A Framework for Predictive Capability Maturity Quantification

Paridhi Athe, Nam Dinh
North Carolina State University

July 7, 2015
A framework for Predictive Capability Maturity Quantification

Paridhi Atha, Department of Nuclear Engineering, North Carolina State University
Dr. Nam Dinh, Department of Nuclear Engineering, North Carolina State University

Introduction

Modeling and simulation permeate all fields of science and engineering. In nuclear engineering, numerical results generated by computer simulation codes have long been used in making decisions about nuclear reactor design, operation, and safety. In particular, comprehensive methodologies and systematic processes have been developed, adopted, and applied to guide the development of analytical tools and evaluate their adequacy for applications in nuclear reactor safety assessment. Notable are the “Code Scoping, Applicability, and Uncertainty” (CSAU) methodology, “Evaluation Model Development and Assessment Process” (EMDAP), both developed by the U.S. Nuclear Regulatory Commission (NRC) and “Predictive Capability Maturity Model” (PCMM) developed by Sandia National Laboratories. However, due to lack of data/evidence and the inherent risk associated with various nuclear engineering processes, the assessment process becomes very rigorous and time consuming, and when it comes to decision making it’s hard to answer in “Yes” or “No” if a certain code is “good enough” to describe a particular application of interest. This work describes an initial effort to develop and demonstrate a systematic, formalized and computerized framework for quantification of reliability of a simulation tool for (a given) nuclear reactor engineering or safety application. It is now noted that the present study is limited to “validation”, which is an important component in PMM also, with a broader scope considered in PCMM.

Fuzzy logic and code maturity quantification

A fuzzy set can be defined as a set with fuzzy boundaries. Fuzzy logic is theory of sets that calibrate obscurity or uncertainty. Unlike Boolean algebra that works on binary logic (True or False, 0 or 1), fuzzy logic characterizes the data using degree of membership. It provides a precise way of representing approximations of natural language terms. A fuzzy set, e.g., fuzzy set “X” of universe of discourse Y, with elements represented by x, is defined by its membership function (MF), $\mu_x(x) = \frac{ax + b}{c}$, where, $\mu_0(x) = 0$, if x does not belong to A, and $0 \leq \mu_x(x) \leq 1$, if x partially belongs A.

Since its conception, fuzzy logic has find wide application in different areas of engineering and science. There are several feature of fuzzy logic that makes it suitable choice for developing a quantifiable framework for assessing the reliability of simulation tools used in nuclear engineering.

- Fuzzy logic provides an efficient methodology for representing incomplete knowledge and imprecise information using degree of belief or degree of confidence in the truth value of the information.
- It can handle heterogeneity very well. Extraction of important information from large amount of data is easier.
- Availability of large number of mathematical operation (min, max, product, average, sum, etc.) makes it a very efficient tool for developing an expert inference system.
- Simplicity and flexibility of fuzzy logic makes it possible to combine fuzzy logic with other methodologies. e.g. Genetic fuzzy systems are fuzzy system developed by combining fuzzy logic and genetic algorithms. Similarly, neuro-fuzzy system are developed by combining fuzzy logic with neural networks.

CSAU, EMDAP and PCMM are all expert elicitation methodologies. Expert judgement plays a critical rule at each level/stage of these methodologies. Fuzzy logic provide a unique technique to codify the expert knowledge which makes it a very attractive tool for developing a quantifiable framework for code’s maturity evaluation.

Governing factors and Membership function

Validation Pyramid: In lieu of full-scale plant testing, a historical application of the capability for code validation. A set of relevant separate-effect tests (SET), mixed-effect tests (MET) and integral-effect tests (IET) is used to construct a validation pyramid. We assume that each layer consist of a finite number of independent experiments, that together describe the knowledge base of that layer in the validation hierarchy.

Coverage: The range of applicability or coverage of a particular application is decided on the basis of operating range of its primary control parameters (like pressure, temperature, and mass flow rate and temperature). We assume that our application of interest is governed by a single control parameter P. Coverage ratio is defined by: $x_c = x_{LP} - x_{UP}$, where, $x_{LP}$ and $x_{UP}$ represents the lower and upper limit of P for the experiment. $x_{LP}$ and $x_{UP}$ represents the lower and upper limit of P for the application. Therefore, $x_c$ lies between 0 and 1, when experiment does not lies in the range of experiment and 1 when experiment covers the entire range of application.

Measurement Uncertainty: Code validation is based on the comparison of code prediction and measurements. However, due to instrumentation errors and limited resolution of measurement technique, there is a finite measurement uncertainty associated with each experiment. Experiments with higher measurement uncertainty cannot be used for code validation analysis. Confidence in the measurements is expressed by a variable $\mu_x$, a simplified definition of $\mu_x = 1 - MU$, here, $MU$ is the measurement uncertainty which is assumed to lie between 0-100%. Hence, the variable $x_c$ lies between 0 and 1.

Bias Uncertainty: Bias uncertainty can be defined as the statistical measure of the difference between code prediction and measurement. If measurement uncertainty is small, bias uncertainty is an adequate measure of the code deficiency. Confidence in the code prediction is expressed by a variable $\mu_x$ defined by $x_b = 1 - BU$ here, $BU$ is the bias uncertainty that is assumed to lie between 0-100%. Hence, the variable $x_b$ lies between 0 and 1.

Scaling: Experiments used for validating the code are scaled version of the real application. The relevance of these experiment to the real application is determined on the basis of different scaling factors like material scaling, geometric similarity and physics scaling. We assume that all experiments used in the validation pyramid are of reactor prototype level. Therefore, we do not consider effect of scaling in this analysis.

Evaluation of assessment base

This step assess the completeness of a particular layer in the validation pyramid on the basis of code bias understanding and measurement uncertainty of all the experiments in that layer. The assessment result gives a number y that represents the evaluation of the respective layer on a scale of 0 to 1 using all the evidences that constitute the knowledge base of that layer. Steps of evaluation:

- Step 1 (Specification of input): Crisp inputs, $x_c$, $x_b$ and $x_M$ for each validation test are specified.
- Step 2 (Fuzzification): Fuzzification helps in weighting the evidence on the basis of expert knowledge. The fuzzified input variable corresponding to $x_c$, $x_b$ and $x_M$ are represented by $\mu_c(\mu_c)$, $\mu_b(\mu_b)$ and $\mu_M(\mu_M)$ respectively (where i is the experiment index).
- Step 3 (Rule): $\mu_i = \mu_c(\mu_c) \cdot \mu_b(\mu_b) \cdot \mu_M(\mu_M)$
- Step 4 (Normalization): $\mu_i = \frac{\mu_i}{\sum \mu_i}$
- Step 5 (Defuzzification): Defuzzification is performed according to the following:
  $d_i = \frac{\mu_i}{\sum \mu_i} \cdot (x_c + x_b + x_M) \cdot \frac{1}{3}$

In this section, we construct a maturity function that provides a graphical representation of the code’s predictive capability over the entire range of operation of the application.

Here, we consider three additional MF function in addition to the MF’s described in the previous sections: (1) MF for number of experiments, $\mu_N$: This MF is defined according to the number of experiments in its constitutive layer, e.g. if the bottom layer consists of M experiments, then $\mu_N$ is defined as $\frac{1}{M}$ for M experiments belonging to this layer; (2) Based on the subjective knowledge about a particular type of experiment (viz. SET or MET or IET), we assign weight to each type of experiments using expert’s knowledge about the importance of each type of experiment. Assumed membership values for each type of experiment are: $\mu_{SET} = 0.5$ for IETs, $\mu_{MET} = 0.3$ for METs and $\mu_{SET} = 0.2$ for SETs.

The modified MF ($\mu_{final}$) of all the experiments are added to obtain a global maturity function for the application of interest.

Summary Remarks

This work presents a quantitative framework for assessing the maturity of a simulation code. In essence, it provides an efficient methodology that assimilates objective data (evidence) with subjective data (based on expert knowledge) for a codes maturity quantification. This method integrates expert’s knowledge in the framework by using fuzzy membership function. These membership functions have advantage that the evidence based a degree of confidence in their truth value. These evidences are combined using different fuzzy operations to obtain the total maturity function for the code for a specific application of interest.

Acknowledgements

This work is performed for the U.S. Department of Energy’s Consortium for Advanced Simulation of Light Water Reactors (CASL) to support application of the Predictive Capability Maturity Model to the CASL’s Virtual Environment for Reactor Applications (VERA) codes.

Assessment of entire knowledge base

The modified Maturity function ($\mu_{final}$) represents the degree of confidence in the code’s predictive capability over the entire operating range of the application.