RAVEN: a GUI and an Artificial Intelligence Engine in a Dynamic PRA Framework

C. Rabiti, D. Mandelli, A. Alfonsi, J. Cogliati, R. Kinoshita, D. Gaston, R. Martineau, C. Smith

Idaho National Laboratory, 2525 North Fremont Street, Idaho Falls (ID)
{CristianRabiti, Diego.Mandelli, Andrea.Alfonsi, Joshua.Cogliati, Robert.Kinoshita}@inl.gov
{Derek Gaston, Richard.Martineau, Curtis.Smith}@inl.gov

INTRODUCTION

Increases in computational power and pressure for more accurate simulations and estimations of accident scenario consequences are driving the need for Dynamic Probabilistic Risk Assessment (PRA) [1] of very complex models. While more sophisticated algorithms and computational power address the back end of this challenge, the front end is still handled by engineers that need to extract meaningful information from the large amount of data and build these complex models. Compounding this problem is the difficulty in knowledge transfer and retention, and the increasing speed of software development.

The above-described issues would have negatively impacted deployment of the new high fidelity plant simulator RELAP-7 (Reactor Excursion and Leak Analysis Program) at Idaho National Laboratory. Therefore, RAVEN that was initially focused to be the plant controller for RELAP-7 will help mitigate future RELAP-7 software engineering risks.

In order to accomplish such a task Reactor Analysis and Virtual Control Environment (RAVEN) has been designed to provide an easy to use Graphical User Interface (GUI) for building plant models and to leverage artificial intelligence algorithms to reduce computational time, improve results, and help the user to identify the behavioral pattern of the Nuclear Power Plants NPPs.

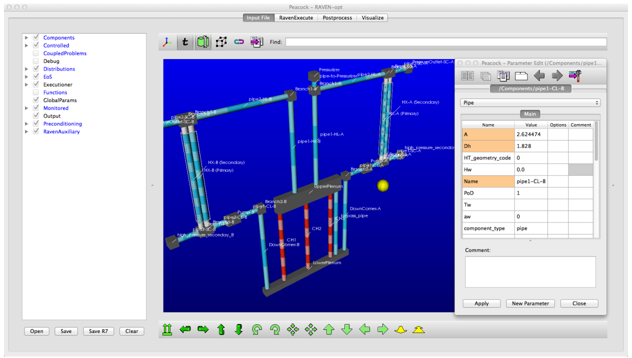
In this paper we will present the GUI implementation and its current capability status. We will also introduce the support vector machine algorithms and show our evaluation of their potentiality in increasing the accuracy and reducing the computational costs of PRA analysis. In this evaluation we will refer to preliminary studies performed under the Risk Informed Safety Margins Characterization (RISMC) project of the Light Water Reactors Sustainability (LWRS) campaign [3]. RISMC simulation needs and algorithm testing are currently used as a guidance to prioritize RAVEN developments relevant to PRA.

GUI OVERVIEW

The RAVEN GUI for RELAP-7 is based on Peacock, a general GUI for Multiphysics Object Oriented Simulation Environment (MOOSE) [4] applications. MOOSE is a scientific simulation library on which RELAP-7 and RAVEN are built. Every component of the plant, in software terms, is a C++ class (object) registered with a factory owned by MOOSE. As part of this registration process, MOOSE knows all the needed parameters to generate the numerical model of the plant components. RAVEN also uses this approach for any other C++ classes describing control logic action.

One of the many capabilities of the MOOSE factory is to dump the parameters needed to define the instance of one object in a structured format file in Yet Another Markup Language (YAML) format. This standard format of the information needed to construct all the components and initialize all functions needed to run a RAVEN RELAP-7 simulation allows Peacock to use a generic interface and mold itself to the given application.

Through Application Programming Interfaces (API)s, Peacock can be specialized for both RELAP-7 and RAVEN applications so that it is able to build a visual model of the components immediately after the user generates them. The user can in fact see the plant design while he is building it.

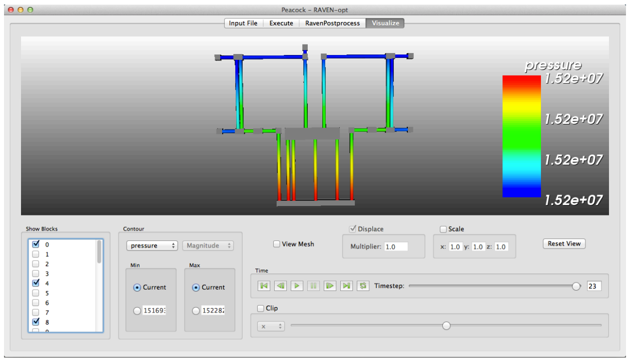


Name	Value	Options	Comment
A	2.624474		
Dh	1.828		
HT_geometry_code	0		
Hw	0.0		
Name	pipe1-CL-B		
PoD	1		
Tw			
aw	0		
component_type	pipe		

Figure 1: Input/Plant Layout visualization tab.

Figure 1 shows the input tab of Peacock-RAVEN. The list of the possible objects that could be added to the simulation input is on the left, while on the right we have one of the sub-tabs allowing construction of a specific component. Also in Figure 1 in background we have a 3D layout of the plant being built. On the top and the bottom of the 3D layout there are several functions that allow searching for specific components (which are highlighted when found) and manipulating the view. To ease inspection of the plant layout the 3D visualization is active and visually searchable. Clicking one of the components will open a corresponding property tab.

Figure 2 shows the projection of the solution field for the pressure into the plant layout during a transient analysis. In fact, Peacock can read the file produced by the MOOSE platform at each time step and thus displays the simulation while running.



Multiplier: 1.0

x: 1.0 y: 1.0 z: 1.0

Timestep: 23

Min: 151693, Max: 152282

Figure 2: Pressure solution during transient analysis.

ARTIFICIAL INTELLIGENCE ALGORITHMS

Construction of Limit Surfaces Using Support Vector Machines to Investigate Failure Probability

Correct estimation of the failure probability in a complex system such as a nuclear power plant is an almost overwhelming challenge. The very low likelihood of a failure scenario adds to the difficulty of building sampling strategies that produce trustable statistics of these events.

To fix the ideas we can consider:

- An input space
  $\Omega = \left\{ \vec{p} = \left(p_1,...,p_i \right) \in \Omega \right\} \quad i = 1,....,Dim\left[\Omega\right]$
  (All the parameters used to define the RELAP-7 and RAVEN simulation)
- A feature space
  $\vec{f} \in O$
  (The set of variable of our output that we decide to be relevant, e.g. the success, failure of the plant)
- A kernel function
  $\vec{\varphi}\left(\vec{r},t,\vec{p}\right) = \vec{f} \in O$
  (The numerical model of the NPP solved by RELAP-7 and RAVEN projected to the feature space O)

If, as in most of the cases, the inputs parameters are subject to probabilistic behavior, the values of $\vec{f}$ are also characterized by a probabilistic distribution.

Usually in reliability analysis the feature space is just a Boolean variable (Success/Failure).

The problem of determining the probability of failure P_f is then equivalent to finding the probability to be in the input space leading to a failure:

$$P_f = Pb\left(\vec{p} \in \Omega'\right)$$

$$\Omega' = \left\{ \vec{p} \mid \varphi\left(\vec{r},t,\vec{p}\right) = Failure \right\}$$

$$P_f = \int_{\Omega'} d\vec{p}\,' Pb\left(\vec{p}\,'\right)$$

Clearly, the challenge is in the definition of the input subspace Ω' since the probability distribution of the input parameters $Pb\left(\vec{p}\right)$ is a known function (at least so it is assumed).

The deterministic solution of the problem would require the direct solution of $\varphi\left(\vec{r},t,\vec{p}\right) = Failure$. This is in most of the time impractical and ill-posed; therefore, Monte Carlo approaches are used to evaluate the integral P_f. Unfortunately, when P_f is very low and the input space has a large dimensionality the number of forward simulation $\varphi\left(\vec{r},t,\vec{p}\right) = \vec{f}$ needed to achieve a reasonable confidence in the estimation of P_f is overwhelming, even using large computational clusters.

Exactly in this situation Support Vector Machines come in handy [4].

Without going into the details, also because several different algorithms are available, the process is the following:

- A training set of sampling of the function φ is generated
- The training set is than used to define hyperplanes in the input parameter space that separate, according to a given criteria, the feature space (in our case success/failure)
- Sampling points, estimated laying on such hyperplanes, are than chosen for sub-sequential evaluation of $\varphi\left(\vec{r},t,\vec{p}\right) = \vec{f}$
- The outcome of these newly generated sampling points is then used to refine the definition of the hyperplanes
- The process is repeated until a proper convergence criteria on the hyperplane definition is achieved
- After convergence, the algorithm locating an input in the Ω space with respect the hyperplane is now classifier

The classifier is essentially a set of equations that will determine where an input point lays with respect to the hyperplanes (if it will leads to Success or Failure).

The evaluation of the classifier is very fast; therefore, more convenient than to evaluate the kernel function $\vec{\varphi}$.

At this point, the evaluation of the integral P_f is feasible using the classifier rather than the evaluation of the kernel, and given its speed, the number of sample could be so high that is easy to achieve a satisfactory statistic.

The drawback of this approach is that hyper-surfaces surrounding small hyper-volumes (e.g. a small region leading to failure in a large region leading to success) could be missed in the initial training set; therefore, the classifier will never become aware of these.

Several strategies are available for the generation of the initial training set that range from standard Monte Carlo (MC), Latin Hypercube (LH), and Centroidal Voronoi Tessellation (CVT) or search patterns gradient based and many more.

As a part of a synergy effort the investigation of these algorithms has been started last year by the RISMC project where Matlab® has been used to build pilot tests. Figure 1 shows the limit surface build in one of these tests after the initial evaluation of the training set, while figure 2 shows the final form of the limit surface when the classifier has converged to the final hyper-surface.

RAVEN already has a Monte Carlo sampling capability, and it is currently used to sample the RELAP-7 input space. We are in the process of implementing a limit surface approach based on the present Monte Carlo to generate the training set and the SVM algorithm for generation of the classifier.

In the future the introduction of the more general structured SVMs [5] could be also tested to generate surrogate models for plant emulator purpose.

We are also investigating the possibility of extending this approach to generate limit surfaces classifiers starting from adaptive branching tree.

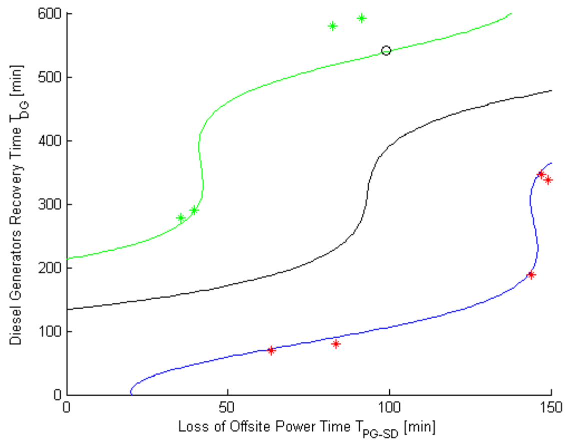


Figure 3: Initial limit surface after a very small training set (failure red, success green) [4].

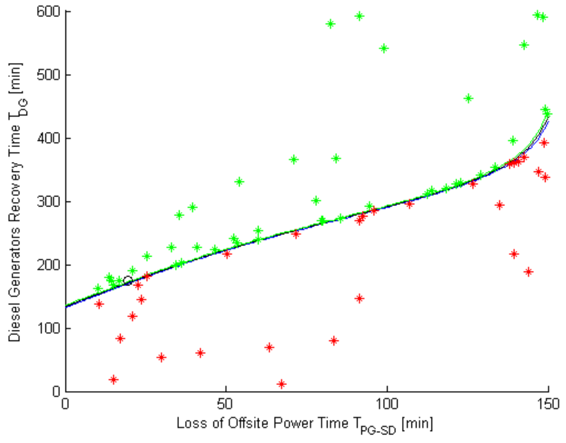


Figure 4: Final limit surface [4].

CONCLUSION

We have presented the advancements in the design and implementation of the RAVEN-Peacock GUI meant to ease the analyst work necessary to generate complex plant models and also to enhance the understanding of plant behavior in accident scenarios. This is realized via a graphical tool that both assists the user generating the plant layout and allows performing an online monitoring of the simulation results.

Another step forward in the development of the RAVEN project has been presented here. The preliminary analysis of SVM has revealed a great potentiality in reducing of computational time and increasing the accuracy of failure probability estimation.

The implementation of this technique is currently ongoing, and we are planning to illustrate preliminary results during the ANS meeting.

ENDNOTES

This work is supported by the U.S. Department of Energy, under DOE Idaho Operations Office Contract DE-AC07-05ID14517. Accordingly, the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.

REFERENCES

1. N. SIU, “Risk assessment for dynamic systems: an overview,” *Reliability Engineering and System Safety*, 43, 1, 43–73 (1994).
2. C. RABITI, A. ALFONSI, D. MANDELLI, J. COGLIATI, and R. MARTINEAU, “RAVEN as Control Logic and Probabilistic Risk Assessment Driver for

RELAP-7," in Proceeding of American Nuclear Society (ANS), San Diego (CA)," (2012), vol. 107, pp. 333–335.
3. IDAHO NATIONAL LABORATORY, "Light Water Reactor Sustainability Research and Development Program Plan (Fiscal Years 2009-2013)", INL/MIS-08-14918, Rev. 1 (2009).
4. D. Gaston, C. Newman, G. Hansen, D. Lebrun-Grandié, "MOOSE: A parallel computational framework for coupled systems of nonlinear equations", *Nuclear Engineering and Design*, Vol. 239, Issue 10, pp. 1768-1778 (2009).
5. D. MANDELLI and C. SMITH, "Adaptive Sampling Using Support Vector Machines," in "Proceeding of American Nuclear Society (ANS), San Diego (CA)," (2012), vol. 107, pp. 736–738.