OpenFOAM in Nuclear Applications

SERPENT and Multiphysics, LPSC, Grenoble, February 26-27, 2015
What this talk is:

- An introduction to OpenFOAM within the nuclear context
- A very brief summary of OpenFOAM development work over the years (work in which I have been involved)

What this talk is not:

- A comprehensive overview of all nuclear-related work using OpenFOAM
Outline

• Introduction to OpenFOAM - a nuclear perspective
• OpenFOAM for nuclear applications
  – Pebble Bed Modular Reactor (PBMR)
  – Prismatic High Temperature Reactor (Pennstate University)
  – Light Water Reactors (STARS group at PSI)
• Summary and Outlook
- OpenFOAM stands for Open Field Operation and Manipulation
- Officially described as an open-source CFD toolbox
  - Capabilities mirror those of commercial CFD
  - Free-to-use software without paying for licensing and support
- Far more than simply a set of CFD solvers and tools
- Underlying design - a comprehensive multi-physics framework for solving sets of PDEs using cell-centred polyhedral finite-volume methods
Equation Mimicking

- Natural language of continuum mechanics: partial differential equations
- Example: turbulence kinetic energy equation

\[
\frac{dk}{dt} + \nabla \cdot (\vec{u}k) - \nabla \cdot [(\nu + \nu_t)\nabla k] = \nu_t \left[ \frac{1}{2} (\nabla \vec{u} + (\nabla \vec{u})^T) \right]^2 - \frac{\epsilon_0}{k_0} k
\]

- Objective: represent PDEs in their natural language

```cpp
solve
{
    fvm::ddt(k)
    + fvm::div(phi, k)
    - fvm::laplacian(nu() + nut, k)
    ==
    nut*magSqr(symm(fvc::grad(U)))
    - fvm::Sp(epsilon/k, k)
};
```

- Correspondence between implementation and equation is clear
• Structure of OpenFOAM
  - Libraries
    - Foundation libraries: underlying mesh, field, matrix, etc. functionality
    - Physical modelling libraries: physics-specific functionality, eg. Material properties, specialized boundary conditions, turbulence models (largely focused on CFD)
    - Written in a reusable form
  - Executables
    - Solver applications (mostly focused on CFD)
    - Tools and utilities for creating, manipulating and post-processing meshes and field data
• Executables rely heavily on functionality provided by the underlying libraries
  - Typically only a few 100s of lines of code
  - Top level executables are easy to read, understand and modify
  - Low-level functions, eg. mesh handling, parallelisation, data I/O handled transparently: no need for special coding at top level
• Written from the base up in object-oriented C++
• Special Note: OpenFOAM is not a high-level language for scientific research like Matlab. This is production level optimized code suitable for large-scale computing
  - Fully parallelized: has been run successfully using thousands of cores
OpenFOAM

Physical Modelling Capability Highlights

• Incompressible & compressible flow: segregated pressure-based algorithms
• Heat transfer: Buoyancy driven flow, conjugate heat transfer, thermal radiation
• Multiphase flows: Euler-Euler, VOF free surface capturing and surface tracking
• Turbulent flows: RANS, DES, LES
• Combustion and chemical reactions: Engine simulations
• Turbomachinery, Stress analysis, fluid-structure interaction, electromagnetics, MHD, particle tracking, etc.
• Signification CFD jargon is used when talking about OpenFOAM
  – This can be confusing
  – Not immediately obvious how OpenFOAM may be applicable in the nuclear and reactor analysis context

  I will try to give some basic insight!

• A typical OpenFOAM case
  – Time control
  – One or more meshes: Spatial description of the domain with boundaries defined
  – One or more fields: Unknown variables associated with a mesh with boundary conditions defined
    – Fields on the same mesh can be manipulated and combined directly
    – Fields on different meshes can be mapped between meshes and thereafter manipulated and combined directly
  – One or more equations: Partial differential equations in terms of the unknown variables with source terms
    – Coupling between equations typically uses segregated algorithms
    – Newton-based nonlinear solvers are not currently available (eg. MOOSE platform)
Conceptual demonstration: A simple nuclear reactor
Massively Parallel Scaling

- Mesh size and resolution environment in 21st century are revolutionary different from (the comfortable) 1990s: thousands of processors, a billion cell mesh
- Complex hardware architecture without proper programming support complicates the programming and roll-out into real-world applications
- Still learning GPU lessons: potentially changing the way we write CFD codes back towards structured mesh codes and fixed internal infrastructure
- What about parallel CFD clouds: do we need custom infrastructure?

Scaling Test, VOF Free Surface Flow, 50m Cells

OpenFOAM Scalability on Firestorm

Cyprus Advanced HPC Workshop Winter 2012. Tracking: FH6190763 Feb/2012
OpenFOAM

Advantages

• Open architecture
  − Access to complete source: no secret tricks or cutting corners
  − Readily extendable – Custom boundary conditions, physical models, etc.
  − Significant community-based support available

• Open-source
  − No license costs

• Problem-independent numerics and discretization
  − Tackle non-standard continuum mechanics problems
  − Tackling complex multi-physics problems is made easier through equation mimicking
  − Runtime selection of discretization schemes and physical models
    − Case setup is highly flexible
  − Good track record in non-linear and strongly coupled problems

• Excellent piece of C++ code and software engineering
OpenFOAM

- Library is LARGE: 1000s of classes
  - Steep learning curve for non-standard physics
  - Good knowledge of C++ programming is essential
- Finite-volume methods have relatively low order accuracy (2nd order)
  - Significant mesh refinement may be needed to achieve the desired accuracy
  - This disadvantage is partly offset by the simplicity of the method
- Limited number of users in the nuclear community
  - Less community support for your particular project
  - Less existing code from which to borrow ideas
- In-house verification and validation is typically needed: community-based efforts may not consider your particular problem
- Recently: The OpenFOAM project has branched into two separate groups and development tracks (OpenFOAM Foundation and OpenFOAM Extend Project)
  - The developments are no longer closely compatible
  - The choice of which project to follow can have unexpected consequences down the line
Code Verification

- Primary responsibility of code developer

- In many respects, code verification also carried out by user community: with so many users, errors inevitably get identified quickly
  - In a sense: Direct Peer Review by the community

- Layered development assists verification efforts
  - Underlying libraries don’t change
  - If you have not recompiled or re-linked the code, you have not broken it!
  - Object-oriented code design
    - Component-wise V&V (unit tests) can easily be carried out
    - Data protection prevents unexpected data corruption from higher levels
Validation

• At basic level, validation of open-source code is identical to other efforts
  – Choose well established validation examples (experiments)
  – Perform detailed studies, varying model parameters and studying mesh dependence effects
  – Compare results with reference values

• Huge amount of effort: cannot be avoided

• Code validation done in a traditional way… helped by the fact that top-level solvers are very simple
  – single-physics solvers can be used to validate individual physical models
  – All critical functionality is shared and reused from library components
• Legacy pebble bed reactor analysis codes originally developed at Forschungszentrum Jülich (FZJ) in 1980s
  – Fast and effective but severely limited
    – Inherently 2D (axisymmetric)
    – Rectangular r-z discretization
    – Difficult to implement new methods
    – Difficult to couple with other codes

• PBMR design
  – New reactor → New codes and methods

• Modern simulation platform was conceived

• OpenFOAM chosen as basis for reactor dynamics solver (Vulashaka)
  – Steady-state (eigenvalue) and transient solver
  – Coupled multigroup neutron diffusion and coarse mesh thermal-fluids
PBMR

Multigroup Neutron Diffusion

- OpenFOAM is proven and tested for CFD, i.e. thermal-fluids
- Neutronics was never considered by the developers or community
  
  Can it be done?

- Multigroup neutron diffusion equation

\[
\frac{1}{v_g} \frac{d\phi^g}{dt} - \nabla \cdot D^g \nabla \phi^g + (\Sigma^a_a + \Sigma^g_s) \phi^g = S^g_s + \frac{1}{k} X^g_p S_p + X^g_d S_d
\]

- In OpenFOAM this is easily implemented for each equation

```plaintext
solve
(
    fvm::ddt(invV[g], phi[g])
    - fvm::laplacian(D[g], phi[g])
    + fvm::Sp(sigma_a[g]+sigma_s[g], phi[g])
    ==
    S_s[g]
    + (1/k)*chi_p[g]*S_p
    + chi_d[g]*S_d
);```

http://www.psi.ch/stars  PSI, 14-18 December 2014
• Simple segregated approach
  - Works for steady-state problems (simple power iteration)
  - Fails miserably in many cases for time-dependent problems (stiff equations)

  Back to the drawing board?

  Not quite!

• Block-coupled solver recently developed for OpenFOAM
  - Solves systems of linearly coupled PDEs implicitly

  $$\nu^{-1} \frac{d\Phi}{dt} - \nabla \cdot D \nabla \Phi + (\Sigma_a + \Sigma_s) \Phi = \cdots$$

  - Solution variables and coefficients become vectors and tensors, eg.

  $$\Phi = \begin{bmatrix} \phi_1 \\ \vdots \\ \phi_G \end{bmatrix}, \Sigma_a = \begin{bmatrix} \Sigma_a^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \Sigma_a^G \end{bmatrix}$$

  - Additional work needed since the equation mimicking cannot be readily applied to general coupled sets of equations

• Block-coupled approach works for both steady-state and transient simulations
PBMR400 Benchmark: Single Control Rod Ejection Transient

- Fast Flux ($t=3s$)
- Thermal Flux ($t=3s$)
PBMR400 Benchmark: Neutron flux profiles for steady-state operation

Fast  Thermal
Multi-scale solution of solid temperatures in prismatic HTG cores

- Reactor design has multiple distinct spatial scales

- Basic concept: Hierarchical multi-scale solution for temperature in solid materials
  - Borrows concepts from homogenization theory (unit cell calculations, discontinuity factors, etc.)
  - Subscale temperatures modelled using reduced order models
MHTGR-350MW Benchmark: Hierarchical Model Development
MHTGR-350MW Benchmark: Multi-scale solid temperature solution

Velocity

Homogeneous temperature

Reconstructed temperature (z=8.2m)
• STARS project at PSI
  - Development of analysis models for the Swiss reactors
  - Continually strive to improve modeling techniques
  - Continually expanding our validation database
  - Participate in several OECD international programs
• Hi fidelity multi-physics simulations are key to better understanding many phenomena
  - Fluence, activation and ageing
  - Recriticality following LOCA
• OpenFOAM use within STARS is focused on CFD analysis
• OpenFOAM is the current preferred CFD software for coupled simulations
  - Flexibility and availability of source code
  - More implicit coupling strategies can be investigated
TRACE/ANSYS-CFX

CFX: “Master” Program
- Data Exchange
- Data Manipulation
- Data Conversion

TRACE: “Slave” Program
- Data Exchange

Master

Data exchange

BCs
Comparison of commercial vs. open-source CFD and experimental data

- Snapshots of dimensionless temperature maps @ Downcomer sensors, \( t = 0.875 \) [-]. Width of the plume is largely over-estimated by the commercial code when standard water properties are used.

- OpenFOAM solution obtained with physical property correction and variable turbulent Prandtl model shows flow patterns and features that are more similar to the experiment.
Flow separation from hot leg

Stagnant zone below cold leg

KKG Downcomer (Preliminary)

Fournier et al. (2007)  Jeong & Han (2008)

Higher velocity: oscillations damped by turbulence

KKG Bypass (Preliminary)

Flow jets with vortex shedding

Rupp et al. (2008)
Development CFD-TRACE/S3K-COBRA-TF Coupling for KKG MSLB

**STEP 1**
TRACE PK for BC
CFD for Coolant Mixing Matrices (CMM)

**STEP 2**
TRACE/S3K + COBRA-TF
with CMM

**STEP 3**
TRACE/S3K + COBRA-TF
with TRACE/CFD Coupling

**STEP 4**
TRACE/S3K + COBRA-TF
with TRACE/CFD Coupling

**STEP 5**
Zoom Analysis for Hot-Assembly (HA)
with Higher-Order Transport/CFD Coupling

Coordination with Participation to NURESAFE
SN Projects

Development and Validation of CORE Models for the Swiss Reactors

Neutron Source \( S(\vec{r}, t) \) for the Swiss Reactors

Development and Verification of Solid CAD and CFD Models for the Swiss Reactors

RADAR – Fluence/Activation

MCNP
Neutron/Radiation Transport from core to vessel

CF-DYNAMO - Coupling MCNP/CFD
Temperatures/Densities of Coolant/Structures
Heat Sources to Coolant/Structures

CFD
Flow/Heat Transfer in Bypass and Downcomer

\( \varphi(\vec{r}, E, t) \) \(_{RPV}\)

\( T(\vec{r}, t) \) \(_{RPV}\)

PREVENT
RPV Embrittlement
Other nuclear-related development efforts (that I know about)

- Politecnico di Milano (Manuele Aufiero et al.)
  - Molten salt reactors
    - Transport of delayed neutron precursors
  - Coupled CFD / Monte Carlo simulations
- FAST group at PSI (Carlo Fiorina et al.)
  - Sodium-cooled fast reactors
  - Discrete ordinates neutron transport
- GRS (Joachim Herb)
  - Coupled CFD / system code (ATHLET) simulations for LWRs
- KTH Sweden (Klas Jarateg)
  - Multigroup neutron diffusion for LWRs
  - Discrete ordinates neutron transport for LWRs
- Other: Have a look at
  http://openfoamwiki.net/index.php/SIG_Nuclear_/_Publications
Summary

• OpenFOAM is a free software, available to all at no charge: GNU Public License

• Object-oriented approach facilitates model implementation

• Equation mimicking opens new grounds in Computational Continuum Mechanics
  - Far more than simply a set of CFD solvers and tools
  - A comprehensive multi-physics framework for solving sets of PDEs

• Extensive capabilities already implemented (mostly CFD-related); open design for easy customisation

• Nuclear-related developments are still at an early stage
  - Existing examples demonstrate that ground-breaking developments are possible
  - Development time for new solvers and physical models can be surprisingly short