo) zones (each color corresponds to a different material)

A RZ division is under development

- Pebbles in different radial zones can have different residence times: wall effects
- Zone creation for complex geometries made simple: converging regions
- Define radial channel with:
  - Individual number of subzones
  - Possibility to merge subzones
  - r = f(z) outer limit polynomial dimension and coefficients

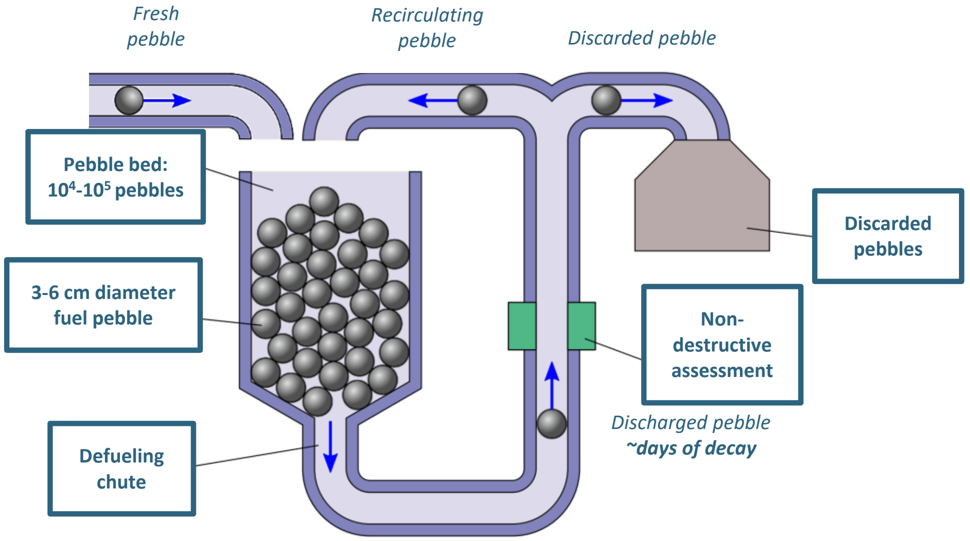


Generic FHR (gFHR) model, pebbles divided into 4 radial zones, with 5,6,7 and 10, zones (each color corresponds to a different material)

Examples of applications

Hyper-fidelity can follow pebbles depletion and motion individually

- Pebbles division into individual depletion zones
- Between each burnup step: motion step (DEM or other)
- Recirculating pebbles:
  - Test for discarding
  - Possible decay step
- Discarded pebbles:
  - Replaced with fresh fuel: reset
  - Store information
- Key parameters saved in csv, hdf5, etc. files

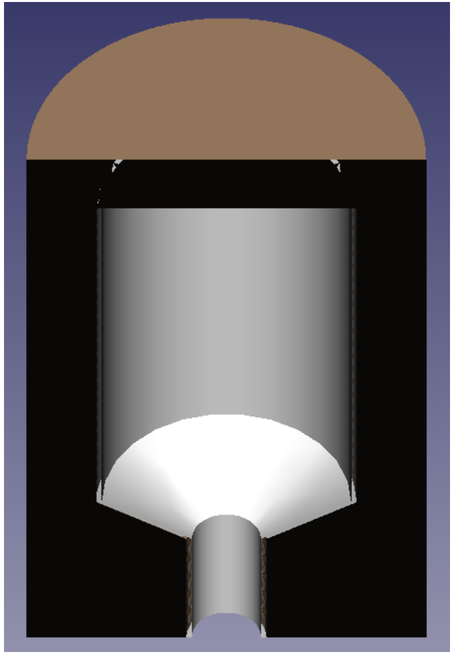


Schematic view of multipass PBR operation

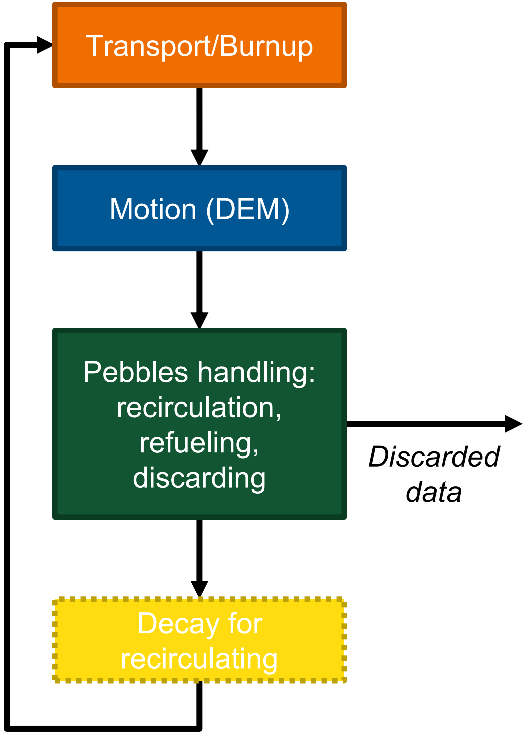
Note: Motion is inverted for FHRs
Initial figure: Tang, Nuclear Engineering and Design 2019

Hyper-fidelity example with small pebble bed reactor

- 5000 pebbles, HTGR
- STL geometry
- Pebbles positions: DEM
- Pebbles individually divided
- No decay when recirculating here
- Discarded pebbles based on burnup (50 MWd/kg)
- Python library written for reading/processing/plotting

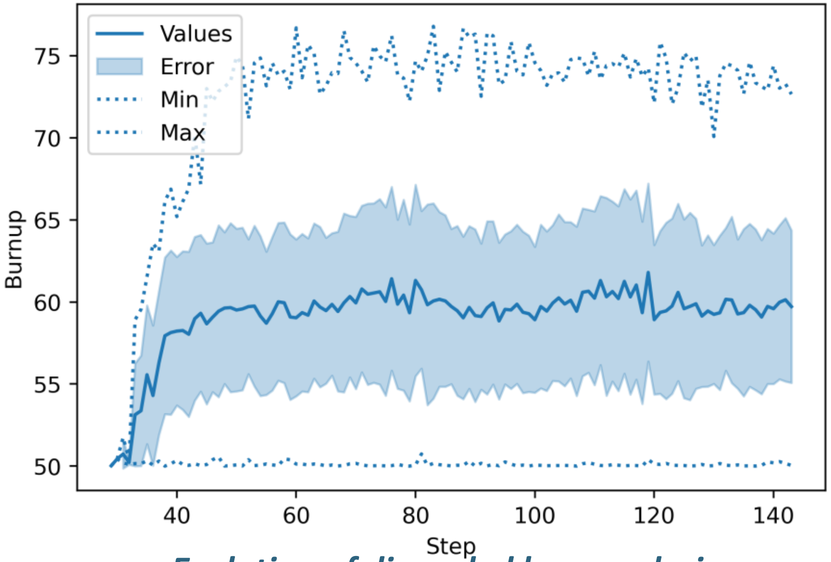


Small PBR STL core/reflector models

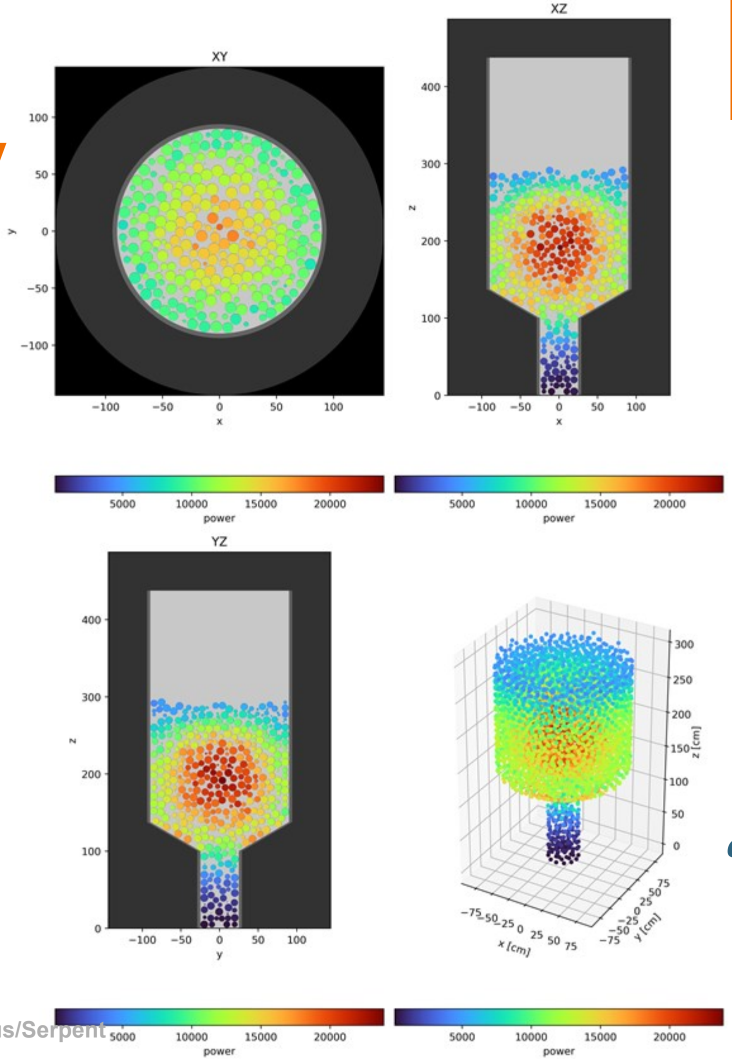


Hyper-fidelity flowchart

Hyper-fidelity preliminary results



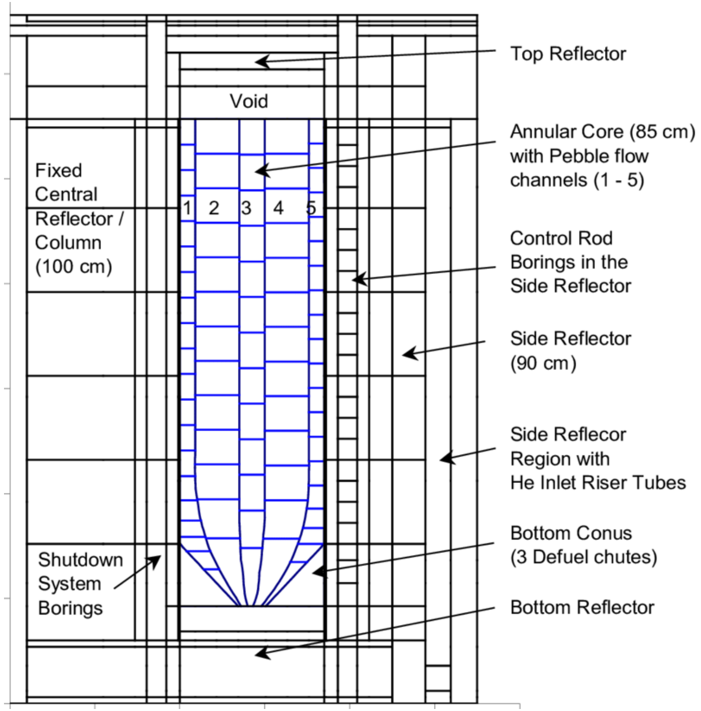
Evolution of discarded burnup during approach to equilibrium



Power at equilibrium (values are indicative)

Macro-zones and search for equilibrium is a quicker lower-fidelity approach

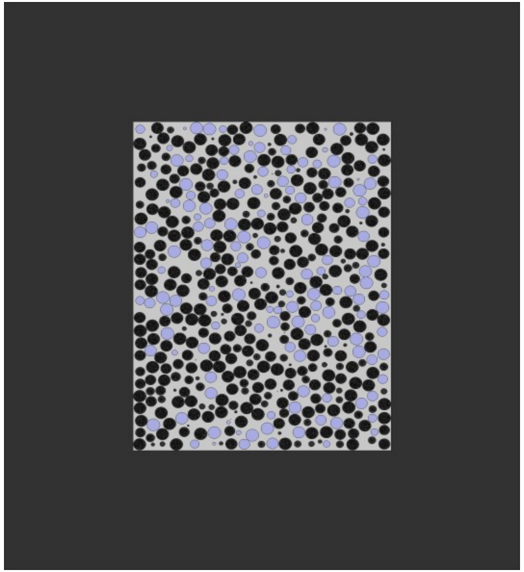
- Core divided into spatial zones
- Circulation matrix controlling composition flow with Cerberus
- Recirculating compositions:
  - Possible decay step
- Discarded composition:
  - Store mixed composition at last pass/last axial zone
- Approach to equilibrium



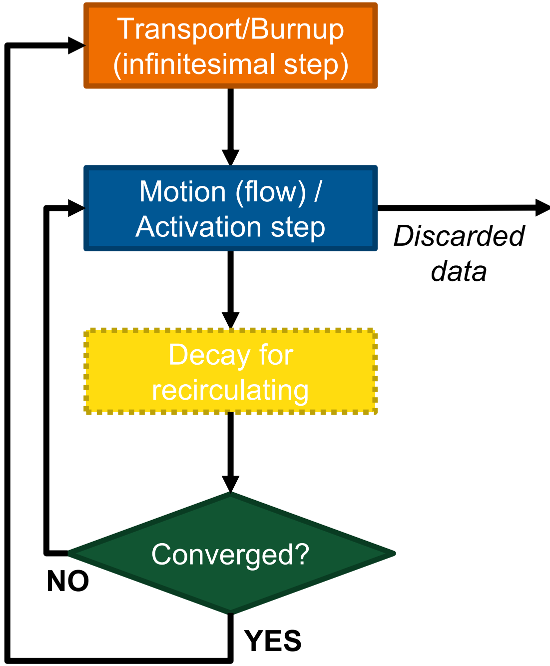
PBMR VSOP model (Reitsma, PHYSOR 2004)

Macrozones example with small pebble bed reactor

- 5000 pebbles, FHR
- 6 radial, 10 axial material zones
- 4 passes:
  - Refueling from the bottom
  - Discarding at the top
  - Continuous motion up for the rest
- Two types: fuel and graphite pebbles
- Circulation matrix mimics mflow
- Simple motion
- Combination of burnup and activation steps

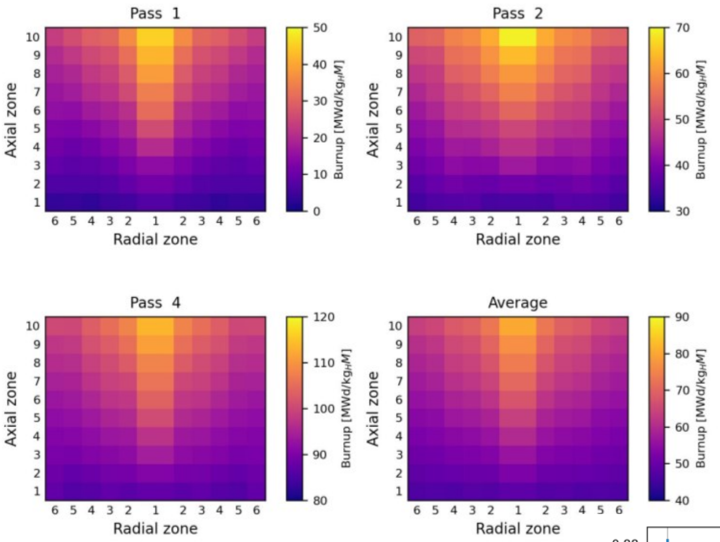


Small PBR Serpent model

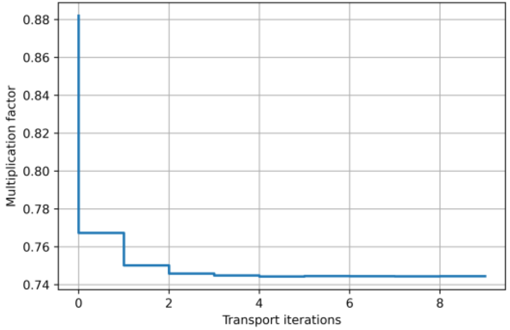


Macrozones approach flowchart

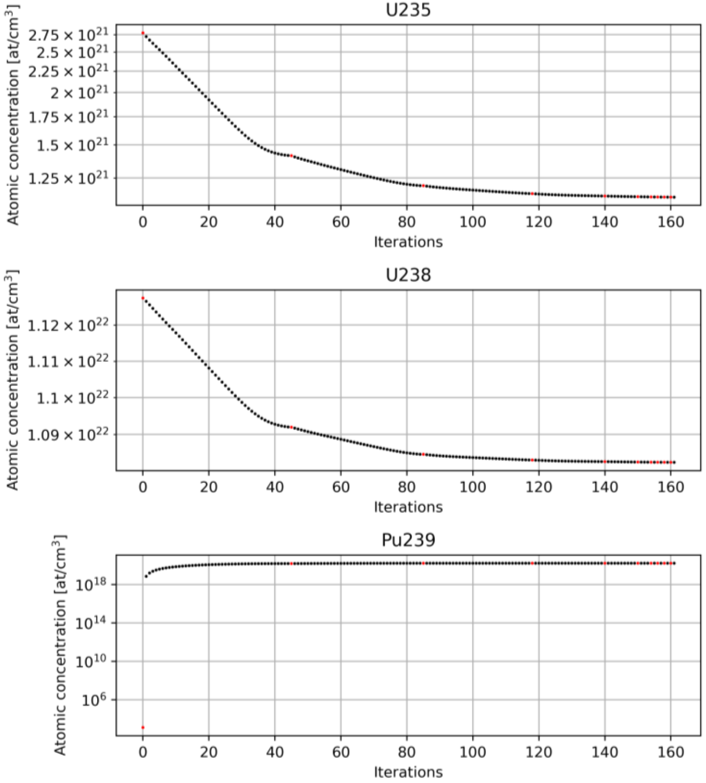
Macrozones approach preliminary results



Burnup at equilibrium (values are indicative)



Multiplication factor throughout transport iterations



Discarded atomic concentrations (red: transport steps)

VTT

Conclusions

- New features developed to support the growing need for pebble bed reactors simulations:
  - Additional input/output variables regarding depletion and pebble bed specifics
  - Update and re-plot geometry
  - Use of domain decomposition with Cerberus and user-defined domains
  - Revised pebble bed division for more control and faster creation
- Possibility in the future to couple Serpent and thermal solver (GenFOAM, Goose, Pronghorn, etc.)
- Applicability to hyper-fidelity depletion coupled with DEM, or macrozones approach
- Some features can be applied to other types of reactors

Macrozones → **Hyper-fidelity**

Accuracy/Time

Thank you
Questions?