A SERPENT2-SUBCHANFLOW-TRANSURANUS coupling for pin-by-pin depletion calculations in Light Water Reactors

Garcia, M.; Tuominen, R.; Gommlich, A.; Ferraro, D.; Valtavirta, V.; Imke, U.; van Uffelen, P.; Mercatali, L.; Sanchez, V.; Leppänen, J.; Kliem, S.;

Originally published:

November 2019

Annals of Nuclear Energy 139(2020), 107213

DOI: https://doi.org/10.1016/j.anucene.2019.107213

A Serpent2-SUBCHANFLOW-TRANSURANUS coupling for pin-by-pin depletion calculations in Light Water Reactors

Manuel García[a,*], Riku Tuominen[b], Andre Gommlich[c], Diego Ferraro[a], Ville Valtavirta[b], Uwe Imke[a], Paul Van Uffelen[d], Luigi Mercatali[a], Victor Sanchez-Espinoza[a], Jaakko Leppänen[b], Sören Kliem[c]

[a] *Karlsruhe Institute of Technology, Institute of Neutron Physics and Reactor Technology, Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen, Germany*
[b] *VTT Technical Research Centre of Finland Ltd., P.O. Box 1000, FI-02044 VTT, Finland*
[c] *Helmholtz-Zentrum Dresden-Rossendorf, Institute of Resource Ecology, Bautzner Landstraße 400, 01328 Dresden, Germany*
[d] *Joint Research Centre, European Commission, Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen, Germany*

Abstract

This work presents the development of a coupling scheme for Serpent2, a continuous-energy Monte Carlo particle transport code, SUBCHANFLOW, a subchannel thermalhydraulics code, and TRANSURANUS, a fuel-performance code, suitable for large-scale high-fidelity depletion calculations for Light Water Reactors. The calculation method is based on the standard neutronic-thermalhydraulic approach, replacing the simple fuel-rod solver in SUBCHANFLOW with the more complex thermomechanic model of TRANSURANUS. The depletion method is fully coupled and semi-implicit, and the implementation relies on an object-oriented design with mesh-based feedback exchange. The results of the three-code system for a 360-day depletion calculation of a VVER-1000 fuel assembly with a pin-by-pin modelling approach are presented and analyzed. The performance of this tool, as well

as the bottlenecks for its application to full-core problems, are discussed.
Keywords:
Serpent2, SUBCHANFLOW, TRANSURANUS, Multiphysics, LWR

1. Introduction

Driven by the growing interest in the nuclear industry in high-fidelity simulations for design and safety analysis of nuclear reactors, the EU Horizon 2020 McSAFE project [1] was set to tackle the implementation of multiphysics tools based on the Monte Carlo particle transport method. The general objective of the project is to improve the prediction of local safety parameters at pin level in Light Water Reactors (LWRs) solving large-scale pin-by-pin depletion and transient problems.

In this framework, a coupling scheme for Serpent2 [2], a continuous-energy Monte Carlo code, and SUBCHANFLOW (SCF) [3], a subchannel thermalhydraulics code, has been developed and tested for PWR [4] and VVER [5] problems. This tool relies on the traditional neutronic-thermalhydraulic iterative approach and is based on an object-oriented design with mesh-based feedback exchange. Each code is modularized using a well-defined interface structure and the coupling is implemented in a supervisor program independent from the solver modules.

In order to add fuel-performance analysis capabilities to this coupled system, the TRANSURANUS (TU) code [6] has been included, and a first version of Serpent2-SCF-TU has been developed and is being tested and

*Corresponding author
Email address: manuel.garcia@kit.edu (Manuel García)

optimized. The coupling of TU with Serpent2-SCF is based on replacing the simplified fuel model in SCF with the more sophisticated thermomechanic model in TU, and relies on the same object-oriented methodology. This three-code coupling is to be applied to large-scale steady-state and depletion problems, with the aim at performing fully coupled full-core pin-by-pin simulations for LWRs.

The motivation to add fuel-performance and thermomechanic analysis to the traditional neutronic-thermalhydraulic approach is manifold. Firstly, the detail of the solution for the fuel rods is greatly increased, and safety-relevant phenomena such as pellet-cladding interaction and fission-gas release can be accurately modelled. In addition, the calculation of fuel temperature profiles is improved by using more sophisticated models for the fuel-cladding gap and for the thermomechanic behavior. This in turn can potentially improve the Doppler feedback for the neutronic calculation.

The objective of this work is to describe and discuss in detail the coupling methodology for this three-code system, as well as to present results showing the current capabilities and future challenges.

The paper starts with a brief description of the three codes in Section 2, focusing on the methods and features relevant for the rest of the discussions. The inheritance-based modularization of the codes and the mesh-based feedback exchange, the two key aspects of the implementation approach, are presented as well.

Section 3 explains the coupling methodology in full detail. The scheme to replace the fuel solver in SCF by TU is described first, since it is the most innovative aspect of the coupling. The fully coupled depletion method is then presented, along with its implementation in the supervisor program.

To show the current capabilities and analyze the methodology, a depletion calculation for a VVER-1000 fuel assembly is presented in Section 4. The pin-by-pin modelling of complex reactor geometries in the three-code system is presented and the general implementation of the feedback and depletion schemes is verified. The results are compared with Serpent2-SCF without TU to analyze the differences and potential advantages of the new scheme. These results set the stage for the further application of Serpent2-SCF-TU to large-scale depletion problems, including full-core analysis. With this in mind, the performance of the coupled system and its current bottlenecks are examined.

2. Description of the codes

This section presents a brief description of the three codes used in this work, with emphasis on their multiphysics capabilities. A full description of the features of each code can be found in the references provided. The approach used to modularize the codes using a fixed interface format is described in Section 2.4, along with the exchange of variables between codes through unstructured meshes.

2.1. Serpent2

Serpent2 [2] is a continuous-energy Monte Carlo particle transport code for steady-state, depletion and transient calculations. The definition of the geometry for the neutronic simulation can be done using Constructive Solid Geometry (CSG), unstructured meshes and stereolithography (STL)-based geometries, such that the creation of virtually any reactor geometry is possible. For the solution of the Bateman equations coupled with the

transport simulation, Serpent2 has several schemes based on the predictor-corrector method and an implementation of the Stochastic Implicit Euler (SIE) method [7].

The feedback fields in multiphysics applications are not defined directly in the geometry used to track particles, but rather using meshes superimposed to the model [8]. Each mesh is used to define density and temperature distributions for one or more materials and to tally power. As an example, Figure 1 shows a Serpent2 model for a typical PWR fuel assembly, for which the densities and temperatures of coolant and fuel materials can be defined with a mesh like the one shown in Figure 2. The three-code coupling scheme relies on this capability, in essentially the same way as the Serpent2-SCF coupling does [5].

2.2. SUBCHANFLOW

SUBCHANFLOW (SCF) [3] is a subchannel analysis code capable of performing steady-state and transient calculations, and extensively used and validated for analysis of LWRs. A SCF model is composed of a set of subchannels defined by their hydraulic parameters (flow area, and heated and wetted perimeters) and by connections to other subchannels, and by a set of rods in which power is deposited and transferred to the subchannels. Figure 3 shows a typical subchannel model, in this case for a PWR fuel assembly.

The solution algorithm to calculate the pressure, velocity and temperature fields for the coolant, as well as the fuel temperatures is:

1. Solve the radial fuel temperature profile $T^{ik}_{fuel}(r)$ for each rod i and axial position k, using the cladding-coolant heat-transfer coefficient

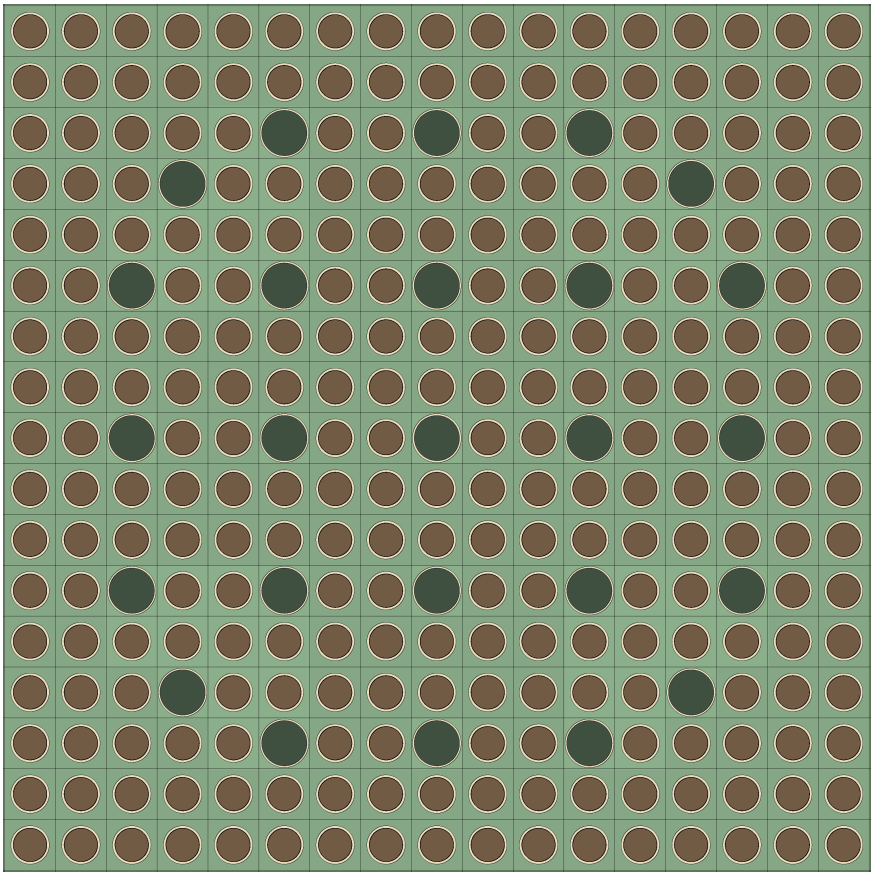


Figure 1: Serpent2 model for a PWR fuel assembly. Guide tubes are shown in green.

$h^{ik}_{clad-cool}$ and the coolant temperature T^{ik}_{cool} as boundary conditions and the power P^{ik} as heat source.

2. Calculate the power going to each subchannel j and axial position k as $q^{jk} = \sum_{i(j)} A^{ij} h^{ik}_{clad-cool}[T^{ik}_{clad} - T^{ik}_{cool}]$, i. e. adding the power from each rod i with a contact area A^{ij} to channel j.

3. Solve the coolant pressure p, velocity v and temperature T_{cool} from the mass, energy and momentum conservation equations.

4. Evaluate the convergence and iterate (go to 1) if needed.

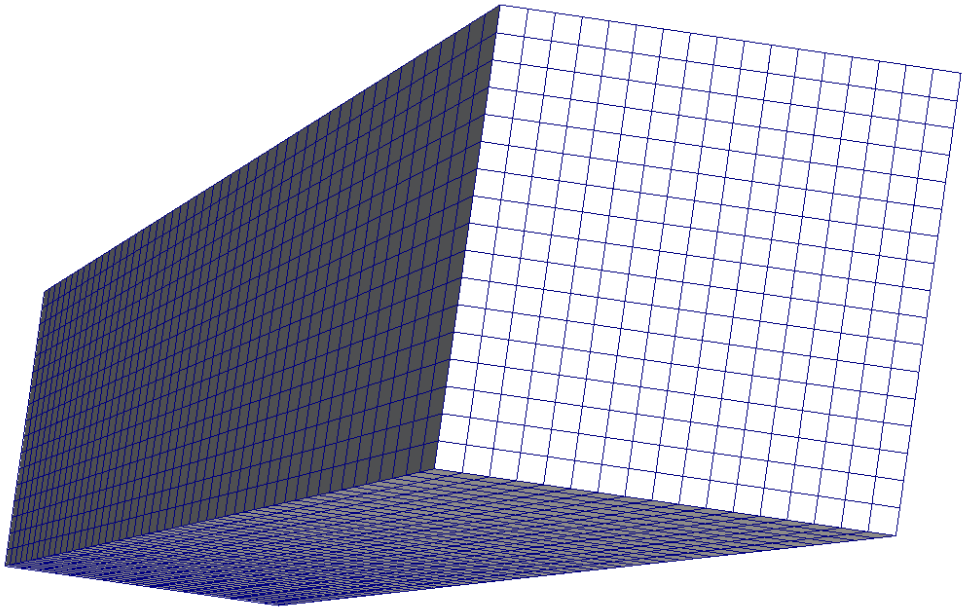


Figure 2: Serpent2 mesh for a PWR fuel assembly.

This method is used for steady-state and transient steps, with the time derivatives set to zero in steady state, as well as for each iteration in multiphysics simulations.

For depletion calculations involving SCF only steady-state solutions are needed, as the time dependence is not modelled explicitly but a quasi-stationary approach is used.

The correlations for the temperature-dependent specific heat capacity c_P, thermal conductivity k, thermal expansion coefficient α and emissivity ϵ used to perform the fuel calculation are taken from MATPRO, version 11 [9]. The gap width is calculated using a simplified fuel-performance model, considering thermal expansion and burnup-dependent fuel relocation by cracking [10] and swelling [9]. For the fuel-cladding gap conductance a simple model considering radiation and conduction through the filling gas

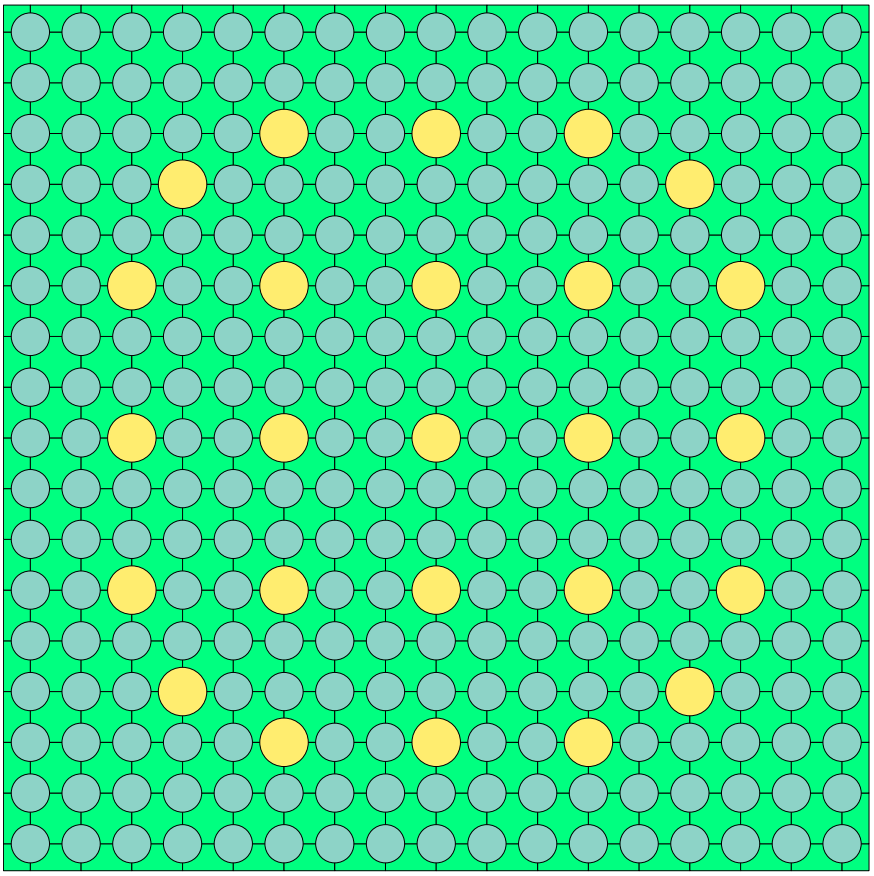


Figure 3: SCF model for a PWR fuel assembly. Guide tubes are shown in yellow.

is used.

As in Serpent2, the exchange of feedback with other codes is done superimposing meshes to the calculation geometry [11]. Two types of meshes are used, one for the coolant and one for the fuel, as shown in figures 4 and 5, respectively, for the case in Figure 3. The coolant mesh is used to get subchannel variables calculated by SCF, such as T_{cool} or p, while the fuel mesh is used to set the power and get rod variables, e. g. T_{fuel} and T_{clad}.

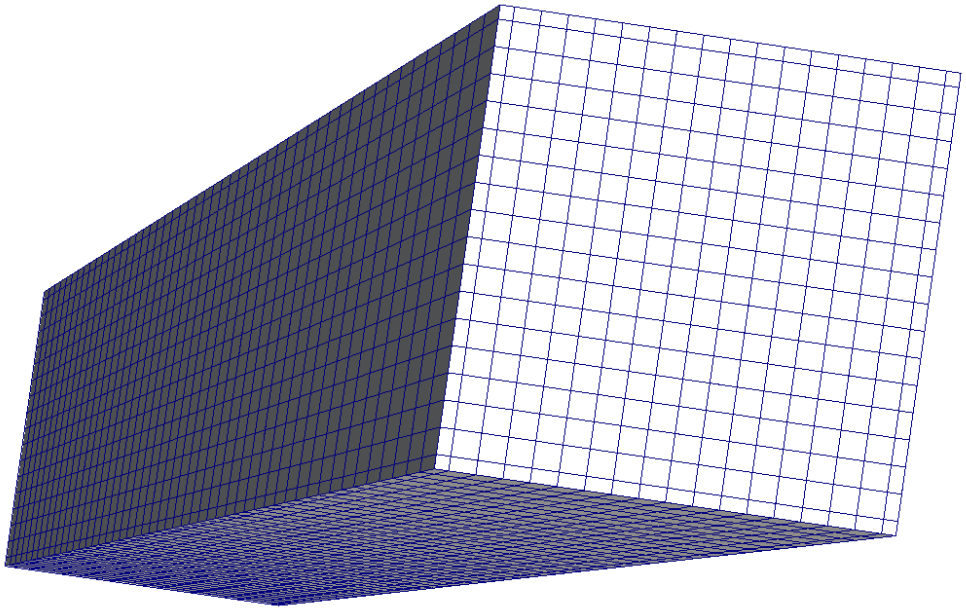


Figure 4: SCF coolant mesh for a PWR fuel assembly.

2.3. TRANSURANUS

TRANSURANUS (TU) [6] is a fuel-performance code for thermal, mechanical and neutron-physical analysis of a cylindrical fuel rod. Depletion is simulated solving a simplified system of Bateman equations which accounts for the most relevant isotopes. The neutronic parameters, i. e. radially dependent power, flux and reaction rates, needed for the rest of the calculation, are obtained using a low-order method.

A wide variety of physics are included in the thermomechanic model, i.e. thermal and irradiation-induced densification of fuel, swelling due to solid and gaseous fission products, creep, plasticity, pellet cracking and relocation, oxygen and Pu redistribution, volume changes during phase transitions, formation and closure of central void and treatment of axial friction forces. The fuel-cladding gap conductance is calculated using the URGAP

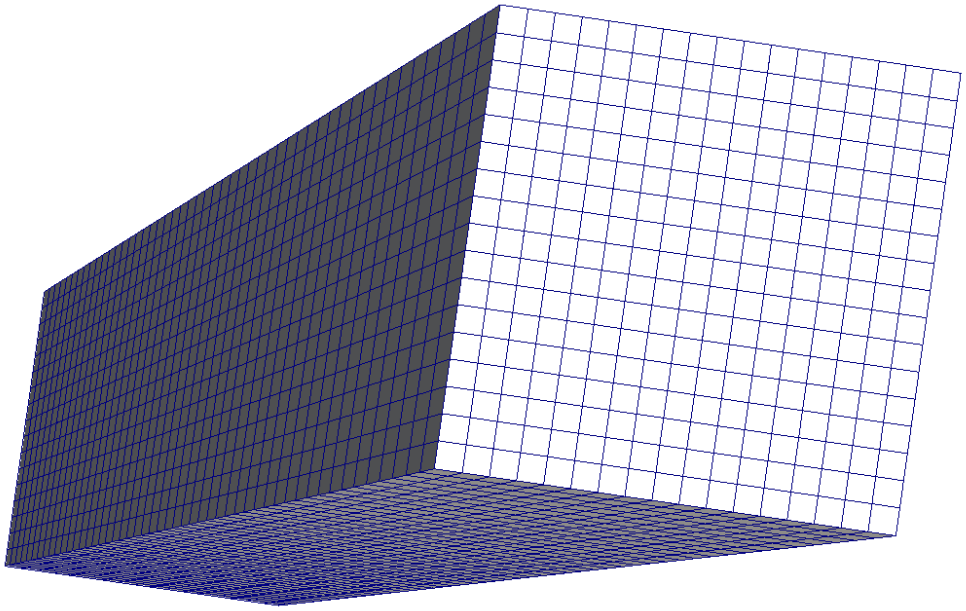


Figure 5: SCF fuel mesh for a PWR fuel assembly.

model [12], and depends on the gap width or contact pressure between the fuel and the cladding, the gas pressure and composition and the surface characteristics of the cladding and the fuel.

The calculation scheme deals with a single fuel rod discretized axially and radially, for which different sets of boundary conditions can be given. In the present work, the boundary conditions are $h_{clad-cool}$, T_{cool} and p. To perform the coupling with Serpent2 and SCF, a TU module that deals with more than one rod was developed, but the solution of each rod remains independent.

To combine the results of all rods in a single field and to exchange variables with the other two codes unstructured meshes are used, as is shown in Figure 6 for the same PWR example used for Serpent2 and SCF, where the guide tubes are not simulated in TU at all.

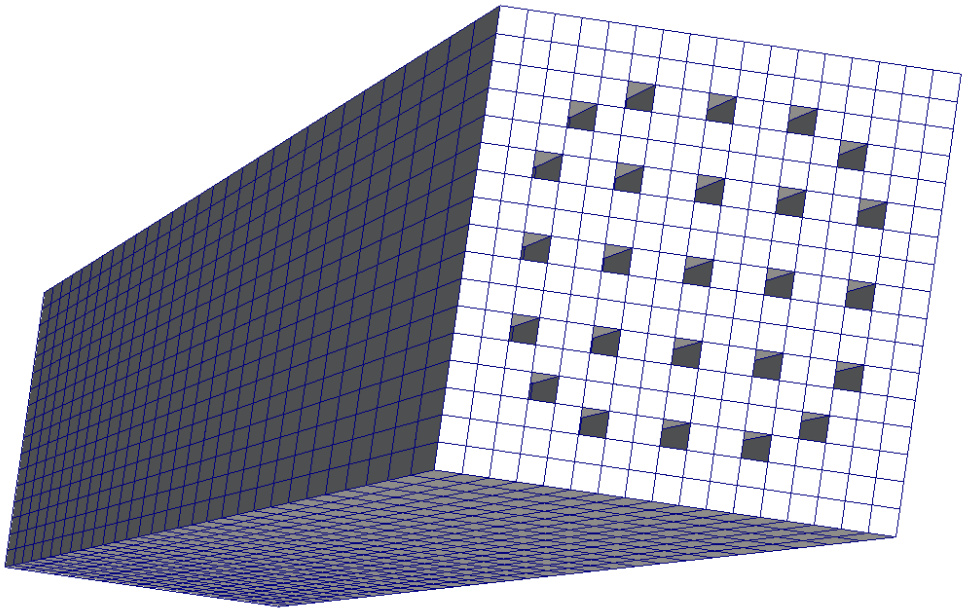


Figure 6: TU mesh for a PWR fuel assembly.

2.4. ICoCo-based modules

The programming approach used in this work is based on modularizing each code following a well-defined methodology, as schematized in Figure 7.

Each source code is first restructured as a solver library that implements the capabilities required in multiphysics simulations, i. e. initialization, termination, time-step control, calculation control for steady-state, depletion and transient calculations and exchange of feedback variables. To define the Application Programming Interface (API) for each code, these libraries are then wrapped into C++ classes which are derived from an abstract base class that represents a generic code suitable for multiphysics simulations. This base class is defined by the Interface for Code Coupling (ICoCo) specification from the SALOME open-source platform [13]. At this point, the three codes are implemented as solver C++ classes with a common format,

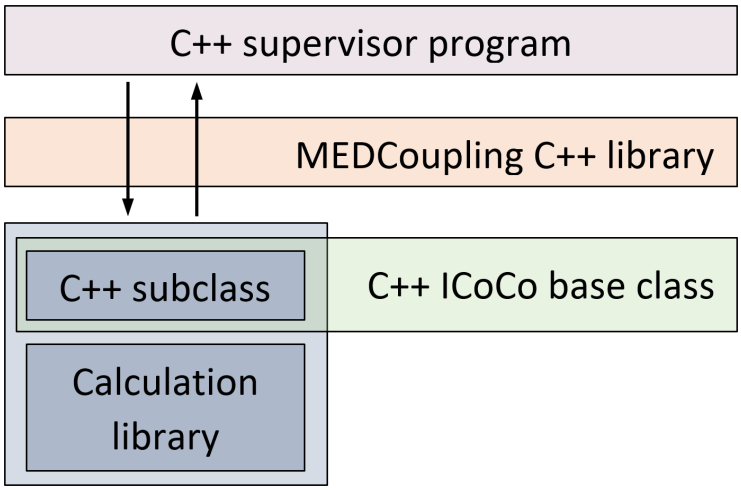


Figure 7: Software design of the calculation modules.

with the native language masked by the C++ interface.

Another key aspect of the API of the codes is the use of a common format for the exchange of feedback variables, which is done using the MEDCoupling library [14], also from the SALOME platform. This library implements unstructured meshes and fields defined on them, as well as advanced interpolation methods and output capabilities. The meshes presented in sections 2.1 through 2.3 are all handled in this way, and when a variable is retrieved from or set to a code the data is represented as a field defined on a mesh. The exact way in which variables are exchanged and interpolated is described in Section 3.3.

With this methodology, the three codes involved in the coupling have identical interfaces, i. e. C++ classes derived from a common base class and therefore with the exact same calculation methods and format for feedback exchange. This greatly simplifies the implementation of the coupling scheme, since all the codes behave in the same way and the mapping of feed-

back variables between the codes is done using mesh-based interpolation at runtime and not through index-based mappings. It is also key to note that up to this point the three codes remain completely separate, and hence can be maintained independently without affecting the coupled system as long as the ICoCo-based multiphysics interface is kept constant.

3. Coupling scheme

The methodology being used to couple Serpent2, SCF and TU is derived from the traditional neutronic-thermalhydraulic coupling scheme, replacing the fuel solver in SCF by TU. The physical feedback scheme is presented in this section, as well as the programming methodology. First, the SCF-TU calculation scheme is defined in Section 3.1, while Section 3.2 presents the three-code scheme. The complete scheme is shown in Section 3.3 from the software perspective.

3.1. SCF-TU coupling

While the Serpent2 side remains unchanged when adding TU to the neutronic-thermalhydraulic coupling scheme, the SCF-TU interaction is more involved.

Since TU calculates $T_{fuel}(r)$ with the same boundary conditions as SCF, plus p, it can be readily integrated into the SCF calculation scheme replacing the fuel solver in step 1 of the algorithm in Section 2.2. Once the thermomechanic calculation is done, T_{clad} is available for all rods and axial levels, and the SCF solution can be resumed from step 2, i. e. skipping the rods and performing only the calculation for the coolant.

In a transient calculation, the coupled solution for each time step can be obtained iterating according to this method and setting a convergence criterion over T_{clad} for instance. For steady-state, the scheme can be simplified noting that, if the heat conduction inside the rods in the axial direction is neglected, the power going to the coolant at each axial level is in fact the power generated there, this is $q^{jk} = \sum_{i(j)} \frac{A^{ij}}{A^i} P^{ik}$, i. e. a sum of the rod powers multiplied by the fraction of each rod area A^i in contact with each channel. Therefore, SCF can calculate the coolant conditions directly using the power coming from the neutronics and pass the boundary conditions to TU to solve the fuel pins, with no need to perform iterations. It is important to note that with this method there is no feedback from TU to SCF. This is not true in a transient, since energy can accumulate in the fuel and the original equation in step 2 has to be used.

Hence, for the calculations presented in this work, which correspond to steady-state and depletion (quasi-stationary) calculations, a simplified scheme is used. The SCF-TU side of the coupling is:

1. Solve the coolant pressure p, velocity v and temperature T_{cool} from the mass, energy and momentum conservation equations in SCF, with the power P coming from Serpent2 going directly to the coolant.
2. Transfer $h_{clad-cool}$, T_{cool} and p from SCF to TU.
3. Perform the thermomechanic calculation for each rod in TU, with the boundary conditions from SCF and the power from Serpent2.
4. Transfer T_{cool}, T_{fuel} and the coolant density ρ_{cool} to Serpent2.

To use this scheme a new calculation mode was implemented in SCF, in which the rods are not simulated and the power goes directly to the coolant. This mode is of course only applicable to steady-state problems.

3.2. Full coupling

The fully coupled depletion scheme for the three-code system is shown in Figure 8, where t_n and t_{n+1} are depletion steps n and $n+1$.

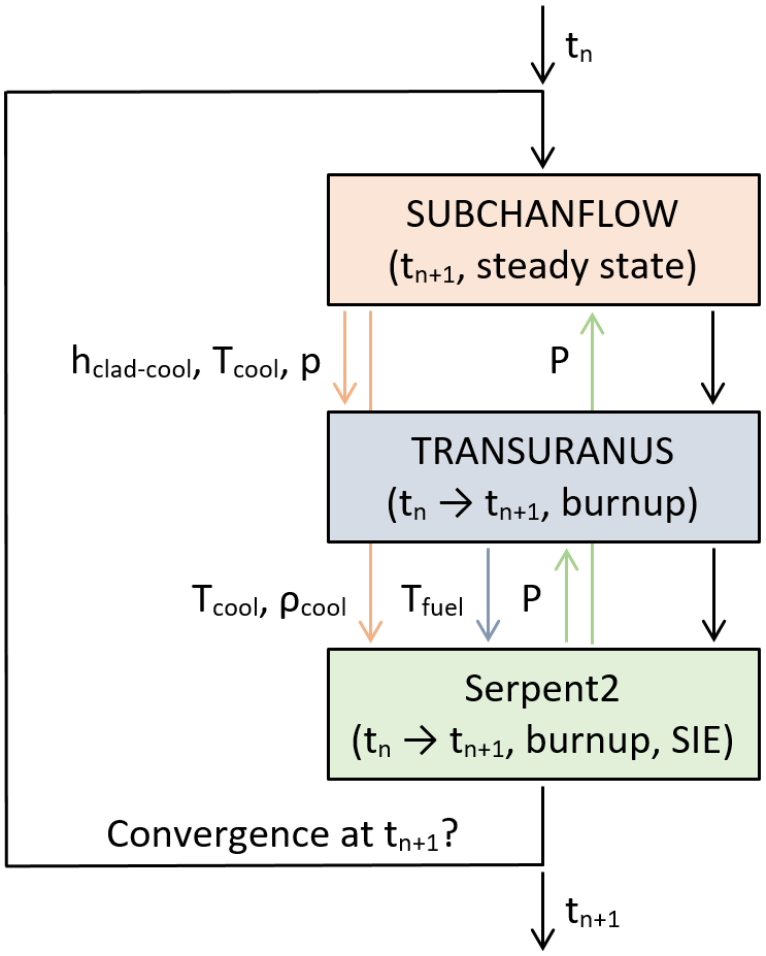


Figure 8: Depletion scheme.

The calculation scheme is the standard Picard iterative method, with each iteration running first a SCF steady-state calculation, and then TU and Serpent2 burnup steps from t_n to t_{n+1}. This order was chosen to run SCF and TU following the algorithm in Section 3.1 and Serpent2 last so updated

thermalhydraulic conditions are used for the Monte Carlo calculation, which is the most time-consuming part. The calculation order is flexible however, and it is a user-defined option in the current version of the coupling.

The coupling scheme for depletion is semi-implicit, meaning that for each burnup step the feedback variables at the end of the step from t_n to t_{n+1} are used and the solution is converged at t_{n+1}. In Serpent2, the isotope compositions are solved using the SIE method with thermalhydraulic feedback [15] which is based on iterating the solution for each step, averaging the flux distribution and the reaction rates at the end of the step and solving the Bateman equations with this values. As part of the transient thermomechanic evolution in TU, a simplified burnup calculation independent of Serpent2 is performed, and there is no information about the material compositions going from Serpent2 to TU.

To evaluate the convergence of the solution, criteria in L_2 or L_∞ norm over any feedback variable can be used. In the present work, the L_∞ error in P, T_{fuel}, ρ_{cool} and the effective multiplication factor k_{eff} are used. Furthermore, the iterative solution can be accelerated and stabilized using under- and over-relaxation and Nonlinear Krylov Acceleration (NKA), although from previous calculations it seems that this methods do not improve significantly the convergence, at least not for LWR fuel-assembly cases at nominal operating conditions [4].

3.3. ICoCo-based supervisor

Figure 9 shows the overall design of Serpent2-SCF-TU, which relies on modularization and object-oriented programming. With the three codes wrapped into interfaces as explained in Section 2.4, the coupling scheme is

implemented in a C++ supervisor program that manages the multiphysics calculation scheme, the feedback exchange and the output of the simulation.

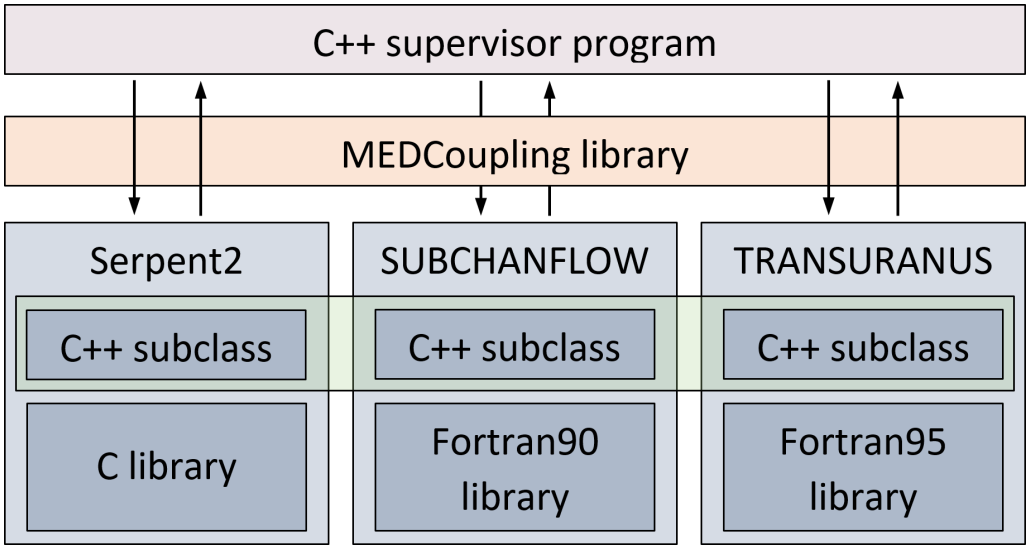


Figure 9: Software design of the full coupling.

Given that all the APIs are derived from the same base class, the supervisor program is implemented using a set of classes that deal with this base class and implement the main methods, e. g. feedback exchange, calculation schemes and convergence control, in a problem-agnostic way, i. e. independent of the particular codes being coupled. This is a result of the inheritance-based definition of the APIs, and greatly enhances code reusability. In fact, the same supervisor program can be used to run any set of codes, provided that they have ICoCo-based interfaces. This means that to add TU to the preexisting Serpent2-SCF coupling the only major development was to implement TU as an ICoCo-based API.

Regarding the input files, the model for each code is defined in the same way as in the standalone versions, i. e. the same input formats are used. The

meshes used to transfer fields are also given as input, and are built with a special preprocessor, which also generates the geometrical model of the three codes. The coupling scheme is defined in the supervisor input file, where the calculation methods, time discretization, feedback fields, convergence tolerances and acceleration methods are specified.

The feedback exchange for all fields works in the exact same way, and can be summarized as:

1. Get the field from the code that calculated it with its associated mesh.
2. Evaluate convergence, change units, accelerate, generate output.
3. Get a template of the field in the second code with its associated mesh.
4. Interpolate the field from the source mesh to the target mesh.
5. Set the interpolated field to the second code.

In this scheme the classes that define the fields and meshes, as well as the interpolation methods, are provided by the MEDCoupling library.

This coupling methodology enhances the flexibility and maintainability of the tool with respect to the traditional master-slave approach, in which one or more *slave* codes are embedded into a *master* code that manages the coupled calculation. Therefore, it is particularly suitable for this three-code coupling and for the cooperation in code development between institutions in the McSAFE project.

4. Results

This section presents the results obtained with Serpent2-SCF-TU for a VVER-1000 fuel assembly. The test case and the modelling approach are described in Section 4.1. The results for this problem are shown in Section

4.2, along with a comparison with the Serpent2-SCF solution without TU to verify the implementation of the three-code coupling and assess the impact of the new fuel calculation methodology. The performance and potential bottlenecks for larger cases are analyzed and discussed in Section 4.3.

4.1. Test case

The test problem considered is the TVSA-30AV5 fuel-assembly design from the AER benchmark for VVER core burnup calculations [16]. It consists of a VVER-1000 fuel assembly with 303 regular fuel pins (UO_2 with 2.99% weight enrichment of U^{235}), 9 fuel pins with burnable poison (UO_2 with 2.4% weight enrichment, 5% Gd_2O_3 mass fraction), both with a central hole, 18 guide tubes and a single instrumentation tube. The fuel assembly has 15 spacer grids, 13 of them within the active length of the core, and stiffening angle plates. A 360-day depletion calculation was performed using Hot Full Power (HFP) operating conditions at 18.4MW.

4.1.1. Serpent2 model

The Serpent2 model consists of a single fuel assembly with reflective boundary conditions in the radial directions and three reflector layers at the top and bottom of the fuel assembly with vacuum boundary conditions, as described in the benchmark definition. The active length consists of two types of sections, namely with and without grid spacers, which are shown in figures 10 and 11. The spacers are simulated in such a way as to maintain their mass and volume, and the stiffeners in the corners are modelled explicitly.

Each transport calculation was done with 1000 active cycles of 100,000 particles, resulting in a statistical uncertainty in the power of less than

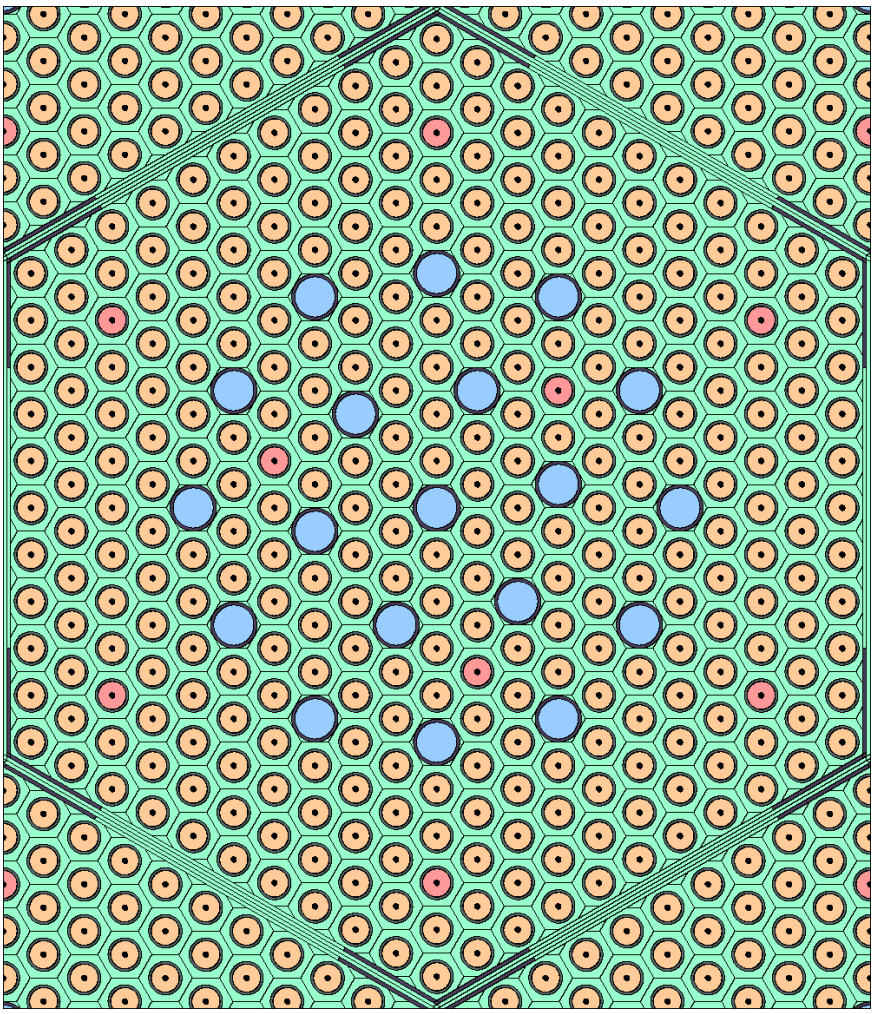


Figure 10: Serpent2 model for sections without spacers. Pins with Gd_2O_3 are shown in red, guide tubes in blue.

1% for every pin and axial level. The fission source was obtained using 250 inactive cycles for the first transport calculation and corrected with 50 cycles for each subsequent iteration. Given that the SIE method used for the burnup calculation is based on relaxation of the solution, the uncertainty of the power distribution cannot be computed directly, and therefore the

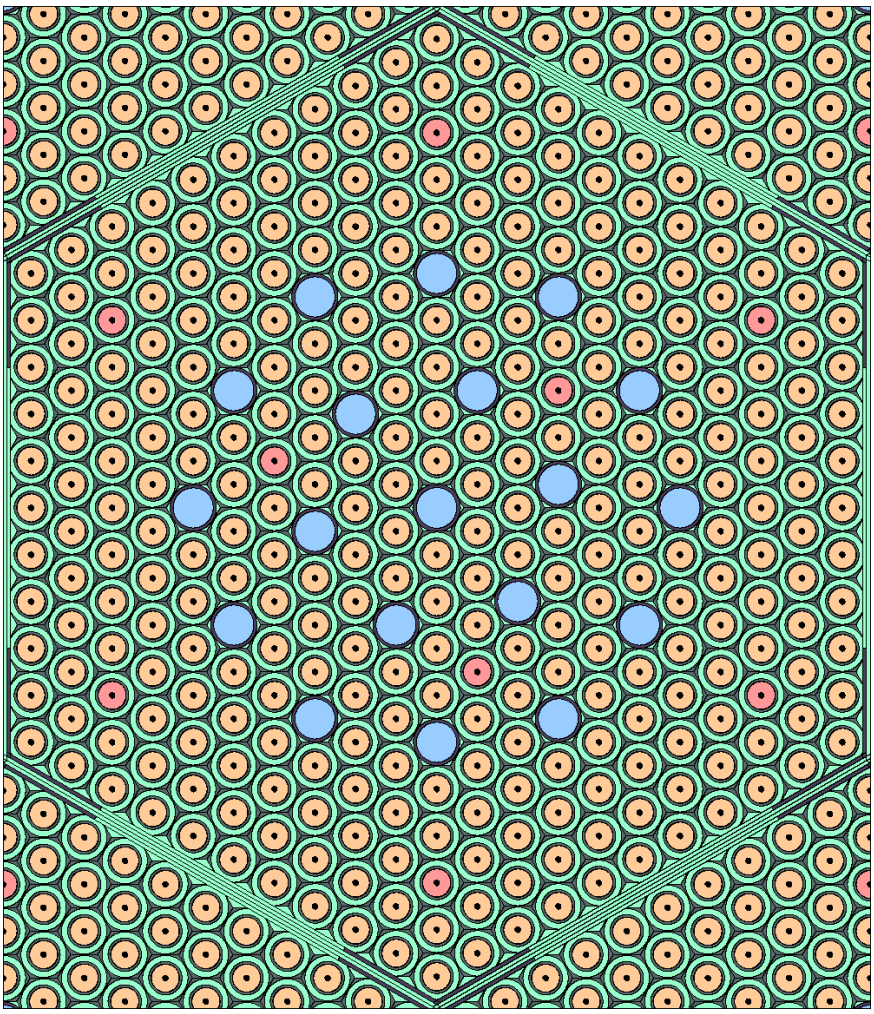


Figure 11: Serpent2 model for sections with spacers. Pins with Gd_2O_3 are shown in red, guide tubes in blue.

results for the power are reported without their uncertainty and a 1% limit in the local statistical uncertainty can be assumed.

The densities and temperatures for the coolant and fuel materials are given in the superimposed mesh shown in Figure 12. This mesh corresponds to a newly developed Serpent2 feature which allows nesting regular meshes

to build full-core pin-by-pin geometries. The temperature and density for a given position are retrieved in the same way as materials are found in the traditional universe-based geometry treatment used in Monte Carlo particle transport, i. e. starting at the root level and going down nested levels. For feedback exchange, this multi-level mesh is represented as a normal unstructured mesh. In this case, a regular y-type hexagonal pin-level mesh is nested inside an x-type fuel-assembly-level mesh, with 50 axial levels.

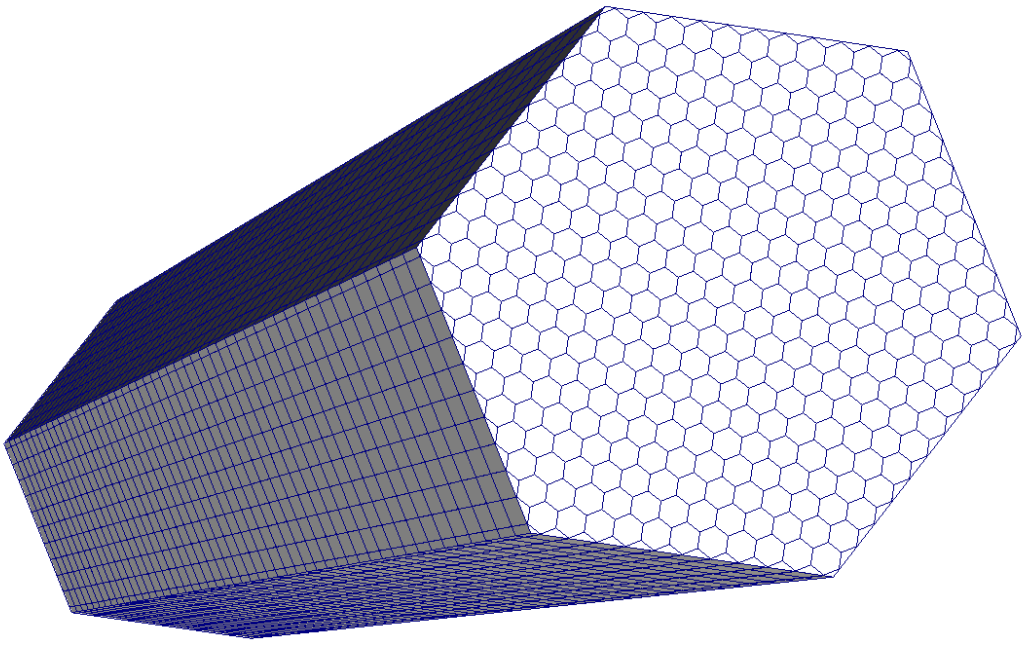


Figure 12: Serpent2 mesh.

4.1.2. SUBCHANFLOW model

A standard coolant-centered subchannel model for hexagonal geometry is used in SCF, as shown in Figure 13. The spacer grids are accounted for as local pressure drops with a fixed loss coefficient, in this case 1.0. The stiffener plates are considered in the calculation of the hydraulic parameters

of the subchannels in contact with them, and have the effect of increasing the friction in those subchannels. For the flow calculation the axial length is discretized uniformly in 50 levels.

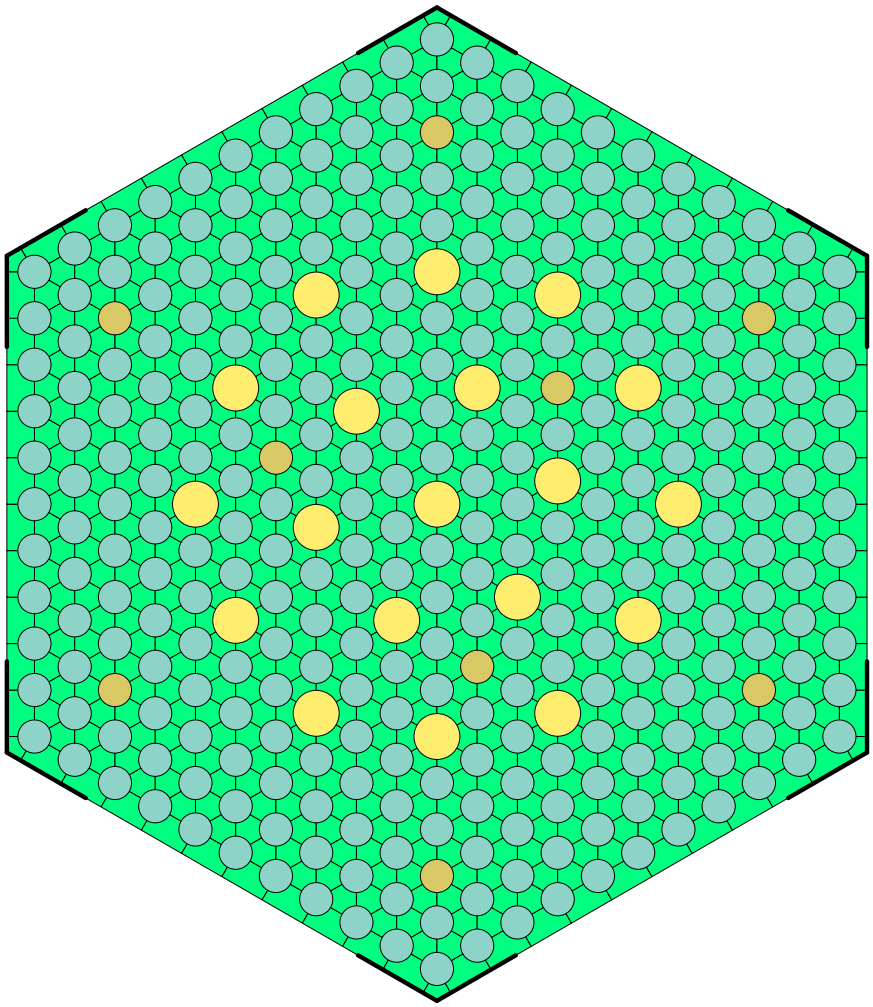


Figure 13: SCF model. Pins with Gd_2O_3 are shown in orange, guide tubes in yellow.

As previously stated, the exchange of feedback fields is done superimposing two unstructured meshes to the SCF model, one to define the shape of the subchannels and another one to give the rod structure. The subchan-

nel mesh is shown in Figure 14, where it can be seen that the cells have the shapes as in Figure 13. The fuel mesh is shown in Figure 15, where it is clear that the actual shape of the rods is not considered, but rather cells containing the rods are defined. The rod geometry is considered through the flow area and the hydraulic and heated perimeters that define the sub-channels used for the flow calculation, as well as in the heat-conduction solver for the fuel. The fuel variables are represented as scalars in these cells, and in particular the fuel temperature, for which a radial profile is calculated, is condensed into an effective Doppler temperature for each cell using a volume average.

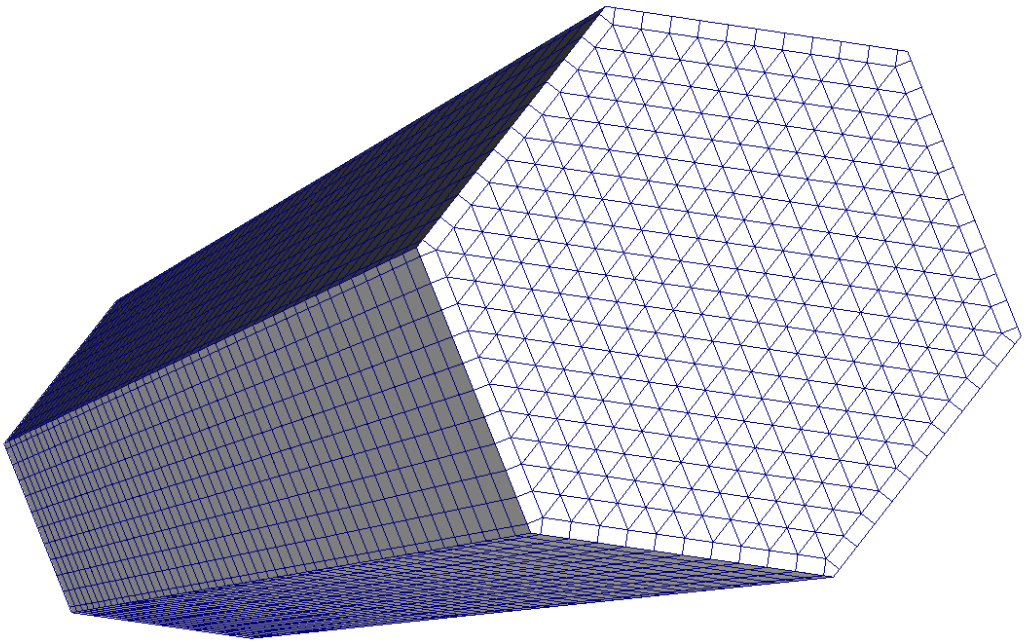


Figure 14: SCF coolant mesh.

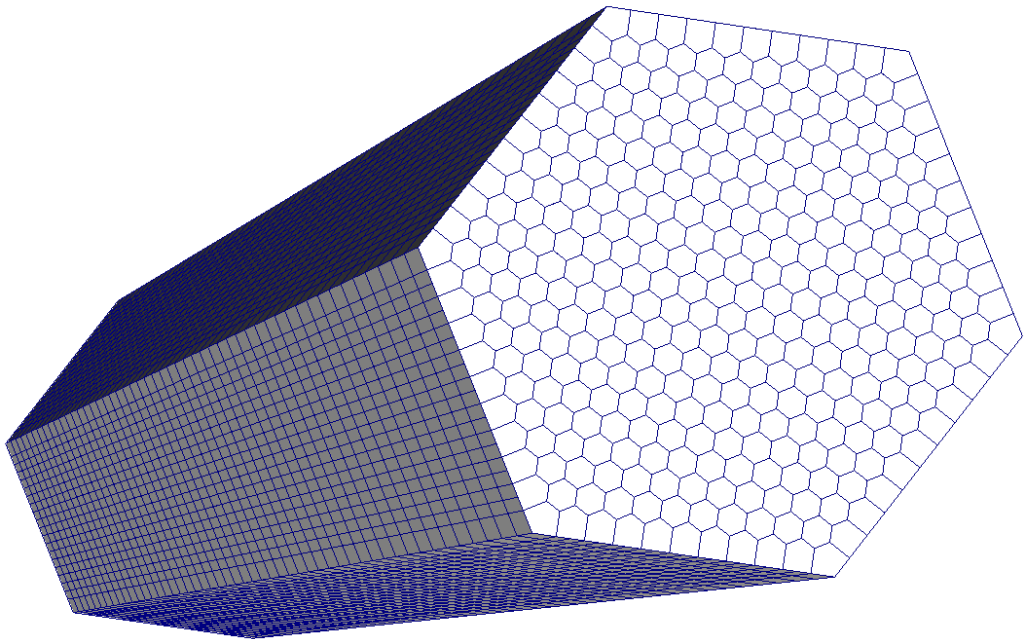


Figure 15: SCF fuel mesh.

4.1.3. TRANSURANUS model

The TU model is comprised of all fuel rods, leaving out the guide tubes. Two types of rods are modelled, with and without Gd_2O_3. For each rod, 4 radial coarse zones for the fuel and 2 for the cladding are used, where mechanical properties are taken as uniform. Each of these zones is further subdivided in 5 to 10 fine radial nodes for the numerical solution. The axial discretization consists of 30 equidistant nodes. The mesh for feedback exchange is shown in Figure 16.

4.1.4. Coupling parameters

To evaluate the convergence of the iterative solution at the end of each burnup step, limits of 30 pcm for the multiplication factor k_{eff} and of 1% in L_∞ norm for ρ_{cool}, T_{fuel} and P are used.

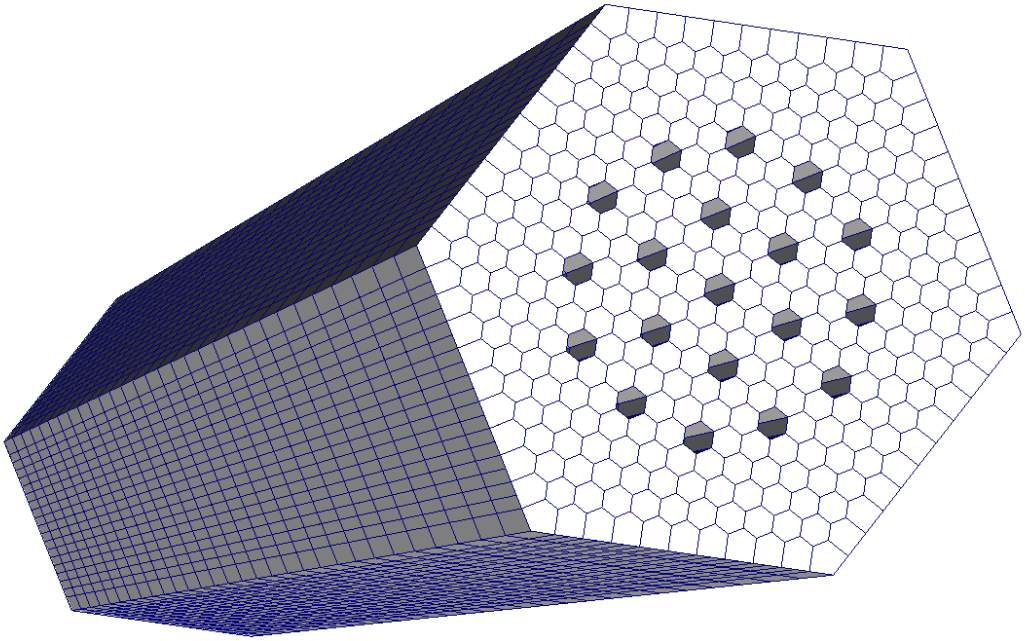


Figure 16: TU mesh.

For the depletion calculation the SIE method is used. This method performs an average of the power at the end of each burnup step, which is enough to stabilize the solution in this case, and therefore no under-relaxation is used for other feedback variables.

4.2. Selected results

In order to analyze the new calculation scheme, the results of Serpent2-SCF-TU are compared in this section with the ones without TU. Given that at steady state there is no accumulation of energy in the fuel and that in the calculation scheme used here there is no direct feedback from TU to SCF, the results for the coolant are expected to be very similar for the two simulations, as only changes in the power distribution can affect the coolant calculation. This means that from the neutronic side no significant changes

produced by coolant feedback are expected. From the physical point of view, the main difference in the modelling approach is the solution of the fuel behavior, and therefore the analysis focuses on the fuel temperatures and gap parameters, and on the Doppler feedback to the neutronic calculation.

Figure 17 shows a summary of the global results for the 360-day burnup calculation. The main axis of each plot shows mean, maximum and minimum values taken over the fuel assembly. The secondary axis shows the global differences between Serpent2-SCF (SSS2-SCF) with and without TU, defined locally as

$$\delta_x^{abs}(\vec{r}) = x_{TU}(\vec{r}) - x_0(\vec{r}), \tag{1}$$

where x_{TU} and x_0 are the results with and without TU at position $\vec{r}$. Relative differences are normalized with the mean value without TU, as

$$\delta_x^{rel}(\vec{r}) = \frac{\delta_x^{abs}(\vec{r})}{\bar{x_0}}. \tag{2}$$

The power and fuel temperature axial profiles are shown in Figure 18 for selected burnup steps.

The difference in the multiplication factors calculated with and without TU (Figure 17a) is smaller than 100pcm, with the calculation with TU predicting slightly lower values. This can be attributed to the average fuel temperature, which is higher for Serpent2-SCF-TU (Figure 17c).

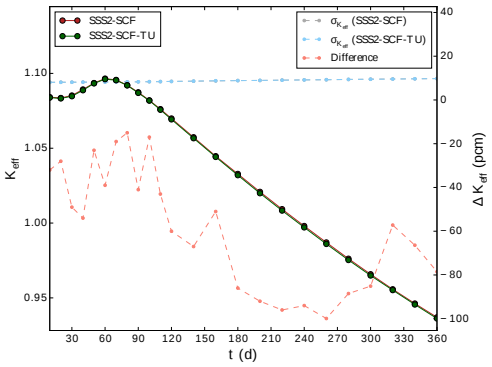
The normalized maximum power, i. e. the peaking factor (Figure 17b) is quite similar for both calculations. The differences in Root Mean Square (RMS) in the power distributions are below 5% for the entire depletion range. The maximum local differences reach 20% for some burnup steps, though these are located in pins with burnable poisons at the top and

bottom of the fuel assembly (Figure 18a), where the power is lower, and cannot be attributed to TU but rather to the Monte Carlo uncertainty in the power calculation.

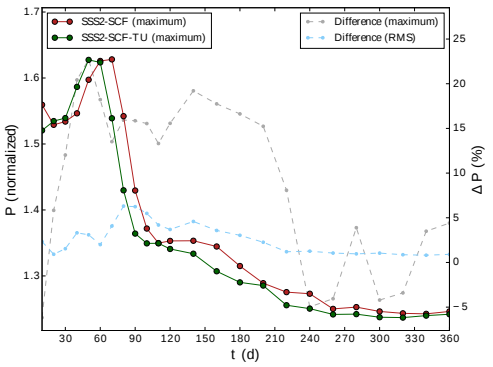
As expected, the coolant temperature calculated by SCF does not change significantly when using TU, as can be seen in Figure 17d, and no significant impact has been observed in any coolant parameter. The Departure from Nucleate Boiling Ratio (DNBR), shown in Figure 17e, is essentially the same for the two simulations.

Finally, Figure 17f shows the fission gas release calculated by TU, which in this case seems dominated by the burnup-dependent athermal release fraction (linear with burnup). SCF does have a fission gas release model, though the correlation has a threshold for the linear heat rate that is never reached for this case, and therefore no release is predicted.

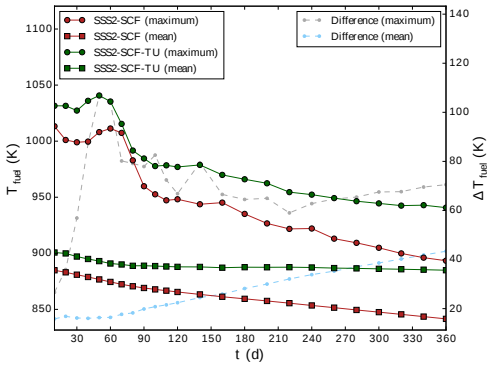
A summary of the solution for the fuel rods is presented in Figure 19. The cladding-coolant heat-transfer coefficient (Figure 19a) is essentially the same for both simulations, as expected, resulting in very small differences in the cladding temperatures (about 2K on average, see Figure 19b). The differences in the temperatures at the pellet surface are quite small, the maximum being around 20K, as can be observed in Figure 19e. This is due to the surprisingly good agreement in the fuel-cladding gap solution between SCF and TU, which can be seen in figures 19d and 19c. Overall the improvement in the modelling of the fuel-cladding gap behavior does not seem to have a significant impact, at least not for this case at this level of burnup. Relatively large differences can be observed in the fuel centerline temperature (Figure 19f), where the calculation using TU predicts larger values, though this is likely due to the thermomechanic properties of the



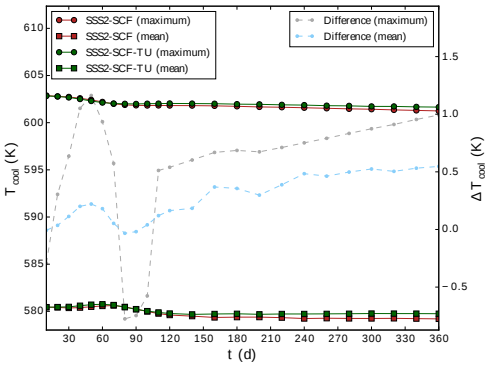
(a) Multiplication factor.



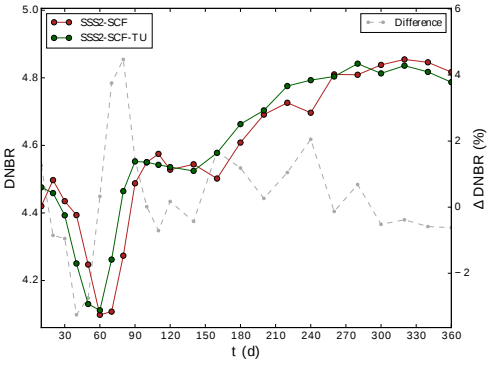
(b) Power.



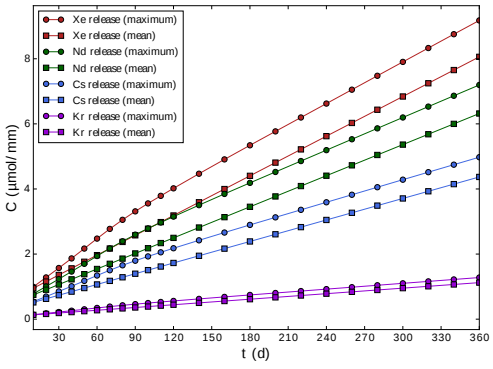
(c) Average fuel temperature.



(d) Coolant temperature.

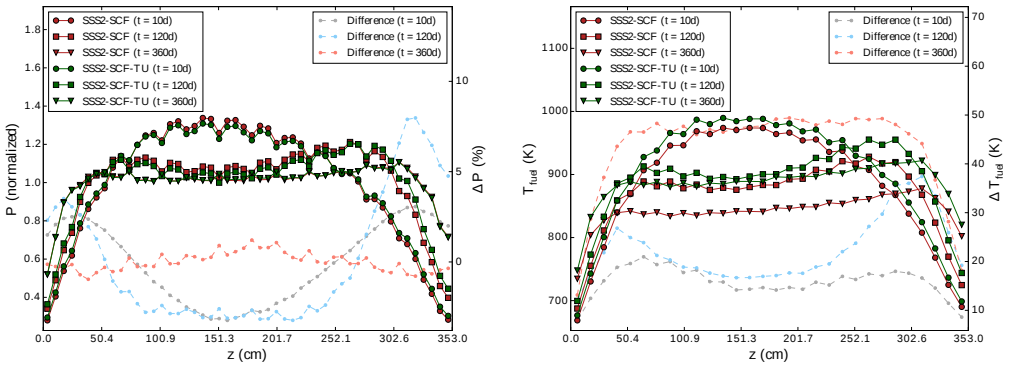


(e) DNBR.



(f) Fission gas release.

Figure 17: Global results for Serpent2-SCF with and without TU.



(a) Power. (b) Average fuel temperature.

Figure 18: Axial profiles for Serpent2-SCF with and without TU.

fuel, in particular the thermal degradation of the conductivity, not the gap model.

Figure 20 shows the evolution of the power distribution for Serpent2-SCF-TU and the difference with the simulation without TU. Overall the results show good agreement, and the differences seem to be related with the axial convergence of the solution, which is a combination of the neutronic, thermalhydraulic and depletion calculations and the propagation of the statistical uncertainty of the Monte Carlo simulation. In any case, the differences observed cannot be attributed directly to the method used to simulate the fuel. The fuel temperature, shown in Figure 21 tends to follow the power, and here again no clear impact from the fuel model is observed.

4.3. Performance and bottlenecks

The simulations shown in this work were performed in the ForHLR II high-performance computer of the Karlsruhe Institute of Technology (KIT) [17], which features Intel(R) Xeon(R) E5-2660 v3 (2.60GHz) CPUs. Table

Code	SSS2-SCF	SSS2-SCF-TU
Time / it. (SSS2)	1825.3s	1828.1s
Time / it. (SCF)	3.9s	3.3s
Time / it. (TU)	-	0.2s
Time / it. (total)	1829.2s	1831.6s
Total time	168108.7s	218092.9s
Num. iterations	4.8	5.9

Table 1: Partial and total calculation times and average number of iterations per burnup step.

1 shows the average calculation time per iteration for each code, as well as for a complete iteration, the total calculation time and the average number of iterations per burnup step, using 40 nodes with 20 cores per node. It is clear that the calculation time added by TU to each iteration is completely negligible, as Serpent2 (SSS2) takes the overwhelming majority of the time. The simulation with TU does take more iterations to converge, which results in an increase in the total runtime of about 30%, though this can likely be mitigated optimizing the convergence criteria and applying acceleration methods, and is not expected to be a problem when tackling larger problems.

Regarding memory utilization, the problem size goes from 25.1GB to 26.4GB, i. e. 1.3GB (5,2%) more are required, when adding TU to Serpent2-SCF, which does not constitute a significant increase. Furthermore, the multi-rod TU module used in this coupling splits the rods across nodes (MPI tasks), scaling down the in-node memory demand. Therefore, the memory demand by TU should not be a problem when moving to full-core cases either.

The most important bottleneck to apply Serpent2-SCF-TU to full-core cases is the memory required by Serpent2 to store the material data in depletion calculations. The simulation of a VVER-1000 core, which typically contains 163 fuel assemblies, with the modelling approach used in this work, can be expected to take around 4TB, which clearly exceeds the in-node memory of any current supercomputer. To tackle this issue, the development of a domain decomposition scheme for Serpent2 is underway in the framework of the McSAFE project.

5. Conclusions

A Serpent2-SCF-TU coupling for high-fidelity depletion simulations has been developed using an object-oriented mesh-based implementation approach. The calculation scheme is based on replacing the simple fuel-rod model in SCF with the more sophisticated thermomechanic model in TU, with the aim at adding fuel-performance analysis capabilities to the traditional neutronic-thermalhydraulic methodology.

To test the first version of this three-code system, a 360-day depletion calculation of a VVER-1000 fuel assembly was performed using a pin-by-pin model and the results were compared with the ones obtained with Serpent2-SCF without TU. The results of both simulations were shown to be consistent, which serves to verify the implementation of the three-code coupling. As expected, essentially no changes were observed in the coolant calculation. Significant differences were found in the width and conductance of the fuel-cladding gap, as well as in the maximum and average fuel temperatures, which can be expected to increase for higher burnup. The effect of this differences in the neutronic calculation using radially averaged fuel

temperatures is relatively small, with changes in the multiplication factor of less than 100pcm, and a feedback scheme considering the radial dependence of these temperatures merits further investigations.

To summarize, the general implementation of Serpent2-SCF-TU has been verified, though there still remain a couple of issues to analyze, namely the treatment of the Doppler feedback and the influence of high-burnup effects that could lead to larger differences between the fuel models of SCF and TU.

5.1. Further work

While the proposed coupling approach was used to successfully develop a first version of the three-code system and start the testing and analysis of the results, some important questions remain.

First, the current methodology relies on the modelling in TU of all the fuel rods in the system, which is implicitly assumed in Section 3.1. This poses a problem not only in terms of performance, in particular regarding memory demand, but also when analyzing the results, since the amount of output data for a large problem can become overwhelming (a full-core PWR or VVER model would have more than 50,000 rods). A sensible modelling approach for TU is therefore needed when moving towards full-core cases. An option could be the use of TU at fuel-assembly level modelling average pins, as was done in a coupling with the nodal code PARCS and SCF [18], combined with a hot-channel pin-by-pin methodology to calculate safety parameters. It is not clear how the coupling with SCF would be done in this case, and a consistent method needs to be formulated.

Moreover, the feedback scheme used for depletion is suitable for a first

version, but could potentially be improved. In the current method, the only feedback from the neutronics to the fuel-performance is the linear heat rate, and the simplified neutronic model implemented in TU is still used to calculate the radial profiles of the flux and the power inside each pin. In addition, the simple depletion solver in TU is used to calculate the isotope concentrations relevant to the thermomechanic solution. TU basically simulates the depletion in the same way as in a standalone simulation, only with the power given by Serpent2. This method could be improved using radial power and flux profiles, as well as isotope densities, coming from Serpent2 instead.

The last issue is the Doppler feedback for the neutronic calculation. Even though the calculation of the fuel temperature profiles is in principle improved when TU is used, this does not necessarily increase the accuracy of the Doppler feedback, as was shown in this work. This is due to the fact that the fuel temperature is being averaged radially and a uniform temperature is used in Serpent2 for each rod and axial level, either using a volume average or an empirical formula of the form $T_{fuel} = wT_{surface} + (1 - w)T_{centerline}$ using the fuel surface and centerline temperatures $T_{surface}$ and $T_{centerline}$ and a weight w for which several models exist [19]. A significant improvement in the Doppler feedback could be made replacing this average temperature by some type of radial distribution. This is particularly important for fuels with burnable absorbers.

Acknowledgments

This work was done within the McSAFE project which is receiving funding from the Euratom research and training programme 2014-2018 under

grant agreement No 755097.

This work was performed on the computational resource ForHLR II funded by the Ministry of Science, Research and the Arts Baden-Württemberg and DFG ("Deutsche Forschungsgemeinschaft").

Conflict of interest

The authors declare no conflict of interest.

References

[1] L. Mercatali, et al., The EC McSAFE Project: High Performance Monte Carlo Methods for Safety Demonstration - Status and Perspectives, International Multi-Physics Validation Workshop, North Carolina State University, Raleigh, USA, June 14-15, 2018.

[2] J. Leppänen, et al., The Serpent Monte Carlo code: Status, development and applications in 2013, Annals of Nuclear Energy 82 (2015) 142–150.

[3] U. Imke, et al., Validation of the Subchannel Code SUBCHANFLOW Using the NUPEC PWR Tests (PSBT), Science and Technology of Nuclear Instalations 2012.

[4] M. García, et al., Development of an Object-oriented Serpent2-SUBCHANFLOW Coupling and Verification with Problem 6 of the VERA Core Physics Benchmark, International Conference on Mathematics and Computational Methods applied to Nuclear Science and Engineering (M&C 2019), Portland, Oregon, USA, August 25-29, 2019.

[5] M. García, et al., Serpent2-SUBCHANFLOW pin-by-pin modelling capabilities for VVER geometries, Annals of Nuclear Energy 135 (2019) xx.

[6] P. V. Uffelen, et al., Extending the application range of a fuel performance code from normal operating to design basis accident conditions, Journal of Nuclear Materials 383 (2008) 137–143.

[7] J. Dufek, et al., The stochastic implicit Euler method - A stable coupling scheme for Monte Carlo burnup calculations, Annals of Nuclear Energy 60 (2013) 295–300.

[8] V. Valtavirta, Development and applications of multiphysics capabilities in a continuous energy Monte Carlo neutron transport code, Ph.D. thesis, Aalto University, School of Science, Department of Applied Physics, Finland (2017).

[9] D. L. Hagrman, et al., MATPRO-Version 11: a handbook of materials properties for use in the analysis of light water reactor fuel rod behavior, NUREG/CR-0497, TREE-1280 Technical Report, Idaho National Laboratory.

[10] M. E. Cunningham, et al., GT2R2: An Updated Version of GAPCONTHERMAL-2, NUREG/CR-3907, PNL-5178 Technical Report, Pacific Northwest Laboratory.

[11] M. García, et al., Advanced Modelling Capabilities for Pin-level Subchannel Analysis of PWR and VVER Reactors, 18th International Topical Meeting on Nuclear Reactor Thermal Hydraulics (NURETH-18), Portland, Oregon, USA, August 18-23, 2019.

[12] K. Lassmann, et al., The revised URGAP model to describe the gap conductance between fuel and cladding, Nuclear Engineering and Design 103 (1987) 215.

[13] CEA/DEN, EDF R&D, OPEN CASCADE, SALOME Platform Documentation: Documentation of the Interface for Code Coupling (ICoCo), `https://docs.salome-platform.org/latest/extra/Interface_for_Code_Coupling.pdf`, accessed: 2019/05/21.

[14] CEA/DEN, EDF R&D, OPEN CASCADE, SALOME Platform Documentation: MEDCoupling User's Guide, `https://docs.salome-platform.org/7/dev/MEDCoupling/index.html`, accessed: 2019/05/21.

[15] J. Dufek, et al., Derivation of a stable coupling scheme for Monte Carlo burnup calculations with the thermal-hydraulic feedback, Annals of Nuclear Energy 62 (2013) 260–263.

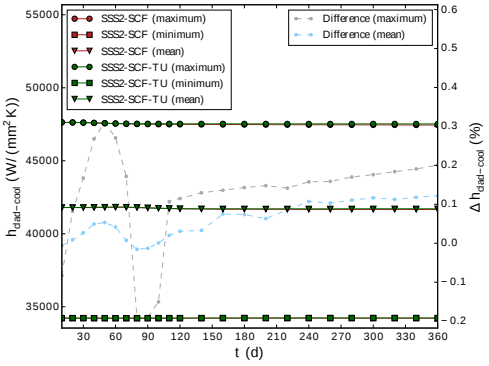
[16] T. Lötsch, et al., Corrections and Additions to the Proposal of a Benchmark for Core Burnup Calculations for a VVER-1000 Reactor, Proceedings of the Twentieth Symposium of Atomic Energy Research, Hungary, Kiadja and KFKI Atomenergia Kutatointezet.

[17] Steinbuch Centre for Computing (SCC), ForHLR II Documentation, `https://www.scc.kit.edu/dienste/forhlr2.php`, accessed: 2019/05/21.

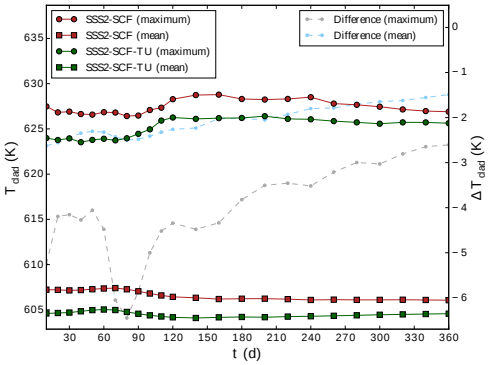
[18] J. R. Basualdo, et al., PARCS-SUBCHANFLOW-TRANSURANUS Multiphysics

Coupling for Improved PWR's Simulations, PHYSOR 2010 - Advances in Reactor Physics to Power the Nuclear Renaissance, Fukui and Kyoto, Japan, April 24-28, 2017.

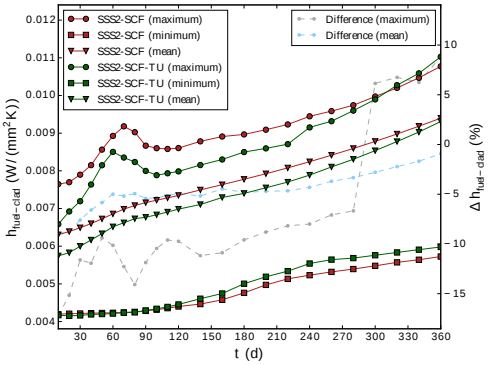
[19] G. Grandi, et al., Effect of CASMO-5 Cross-section Data and Doppler Temperature Definitions on LWR Reactivity Initiated Accidents, The International Congress on Advances in Nuclear Power Plants (ICAPP-2017), Pittsburgh, Pennsylvania, USA, May 9-14, 2010.



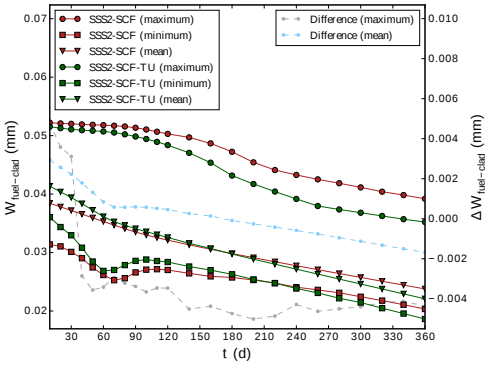
(a) Cladding-coolant heat-transfer coefficient.



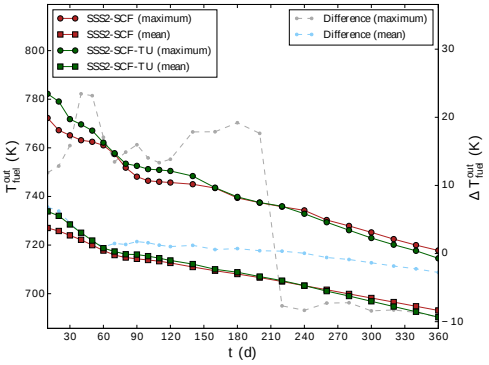
(b) Cladding temperature.



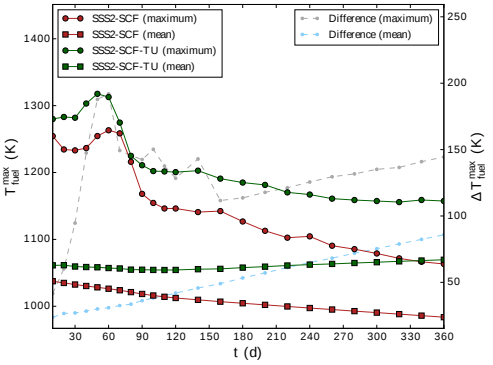
(c) Fuel-cladding gap conductance.



(d) Fuel-cladding gap size.



(e) Outer fuel temperature.



(f) Centerline fuel temperature.

Figure 19: Fuel solution for Serpent2-SCF with and without TU.

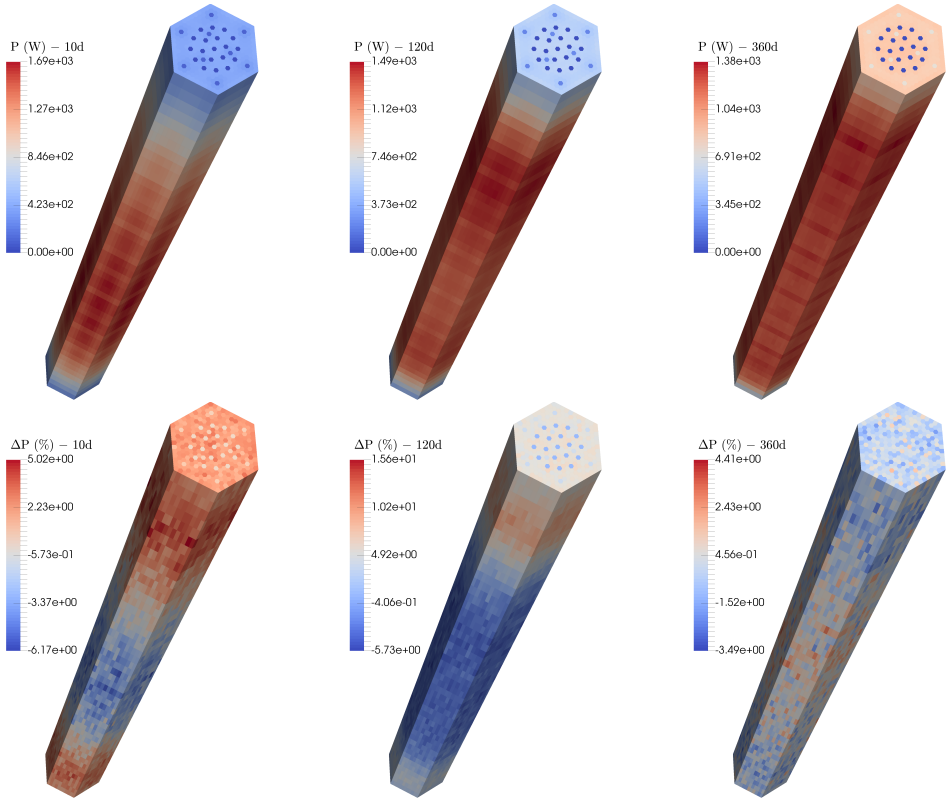


Figure 20: Serpent2-SCF-TU power (top) and difference with Serpent2-SCF (bottom).

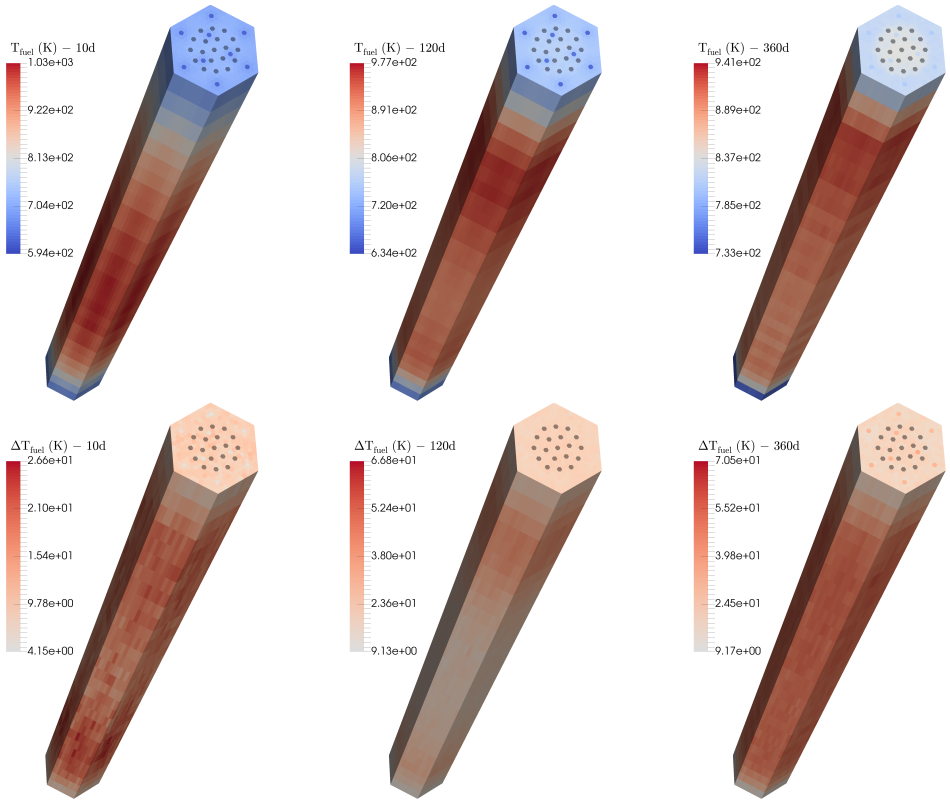


Figure 21: Serpent2-SCF-TU fuel temperature (top) and difference with Serpent2-SCF (bottom).

***Declaration of Interest Statement**

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: