Full-core multiphysics analysis of Molten Salt Reactor Experiment using parallel code

Andrei Rykhlevskii

andreir2@illinois.edu

INTRODUCTION

The Molten Salt Reactor (MSR) is an advanced type of reactor which was developed at Oak Ridge National Laboratory (ORNL) in the 1950s and was operated in the 1960s. In the MSR, fluorides of fissile and/or fertile materials (i.e. UF_4, PuF_3 and/or ThF_4) are mixed with carrier salts to form a liquid fuel which is circulated in a loop-type primary circuit [1]. This innovation leads to immediate advantages over traditional, solid-fueled, reactors. These include near-atmospheric pressure in the primary loop, relatively high coolant temperature, outstanding neutron economy, a high level of inherent safety, reduced fuel preprocessing, and the ability to continuously remove fission products and add fissile and/or fertile elements [2].

LITERATURE REVIEW

MSR modeling efforts describe steady-state and transient behavior. Krepel et al. extended the in-house Light Water Reactor (LWR) diffusion code DYN3D to consider drift of delayed neutron precursors alongside the reactor temperature profile, re-casting the extended code as DYN3D-MSR [3]. That work compared DYN3D-MSR against experimental Molten Salt Reactor Experiment (MSRE) data and then used it to simulate local fuel channel blockages as well as local temperature perturbations. In a similar vein, Kophazi et. al. used iterative coupling between in-house three-dimensional neutronic and one-dimensional heat conduction models DALTON and THERM to analyze normal MSRE operation as well as channel-blocking-incident transients [4]. The Kophazi model added entrance effects of heat transfer coefficients as well as thermal coupling between fuel channels through moderator heat conduction. More recently, Cammi et. al. performed a 2D-axisymmetric single-channel analysis of the Molten Salt Breeder Reactor (MSBR) using the commercial finite element package COMSOL Multiphysics [5]. That work directly solved the fuel salt velocity field, used heterogeneous group constants in fuel and moderator regions, and employed a software package (COMSOL) intrinsically designed for coupled multi-physics simulation. Additionally, Aufiero et. al [6] have begun to approach transient simulations in the Molten Salt Fast Reactor (MSFR) by directly coupling Serpent 2 Monte Carlo neutronics with OpenFOAM [7] thermal-hydraulics.

APPROACH AND METHOD

Recently developed multiphysics code Moltres [8] could be employed for simulating MSRs. By implementing deterministic neutronics and thermal hydraulics in the context of the Multiphysics Object-Oriented Simulation Environment (MOOSE) finite element modeling framework, Moltres solves arbitrary-group neutron diffusion, temperature, and precursor governing equations in anywhere from one to three dimensions and can be deployed on an arbitrary number of processing units. Through its computing power, Moltres is devoted to previously unmatched fidelity in coupled neutronics and thermal hydraulics MSR simulation.

Moreover, Moltres depends on the MOOSE framework, [9] which is Lesser GNU Public License (LGPL) code that itself leans on LibMesh [10], a LGPL finite element library, and PetSc [11], a Berkeley Software Distribution (BSD)-licensed toolkit for solving nonlinear equations yielded by discretizing PDEs. MOOSE and LibMesh translate weak PDE forms defined by applications (e.g. Moltres) into residual and Jacobian functions. These functions are the inputs into PetSc Newton-Raphson solution routines. All codes use MPI for parallel communication and are easily deployed on massively-parallel cluster-computing platforms. MOOSE applications by default use monolithic and implicit methods ideal for closely-coupled and multi-scale physics, such as the model problem described in this work. However, Moltres can also use explicit time-stepping routines as well as segregated solution methods, making it extensible to myriad future modeling challenges.

In Moltres, neutrons are described with time-dependent multi-group diffusion theory:

$$\frac{1}{v_g}\frac{\partial \phi_g}{\partial t} - \nabla \cdot D_g \nabla \phi_g + \Sigma_g^r \phi_g = \chi_g^p \sum_{g'=1}^{G}(1-\beta)\nu\Sigma_{g'}^f \phi_{g'} + \sum_{g \neq g'}^{G} \Sigma_{g' \to g}^s \phi_{g'} + \chi_g^d \sum_i^I \lambda_i C_i \tag{1}$$

v_g = speed of neutrons in group g
ϕ_g = flux of neutrons in group g
t = time
D_g = diffusion coefficient for neutrons in group g
Σ_g^r = macro removal cross-section from group g
$\Sigma_{g' \to g}^s$ = macroscopic cross-section of scattering from g' to g
χ_g^p = prompt fission spectrum, neutrons in group g
G = number of discrete groups, g
ν = number of neutrons produced per fission
Σ_g^f = macro cross-section for fission in group g
χ_g^d = delayed fission spectrum, neutrons in group g
I = number of delayed neutron precursor groups
β = delayed neutron fraction
λ_i = average decay constant of delayed neutron precursors in i
C_i = concentration of delayed neutron precursors in group i

Delayed neutron precursors are described by equation:

$$\frac{\partial C_i}{\partial t} = \sum_{g'=1}^{G} \beta_i \nu \Sigma_{g'}^f \phi_{g'} - \lambda_i C_i - \frac{\partial}{\partial z} u C_i \tag{2}$$

with the last term representing the effect of fuel advection. The governing equation for the temperature is given by:

$$\rho_f c_{p,f} \frac{\partial T_f}{\partial t} + \nabla \cdot \left(\rho_f c_{p,f} \boldsymbol{u} \cdot T_f - k_f \nabla T_f \right) = Q_f \qquad (3)$$

ρ_f = density of fuel salt
$c_{p,f}$ = specific heat capacity of fuel salt
T_f = temperature of fuel salt
$\boldsymbol{u}$ = velocity of fuel salt
k_f = thermal conductivity of fuel salt
Q_f = source term

in the fuel, where the source term Q_f is defined by:

$$Q_f = \sum_{g=1}^{G} \epsilon_{f,g} \Sigma_{f,g} \phi_g \qquad (4)$$

In the moderator, the governing equation for temperature is given by:

$$\rho_g c_{p,g} \frac{\partial T_g}{\partial t} + \nabla \cdot \left(-k_g \nabla T_g \right) = Q_g \qquad (5)$$

Group constants must be generated by the modeler with either Serpent [12] or SCALE [13]. Moltres interpolates group constant temperature dependence from prepared tables, which must be constructed separately for fuel and moderator regions.

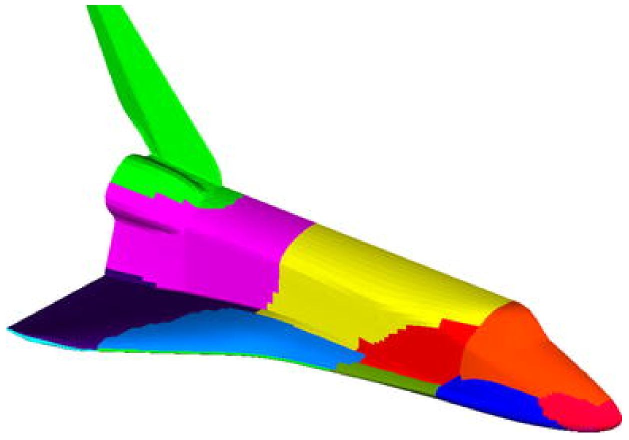

Fig. 1: Element-based domain decomposition of a surface mesh into 16 subdomains [14].

OBJECTIVES

The proposed work will solve set of equations (1-5) in 2-D geometry (r-z) for well-studied MSRE, verify and validate results with reference [15]. Moreover, strong and weak scaling will be performed to find out appropriateness using Moltres for heavy computing full-core analysis.

Additionally, finite-difference parallel solver will be developed to solve one-group neutron diffusion equation for the 2-D case for cross-verification Moltres' results.

Finally, strong and weak scaling study will be done for both codes to compare scalability finite-element and finite-difference methods. The proposed work will contribute to the coupled neutronics / thermal-hydraulics simulations for MSRs and will provide information for future development of multiphysics parallel solver for this type of reactors.

PARALLELIZATION

MOOSE framework uses LibMesh finite element library to solve set of physics equations. Different finite element formulations may be applied including Galerkin, Petrov-Galerkin, and discontinuous Galerkin methods [14]. Parallelism is achieved using domain decomposition through mesh partitioning, in which each processor contains the global mesh but in general computes only on a particular subset. Parallel implicit linear systems are supported via an interface with the PETSc library.

A standard non-overlapping domain decomposition approach is used in LibMesh to achieve data distribution on parallel computers as shown in Fig. 1. The elements in each subdomain are assigned to an individual processor. The two primary metrics in judging the quality of a partition are the subdomain mesh size and the number of "edge cuts" in the resulting partition. For a mesh composed of a single type of element, each subdomain should contain an equal number of elements so that the resulting domain decomposition is load balanced across all available processors. Usually, for MSR simulation adaptive mesh refinement and coarsening (AMR/C) scheme provided by LibMesh is employed. AMR/C allows do not explicitly specify domain decomposition, and provides optimal parallel performance with minimal user's efforts.

Developing C++/MPI code will employ finite-difference iterative method (i.e. Jacobi) to solve one-group neutron diffusion equation in 2-D mesh. Parallelization will be implemented using 2-D domain decomposition, when the 2D domain is divided into multiple sub-domains using horizontal and/or vertical axis depending on the available number of computer nodes.

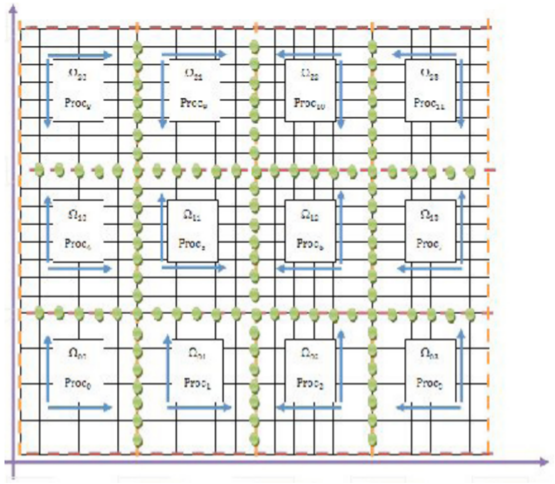

Fig. 2: The division of Ω_n into 12 subdomains for 2-D case [14].

For interior points neutron diffusion equation will be solved implicitly by Jacobi iterative scheme in combining with the boundary conditions. At the interface points of interior subdomains, equation will be solved by explicit iterative schemes. For illustration, consider a system N_x=N_y=24 with C=P×Q = 3×4 cluster nodes. The domain Ω_n can be decomposed to twelve subdomains as shown in Fig. 2. The proposed approach fulfills the suitability for the implementation on Linux PC cluster through the minimization of inter-process communication by restricting the exchange of data to the interface between the sub-domains. To examine the efficiency and accuracy of the iterative algorithm, several numerical experiments using different number of nodes of the Linux PC cluster will be tested. The performance metrics should clearly show the benefit of using multicore system in terms of execution time reduction and speedup with respect to the sequential running in a single core.

REFERENCES

1. P. N. HAUBENREICH and J. R. ENGEL, "Experience with the Molten-Salt Reactor Experiment," *Nuclear Technology*, **8**, *2*, 118–136 (Feb. 1970).
2. D. LEBLANC, "Molten salt reactors: A new beginning for an old idea," *Nuclear Engineering and Design*, **240**, *6*, 1644–1656 (Jun. 2010).
3. J. KÅŹEPEL, U. ROHDE, U. GRUNDMANN, and F.-P. WEISS, "DYN3D-MSR spatial dynamics code for molten salt reactors," *Annals of Nuclear Energy*, **34**, *6*, 449–462 (Jun. 2007).
4. J. KOPHAZI, D. LATHOUWERS, and J. KLOOSTERMAN, "Development of a Three-Dimensional Time-Dependent Calculation Scheme for Molten Salt Reactors and Validation of the Measurement Data of the Molten Salt Reactor Experiment," *Nuclear Science and Engineering*, **163**, *2*, 118–131 (2009).
5. A. CAMMI, V. DI MARCELLO, L. LUZZI, V. MEMOLI, and M. E. RICOTTI, "A multi-physics modelling approach to the dynamics of Molten Salt Reactors," *Annals of Nuclear Energy*, **38**, *6*, 1356–1372 (Jun. 2011).
6. M. AUFIERO, A. CAMMI, O. GEOFFROY, M. LOSA, L. LUZZI, M. E. RICOTTI, and H. ROUCH, "Development of an OpenFOAM model for the Molten Salt Fast Reactor transient analysis," *Chemical Engineering Science*, **111**, 390–401 (May 2014).
7. H. G. WELLER, G. TABOR, H. JASAK, and C. FUREBY, "A tensorial approach to computational continuum mechanics using object-oriented techniques," *Computers in physics*, **12**, *6*, 620–631 (1998).
8. A. LINDSAY, "Moltres, a code for simulating Molten Salt Reactors," (2017), https://github.com/arfc/moltres.
9. D. R. GASTON, C. J. PERMANN, J. W. PETERSON, A. E. SLAUGHTER, D. ANDRÅĄ, Y. WANG, M. P. SHORT, D. M. PEREZ, M. R. TONKS, J. ORTENSI, L. ZOU, and R. C. MARTINEAU, "Physics-based multiscale coupling for full core nuclear reactor simulation," *Annals of Nuclear Energy*, **84**, 45–54 (Oct. 2015).
10. B. S. KIRK, J. W. PETERSON, R. H. STOGNER, and G. F. CAREY, "libMesh: a C++ library for parallel adaptive mesh refinement/coarsening simulations," *Engineering with Computers*, **22**, *3-4*, 237–254 (Dec. 2006).
11. SATISH BALAY, SHRIRANG ABHYANKAR, MARK ADAMS, JED BROWN, PETER BRUNE, KRIS BUSCHELMAN, LISANDRO DALCIN, VICTOR EIJKHOUT, WILLIAM GROP, DINESH KAUSHIK, MATTHEW KNEPLEY, LOIS CURFMAN MCINNES, KARL RUPP, BARRY SMITH, STEFANO ZAMPINI, and HONG ZHANG, "PETSc Users Manual," Tech. Rep. ANL-95/11 - Revision 3.6, Argonne National Laboratory (2015).
12. J. LEPPÄNEN, M. PUSA, T. VIITANEN, V. VALTAVIRTA, and T. KALTIAISENAHO, "The Serpent Monte Carlo code: Status, development and applications in 2013," *Annals of Nuclear Energy*, **82**, 142–150 (Aug. 2015).
13. M. D. DEHART and S. M. BOWMAN, "Reactor Physics Methods and Analysis Capabilities in SCALE," *Nuclear Technology*, **174**, *2*, 196–213 (May 2011).
14. B. S. KIRK, J. W. PETERSON, R. H. STOGNER, and G. F. CAREY, "libMesh : a C++ library for parallel adaptive mesh refinement/coarsening simulations," *Engineering with Computers*, **22**, *3*, 237–254 (Dec 2006).
15. R. B. BRIGGS, "Molten-Salt Reactor Program semiannual progress report for period ending July 31, 1964," Technical Report Archive and Image Library ORNL-3708, Oak Ridge National Laboratory, Oak Ridge, TN, United States (1964).