Business from technology

Introduction to Serpent

Jaakko Leppänen
VTT Technical Research Centre of Finland

Outline

- Background – Monte Carlo method in reactor analysis
- Brief history of Serpent development
- Methods:
  - Geometry and neutron tracking
  - Interaction physics
  - Burnup calculation
  - Group constant generation
- Current status of Serpent development
- Serpent user community
- This workshop

Background – Monte Carlo method in reactor analysis

- Computational modeling of an operating nuclear reactor relies on a two-stage calculation scheme:

  1) Homogenization at fuel assembly level – production of few-group interaction parameters for every assembly type and state point in the reactor core
  2) Simulator calculation – coupled neutronics / thermal hydraulics calculation, typically based on nodal diffusion methods

- The reason for this approach is the complexity of the task – we do not have the calculation tools or the computing power to solve the full-scale transport problem without approximations

Background – Monte Carlo method in reactor analysis

- Apart from criticality calculations, Monte Carlo neutron transport codes are not widely used in reactor analysis
- Instead, Monte Carlo codes are traditionally used for:
  - Detector modeling
  - Radiation shielding and dosimetry
  - Validation of deterministic transport codes
- Monte Carlo codes are often viewed as general-purpose tools that are used when deterministic transport methods cannot be applied
- Because of this role, the calculation methods are designed as versatile as possible, without considering the *specific* requirements of any particular application

Background – Monte Carlo method in reactor analysis

- It is easy to see that the use of the Monte Carlo method could, at least in principle, be extended to homogenization, burnup calculation and other reactor physics applications
- But what about in practice?
  - Can we really produce *all* homogenized group constants required for simulator calculations?
  - Are there any real advantages over the presently-used, computationally more efficient deterministic lattice codes?
  - If the method is applicable in theory, are the codes up for the task in practice?
  - If we have the methods and codes, do we really have the computer capacity?

Background – Monte Carlo method in reactor analysis

Lattice physics applications:

- Relatively simple assembly-level geometry in 2D
- Large number of standard output parameters, similar requirements for every calculation case
- Main goal is to generate input data that represents the neutron physics in the next stage of the calculation chain
- The same calculation is repeated for every assembly type in different reactor operating conditions
- Thousands of runs → computing time becomes a limiting factor

Traditional MC applications:

- Complicated 3D geometries
- Criticality eigenvalue and/or user-defined tallies, requirements depend on case
- Main goal is to simulate the neutron transport process in a realistic geometry to within maximum accuracy
- Often a single run with a few parameter variations
- Accuracy is usually more important than computing time

Background – Monte Carlo method in reactor analysis

- So why use Monte Carlo codes for group constant generation in the first place?
  - Capability to model geometry and physics without major approximations.
  - Capability to use the best available knowledge on neutron interactions without major approximations.
  - The same code and data can be used with any fuel or reactor configuration without application-specific approximations.
  - The calculation chain from neutron interactions to full-core simulation could become simpler and more transparent.
  - The method could open up new possibilities in reactor modeling

Background – Monte Carlo method in reactor analysis

- Practical challenges:
  - Monte Carlo codes are slow, especially when used for burnup calculation.
  - Combining the Monte Carlo simulation with the deterministic calculation chain requires special techniques:
    - How to calculate diffusion coefficients?
    - How to perform critical spectrum calculations?
    - How to perform adjoint calculation?
    - How to calculate current integrals?
    - How to calculate differential quantities?

Background – Monte Carlo method in reactor analysis

- Examples of previous work in the field:
  - R. C. Gast (1981): Calculation of diffusion coefficients using Monte Carlo code RCP01
  - E. L. Redmond II (1997): Homogenized multi-group cross sections and Legendre scattering cross sections using MCNP4B
  - H. R. Trellue (1999): Monteburns - probably the first Monte Carlo burnup calculation code, based on the coupling of MCNP4C and ORIGEN
  - Cetnar et al. (1999): MCB - a burnup extension to MCNP4C.
  - G. Ilas and F. Rahnema (2003): Diffusion coefficients using MCNP for spent fuel storage lattice calculations

Background – Monte Carlo method in reactor analysis

- M. Tohjoh et al. (2005): BWR group constants using MVP-BURN
- Pounders (2006): Calculation of diffusion coefficients using Monte Carlo
- S. C. van der Marck et. al. (2006): Homogenized group constants for Petten High Flux Reactor using MCNP
- J. Leppänen (2007): Development of the PSG code for homogenized multi-group constant generation.
- Shim et al. (2008): Calculation of group constants using the McCARD code, including methodology to account for the critical spectrum correction.

- So Serpent is not the first and only Monte Carlo code used for homogenization, but probably one of the first codes specifically developed for the task

Serpent – a brief history

- The development of Serpent started in September 2004, under the working title “Probabilistic Scattering Game”, or PSG
- The first idea was to see whether it was possible to develop a simplified Monte Carlo neutron transport code for group constant generation
- The first version consisted of a simplified physics model and a simple pin-lattice geometry routine:
  - Cross sections read from ACE format data files and reconstructed on a uniform lethargy-width energy grid
  - Geometry routine based on Woodcock delta-tracking
  - Output: k-eff, homogenized group constants
  - Development carried out as free-time activity

Serpent – a brief history

- The results were getting close to reference MCNP5 calculations by early 2005, and it was decided to take a more serious approach:
  - Source code was completely re-written
  - Improved geometry model for the treatment of localized heavy absorbers (a bottleneck for the delta-tracking routine)
  - Improved physics → differences to MCNP below 1%
  - Parallel calculation mode using MPI
- The project was officially included in the VTT research programme in the spring of 2005
- Topic for a D.Sc. thesis (completed in 2007)[1]

[1] J. Leppänen, *"Development of a New Monte Carlo Reactor Physics Code."* D.Sc. Thesis, Helsinki University of Technology, 2007.

Serpent – a brief history

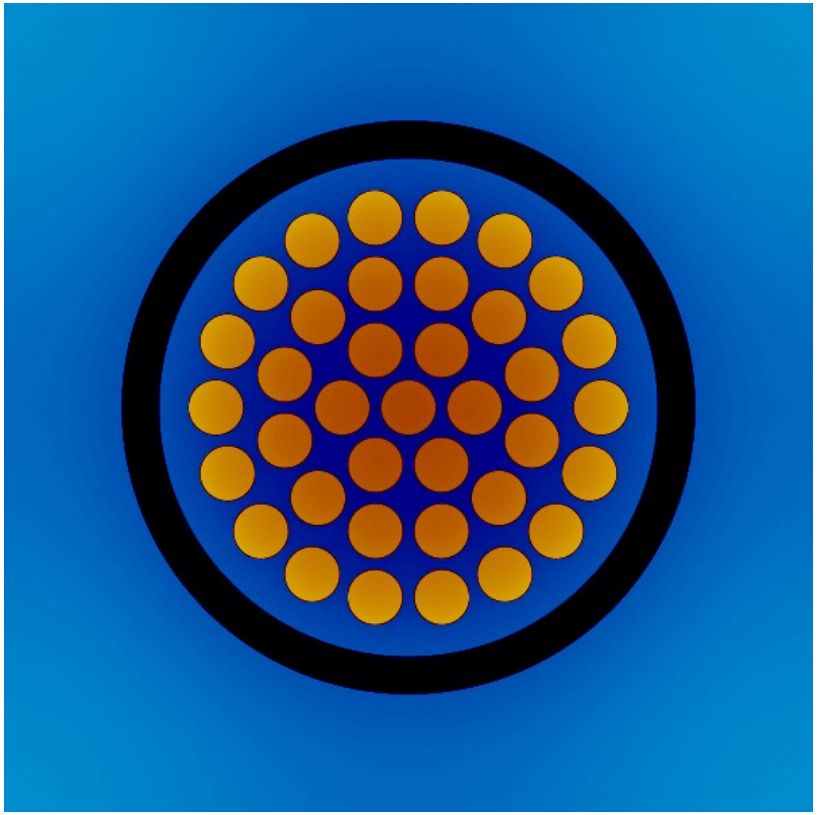
- Current version of Serpent began to formulate when the source code was re-written for the second time by the end of 2008:
  - More improvements in physics → differences to MCNP to the level of statistical accuracy
  - New geometry routine based on the combination of delta-tracking and conventional surface tracking
  - Implementation of built-in burnup calculation routines (TTA and CRAM) by the end of 2008
  - Name changed to “Serpent” and public distribution in 2009
- Some of the methods, originally implemented for the sake of simplicity (delta-tracking, unionized energy grid), have later turned out particularly efficient in lattice physics applications

Methods – overview

- Serpent can be characterized as a three-dimensional continuous-energy Monte Carlo reactor physics code
- Specialized in calculations at the fuel assembly level:
  - Built-in capabilities for group constant generation and burnup calculation
  - Calculation routines optimized for lattice physics applications
- However:
  - Geometry model is not limited to two-dimensional problems
  - Additional features similar to general-purpose codes: external source simulation mode, user-defined detectors (tallies) available for calculating integral reaction rates, etc.

Methods – geometry and neutron tracking

- Three-dimensional universe-based geometry model (similar to MCNP and KENO-VI)
- Additional geometry types for reactor physics calculations:
  - Cylindrical fuel pin and spherical particle
  - 4 basic lattice types
  - Explicit model for stochastic HTGR geometries
- Criticality and external source simulation modes
- Transport simulation limited to neutrons



Methods – geometry and neutron tracking

- Tracking routine based on Woodcock delta-tracking method:
  - A rejection sampling technique that enables neutron path lenghts to be extended over one or several material boundaries
  - Significant reduction in CPU time in geometries where the dimensions are small compared to neutron mfp
  - Necessitates the use of collision flux estimator for calculating integral reaction rates (instead of track-length estimator)
  - Good efficiency in reactor calculations, but poor statistics when integrated over small volumes (detector modeling, etc.)
  - Used in combination with conventional surface-tracking to improve performance in the presence of localized heavy absorbers

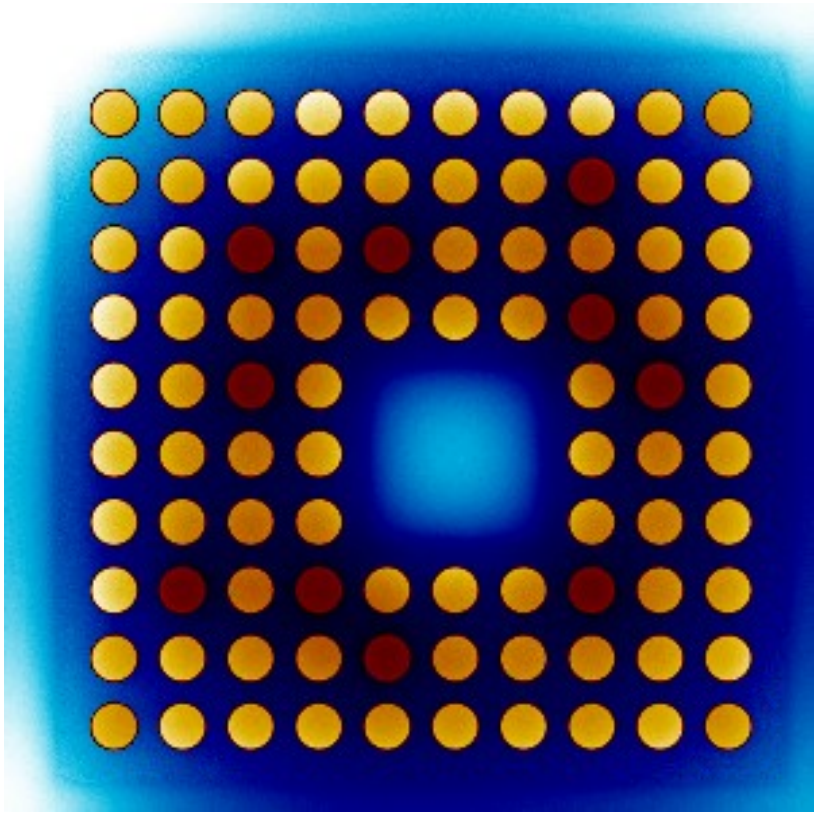
Methods – interaction physics

- Serpent reads continuous-energy cross sections from ACE format data libraries:
  - Cross section libraries generated using NJOY
  - Interaction physics modeled according to classical collision kinematics and ENDF reaction laws
  - Thermal scattering libraries for important moderator materials
  - Probability table sampling in the unresolved resonance region
- Data format is shared with MCNP → Serpent can be validated by direct comparison to reference results without additional uncertainties from nuclear data

Methods – interaction physics

- All reaction cross sections are reconstructed on a unionized energy grid:
  - Combination of nuclide-wise energy grids into a single large grid, with optional grid thinning to reduce the number of points
  - Material-wise total cross sections pre-calculated before the transport cycle
  - An extremely efficient way of saving CPU time in calculations involving a lot of nuclides (burnup calculation)
  - Significant increase in memory demand, which eventually becomes a major limitation
  - More about this topic in tomorrow's presentation

Methods – burnup calculation



- Fully-automated, built-in depletion routines for burnup calculation:
  - Transmutation cross sections calculated during the transport cycle (2 methods)
  - Radioactive decay and fission yield data read from ENDF format files
  - Decay / transmutation chains generated automatically without user effort
  - Two methods for solving the Bateman depletion equations
  - Option for predictor-corrector calculation

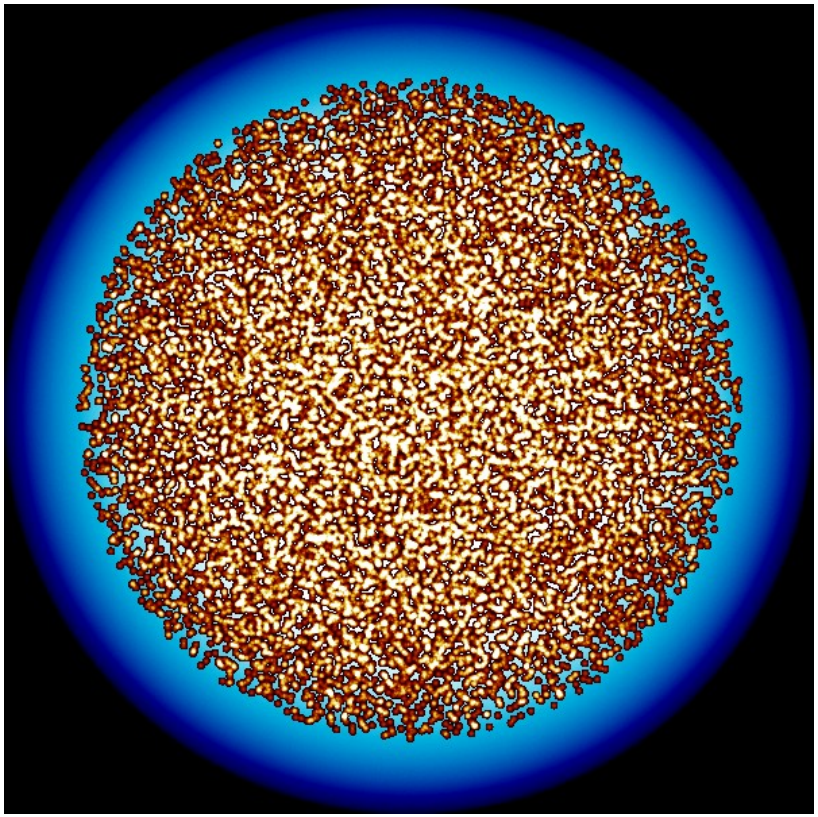
Methods – group constant generation

- Serpent has the capability to produce all input parameters needed for few-group nodal diffusion calculations:
  - Homogenized multi-group cross sections
  - Scattering matrices
  - Point-kinetic parameters (forward flux-weighed)
  - Assembly discontinuity factors
  - Pin-wise power distributions
  - Effective delayed neutron fractions (Meulekamp's method)
  - Diffusion parameters
- Newly implemented capability to homogenize group constants in critical spectrum (B1 fundamental mode calculation)

Methods – additional output and tallies

- Fission source entropy for convergence studies
- User-defined detectors (tallies) for calculating volume-integrated reaction rates:
  - Spatial domain defined by a combination of cells, universes, lattices and materials or using a super-imposed 3D mesh
  - Arbitrary energy bin structure
  - Various response functions (material total and isotopic reaction cross sections, ACE format dosimetry data)
  - Differential and integral flux and reaction rate spectra
- Geometry and thermal flux / fission rate plotter producing png-format graphics

Methods – HTGR capabilities

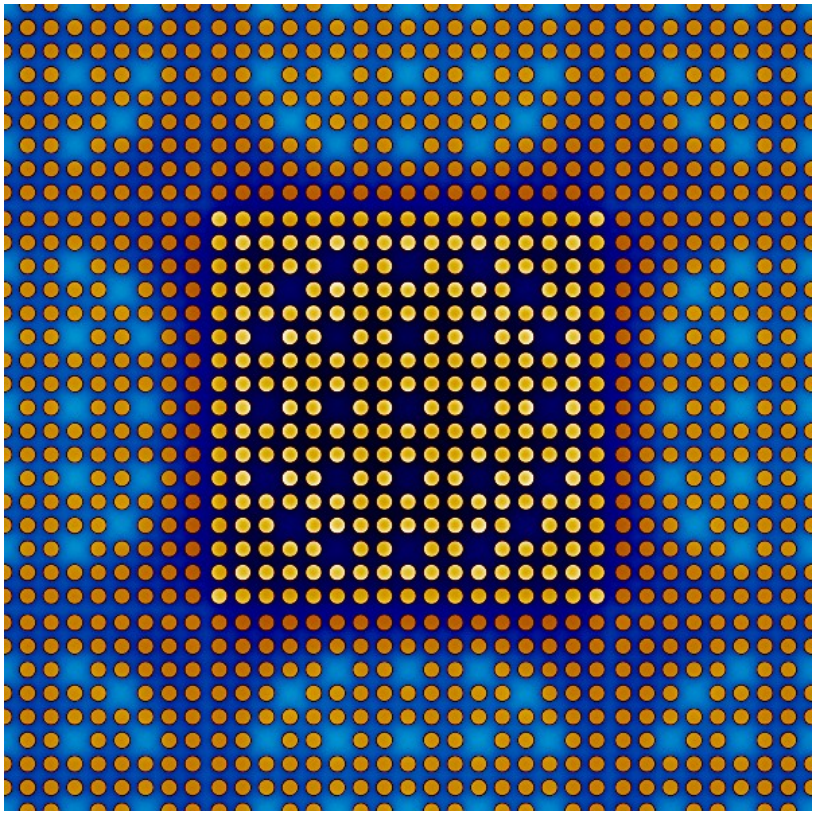


- The woodcock delta-tracking method works exceptionally well in HTGR particle fuel geometries
- Explicit model for stochastic geometries:
  - Coordinates of spherical objects are read from a separate input file
  - Works at several levels (TRISO particles in graphite / fuel pebbles in pebble-bed core)
  - Tested with over 2 million random units, no major increase in calculation time compared to a regular-lattice calculation
  - Pebble-wise power distributions printed in separate output files

Methods – additional capabilities

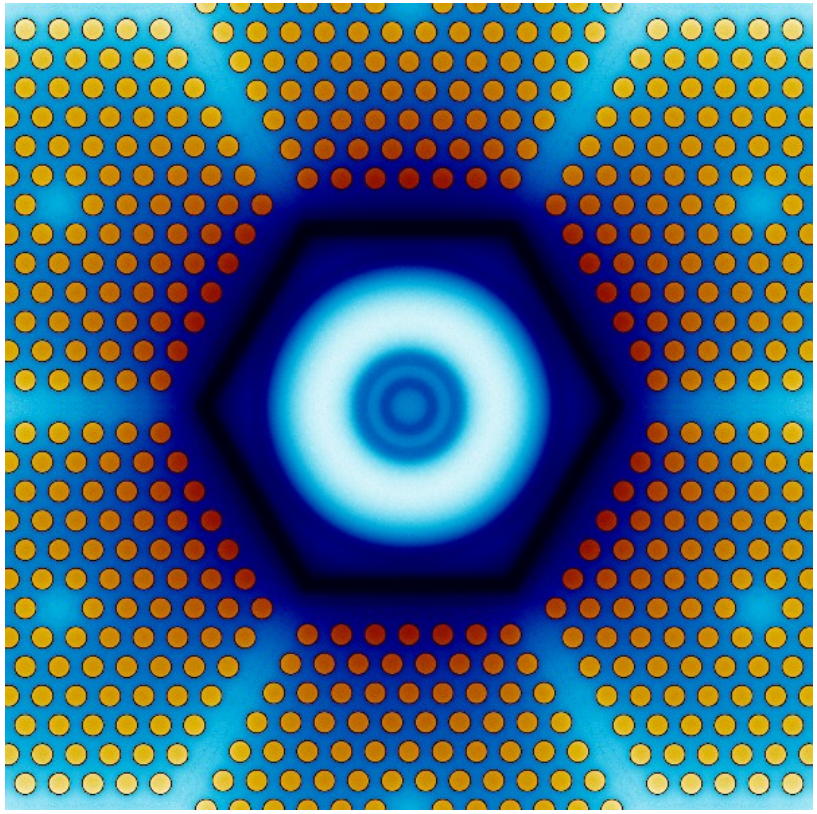
- Built-in Doppler-broadening preprocessor routine
- Doppler-broadening rejection correction (DBRC) method for resonant scattering kernels
- Iteration of equilibrium xenon concentration (in both transport and burnup calculation modes)
- Iteration of soluble absorber concentration (in both transport and burnup calculation modes)
- Parallel calculation mode based on distributed memory using the Message Passing Interface (MPI)
- Serpent installation package contains ACE format cross section libraries generated using NJOY for 432 nuclides at 6 temperatures based on ENDF/B-VI, ENDF/B-VII, JEF-2.2, JEFF-3.1 and JEFF-3.1.1 evaluated nuclear data files

Typical applications



- Serpent is specifically developed for:
  - Generation of homogenized multi-group constants
  - Fuel cycle studies involving assembly-level burnup calculations
- But also:
  - Research reactor applications
  - Validation of deterministic transport codes
  - Educational purposes
  - Versatile Platform for testing and developing new methods and ideas
- Examples of user applications: http://montecarlo.vtt.fi/download/VTT-R-01362-11.pdf

Website and discussion forum



- A complete and up-to-date description of the project, including manual and methodological references, can be found at Serpent website:

  http://montecarlo.vtt.fi

- A discussion forum for Serpent users:

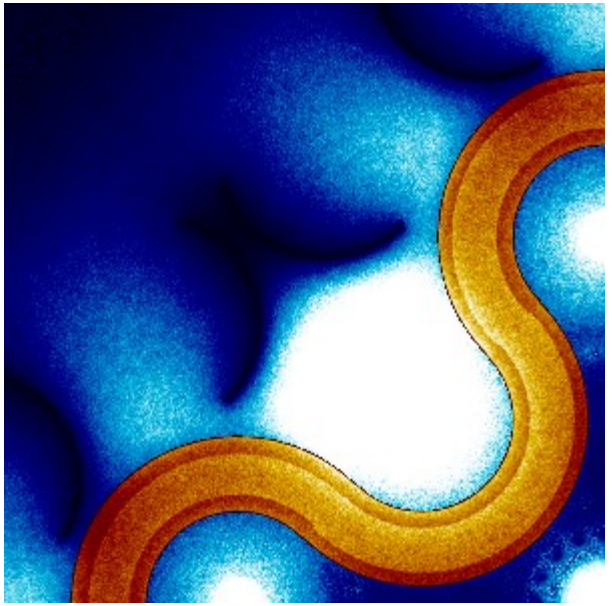
  http://ttuki.vtt.fi/serpent/

- Code development is a continuous effort, and currently focused on Serpent 2, which is the topic of tomorrow's introductory presentation

Current status of Serpent development

- Serpent source code consists of about 220 subroutines and almost 80,000 lines of C-code.
- Development is currently funded from the Finnish national research programme on nuclear power plant safety (SAFIR-2014)
- SAFIR-2014 / KÄÄRME -project:
  - Four-year project started in 2011, with total budget of 955 k€
  - 16 person-months of work allocated for 2011, divided between three scientist:

    Jaakko Leppänen
    Maria Pusa
    Tuomas Viitanen



Current status of Serpent development

- Additional funding from EU HPMC project (High Performance Monte Carlo Reactor Core Analysis):
  - Joint EU project with KTH, KIT and Delft Nuclear Consultancy
  - Three-year project starting in October 2011, with total budget of 209 k€ reserved for VTT
  - Main focus (for us) will be on multi-physics, and the work will combine Serpent development and VTT's experience in TH modeling
- In addition to VTT, some significant code development has been carried out in collaboration with universities and other research organizations*

* It should be noted that Serpent is not an open-source project, despite the collaboration in code development. Distribution is subject to export control limitations, and the software license agreement practically prohibits independent code development outside VTT.

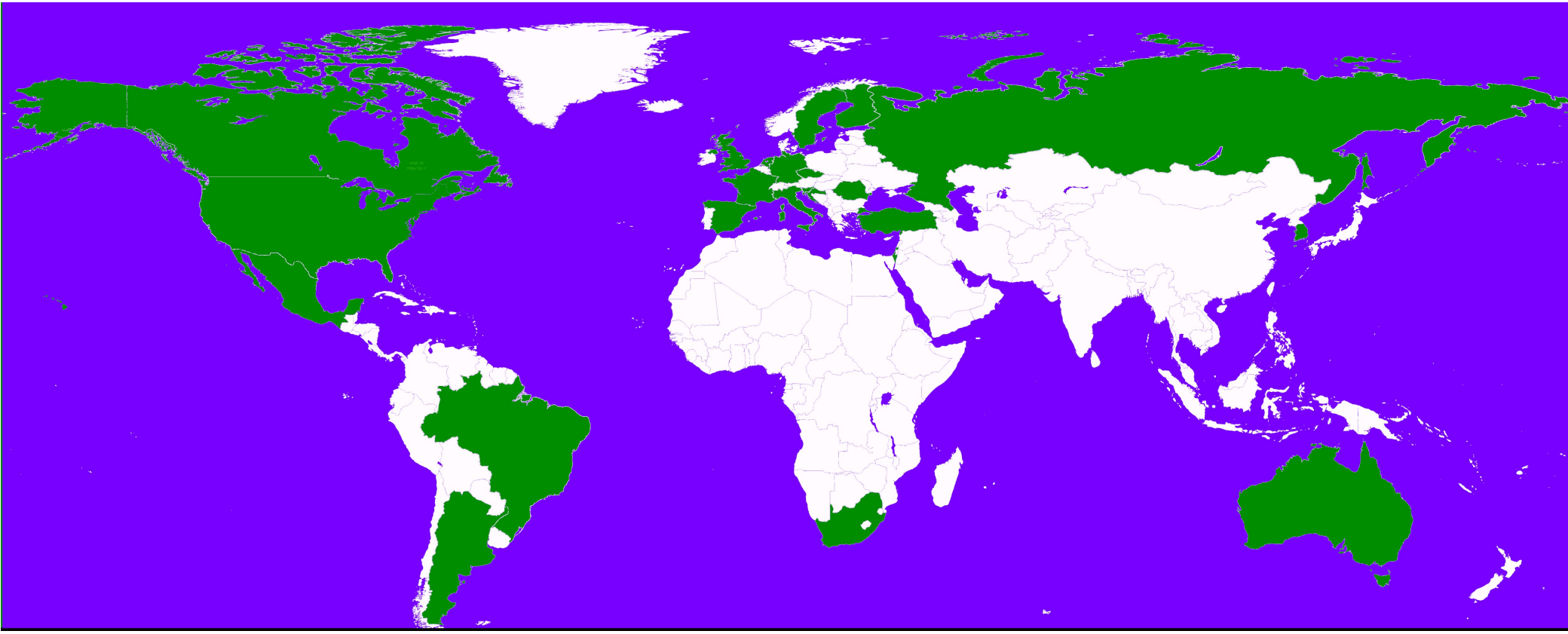
Serpent user community

- Serpent source code for base version 1.1.7 is distributed free of charge for non-commercial research and educational use by two data centers:

  OECD/NEA Data Bank (since May 2009)
  RSICC (since March 2010)

- Updates are distributed to registered users by e-mail (current update 1.1.16, distributed Sept. 5, 2011)
- Serpent is used in 53 organizations in 24 countries around the world
- 115 registered users, 78 members in discussion forum

Serpent user community



This workshop – participants and presentations

- This is the first international workshop / user group meeting for Serpent users
- We have 35 participants from 17 organizations in 10 countries: *Finland, Germany, Israel, Italy, the Netherlands, South Africa, Spain, Sweden, Switzerland and the U.S.*
- Topics for day 1 are related to applications and methods in Serpent 1:
  - Group constant generation
  - Next-generation reactors: SFR, thorium cycle, HTGR, high-conversion reactors
  - Methods and physics

This workshop – participants and presentations

- Topics for day 2 are related to the development of Serpent 2:
  - Current status and near-term plans
  - New methods and capabilities
  - Possibilities for future development
- Each presenter has 30 minutes, including questions and discussion, but let's keep things flexible and informal
- Presentations will be made available at the Serpent website, so let me know if you don't want your atomic secrets in public distribution!

This workshop – topics for discussion

- Tomorrow afternoon is reserved for discussion. Feel free to ask questions and talk about whatever you like, but here are a few topics I would like bring to your attention:

  1) Group constant generation in non-fissile zones – how to generate group constants for reflectors and other regions without source points? How is it done with deterministic codes?

  2) Parallelization – Tomorrow I will talk about parallelization of Serpent 2 using OpenMP. The scalability is not very impressive. Why is that?

This workshop – topics for discussion

3) Production runs – How to manage the calculations and data flow needed for actual group constant generation? Should we have a separate project for developing a common driver program for this task?

4) Scripts – Many users are writing their own processing scripts for managing input and output. Should we have some coordinated collaboration instead of re-inventing the wheel over and over again?

5) Multi-physics – This is a hot research topic that could take whole afternoon, but what I would really like to talk about is the development of a *universal interface* for communication between codes. What should it look like?

Thank you for listening!

and

Welcome to the first international Serpent User Group Meeting!