Asociación Argentina
de Mecánica Computacional

Mecánica Computacional Vol XXXII, págs. 1501-1517 (artículo completo)
Carlos G. García Garino, Aníbal E. Mirasso, Mario A. Storti, Miguel E. Tornello (Eds.)
Mendoza, Argentina, 19-22 Noviembre 2013

A SHARED-MEMORY-BASED COUPLING SCHEME FOR MODELING THE BEHAVIOR OF A NUCLEAR POWER PLANT CORE

Germán Theler, Juan Pablo Gómez Omil and Esteban Pellegrino

TECNA Estudios y Proyectos de Ingeniería S.A.
Encarnación Ezcurra 365, C1107CLA Buenos Aires, Argentina

Keywords: Nuclear Reactor, Coupled Calculations

Abstract. The different phenomena that take place in the core of a nuclear reactor comprise a wide variety of physical aspects, which differ both in their nature as in their complexity as well. In general, these effects can be divided into four main disciplines, namely neutronics, thermal-hydraulics, plant conditions and control. Even though neutrons interact with matter in a pretty well-known way, the equations that model how they behave in a nuclear reactor derived from basic mechanistic considerations cannot be yet—and probably ever–be solved completely. And, even more, first-principle equations for turbulent two-phase flow still are not even available. Therefore, to design and analyze nuclear reactors some kind of simplifications ought to be used, which—despite its usual meaning—may still be challenging from a computational point of view. Being the equations behind these four disciplines radically different in their mathematical characteristics, there usually exist dedicated codes that solve each problem separately, especially for the most-complex tail of the spectrum of simplifications. It is therefore desired to devise a mechanism for coupling these particular codes in such a way that the different dependencies are explicitly addressed. A shared-memory-based coupling mechanism was developed in order to perform the aforementioned tasks, which is currently being used in the nuclear industry although it may have applications in other fields as well.

1 INTRODUCTION

In a fission nuclear power reactor, electric energy is provided to the grid by converting the energy released when splitting heavy nuclei into smaller parts, which is then transformed into mechanical energy by means of a turbine and ultimately into a current of electrons by a generator. The process is similar to other chemical reactions in which one starts with a fresh fuel source with a high level of potential energy such as methane and ends up with low potential energy products such as carbon dioxide. In the nuclear case, the fresh fuel is usually uranium—maybe enriched in its 235 isotope—and the results are fission products which should be treated as waste. However, there is a central difference regarding the dynamics of the power generation. Heavy nuclei are split by means of a self-sustained chain reaction in the reactor core, in which a neutron causes a fission and at the same time generates a number ν—between two and three—of new neutrons that are able to cause a new fission, repeating again the process. Figure 1 illustrates the lifetime of a neutron within a nuclear reactor core. Out of the ν newly-born neutrons, exactly one has to cause a new fission and exactly $\nu - 1$ should be absorbed or leaked out of the reactor. The rate at which neutrons cause fissions and the rate at which they are born are regulated by the proper motion of absorbing materials known as control rods. However, these rates also depend on the nuclear properties of the materials of the core—fuel, coolant, moderator, control rods and structural materials—which in turn depend on its macroscopic properties such as temperature and density, which in turn depend on the plant conditions, as we schematically indicate in figure 2. One illustrative example arises when the temperature of the cold thermal sink—i.e. a river or lake—changes by a few degrees with respect to the nominal conditions. As the efficiency of the thermal cycle changes, if a constant power outcome is desired then the control system has to take actions in order to adapt the conditions at which the steam is fed into the turbine. Therefore, the conditions of the coolant—say heavy water—at both the reactor core inlet and outlet change and thus the temperature and density distributions within the core also do. As the nuclear properties depend heavily on how the target nuclei oscillate (i.e. the material temperature) and on the number of nuclei per volume unit (i.e. the density), it is expected that the dynamics of the fission chain reactions that take place within the core should also change in order to keep the overall electric power that the plant gives to the grid constant.

To answer the question of what changes and how to cope with those changes—for example to design a control system able to deal with such operational transients—we need to be able to model as accurately as possible each of the mechanisms involved in the chain of figure 2, as well as the dependencies between them. Unfortunately, the mathematical characteristics of the problems involved in each main aspect of a nuclear power plant—namely the neutronics, the thermalhydraulics, the plant conditions and the control system—are very complex and different,

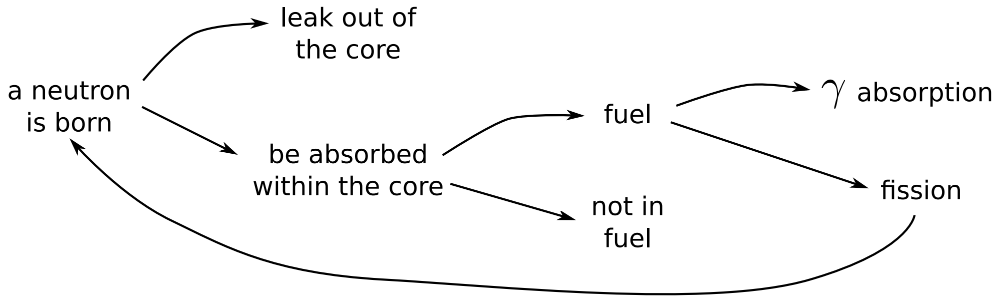


Figure 1: Lifetime of a fission-born neutron

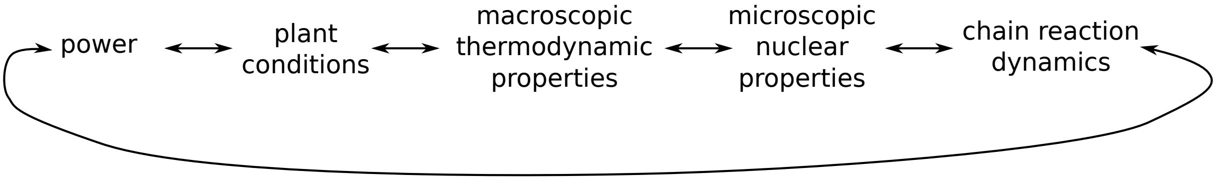

power
plant conditions
macroscopic thermodynamic properties
microscopic nuclear properties
chain reaction dynamics

Figure 2: Dependence of the generated power with the chain reaction dynamics through a series of intermediate conditions and properties.

requiring dedicated tools aimed specifically toward particular cases. It is under these considerations that a huge effort is being made by the nuclear industry to devise and develop schemes to couple the specific computational codes that solve each part of the whole plant in order to obtain faithful models which can be used to analyze, design and optimize nuclear power plants.

In the next section we briefly discuss the mathematical models behind each aspect of a nuclear power plant and their distinctive features. In section 3 we introduce one of the many schemes that can be used to solve the coupled problem which was developed by TECNA and has been successfully used in the nuclear industry for four years now.

2 MATHEMATICAL MODELS

From the modeling point of view, a nuclear power plant can be divided into neutronics, thermal-hydraulics, control system and plant conditions. The first part deals with computing the neutron flux distribution within the core by knowing the instantaneous spatial distributions of the materials within the core—for example how inserted or withdrawn the control rods are—and its macroscopic temperature distributions. Other particular distributions such as xenon concentration should be also computed and taken into account. The thermalhydraulic calculation uses the power distribution in order to compute what the resulting temperature, density and void fraction distributions are. Moreover, it evaluates mass flows and head losses, quantities that in turn change the thermodynamic conditions. This kind of calculation often involves two-flow thermal and hydraulic calculations, which constitute very complex problems on their own.

Nuclear power plants usually have different levels of systems that take actions in order to reach and maintain the desired power level (control system), to keep the reactor under safe operational conditions (limitation system) and to shut the reactor down if something goes wrong (protection system). Given the tightness of safety requirements in nuclear installations, these systems are implemented only in well-known and validated architectures with proper redundancies and employing fail-safe techniques. Therefore, the control system of a nuclear power plant is often quite difficult to model mathematically and tools developed by the manufactures of the actual hardware are usually employed to build computational models.

Finally, whatever system or component in the plant that does not fall into one of the three aspects discussed so far is treated within a generic “plant condition” model. These systems and components include the turbine and the electric generator, the condenser, the steam generators, the pressurizer and auxiliary systems such as volume control systems or, for heavy-water reactors, the moderator cooling system. Also, when analyzing safety transient, emergency systems such as boron or fresh water injection should also be modeled. Moreover, when dealing with loss of coolant accidents, a proper model of the containment should be taken into account. We discuss general aspects of these four groups in the sections that follow.

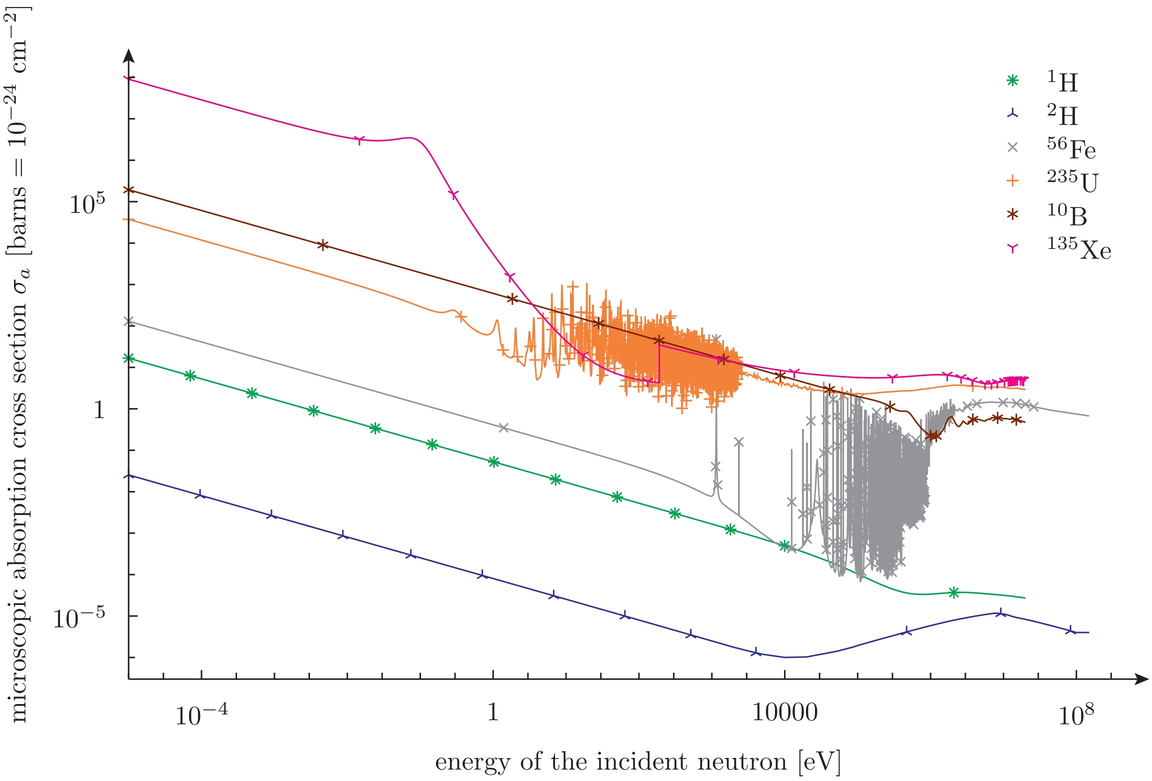


Figure 3: Microscopic absorption cross section for different isotopes as a function of the energy of the incident neutron (Chadwick et al., 2011).

2.1 Neutronics

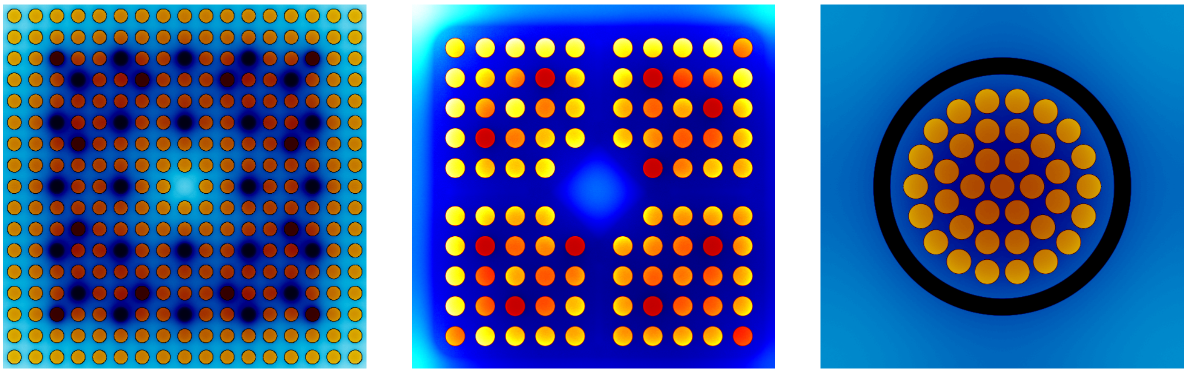
As expected, the computation of the neutronics is the most important task of a power plant model as it lays literally within the core of the model. Individual neutrons interact with the nuclei of the materials that compose the core in a stochastic way. To fix ideas, assume that if a neutron collides with a nucleus, it has a probability p of being absorbed and a probability $1 - p$ of being scattered. However, given the huge number of neutrons that are usually interacting in a reactor core—of the order of 10^{15}—the overall behavior can be regarded as deterministic. Therefore, out of the total number k of neutrons that interact with said nucleus, we can assume that exactly kp will be absorbed and $k(1 - p)$ will be scattered.

These probabilities are characterized by microscopic cross sections, that depend both on the energy of the incident neutron and on the temperature of the target nuclei. The dependence on the neutron energy is far more important than any other one, as can be seen in figure 3. On the one hand, neutrons may span up to eight or nine orders of magnitude of possible energies—from a characteristic fission-born energy of a few MeVs down to energies of cents of eV which are in equilibrium with the thermal spectrum of nuclei. On the other hand, the microscopic cross sections may vary significantly as the energies coincide with resonances of the target nucleus. The heavier and more complex the nucleus is, the more intense and complex its excited states will be. Nevertheless, it is true that in general, all cross sections—and in particular the fission cross section—increase for lower energies. Therefore, most of the nuclear reactors now in operation rely on slowing down the fission-born neutrons so they have more chances to hit an uranium nucleus and trigger a new fission. These kind of reactors are called thermal reactors, and thus these order-of-magnitude changes are roughly taken into account by the engineering design of the reactor. And although plant conditions change the way neutrons are slowed down and therefore the probability that they cause new fissions, operational conditions do not change the behavior of the plant as drastically as figure 3 may suggest.

However, besides depending on the neutron energy, microscopic cross sections depend on the temperature of the target nuclei through a mechanism similar to the classical mechanics effect in which the frequency of a wave is shifted according to the speed of the source. Indeed, we also call it the Doppler effect in nuclear physics. Moreover, the reaction rates at which neutrons interact with matter depend on the macroscopic cross sections, which are the product of the microscopic cross section times the nuclei volumetric concentration. Therefore, the density of a material changes the fission reaction dynamics. This effect is particularly important when phase changes occur in the core—even when only subcooled boiling appears (Theler, 2008)—as the microscopic cross section of water is the same whether it is liquid or vapor, but the volumetric concentration of water atoms—and hence the macroscopic cross section—is not.

Finally, it is appropriate to note that macroscopic cross sections include all the materials present in a certain macroscopic region of the core. Therefore, if the composition of the core varies, the fission chain reaction changes accordingly. This variation can be reasonably small like a small movement of a control rod. But it can be very big, such as the introduction of extraneous materials into the core. Figure 3 shows that boron is a heavy neutron absorber, ten times heavier than uranium. Therefore a small addition of boron may significantly change the macroscopic cross sections. Furthermore, we have to take into account the fact that the fission of heavy isotopes imply transmutation of materials, thus generating new nuclei that did not exist before the fission. The iconic case of this situation is xenon one hundred and thirty five, that has an enormous microscopic absorption cross section and appears as a fission product. The concentration of this "neutronic poison" depends on the dynamics of the neutronics, which in turn depend on its concentration in a non-trivial way.

Due to the complexity of the nature of neutron-matter interactions, engineering neutronic computations are performed in two stages. The first one, called lattice calculations start with a definition of characteristic cells comprising one fuel element that is to appear repeatedly in a regular pattern within the core. Figure 4 shows some common cells for different types of reactor designs. These computations are carried out by the so-called lattice codes, that take as inputs the cell geometry, libraries of detailed microscopic cross sections (such as the one



(a) PWR (b) BWR (c) CANDU

Figure 4: Typical lattice-level cells for three common reactor designs. Images taken from the Monte-Carlo lattice code Serpent (Leppänen, 2012).

shown in figure 3) and the material properties—isotopic composition, isotopic densities and temperature—and compute a set of homogenized macroscopic cross sections that constitute the input of the second stage. The appearance of fission products as the fuel is burnt is also taken into account. Lattice codes are run parametrically so that the macroscopic cross sections can be interpolated and evaluated at any combination of temperatures, densities and fuel burnup. Once this set of cross sections are computed, lattice-level calculations are not needed any further.

The second stage of neutronics calculations comprise the so-called core-level codes, which solve one of the many forms of the discretized Boltzmann transport equation for neutrons. These codes compute the neutron distribution within the core by solving a non-linear partial integro-differential equation that should be discretized both in energy and in space. It is at this level that the influence of the different aspects of the plant over the fission chain are taken into account. Indeed, the non-linearity of the transport equations arises because the cross sections depend on the flux itself—which is the case for any reactor of interest in which the temperatures, densities and xenon distribution depend on the power distribution.

The computation of the steady state involves the solution of a generalized eigenproblem of the form

$$R \cdot \boldsymbol{\phi} = \frac{1}{k_{\text{eff}}} \cdot F \cdot \boldsymbol{\phi}$$

where R and F are matrices of size $NG \times NG$, being N the number of spatials unknowns (of the order of 10^5) and G the number of energy groups (equal to 2 or 3 for thermal reactors). The inverse of the largest eigenvalue is called the effective multiplication factor k_{eff} and is a measure of the criticality of the resulting configuration. The eigenvector $\boldsymbol{\phi} \in \mathbb{R}^{NG}$ give the steady-state flux distribution for the selected spatial discretization.

The elements of the matrices R and F depend on the macroscopic cross sections computed by the lattice code evaluated at the steady-state conditions of the plant, which are not yet known because they are the result of the calculation. This situation needs an iterative scheme to be implemented (see section 3.1). Nevertheless, once the steady-state flux distribution is found, we may proceed to the transient calculation which involves the solution of $NG \cdot (3+I)$ coupled differential equations, being I the number of neutron precursor groups (usually equal to six). The number three represents one equation for the instantaneous flux, one for the ^{135}I concentration and one for the ^{135}Xe concentration (Theler, 2011).

2.2 Thermal-hydraulics

Thermal-hydraulics are nearly as important as neutronic calculations because they are the responsible of addressing the question of whether the core is properly cooled or not under accidental conditions. The computational codes that solve this aspect of the nuclear power plant usually compute the partial differential equations for conservation of mass, momentum and energy for a non-equilibrium two-fluid model with energy and momentum transfer between phases. The spatial discretization involve staggered one-dimensional cells and the models allow cross flows using ad-hoc pressure-drop correlations. Given that in some cases thermalhydraulic codes are used to model both the core and the plant conditions (see section 2.4), specific models for branches and multiple junctions are taken into account to model non-trivial detailed piping designs of auxiliary systems.

These kind of codes include a wide variety of correlations and models for solving the two-phase flow problem, that is indeed a very complex problem on its own. Even though under normal conditions no two-phase flow is involved, usually subcooled boiling occurs at some locations within the core. Under the view of the discussion about macroscopic cross sections above, it is important to be able to correctly predict its occurrence, so a code able to cope with such complex models is desirable.

The one-dimensional nature of thermalhydraulic codes makes the discretization of the full three-dimensional code an important task, as the results will depend on non-trivial ways of the selected nodalization. In most reactors, individual channels can be clearly defined, so a nodalization based on one-dimensional discretization of channels is usually the best choice. However, the core model may result in an extremely big problem that cannot be handled by the code, especially due to the fact that most of these programs date back to seventies and even sixties. The common practice is thus to define larger channels that average the properties of several real channels. It is in this process of averaging that the most extreme care has to be taken, because the appearance of subcooled boiling can be spuriously eliminated from the model. Besides, averaging channels imply that the power distribution as computed by the neutronic core-level code is not distributed as originally conceived and energy is only conserved in an integral fashion.

2.3 Control and limitation systems

As in any other industrial facility, a certain type of closed-loop control system is needed in order to maintain the plant at the desired operational conditions, i.e. to deliver the correct amount of electrical energy per unit time to the grid. Nuclear power plants usually have three levels of closed-loop systems that may take automatic control actions if needed. The first level is the reactor control system whose main goal is to have the reactor, the turbine and the generator to provide the correct amount of power. The usual actions taken by the control system involve moving the control rods inside the core to vary the criticality conditions and thus to change the thermal power extracted from the fuel, and regulating the aperture of the valves that control the steam flow into the turbine. Of course a lot of other actions are needed for the plant operation such as controlling the pressure of the main loop, the temperature of feedwater and even the chemical composition of the coolant.

Operating above the control level there exists the limitation system, which constantly monitors a certain number of variables and makes sure that the reactor always operates within a safe region of the parameter space. In case one or more signals indicate that the plant is approaching a potentially unsafe condition, the limitation system triggers some actions aimed at getting the reactor away of the unsafe zone which mostly consists of asking the control system to decrease the fission power.

At the topmost level, the reactor protection system can be found. Its objective is to shut the reactor down whenever an abnormal condition is detected. The protection system directly trips the reactor by inserting the control rods as fast as possible and, if necessary, by other means such as injecting a heavy neutron absorber—such as boron or gadolinium—into the reactor core.

Depending on the extent up to which these system are to be modeled, the resulting equations may be fairly simple or extremely complicated. In effect, the global behavior of closed-loop control systems can be easily written in high-level computational languages. However, if a faithful representation of the actual logic involved in the actual hardware implementation—which includes redundancies, error checking, conditional use of multiple signals, replacement of data, etc.—may only be possible with active participation of the manufacturer. The level of detail required depends on the other models, as if plant models do not provide redundant signals there is no point in including a voter, second-maximum selection or two-out-of-three logical comparisons.

Control system models become important mainly in transient analysis, as the steady-state conditions may be computed without needing control actions by setting all the actual values equal to their setpoints, or conversely according to the nature of the signal being computed. For example, during the steady-state calculation, the thermal power can be set to its nominal setpoint—say 3000 MWth—whilst the setpoint for the neutronic power indicated by the neutron detectors can be set to the value obtained by the calculation—say 1.45×10^{14} $\text{cm}^{-2} \cdot \text{sec}^{-1}$—so that at $t = 0$ the power error is zero in both cases. The actual mathematical equations depend on the level of detail required, but in general they involve purely algebraic equations as first-order lags and integrators are usually re-written as difference equations using the Tustin or bi-linear approximation, especially when employing digital control systems.

2.4 Plant conditions

The mathematical models needed to simulate and analyze the rest of the systems and components located outside the reactor core are significantly simpler than the neutronic, thermalhydraulics and control system equations. The problem consists of solving adiabatic flow through a network of one-dimensional pipes, heat transfer in one-dimensional exchangers, lumped models of pumps using characteristic curves and ad-hoc integral models for large components such as the pressurizer, the turbine and the generator. The steam generators are an interesting exception, because boiling flow presents some caveats even in the steady-state case. Therefore, either they are treated as special components with ad-hoc equations or they are modeled with the same code used for solving the thermalhydraulic equations.

Last but not least, plant models should take into account the actuators of the different components, that translate the decisions that the control system takes into the form of proper variables employed in the mathematical equations of the model. For example, the control system may set a binary signal representing the action of opening a valve. There should exist one mathematical actuator that ought to integrate the binary signal with the velocity of the motor that drives the stem and store its position so that the head losses can be computed by the piping model. Even the control rod driving mechanisms are often taken into account by the plant model. Conversely, it may be needed to model sensors that conform the signals that the control system needs to take its decisions. Sometimes they are simple lags that take into account the thermal capacities of thermistors but they may include also random noise or simulate redundancies. The reference example of sensors is that of the self-powered neutron detectors which, due to their physical principles, involve a certain transfer function between the input neutron flux and its electric current output. As neutronic codes do not in general model such behavior, the in-core neutron detectors should also be taken into account by the plant conditions code.

3 COUPLED CALCULATIONS

In view of the discussions of the previous section, it is apparent that any engineering analysis of a nuclear power plant requiring a certain degree of accuracy should somehow couple computational codes that solve mathematical problems that are essentially different. Not only are these differences related to the mathematical nature of each of them—such as the eigenvalue formulation of the steady-state neutronic problem and the purely algebraic equations of the control system—but also to the profile of the people that write the codes. Experts in two-phase flow are not usually familiar with core-level neutronic computations, and conversely. The result is that they tackle their problems in different ways, giving rise to further differences in the resulting computational codes.

Either in steady-state or in transient computations (which we discuss in sections 3.1 and 3.2 respectively), it is clear that some kind of data exchange between the codes needs to be devised in order to solve the complete problem. The most archaic way of sending information from one process to another one is by means of files. Needless to say, even if using RAM disks or any other buffering scheme, this solution turns out to be impracticable in most cases. It is true that when no access to the source code is granted by the original developers, this is the only way to couple different codes. However, each program has to be executed from scratch for each step of the computation. We recommend other alternatives to be explored, which may include not using privative codes at all.

A second option is to compile all the source codes into one single big executable and to include ad-hoc routines to map the output of one code to the input of another one. This mapping can be performed by copying data from one memory location to another or, even better, to force the data structures of two—or more—codes to point to the same memory location in user space. The way of invoking the codes and of advancing each step of the computation has to be also rewritten to reflect the fact that instead of several independent codes we now have one single big executable. However, even though it seems a good solution, depending on the nature of the specific codes this approach may not be practicable. And if it is, unless carefully designed, the resulting coupled code may lack some flexibility needed when users of such complex engineering tools start to request changes or features after some months of usage.

A third option—which is the one solution that TECNA S.A. developed for Nucleoeléctrica Argentina S.A. for the completion of the Atucha II Nuclear Power Plant (Mazzantini et al., 2011)—consists of slight modifications of each code so, after a computation step is performed, some of the data stored in the internal structures is copied into a segment of shared memory, which can be accessed by any other process running in the same host. At the same time, before advancing a step, each code should read information from a certain shared-memory object and interpret it as input data. The codes can be synchronized by using shared semaphores, which are successively decremented (waited) and incremented (posted) as the computation evolves.

This shared-memory based coupling scheme has the advantage that relatively little changes are made to the code. Besides, if the routines that export to and import from shared memory segments are coded to read what information to exchange from a certain definition files at runtime, a great deal of flexibility can be achieved. With a careful design of the way the information is written and read from shared memory and the sequence in which the semaphores are set, complex situations can be solved and even parallel computations can be performed. Indeed, the proposed scheme for Atucha II involves one spatial neutron kinetics code, four-hundred and fifty one thermalhydraulic codes (one for each coolant channel of the reactor core), one official emulator of the plant digital control system provided by the manufacturer and one plant code developed by the original plant designer (Di Cesare et al., 2009, 2010, 2011, 2012; Mazzantini et al., 2011). These four-hundred and fifty four codes are commanded by a main code that forks to execute the specific codes and is in charge of posting and waiting the appropriate semaphores. The coolant channels are solved in parallel by sending one process to each available core and as it was needed to change the source files of the codes anyway, OpenMP directives were included in the neutronic code to allow for faster execution (Rivero and Theler, 2010).

3.1 Steady state

Even if we wanted to solve only transient problems, we would need a steady-state solution in order to use it as the initial condition. Moreover, steady-state distributions give a lot of information for the reactor analysis by themselves, especially if comparing results for different nominal conditions, i.e at full power or at eighty-percent power, with fresh or equilibrium core, etc. From a mathematical point of view, the steady-state conditions are obtained by setting all the time derivatives to zero. Using any spatial discretization scheme such as the one proposed by Theler et al. (2012) or by Theler (2011)—except some particular cases of finite elements without lumping the mass matrix—the resulting equations can be expressed as

$$\mathbf{F}(\mathbf{x}^\star) = 0 \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is a vector that contains all the problem unknowns—including all the distributions such as the vector $\boldsymbol{\phi}$ of neutron fluxes—and $\mathbf{F}$ is a vector of n non-linear algebraic functions. The star indicates a steady-state solution. The size n of the state vector is the sum of the sizes of the specific problems. For the sake of clarity, let us assume that we are dealing only with two aspects, say neutronics and thermalhydraulics. We may further simplify the problem and state that the solution of the neutronic problem $\mathbf{x}_N \in \mathbb{R}^{n_N} = \boldsymbol{\phi}$ depends on a certain temperature distribution $\mathbf{x}_T \in \mathbb{R}^{n_T}$, and that the solution of the thermalhydraulic problem $\mathbf{x}_T$ depends on the flux distribution $\mathbf{x}_N$. That is to say, the neutronic code computes a steady-state flux distribution as

$$\mathbf{x}_N = \mathbf{N}^\star(\mathbf{x}_T)$$

and the thermalhydraulic code computes a steady-state temperature distribution as

$$\mathbf{x}_T = \mathbf{T}^\star(\mathbf{x}_N)$$

where $\mathbf{N}^\star$ and $\mathbf{T}^\star$ represent the operations needed to solve the steady-state neutronic and thermalhydraulic problems respectively. The solution of the coupled problem entails finding a certain vector $\mathbf{x}_N^\star$ which when inserted into the thermalhydraulic code, gives rise to a certain vector $\mathbf{x}_T^\star$ that when inserted back into the neutronic code the same vector $\mathbf{x}_N^\star$ is obtained, and conversely. Therefore, the steady-state condition of equation (1) is equivalent to either

$$\mathbf{x}_T^\star = \mathbf{T}^\star\left[\mathbf{N}^\star\left(\mathbf{x}_T^\star\right)\right] \tag{2}$$

or

$$\mathbf{x}_N^\star = \mathbf{N}^\star\left[\mathbf{T}^\star\left(\mathbf{x}_N^\star\right)\right]$$

Sticking to equation (2), we may regard the right-hand member as compound discrete map equal to the composition of operators $\mathbf{T}^\star$ and $\mathbf{N}^\star$ applied to the vector $\mathbf{x}_T$:

$$\mathbf{x}_T^{i+1} = \mathbf{T}^\star\left[\mathbf{N}^\star\left(\mathbf{x}_T^i\right)\right]$$

If the discrete compound map has one stable fixed point, the successive temperature distributions $\mathbf{x}_T^i$ will eventually converge to the steady-state solution $\mathbf{x}_T^\star$. If the discrete map has more than one stable fixed point, the one the vectors $\mathbf{x}_T^i$ will converge to will depend on the topology of the phase space. If the map has only unstable fixed points, no convergence can be guaranteed. Fortunately, we have found that almost any configuration of interest has only one fixed point whose stability is stronger than the stability of the associated transient problem (see next section). That is to say, we found cases where the transient problem is unstable whilst the discrete map that leads to the steady-state solution is stable even in more complex problems that the one presented here.

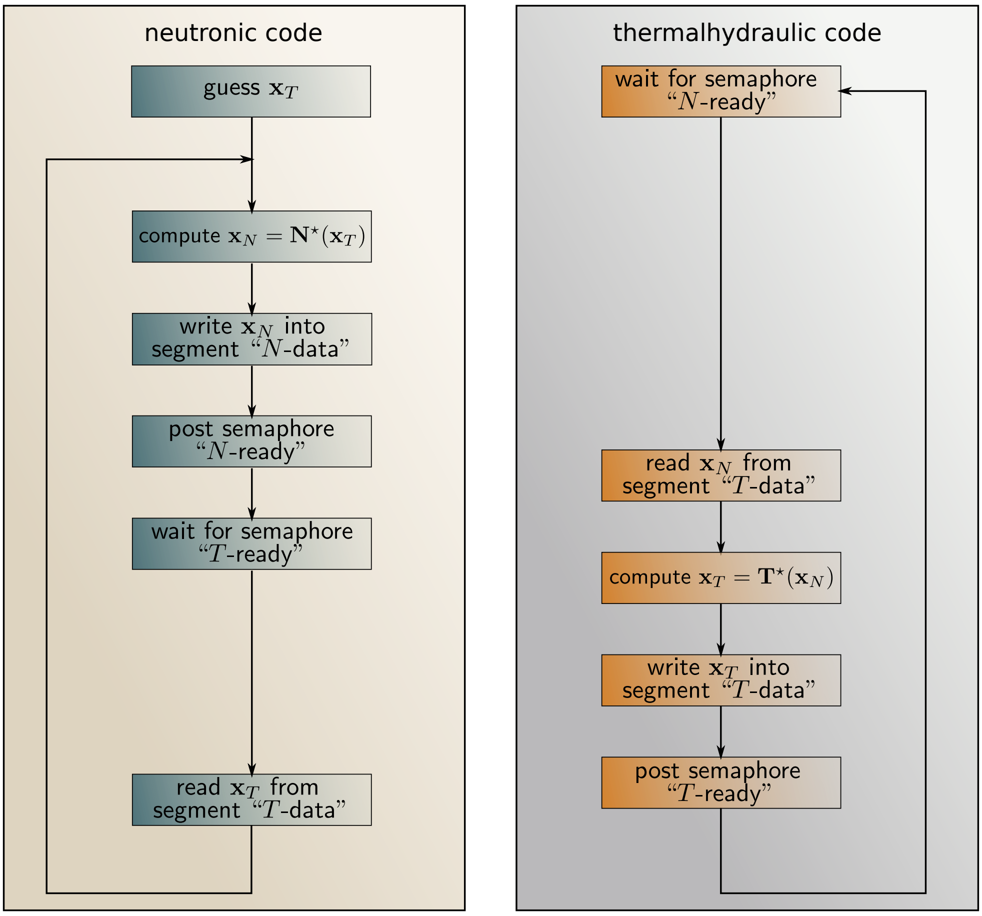

neutronic code
guess $\mathbf{x}_T$
compute $\mathbf{x}_N = \mathbf{N}^\star(\mathbf{x}_T)$
write $\mathbf{x}_N$ into segment "N-data"
post semaphore "N-ready"
wait for semaphore "T-ready"
read $\mathbf{x}_T$ from segment "T-data"
thermalhydraulic code
wait for semaphore "N-ready"
read $\mathbf{x}_N$ from segment "T-data"
compute $\mathbf{x}_T = \mathbf{T}^\star(\mathbf{x}_N)$
write $\mathbf{x}_T$ into segment "T-data"
post semaphore "T-ready"

Figure 5: A simplified schematic representation of the proposed computation of a coupled neutronic-thermalhydraulic steady-state problem using shared-memory objects for data exchange and shared semaphores for synchronization between two separate specific codes. The diagram does not show how the codes choose when and how to exit the iterative loop.

Summing up, to obtain a steady-state solution of a coupled problem where each aspect is solved by a number of specific codes[1] we can proceed in an iterative manner. Starting from a certain initial guess for the inputs of the neutronic code, we compute a first neutron flux distribution. We then feed that distribution into the other codes and collect again the inputs of the neutronic code, which we run again to obtain a second neutron flux distribution, and so on. We can regard the solution as converged when two successive neutron fluxes differ less than a specified tolerance or after a certain number of iterations have elapsed.

From a computational point of view, the coupling scheme may be implemented in several different ways—each of them having a number of advantages and disadvantages, some of which were discussed in section 3. In particular, figure 5 shows a graphic diagram of a possible implementation using two shared-memory objects named N-data and T-data and two shared semaphores named N-ready and T-ready. First, one of the codes has to start with an initial

[1] Only when all of them are able to compute the steady-state solution from a certain input in a single step. For example, this was not the case for the Atucha II project (Mazzantini et al., 2011) so we had to develop a slightly more complicated process to obtain the steady-state solution used as the initial condition of the transient computation.

guess of its input, say it is the neutronic code. Then, after computing its part of the problem, the neutronic code should put its results where the thermalhydraulic code can access it and tell the world that the neutronic calculation has finished so the other codes can proceed. After doing this, the neutronic code should wait for the thermalhydraulic code to finish in order to read its output, transform it to a neutronic input and compute again its part of the problem with the improved input. At the same time, the thermalhydraulic code should wait for the neutronic code to finish, read the neutronic output, compute its part of the problem, write the thermalhydraulic results into shared memory and tell the neutronic code that the thermalhydraulic calculation is over. This process can be generalized to an arbitrary number of codes, therefore splitting a large problem into a series of smaller and very specific calculations without needing to develop one single big code from the individual set of source files. The overhead needed to read from and to write to shared memory objects and to operate over shared semaphores is negligible with respect to the usual computational effort needed to solve each problem, which is not the case when exchanging information through the file system.

3.2 Transient cases

Even though a great deal of information can be extracted and used for analysis and design of nuclear power plants, it is in the study of mathematical models of transient problems where the most interesting features appear. Starting from a steady-state condition, a transient case usually introduces a certain perturbation to said steady-state, although unperturbed transients may be useful when analyzing asymptotic behavior under normal operation such as the frequency of the movement of the control rods. The possible disturbances introduced to trigger the transient problem may be classified into operational and safety cases. The first group includes whatever conditions that are to be expected to occur for the normal operation of the plant such as power cycles, expected shutdowns, changes in setpoints, etc. The second groups encompasses all the incidental and accidental situations that the power plant ought to be able to withstand, such as loss of flow accidents, loss of coolant accidents, station black-outs, etc. This distinction is important for two reasons. First, due to the fact that safety analysis reports are usually to be reviewed by independent bodies and are subject to regulatory conditions whilst studies of operational transients are not. And second, because the mathematical models involved in both groups are essentially different because safety transients involve extreme physical phenomena such as stratified choked flashing two-phase flow in counter-current conditions. Therefore, from the perspective of the analyst engineer that has to develop a coupled model that takes into account as many details as possible but at the same time consuming a reasonable amount of computational effort, the difference between operational and safety transients has to be well understood.

Being nuclear power plants big non-linear complex systems, any reasonable mathematical model ought to be also big, non-linear and complex. Therefore, linear stability analysis are only limited and time-domain studies are to be performed for example when addressing if a certain control algorithm is able to cope with long-term xenon instabilities and maintain oscillations within acceptable bounds for a wide variety of flux-disturbing conditions. It is in cases like this one that coupled transient suites like the one developed by TECNA become a prominent engineering analysis tool.

Mathematically speaking, once the spatial distributions have been discretized into a finite number of values and the spatial differential operators have been replaced by algebraic relations, the transient problem for a continuous variable t representing time can be casted in an analogous form to equation (1) as

$$\frac{d\mathbf{x}}{dt} = \mathbf{F}(\mathbf{x}) + \mathbf{G}(\mathbf{x}, t) \tag{3}$$

where $\mathbf{F}(\mathbf{x})$ is the same function of equation (1) and $\mathbf{G}(\mathbf{x}, t)$ contains all the time derivatives that were set to zero to obtain $\mathbf{F}(\mathbf{x})$ in section 3.1—for example terms involving the contribution due to neutron precursors to the total fission rate in the neutronic problem—plus eventually the particular perturbation under study as a function of time. The theoretical solution, which of course cannot be obtained exactly, is

$$\mathbf{x}(t) = \mathbf{x}(0) + \int_0^t \Big[\mathbf{F}\big(\mathbf{x}(\tau)\big) + \mathbf{G}\big(\mathbf{x}(\tau), \tau\big)\Big] \, d\tau$$

Discretizing the time domain and using an explicit Euler-based scheme, the solution at time $t_{i+1} = t_i + \Delta t$ can be estimated by using the solution at time t_i as

$$\mathbf{x}(t_{i+1}) = \Big[\mathbf{F}\big(\mathbf{x}(t_i)\big) + \mathbf{G}\big(\mathbf{x}(t_i), t_i\big)\Big] \cdot \Delta t \tag{4}$$

If we instead use a fully implicit scheme, the expression is

$$\mathbf{x}(t_{i+1}) = \Big[\mathbf{F}\big(\mathbf{x}(t_{i+1})\big) + \mathbf{G}\big(\mathbf{x}(t_{i+1}), t_{i+1}\big)\Big] \cdot \Delta t \tag{5}$$

A semi-implicit scheme can be obtained by using a certain coefficient $0 < \alpha < 1$ in

$$\mathbf{x}(t_{i+1}) = \bigg[\mathbf{F}\Big(\mathbf{x}\big(\alpha t_i + (1-\alpha) t_{i+1}\big)\Big) + \mathbf{G}\Big(\mathbf{x}\big(\alpha t_i + (1-\alpha) t_{i+1}\big), \alpha t_i + (1-\alpha) t_{i+1}\Big)\bigg] \cdot \Delta t \tag{6}$$

Either in the fully or semi-implicit schemes, advancing one single time step requires an iterative inner scheme because the solution $\mathbf{x}(t_{i+1})$ at t_{i+1} appears in both members of equations (5) and (6). The fully implicit scheme has a great numerical stability allowing for large time steps. The semi-implicit scheme with $\alpha = 0.5$ gives the best accuracy, as this special case turns equation (6) into a second-order approximation instead of the plain first-order cases with other values for α. On the other hand, using the explicit formulation the solution at the next time can be computed from the solution at the current time in a single step, giving the fastest algorithm. Which scheme to use and what time step to choose—which may vary with time—depends on the particular problem being solved. Of course the Euler scheme for solving differential algebraic equations is very basic and inefficient. However, the discussion about the differences between explicit and implicit formulations still apply to other schemes as well.

We now face the situation that we have to solve a big problem of the form (3) by using different codes, and go back to the neutronic-thermalhydraulic simplified problem of section 3.1. Once the codes have obtained the solution $\mathbf{x}(t_i)$ for time t_i (which may be $t_0 = 0$ and $\mathbf{x}(t_0) = \mathbf{x}^\star$ for the initial step) then we can advance one step of the neutronic transient code to compute by using a explicit scheme as

$$\mathbf{x}_N(t_{i+1}) = \mathbf{N}\left(\begin{bmatrix}\mathbf{x}_N(t_i) \\ \mathbf{x}_T(t_i)\end{bmatrix}\right)$$

where now the function $\mathbf{N}$ depends on both the previous thermal and neutron distribution because in the transient problem the new flux $\mathbf{x}_N(t_{i+1})$ depends on the previous flux $\mathbf{x}_N(t_i)$ while

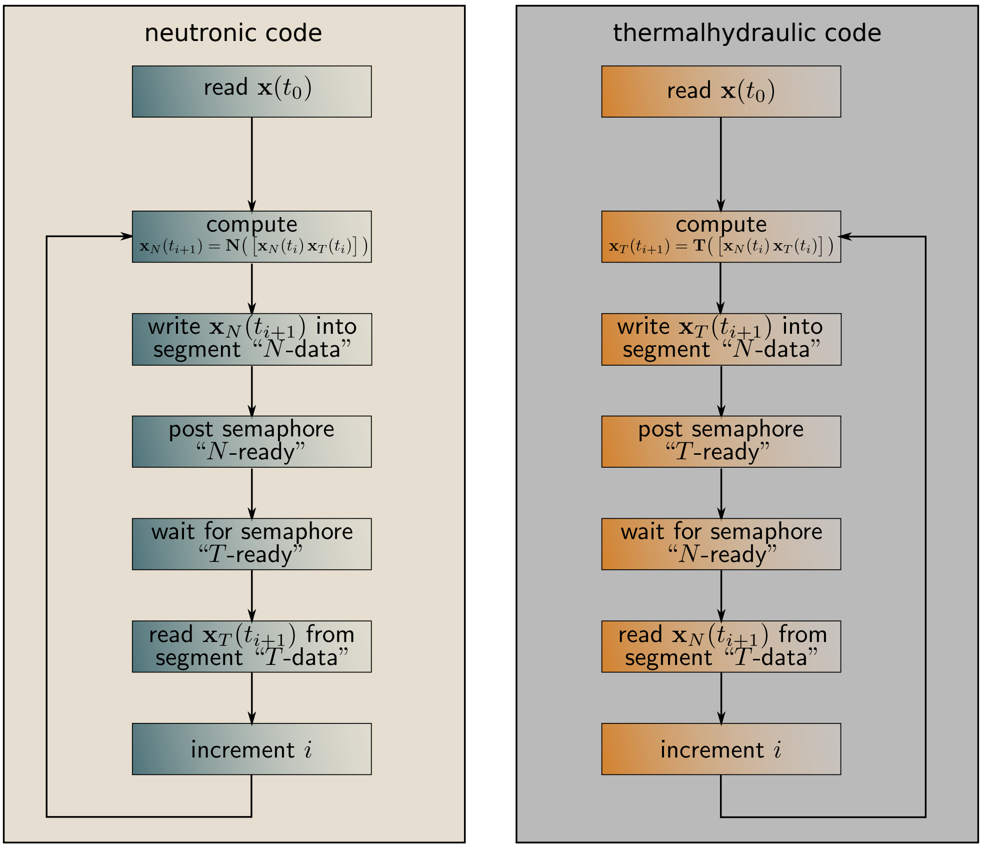

neutronic code
read $\mathbf{x}(t_0)$
compute
$\mathbf{x}_N(t_{i+1}) = \mathbf{N}(\,[\mathbf{x}_N(t_i)\,\mathbf{x}_T(t_i)]\,)$
write $\mathbf{x}_N(t_{i+1})$ into segment "N-data"
post semaphore "N-ready"
wait for semaphore "T-ready"
read $\mathbf{x}_T(t_{i+1})$ from segment "T-data"
increment i
thermalhydraulic code
read $\mathbf{x}(t_0)$
compute
$\mathbf{x}_T(t_{i+1}) = \mathbf{T}(\,[\mathbf{x}_N(t_i)\,\mathbf{x}_T(t_i)]\,)$
write $\mathbf{x}_T(t_{i+1})$ into segment "N-data"
post semaphore "T-ready"
wait for semaphore "N-ready"
read $\mathbf{x}_N(t_{i+1})$ from segment "T-data"
increment i

Figure 6: A simplified schematic representation of the proposed computation of a coupled neutronic-thermalhydraulic transient problem using an explicit scheme. Each code compute its part in parallel and waits until the other one finishes its part before proceeding to the next time step.

that's not the case in the steady-state calculation. Then, we could proceed to compute the thermal-hydraulic step either as

$$\mathbf{x}_T(t_{i+1}) = \mathbf{T}\left(\begin{bmatrix}\mathbf{x}_N(t_i)\\ \mathbf{x}_T(t_i)\end{bmatrix}\right) \tag{7}$$

or as

$$\mathbf{x}_T(t_{i+1}) = \mathbf{T}\left(\begin{bmatrix}\mathbf{x}_N(t_{i+1})\\ \mathbf{x}_T(t_i)\end{bmatrix}\right) \tag{8}$$

Equation (7) represents an explicit scheme, while equation (8) can be thought of a mixed-scheme in which the neutronic calculation is fully explicit while the $\mathbf{T}$ step is a mix between an explicit thermalhydraulic step with implicit neutronic information. Indeed, this is the most common case used to solve problems of a certain complexity as the thermalhydraulic problem where the thermal and hydraulic problems are separated and one is solved using the most recent information the other has computed. However, in view of the shared-memory coupling discussion above, equation (7) has the advantage that both neutronic and thermalhydraulic problems can be solved in parallel, as they both depend on data computed at time t_i.

In effect, figure 6 shows a simplified schematic implementation of the fully-explicit case as written in equation (7). The flow implies that both the neutronic and the thermalhydraulic

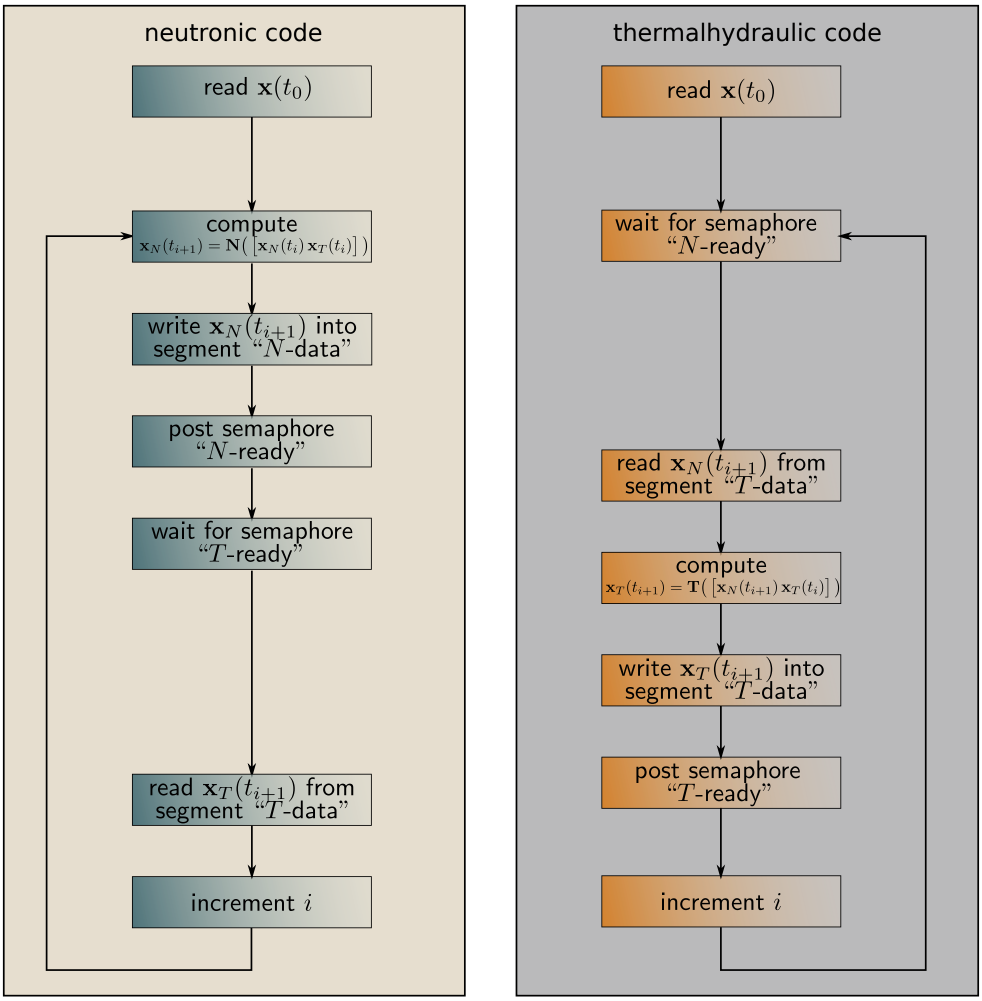

neutronic code
read $\mathbf{x}(t_0)$
compute $\mathbf{x}_N(t_{i+1}) = \mathbf{N}(\,[\mathbf{x}_N(t_i)\,\mathbf{x}_T(t_i)]\,)$
write $\mathbf{x}_N(t_{i+1})$ into segment "N-data"
post semaphore "N-ready"
wait for semaphore "T-ready"
read $\mathbf{x}_T(t_{i+1})$ from segment "T-data"
increment i
thermalhydraulic code
read $\mathbf{x}(t_0)$
wait for semaphore "N-ready"
read $\mathbf{x}_N(t_{i+1})$ from segment "T-data"
compute $\mathbf{x}_T(t_{i+1}) = \mathbf{T}(\,[\mathbf{x}_N(t_{i+1})\,\mathbf{x}_T(t_i)]\,)$
write $\mathbf{x}_T(t_{i+1})$ into segment "T-data"
post semaphore "T-ready"
increment i

Figure 7: A simplified schematic representation of the proposed computation of a coupled neutronic-thermalhydraulic transient problem using a mixed scheme. The neutronic code advances as in figure 6 but the thermalhydraulic code waits until the new neutron flux is available to compute the temperature distribution.

compute their part in parallel. As can be seen, the scheme is completely symmetric between both codes. If, as expected, one codes needs more CPU time to compute its part of the problem, the other one will be idle waiting for it to finish before advancing to the next step.

Figure 7 illustrates how a coupling scheme using shared-memory objects and semaphores would look if using the mixed approach of equation (8). The neutronic code works exactly as in figure 6 but the thermalhydraulic code waits until $\mathbf{x}_N(t_{i+1})$ is available before computing $\mathbf{x}_T(t_{i+1})$. In this case, the problem cannot be solved in parallel but larger time steps can be used.

A careful analysis of figures 6 and 7 shows that the first scheme can be turned into the second one by just shifting the wait over "N"-ready and read $\mathbf{x}_N$ operations three steps up. If the codes allow the user to choose when the operations over the shared objects are performed at run-time, no recompilation is needed to use either method.

The proposed coupling method can be extended to an arbitrary number of specific codes, and both explicit and mixed schemes can be used at the same time. Also, the method can be extended to handle problems in which the specific codes use different time steps to advance the transient solution. For example, in reference Mazzantini et al. (2011) we run the four-hundred and fifty one thermalhydraulic codes explicitly in parallel while the neutronic usesa smaller time step—due to the Courant-Friedrichs-Lewy stability condition—than the one that the control and plant codes use, which in turn use the mixed formulation to exchange information amongst them.

4 CONCLUSIONS

Nuclear power plants are big complex systems that require big complex mathematical models to be described, analyzed and optimized. Depending on the nature of the engineering analysis needed—whether an operational transient with emphasis on control of xenon-induced instabilities or safety transients involving boiling two-phase mixtures in extreme flow conditions—specialized computational codes ought to be used. We have described the mathematical grounds of the four main aspects that should be taken into account when modeling a nuclear power plant, namely neutronics, thermalhydraulics, plant conditions and control systems. Afterward, we discussed how the particular codes that solve these problems independently can be coupled together in order to share information and compute both steady-state and transient plant conditions. A formulation based on shared-memory segments used for data exchange and shared semaphores for synchronization was introduced and compared to other methods. For the analysis of transient cases, two coupling schemes were considered. One is based on a fully explicit formulation and is particularly suitable for parallel computations. The other one is semi-implicit and may allow for larger time steps to be used. We showed that if special care is taken when modifying the original codes to handle shared objects, one scheme can be turned into another one at run-time.

We are currently studying and developing a new coupling scheme which mixes inter-process communication over shared objects for coupling certain codes and the inclusion of several specific codes into a single executable by linking together the compiled objects with general interface routines. Data structures are converted, copied or—if the codes allow it—even directly pointed to user-space memory segments that can be accessed by several numerical routines. This new scheme will allow for even more flexibility and control over the whole coupled analysis.

There are still many unexplored areas and a lot of caveats to be circumscribed. Future works include extending the coupling capabilities to distributed calculations clusters using TCP-based sockets and new studies to apply data exchange to implicit numerical schemes. We are confident that these kind of developments will result in faster and better coupled calculations that will provide nuclear engineers with better tools to design, analyze and optimize nuclear power plants.

REFERENCES

Chadwick M., Herman M., Oblozinsky P., Dunn M., Danon Y., Kahler A., Smith D., Pritychenko B., Arbanas G., Arcilla R., Brewer R., Brown D., Capote R., Carlson A., Cho Y., Derrien H., Guber K., Hale G., Hoblit S., Holloway S., Johnson T., Kawano T., Kiedrowski B., Kim H., Kunieda S., Larson N., Leal L., Lestone J., Little R., McCutchan E., MacFarlane R., MacInnes M., Mattoon C., McKnight R., Mughabghab S., Nobre G., Palmiotti G., Palumbo A., Pigni M., Pronyaev V., Sayer R., Sonzogni A., Summers N., Talou P., Thompson I., Trkov A., Vogt R., van der Marck S., Wallner A., White M., Wiarda D., and Young P. ENDF/B-VII.1 Nuclear Data for Science and Technology: Cross sections, Covariances, Fission Product Yields and Decay Data. *Nuclear Data Sheets*, 112(12):2887–2996, 2011. Special Issue on ENDF/B-VII.1 Library.

Di Cesare J., Garbero R., Schivo M., Rivero M., and Theler G. Desarrollo y acople de códigos de cálculo para la simulación de transitorios de operación de la Central Nuclear Atucha II. In Proceedings of *XXXVI Reunión Anual de la Asociación Argentina de Tecnología Nuclear*. 2009.

Di Cesare J., Garbero R., Schivo M., Rivero M., and Theler G. Desarrollo y acople de códigos de cálculo para la simulación de transitorios de operación de la Central Nuclear Atucha II. In Proceedings of *XXXVII Reunión Anual de la Asociación Argentina de Tecnología Nuclear*. 2010.

Di Cesare J., Garbero R., Schivo M., Rivero M., and Theler G. Inclusión del emulador de Teleperm XS en la suite de cálculo acoplada DyPRA para transitorios de operación de Atucha II. In Proceedings of *XXXVIII Reunión Anual de la Asociación Argentina de Tecnología Nuclear*. 2011.

Di Cesare J., Garbero R., Schivo M., Rivero M., and Theler G. Emulación del control de potencia y nivel de presurizador con la suite de cálculo acoplado DyPRA en transitorios de operación de Atucha II. In Proceedings of *XXXIX Reunión Anual de la Asociación Argentina de Tecnología Nuclear*. 2012.

Leppänen J. *Serpent – a Continuous-energy Monte Carlo Reactor Physics Burnup Calculation Code*, 2012.

Mazzantini O., Schivo M., Di Cesare J., Garbero R., Rivero M., and Theler G. A coupled calculation suite for Atucha II operational transients analysis. *Science and Technology of Nuclear Installations*, 2011:785304, 2011.

Rivero M. and Theler G. Optimization of the numerical solution of the nuclear reactor neutron diffusion equation. In Proceedings of *Congreso Mundial y Exposición Ingeniería*. 2010.

Theler G. Análisis no lineal de inestabilidades en el problema acoplado termohidráulico-neutrónico. 2008. Tesis de la Carrera de Maestría en Ingeniería, Insituto Balseiro.

Theler G. Milonga: a free nuclear reactor core analysis code. 2011. http://ib.cnea.gov.ar/~thelerg/wasora/milonga.

Theler G., Clausse A., and Bonetto F.J. A moving boiling-boundary model of an arbitrary-powered two-phase flow loop. *Mecánica Computacional*, XXXI(5 Multiphase Flows):695–720, 2012.