Parallelization of MARS-KS code for Full-Core Thermal-Hydraulics Simulation of LWRs

N. H. Hoang, Y. K. Kwack, Y. S. Kim, S. H. Bae, and S. K. Sim
Environment and Energy Technology Inc.
hien@en2t.com

1. Introduction

Recent attention in the area of nuclear safety analysis has been paid to the development of high-fidelity and high-performance computing programs for solving challenging problems related to multi-dimensional, multi-scale, and multi-physics in-core phenomena [1-4]. Such a program is a collection of many computer codes for different fields, such as neutronics, thermal-hydraulics, fuel performance, and chemistry. The computer codes can be explicitly integrated into a huge platform (e.g., NURESIM and VERA-CS) or implicitly coupled to be one compact program (e.g., CUPID-RV and MARS-KS/CTF).

Importance for the application of computer programs, which use finite difference methods, to the high-complicated in-core phenomena is the use of fine meshes with a focus on the computational region of interest. The number of computational cells can be in the order of a million for a large region, for example, the nuclear reactor vessel. This is impossible for serial-processing programs which require small computer resources. Due to this fact, parallel processing techniques are usually demanded for modern computer programs. CUPID-RV or CTF code could be an example for the use of parallel processing for the full-core thermal-hydraulic simulation of light water reactors (LWRs). Both the codes use the Message Passing Interface (MPI) and domain decomposition for the code parallelization.

This paper presents an extension of MARS-KS 3D core subchannel module (CTF) parallelization to the MARS-KS regulatory thermal hydraulic system code which has been developed by upgrading the 3D core subchannel module using CTF code through implicit pressure coupling [4]. The parallel processing techniques used for the CTF are utilized for the MARS-KS code with considerations on the coupled 1D/3D interface and the 1D system analysis module of this code. A preliminary assessment of the parallel MARS-KS code is carried out against a verification test of five 3x3 fuel assemblies of the CASL PWR 3x3 HFP (Hot Full Power) [5].

2. MARS-KS Parallelization

The strategy chosen to parallelize the MARS-KS code is the single program multiple data (SPMD) and multiple instruction multiple data (MIMD) with distributed memory and domain decomposition. This strategy is inherited from the CTF subchannel analysis code and it was described in detail by Salko et. al. [5].

The framework of MARS-KS parallelization is briefly illustrated in Fig. 1. The MARS-KS code has two modules corresponding to the 1D system part and 3D core region. An interface formed by implicit pressure coupling exits between these modules. The 3D subchannel analysis module CTF was already parallelized but targeted at open platforms (e.g. Linux distributions) not Windows 32 as the MARS-KS code.

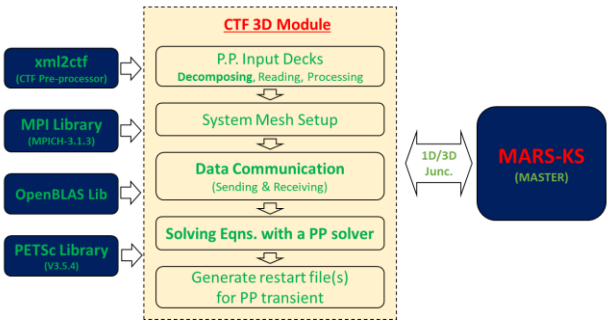


Fig. 1. MARS-KS parallelization framework

To parallelize the MARS-KS code, it is necessary to have the required libraries first and then enable the parallel processing of the 3D module. The parallelization of the 1D/3D interface is the next step. It is important to specify which domain connects to the 1D part, which 1D/3D connection data necessary for setting pressure matrices, and how 1D/3D connections affect numerical solvers. At the final step, it needs to give processors instructions to pass the 1D calculations. Here, only one processor, so-called root processor, will perform all the 1D calculations. Parallelization of the 1D part is not necessary since the size of 1D meshes is usually much smaller than the size of 3D meshes.

2.1. Library preparation

The required libraries include MPI, PETSc, and BLAS/LAPACK. A great difficulty is that the libraries are mostly for LINUX distributions, not Windows 32 bits–the target platform of MARS-KS, and they must compatible with each other. Therefore, MS-MPI version 9.0.1, PETSc version 3.13, and Intel BLAS/LAPACK (or Intel MKL) were chosen. The PETSc library that provides numerical solvers [6] was built for Windows 32 bit using CYGWIN. It was built together with the MS-MPI and Intel MKL (for vector and matrix operations) libraries. Since the current version of PETSc library is different from the original one, the

PETSc functions called through the program should be modified according to the current version.

2.2. Setup of 1D/3D connections

The setup of 1D/3D connections involves several steps. First, MARS-KS reads its 1D input file and automatically sets blind subdomain volumes or SDBVOLs (i.e., pseudo-time dependent volumes) to keep the information of 1D/3D connections. The MARS-KS single junction component is designed to specify the 1D/3D connections. It is noted that the global channel index should be used. This step is performed by the root processor, and this processor should distribute the 1D/3D connection information to other processors after this step. Next, each processor determines which 1D/3D connections belong to itself after having the domain information from a relevant 3D input file. The outcomes of this step are the local indices of 3D subchannels, cells, and gaps at the 1D/3D interface. Finally, each processor collects the information of 3D cells at the 1D/3D interface from other domains using the global indices of the 3D cells. The information is necessary for setting the pressure matrix of this domain. The processors also need to send the 1D/3D connection data to the root processor to set SDBVOLs. The final step is repeated every time step. Figure 2 shows the information of 1D/3D connections that should be considered.

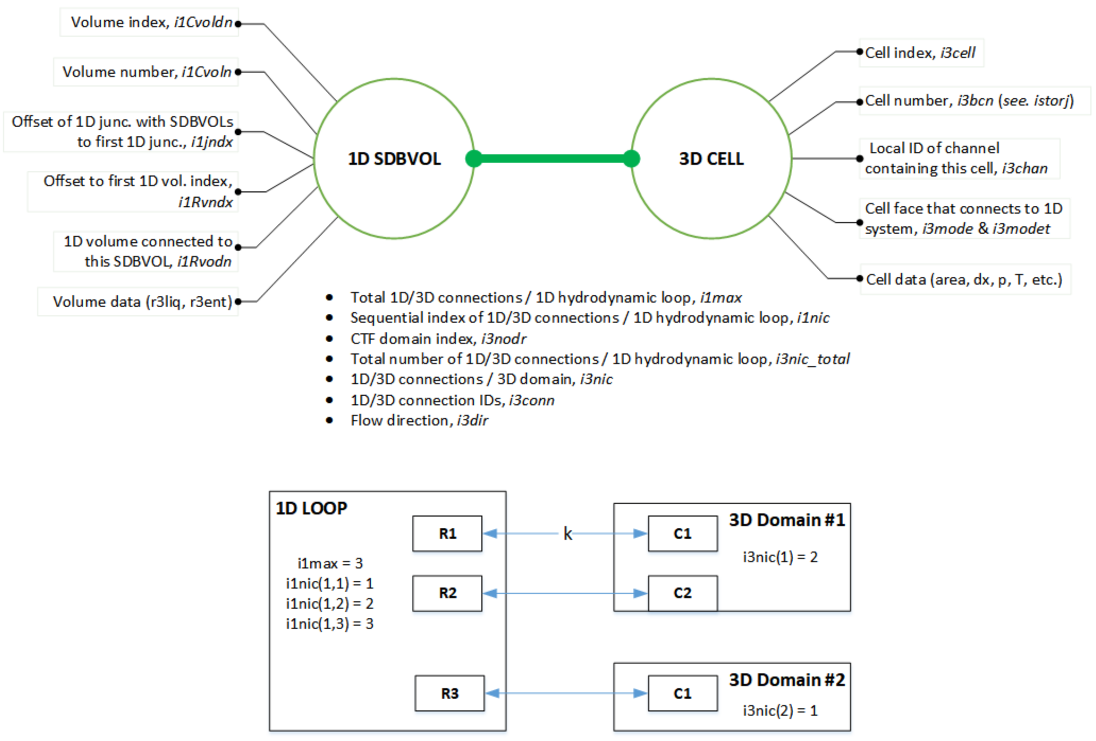

Volume index, *i1Cvoldn*
Volume number, *i1Cvoln*
Offset of 1D junc. with SDBVOLs to first 1D junc., *i1jndx*
Offset to first 1D vol. index, *i1Rvndx*
1D volume connected to this SDBVOL, *i1Rvodn*
Volume data (r3liq, r3ent)
1D SDBVOL
3D CELL
Cell index, *i3cell*
Cell number, *i3bcn (see. istorj)*
Local ID of channel containing this cell, *i3chan*
Cell face that connects to 1D system, *i3mode* & *i3modet*
Cell data (area, dx, p, T, etc.)
Total 1D/3D connections / 1D hydrodynamic loop, *i1max*
Sequential index of 1D/3D connections / 1D hydrodynamic loop, *i1nic*
CTF domain index, *i3nodr*
Total number of 1D/3D connections / 1D hydrodynamic loop, *i3nic_total*
1D/3D connections / 3D domain, *i3nic*
1D/3D connection IDs, *i3conn*
Flow direction, *i3dir*
1D LOOP
i1max = 3
i1nic(1,1) = 1
i1nic(1,2) = 2
i1nic(1,3) = 3
R1
R2
R3
k
3D Domain #1
i3nic(1) = 2
C1
C2
3D Domain #2
i3nic(2) = 1
C1

Fig. 2. Information of MARS-KS 1D/3D connections

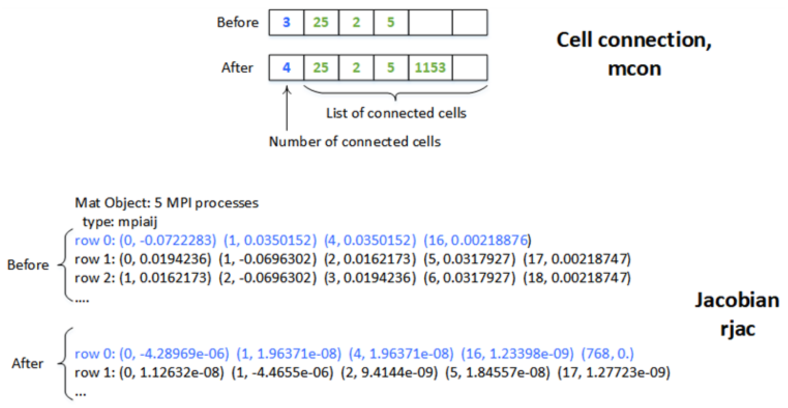
2.3. Solver modifications

The implicit coupling of CTF subchannel module with the MARS-KS code forms a linkage of all 3D cells at the 1D/3D interface by the following equation.

$$V_{k,i}^{n+1} = \alpha_{k,i} + \beta_{k,i}\left(\delta P_{Ci} - \xi_i - \sum_{j=1}^{NC} \eta_{i,j}\delta P_{Cj}\right) \quad (1)$$

where V_k, δP_C, α_k, β_k, and ξ are the phasic velocity, 3D cell pressure change, and coefficients correspondingly. The subscripts *i* and *j* specify the cell indices. Due to the presence of summation term in Eq. (1), the sparse pressure matrices of 3D domains are changed, increasing in size, requiring modifications of the solvers.

In MARS-KS, a global pressure matrix is established over the entire 3D region, and it is solved by the Krylov sub-domain solver that is provided by the PETSc library. Pressure correction coefficients are calculated from the mass and energy equations for each local cell of 3D domains. For 3D cells at the 1D/3D interface, the pressure correction equations include additional terms that come from the other 3D cells at this interface, as illustrated in Fig. 3. It is important to note that the global indices of such 3D cells are stored in the *mcon* variable rather than the local ones to distinguish the 3D domains. Once the pressure correction coefficients of a 3D cell are obtained, they will be sent to the root processor to build and solve the global pressure matrix.



```
Mat Object: 5 MPI processes
  type: mpiaij
Before
row 0: (0, -0.0722283)  (1, 0.0350152)  (4, 0.0350152)  (16, 0.00218876)
row 1: (0, 0.0194236)  (1, -0.0696302)  (2, 0.0162173)  (5, 0.0317927)  (17, 0.00218747)
row 2: (1, 0.0162173)  (2, -0.0696302)  (3, 0.0194236)  (6, 0.0317927)  (18, 0.00218747)
....

After
row 0: (0, -4.28969e-06)  (1, 1.96371e-08)  (4, 1.96371e-08)  (16, 1.23398e-09)  (768, 0.)
row 1: (0, 1.12632e-08)  (1, -4.4655e-06)  (2, 9.4144e-09)  (5, 1.84557e-08)  (17, 1.27723e-09)
...
```

Fig. 3. Modification of global pressure matrix

2.4. Re-construction of MARS-KS program

The MARS-KS program has two folds, input data processing and transient advancement, as shown in Fig. 4. Each part calls different procedures of the 3D subchannel module. The input data processing is also a two-fold process, and it is almost separated between the 1D and 3D parts. Here, the input reading and initialization were separated from the input data processing (InputDataProcess) due to the intermittence of the 1D/3D interface. The 1D part includes ReadNewProblem and InitializeNew. The 3D part becomes ReadCTF. The 3D initialization (init_3d) is performed at the beginning of TransientAdvance. In-betweens are the setups of 1D/3D connections, located in ReadCOBRA, cobrai, and even InitializeNew.

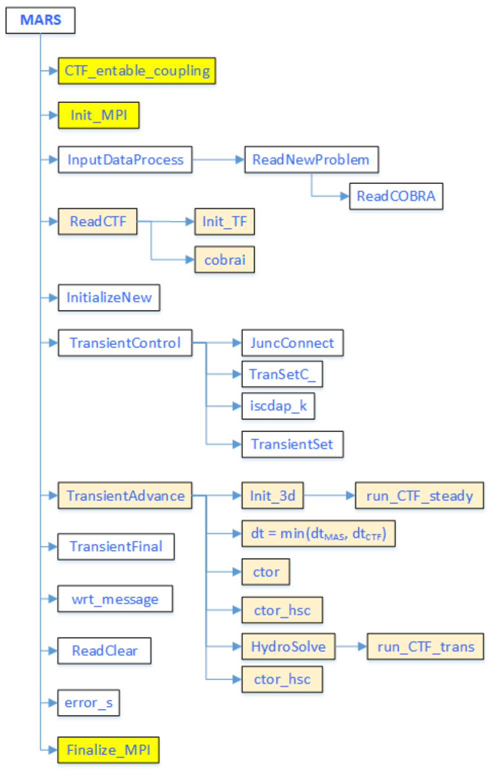


Fig. 4. Re-ordered MARS-KS structure

The transient advancement has a 1D time-step loop, and its main drivers are TransientAdvance, HydroSolve, and PressMatrixSolve. Along the time-step loop, 3D procedures are called for setting the subdomain volumes, solving the 3D governing equations (run_CTF_trans), and data exchange (ctor and ctor_hsc) between the 1D and 3D regions. It means that we cannot separate the 1D and 3D parts inside this loop for parallelization. Instead, all processes have to present in all 1D processes for the exchange of 1D/3D connection data only since the root processor is in charge of these processes. The data exchange is for setting SDBVOLs and calculating phasic velocities at the 1D/3D interface. To do that, detailed instructions were given to each processor. The key point is that only PressMatrixSolve calls run_CTF_trans whereas it is called at several places for implicit or semi-implicit numerical solution schemes of the 1D part. Hence, it needs to track all the subroutines that call PessMatrixSolve to tell whether the processors must be there or not.

3. Verification

The verification of the MARS-KS parallelization is performed for the application of MARS-KS code to the parallel simulation of five 3x3 fuel assembly test of the CASL PW 3x3 HFP [5]. Details of this test and its simulation model are described in Fig. 5 and Table 1. The simulation model was generated with the CTF preprocessor, and it has five domains corresponding to each fuel assembly. In-betweens of the domains are ghost subchannels, gaps, and rods that are shared by the domains. Both MARS-KS and CTF codes were used for this simulation.

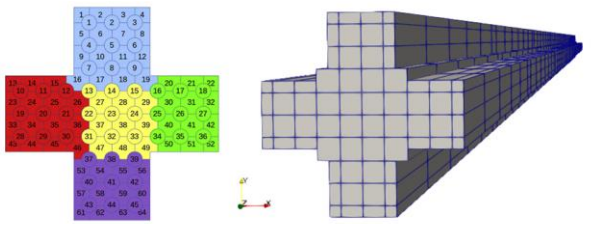


Fig. 5. Domain/mesh configurations of CASL PWR 3x3 HFP

Table 1: Description of verification test

Parameter	*Value*
Number of fuel assemblies (FAs)	5
Size of fuel array	3 x 3
Number of fuel rods per FA	8
Number of guide tube per FA	1
Number of spacer grids per FA	7
Heated length (mm)	3657.6
Assembly pitch (mm)	38.6
Inlet pressure (bar)	155.13
Inlet mass flow rate (kg/s)	13.4
Initial temperature (℃)	292.78
Average linear heat rate per rod (kW/m)	1.6266
Total number of simulated subchannels	64
Total number of simulated gaps	108
Total number of simulated rods	45
Number of axial nodes	49

Figures 6 and 7 show the serial and parallel calculation results of this test. The liquid enthalpy uniformly distributed over the test cross-section and gradually increased as the result of fuel-to-coolant heat transfer along the test section (Fig. 6). Particularly at subchannel 24, the calculation results with MARS-KS and CTF are well agreed as shown in Fig. 7. Only the liquid velocity at the outlet shows a somewhat difference between MARS-KS and CTF. For MARS-KS, 1D parts are connected to the inlet of subchannel 1 of domain 1 and the outlet of subchannel 64 of domain 5. The inlet and outlet of other subchannels are bounded by the same boundary conditions with the CTF calculations. Due to the pressure coupling at the 1D/3D connections, the first cell of subchannel 1 domain 1 and the last cell of subchannel 64 of domain 5 influence each other whereas they do not in the same CTF simulations (see Eq. 1).

The CTF subchannel module, in fact, only supports one vertical section in the parallel calculation. It means the domain decomposition is in the axial direction only. Therefore, it cannot reduce the number of cells at the inlet and outlet of this test to connect to the 1D part. It is a limitation of MARS-KS code. Nevertheless, it is not a significant matter for lateral connections.

The CPU time for a 10-seconds transient calculation with MARS-KS is 467.2 seconds, or roughly 46.72 seconds per one-second transient calculation. This speed is almost the same as the CTF parallel calculation (44.36 seconds).

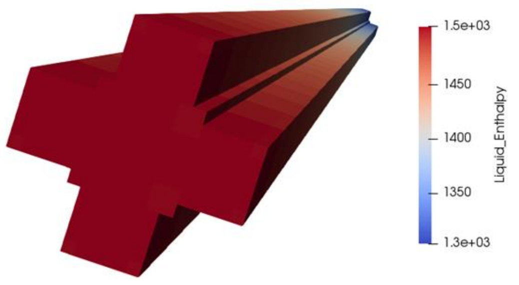


Fig. 6. Calculated liquid enthalpy by MARS-KS

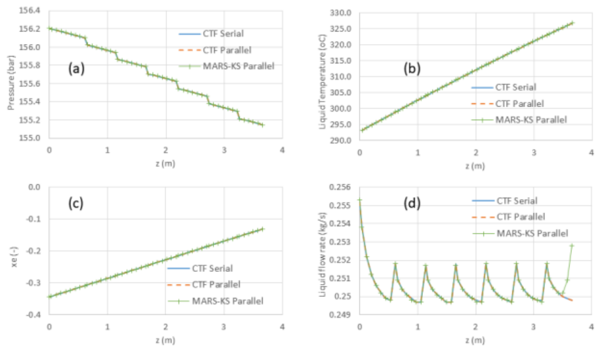


Fig. 7. Calculated parameters at subchannel 24

4. Conclusion

The MARS-KS code with the new 3D subchannel analysis module was parallelized using the Message Passing Interface method and domain decomposition as its CTF subchannel module. The MARS-KS code was also restructured to deal with 1D/3D interface. The verification test showed good performance of parallelized MARS-KS code. Further study is needed to resolve the limitation of domain decomposition in the axial direction as well as further improve the parallelization of the 1D/3D coupling. The parallelized MARS-KS code will be used for the transient DNBR evaluation of the full core simulation of the safety analysis especially for the small and modular reactor (SMRs) and Micro-Reactors under development worldwide.

REFERENCES

[1] C. Chauliac, J. M. Aragones, D. Bestion, D. G. Cacuci, N. Crouzet, F. P. Weiss, and M. A. Zimmermann, "NURESIM – A European simulation platform for nuclear reactor safety: Multi-scale and multi-physics calculations, sensitivity and uncertainty analysis", Nuc. Eng. Des., Vol. 241, pp. 3416-3426, 2011.
[2] J. A. Turner, K. Clarno, M. Sieger, R. Bartlett, B. Collins, R. Pawlowski, R. Schmidt, and R. Summers, "The virtual environment for reactor applications (VERA) design and architecture", J. Computational Physics, Vol. 326, pp. 544-568, 2016.
[3] H. Y. Yoon, I. K. Park, J. R. Lee, S. J. Lee, Y. J. Cho, S. J. Do, H. K. Cho, and J. J. Jeong, "A multiscale and multiphysics PWR safety analysis at a subchannel scale", Nuc. Sci. Eng., Vol. 194, No. 8-9, pp. 633-649, 2020.
[4] N. H. Hoang, Y. K. Kwack, S. H. Bae, Y. S. Kim, and S. K. Sim, "Improvement of MARS-KS sub-channel module using CTF code", Vol. 358, pp. 110431, 2020.
[5] R. K. Salko, R. C. Schmidt, M. N. Avramova, "Optimization and parallelization of the thermal-hydraulic subchannel code CTF for high-fidelity multi-physics applications", Annals of Nuclear Energy, Vol. 84, pp. 122-130, 2015.
[6] https://www.mcs.anl.gov/petsc/