Light Water Reactor Sustainability Program

Industry Use Cases for Risk-Informed System Health and Asset Management

September 2021

U.S. Department of Energy

Office of Nuclear Energy

Industry Use Cases for Risk-Informed System Health and Asset Management

D. Mandelli, C. Wang, M. Abdo, K. Vedros, J. Cogliati, J. Farber, A. Al Rashdan, S. Lawrence (INL)

D. Morton (Northwestern University)

I. Popova (Texas State University)

S. Hess (Jensen Hughes)

C. Pope, J. Miller, S. Ercanbrack (Idaho State University)

September 2021

SUMMARY

Industry equipment reliability and asset management programs are essential elements that help ensure the safe and economical operation of nuclear power plants. The effectiveness of these programs is addressed in several industry-developed and regulatory programs.

The Risk-Informed Asset Management (RIAM) project is tasked to develop tools in support of the equipment reliability and asset management programs at nuclear power plants. These tools are designed to create a direct bridge between component health/lifecycle data and decision making (e.g., maintenance scheduling and project prioritization).

The goal of this report is to provide a guide for specific use cases that the RIAM project is targeting. We have grouped uses cases into three main areas. The first area focuses on the analysis of equipment reliability data with a particular emphasis on condition-based data, such as test/surveillance reports and component monitoring data. The second area focuses on the integration of equipment reliability into system/plant reliability models to determine system/plant health and identify the components that are critical to maintain an operational system. Lastly, the third area manages plant resources, such as maintenance activities and replacement scheduling using optimization methods.

Here the primary focus is on supporting typical system engineer decisions regarding maintenance activity scheduling and component aging management. This is performed in a risk-informed context where the term "risk" is broadly constructed to include both plant reliability and economics. This framework combines data analytics tools to analyze equipment reliability data with risk-informed methods designed to support system engineer decisions (e.g., maintenance and replacement schedules, optimal maintenance posture) in a customizable workflow.

CONTENTS

FIGURES

TABLES

ACRONYMS

AAKR   Auto-Associative Kernel Regression

ANI   American Nuclear Insurers

ASME   American Society for Mechanical Engineers

BE   Basic Event

BOL   Beginning of Life

BWR   Boiling Water Reactor

CAP   Corrective Action Program

CBM   Condition-Based Maintenance

CCW   Component Cooling Water

CDF   Core Damage Frequency

CDF   (Probabilistic) Cumulative Distribution Function

CPM   Critical Path Method

DA   Data Analytics

DAG   Directed Acyclic Graph

DOE   (United States) Department of Energy

ECCS   Emergency Core Cooling System

EFT   Early Finish Time

EPRI   Electric Power Research Institute

ER   Equipment Reliability

EST   Early Start Time

ET   Event Tree

FM   Failure Mode

FMEA   Failure Modes and Effects Analysis

FSAR   Final Safety Analysis Report

FT   Fault Tree

GA   Genetic Algorithm

GL   Generic Letter

GRA   Generation Risk Assessment

IAEA   International Atomic Energy Agency

IDP   Integrated Decision making Panel

INL   Idaho National Laboratory

INPO   Institute of Nuclear Power Operations

IR	Issue Report
LERF	Large Early Release Frequency
LFT	Late Finish Time
LST	Late Start Time
LWR	Light Water Reactor
LWRS	Light Water Reactor Sustainability
M&D	Monitoring and Diagnostic
MA	Maintenance Activity
MBSE	Model Based System Engineering
MCMC	Markov Chain Monte Carlo
MCS	Minimal Cut Set
ML	Machine Learning
MTS	Most Total Successors
NDA	Non Disclosure Agreement
NEI	Nuclear Energy Institute
NEIL	Nuclear Electric Insurance Limited
NLP	Natural Language Processing
NPP	Nuclear Power Plant
NPSH	Net Positive Suction Head
NPV	Net Present Value
NRC	Nuclear Regulatory Commission
NSGA	Nondominated Sorting Genetic Algorithm
OPEX	OPerating EXperience
OPM	Object Process Methodology
OPL	Object Process Language
O&M	Operation and Maintenance
PDF	Probability Density Function
PHM	Plant Health Management
POS	Part Of Speech
PRA	Probabilistic Risk Assessment
PWR	Pressurized Water Reactor
RAVEN	Risk Analysis Virtual Environment
RCP	Reactor Coolant Pump
RCPSP	Resource-Constrained Project Scheduling Problem
RD&D	Research, Development, and Demonstration

RG	Regulatory Guide
RHR	Residual Heat Removal
RI	Risk-Informed
RIAM	Risk-Informed Asset Management
RISA	Risk-Informed Systems Analysis
RMTS	Risk-Managed Technical Specifications
RUL	Remaining Useful Life
R&D	Research and Development
SAX	Symbolic Aggregate Approximation
SFCP	Surveillance Frequency Control Program
SysML	Systems Modeling Language
SR^{2}ML	Safety, Risk, Reliability Model Library
SSCs	Structures, Systems, and Components
STI	Surveillance Test Intervals
TE	Top Event
TEAL	Tool for Economic AnaLysis
TS	Technical Specifications
TSKR	Time Series Knowledge Representation
UML	Unified Modeling Language
UQ	Uncertainty Quantification
VERT	Versatile Economic Risk Tool
WO	Work Order
XML	Extensible Markup Language

INDUSTRY USE CASES FOR
RISK-INFORMED SYSTEM HEALTH AND ASSET MANAGEMENT

1. INTRODUCTION

The Risk-Informed Systems Analysis (RISA) Pathway[1] [1] of the United States Department of Energy (DOE) Light Water Reactor Sustainability (LWRS)[2] [2] program is conducting collaborative research that applies risk-informed technology to assist operating nuclear power plants (NPPs) to reduce costs and support their adaptation to the changing economic and power generation environment. The research is being performed within the framework of specific use cases, which are intended to enable rapid technology development, deployment, and dissemination throughout the operating U.S. NPP fleet to address issues of pressing economic, operational, or safety significance.

One area of research in the RISA pathway is focusing on the development of methods and tools being designed to optimize plant operations (e.g., maintenance/replacement schedules, optimal maintenance postures for plant structures, systems, and components [SSCs]) in a manner that is more cost effective than current approaches and makes better use of available SSC health and cost data. This is accomplished under the Risk-Informed Asset Management (RIAM) project by creating a direct bridge (see Figure 1) between component equipment reliability (ER) data and decision making (e.g., maintenance scheduling and project prioritization). Here we are supporting typical system engineer decisions regarding maintenance activity scheduling and component aging management.
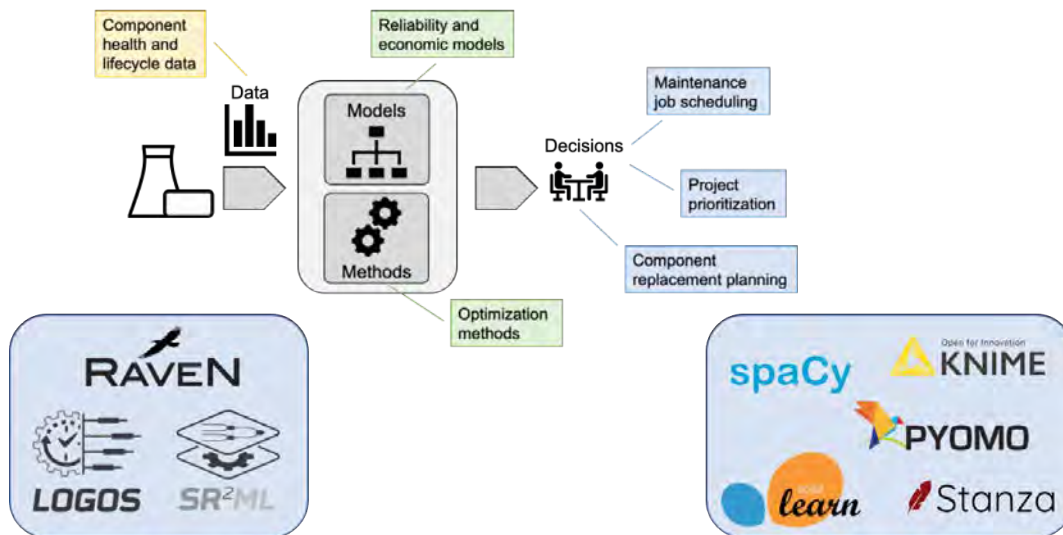
Figure 1. Graphical representation of the RIAM project and its tools: INL developed (bottom left) and open-source libraries (bottom right).

Section 2 provides a broad overview of the specific use cases on which RIAM project is focusing. Here, we also provide information on the RIAM-developed tools that support these use cases and outline references to applications of such tools. Section 3 provides a brief overview of the toolkit that has been developed to tackle the use cases listed in Section 2. Reference [3] provides more details about the structure of the toolkit and how elements of this toolkit can be combined to create customizable workflows.

The overall workflow that can be constructed using the RIAM toolkit is covered in three main tasks. These tasks can be assembled to directly propagate ER data through the decision making process as indicated in Section 4. The first task focuses on the analysis of equipment reliability data with a particular emphasis on condition-based data, such as test/surveillance reports and component monitoring data (see Section 5). The second task focuses on the integration of equipment reliability and simulated data into system/plant reliability models to determine system/plant health and identify the components that are critical to maintain an operational system (see Section 6). And the third task manages plant resources such as maintenance activities (MAs) and replacement scheduling using optimization methods (see Section 7).

2. USE CASES OVERVIEW

A more detailed overview of the RIAM project capabilities is shown in Figure 2 where the main use cases have been grouped into three main area:

1. **ER data analytics.** Targeting the analysis of ER data to adequately measure component health. This area includes all the methods designed to analyze numeric and text ER data (i.e., monitoring data, maintenance activities, work orders and maintenance/issue reports [IRs]), employ historic data to detect abnormal behavior (anomaly detection) and the cause of such abnormal behavior (diagnostic), and predict SSC future performances (prognostic).

Goals: O&M reduction costs
- Reliability/ageing management
- Enhance system performance
- Optimize plant resources

ER Data Analytics
- Anomaly detection
- Diagnostics
- Prognostics

Risk Analytics Platform

Resource Optimization
- Reliability/ageing management
- Enhance system performance
- Optimize plant resources

Digital Modeling
- Integration of reliability and economic models
- Margin reliability solver

Figure 2. Graphical overview of the RIAM project research, development, and demonstration (RD&D) directions.

Table 1. List of use cases per the three RIAM projects RD&D directions.

RD&D Area	Use Case	Models and Methods	References
ER data analytics	Analyze IRs and WOs and extract information	SR^{2}ML NLP methods	Section 5.5
	Retrieve and summarize component performance and reliability history	SR^{2}ML ER data analysis methods	Section 5
	Detect component anomalous behaviors	SR^{2}ML ER data analysis methods	Section 5.4
	Measure component health	SR^{2}ML SSC margin model	Section 6.1
Digital modeling	Measure system/plant health	SR^{2}ML margin-based reliability solver	Section 6.1, [6]
	Monitor risk of loss of power generation	VERT GRA models	Section 6.2, [5], [6]
	Identify most critical SSC	SR^{2}ML margin-based reliability post-processor	Section 6.2, [6]
Resource optimization	Select set of projects that provide highest value	RAVEN multi-objective optimization methods	Section 7.2, [7]
	Prioritize and schedule projects	LOGOS knapsack base models	Section 7.1, [4], [7]
	Preventive maintenance posture optimization	RAVEN single- or multi-objective optimization methods	[7]
	Maintenance job scheduling	LOGOS task optimization models	Section 7.4

2. **Digital modeling.** Designed to model from a risk perspective the considered system/plant and, more importantly, fully integrate ER data into such models. The main feature of these risk models is that they encompass not only reliability/availability of the system/plant but also economic aspects. The real challenge is to inform these models directly with the available ER data (historic and current).

3. **Resource optimization.** Targeting the optimization of plant resources. Here, plant resources include multiple entities, such as plant operation and maintenance (O&M) and capital budgets, workforce tasks, and SSC lifecycle. This area is directly linked to the plant decision making process and is considers both short- and long-term decisions.

Table 1 summarizes the set of specific use cases for each of these three areas that have been considered by the RIAM project. For each use case, we indicated the set of tools that have been developed and the references that provide the direct application of the developed tools. For a few use cases, we have listed previous reports that can provide details of the developed methods, while in this report we provide a short summary for completeness.

3.   RIAM TOOLKIT

In July 2021, the first step toward the development of this framework was taken when two software packages, which are an integral part of the enterprise risk analysis framework, were released with an open-source license: LOGOS and SR^{2}ML [3]. SR^{2}ML is a software package that contains a set of reliability models designed to integrate equipment reliability data (e.g., aging, testing, maintenance) and perform system-level reliability calculations [6]. LOGOS is a software package that contains a set of discrete optimization algorithms that can be employed to effectively manage plant assets and optimize schedules for plant operations [7]. These software packages are plugins that can be interfaced with the Idaho National Laboratory (INL)-developed RAVEN code to propagate data uncertainties in model variables, perform data analysis, and perform model optimization (e.g., via genetic algorithm heuristics). In this respect, SR^{2}ML is designed to propagate equipment reliability knowledge to the system/plant level to identify the components that are most critical to system/plant health. LOGOS uses this information to prioritize and schedule plant operations (e.g., maintenance, testing, replacement) based on budget, reliability, and resource constraints. Lastly, VERT is a new tool designed to contain plant, system and component reliability models.

This risk analytics platform consists of the following tools designed for specific use cases:

- LOGOS: Designed for plant resource optimization (e.g., project prioritization and project scheduling) (https://github.com/idaholab/LOGOS)

- SR^{2}ML: Component and system reliability modeling and analysis of equipment reliability data (https://github.com/idaholab/SR2ML)

- RAVEN: Data analysis computational framework (e.g., model optimization and uncertainty propagation) (https://github.com/idaholab/raven)

- VERT: Plant generation risk assessment modeling (https://hpcgitlab.inl.gov/mandd/vert)

- TEAL: Plant economic analysis (https://github.com/idaholab/TEAL).
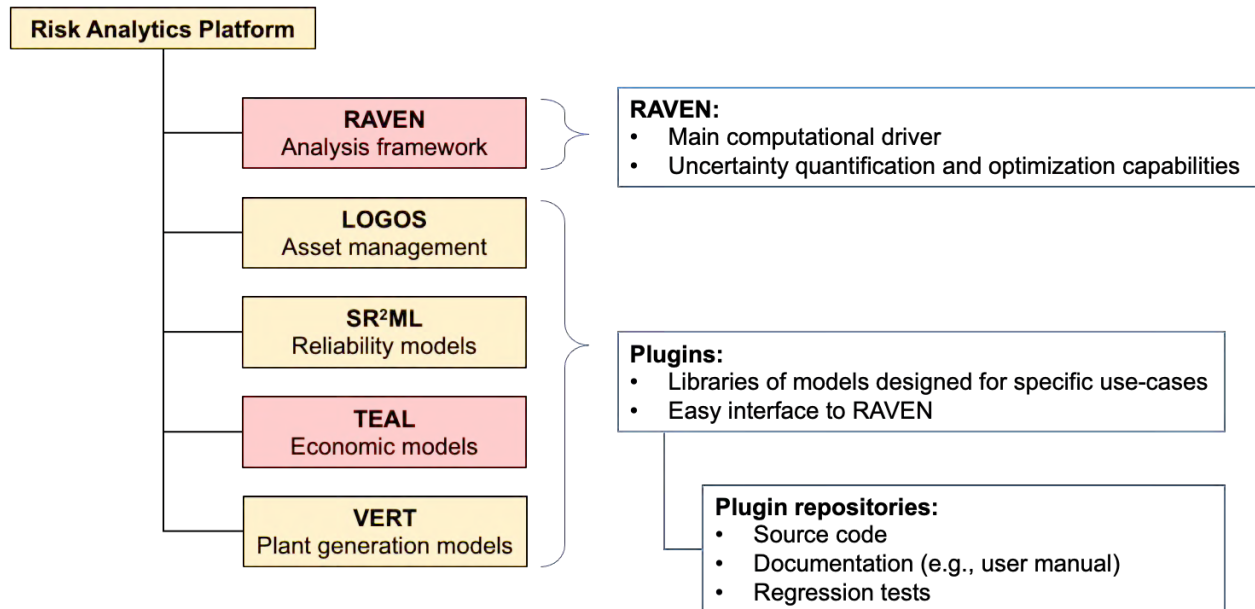
Figure 3. RIAM risk analytics toolkit: elements in yellow were developed within the RIAM project while the development of the elements in red was shared among other DOE programs.

4. THE RISK ANALYTICS PLATFORM WORKFLOW

The scope of this section is to provide a more detailed guide on how the methods and tools developed under the RIAM project (and described throughout this report and in [4, 5, 6, 7]) can be used to bridge ER data with decisions. We refer here to Figure 4 which graphically describes a complete risk analytics platform data workflow.

The starting point is the set of ER data generated by the SSC monitoring system (bottom left of Figure 4), available for example from the plant monitoring and diagnostic (M&D) center. The steps of this workflows are as follows:

1. Analysis of ER data using methods presented in Section 5. The goal is to 1) track SSC health (i.e., performance/degradation), and 2) identify possible anomalies in the SSC behavior (see Section 5.4 and bottom-left portion of Figure 4).

2. Measure SSC health by determining the margin of specific SSC failure modes given current SSC conditions and historic data (see Section 6 and mid-left portion of Figure 4).

3. Once margin values are determined for the failure modes of the chosen SSC, they are propagated through classical reliability models (e.g., fault trees) to determine: 1) the margin at the system/plant level (i.e., plant system/health), and 2) the risk importance measure for each SSC failure mode (see Section 6 and top-left portion of Figure 4).
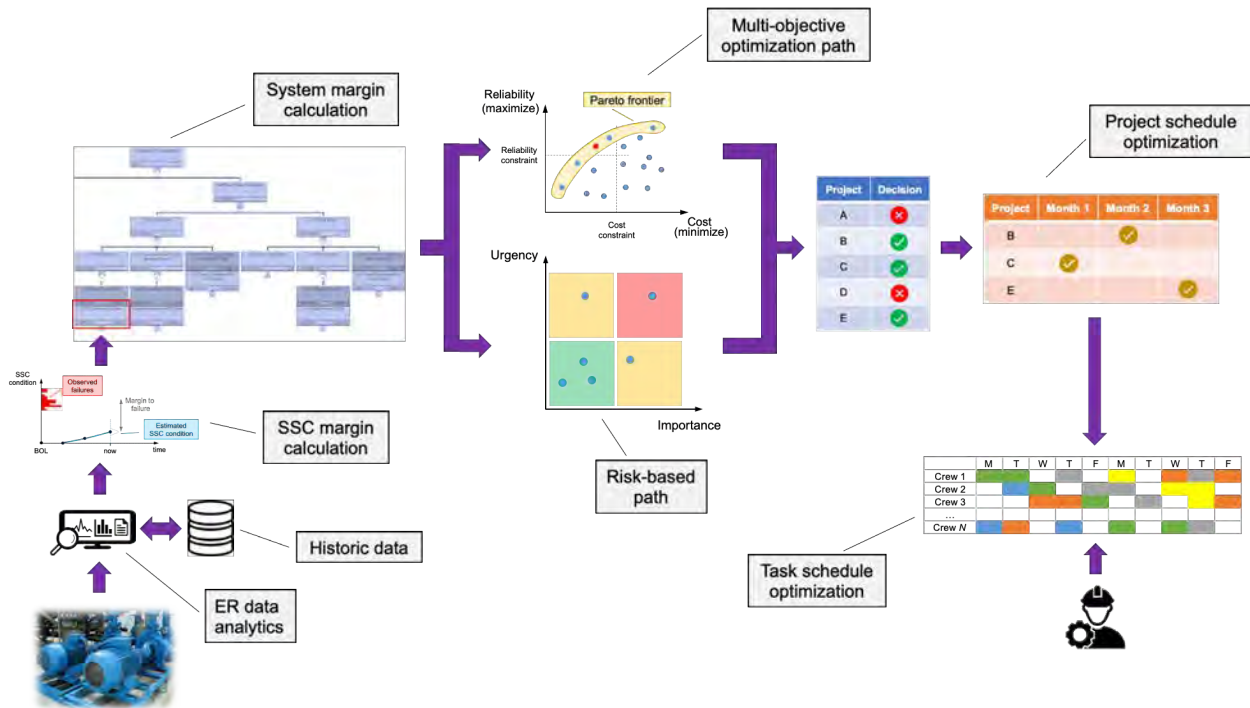
Figure 4. RIAM toolkit as bridge between ER data to decisions: graphical representation of the workflow starting with SSC monitoring data (bottom left) to reliability modeling (top left) to project prioritization (center) and task scheduling (right).

4. The next step is to choose the optimal set of projects (e.g., maintenances activities) given plant system/health information determined in Step 3. Two possible paths can be followed (see center portion of Figure 4):

a. Ranking-based path: the failure modes with highest consequences and importance measures are selected. Economic constraints might be used to filter the chosen project list.

b. Multi-objective optimization path: projects are chosen based on both reliability and economic factors simultaneously (e.g., through a Pareto frontier analysis as indicated in Section 7.1)

5. The next step is to set the optimal schedule for the projects chosen in Step 4 (see top-right portion of Figure 4). Using the methods presented in Section 7.2 and [4,7] it is now possible to set the optimal project actuation schedule based on reliability and economic constraints (i.e., medium- and long-term decisions).

6. Each project is partitioned into tasks and the optimal schedule of each task (i.e., short-term decisions) is determined (see bottom-right portion of Figure 4) using the methods presented in Section 7.3. Here, tasks that might be provided by plant system engineers are added to the list of tasks for the projects chosen in Step 5.

5. ER DATA ANALYTICS

This section addresses the first area of use cases listed in Table 1. Here, we present a set of methods designed to analyze ER data that in both numeric (i.e., monitoring data) and textual (i.e., maintenance/incident reports). The goal is to employ current and historic data to:

- Detect possible abnormal behaviors (i.e., anomaly detection)

- Identify the cause of such abnormal behavior (i.e., diagnostic)

- Predict SSC future performances (i.e., prognostic).

The RIAM project has focused on the development of innovative methods designed to: extract knowledge from IRs and infer more likely failure modes, integrate numeric and text data in a common data structure, and capture sequencing of events from component history that might provide capability to detect occurrence of a component anomaly.

In the next sections, we show how these tasks have been tackled using a blend of system models coupled with natural language processing (NLP) [13, 14] and causal inference [78, 79] methods. We then propose a symbolic data structure to summarize the operational history of an SSC.

5.1 ER Data Taxonomy

Typically, an SSC is a part of a system (see Figure 5 [left]) where such system is designed to provide a specific function, that is, emergence (e.g., electric power generation for a power plant). Each SSC contributes to the system emergence by providing a specified functionality that is being used by other SSCs through a set of connections where operands (e.g., mass, energy, or data) are exchanged. The goal of a system health program is to monitor not only the correct operation of each component but also their health parameters, such as aging and degradation (indicated as $\underline{F}(t)$ in Figure 5 [right]). In addition, a system health program is designed to perform appropriate actions to assure component functionality (indicated as $\underline{T}(t)$ in Figure 5 [right]). In this report, $\underline{T}(t)$ also includes all the external stressors that contribute to altering component aging and degradation.
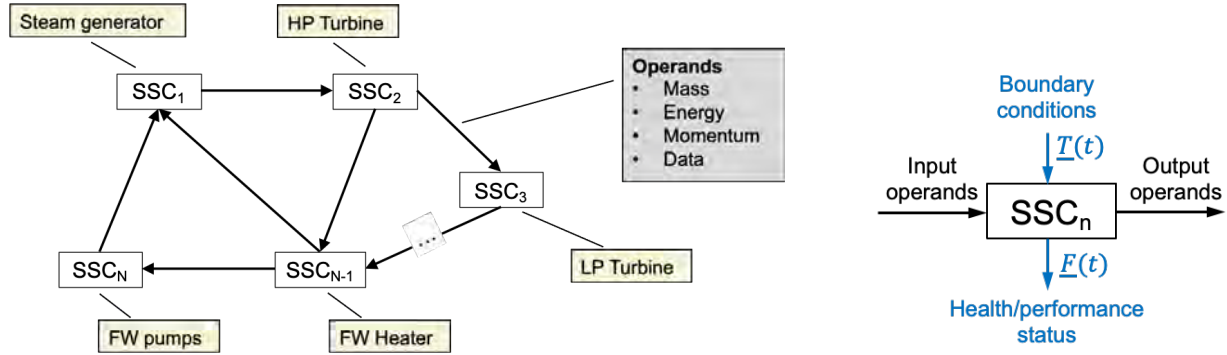
Figure 5. System (left) and component (right) representation.

5.1.1 System Engineer Perspective

When moving to the component level, it is vital to understand and capture the relationship between monitoring/testing data, MAs, and failure modes (FMs). This is typically neglected with the current state of practice of ER data analysis methods. Model-Based System Engineering (MBSE) [8] practices provide several solutions to model component from both *form* (i.e., which elements are part of the SSC) and *functional* (i.e., how SSC elements interact with each other, and which functions they support) points of view.

These solutions are based on MBSE languages that model system/SSC form and functional elements through a set of diagrams. The most commonly used languages are object process methodology (OPM) [9], unified modeling language (UML) [10], and systems modeling language (SysML) [11]. For the scope of this project, we have chosen the OPM language because it provides basic modeling elements for which we are looking for and, more importantly, it is possible to automatically generate textual translation of the OPM diagrams. As indicated in Section 5.5, this textual translation will be "matched" to the content of IRs and WOs.

Figure 6 provides an example of functional/form description of a generic SSC by employing an OPM diagram. An SSC OPM diagram provides an essential description of the SSC from both a form and functional perspective. This diagram explicitly indicates how SSC internal functions ($Func_f, f = 1, \dots, F$) act upon operands and how the elemental components ($ssc_r, r = 1, \dots, R$) support these functions.

From an equipment reliability perspective, monitoring/testing activities (i.e., $\underline{F}(t)$) act on both SSC functions (i.e., rotational frequency recorded for an induction motor) and form (i.e., blade corrosion of centrifugal pump) elements. On the other hand, degradation processes (i.e., $\underline{T}(t)$) directly alter the form-related elements of the component (i.e., ssc_r) that consequently affect SSC functional elements (i.e., $Func_f$,). Typically, from a reliability perspective, component FMs are described in terms of loss of function; hence, in the OPM diagram, FMs are only directly linked to the functional elements of the component (i.e., $Func_f$,). Lastly, note that MAs (such as component replacement, refurbishment, or reconditioning), indicated as MA in Figure 6, act on the form elements of components (i.e., ssc_r).

For the scope of this report, the OPM diagram of a component represents the key point to automatically understand and analyze health data $\underline{F}(t)$ (e.g., IRs). In particular, it clearly links monitored/recorded data with FMs that might affect component performance and MAs that would restore component functionality. We are employing model-based data analysis methods with the goal of linking component models with data rather than using machine learning methods [23], which solely rely on the available data to perform diagnostic/prognostic operations. Note that an OPM diagram extends failure modes and effects analysis tables by providing a form and functional description of the considered system in a graphical form.

In Appendix A we are presenting a small description of main OPM diagram elements and an example of OPM modeling for a generic centrifugal pump.
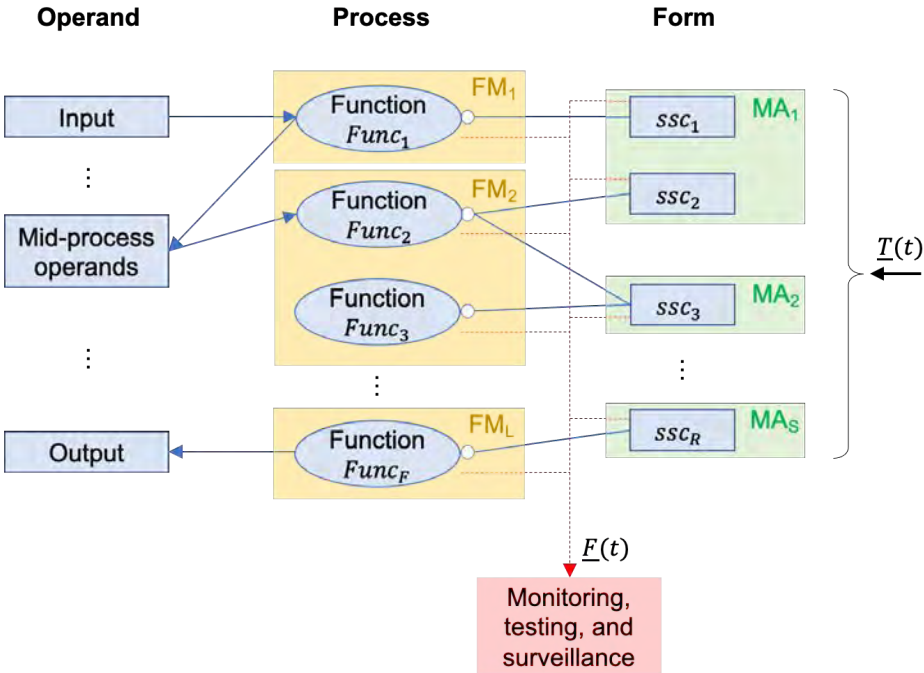
Figure 6. SSC representation through an OPM diagram.

5.1.2 Data Scientist Perspective

The next step is to characterize a generic component SSC from a data scientist point of view. This is shown in Figure 7 where three levels are identified: the component level (which would correspond to what is shown in Figure 6), a sensor/monitoring level (which retrieves and records portions of $\underline{T}(t)$ and $\underline{F}(t)$ in digital form), and data level. Data retrieved from $\underline{T}(t)$ (i.e., $\underline{\theta}(t)$) can be either textual (e.g., work orders) or numeric (e.g., environment temperature). We indicate here with "num" the portion of $\underline{\theta}(t)$ that is numeric while we indicate with "NL" the portion of $\underline{\theta}(t)$ that is textual (NL here stands for natural language). Data retrieved from $\underline{F}(t)$ has been portioned into two portions, component health and performance monitoring ($\underline{\varrho}(t)$ and $\underline{\gamma}(t)$), which can be numeric or textual in nature as well.

Figure 8 provides the temporal evolution of SSC health: starting from its installation point (i.e., at beginning of life [BOL]), its operational use and external factors contribute to SSC degradation. This degradation process affects SSC functional operation, and it materializes when anomalies emerge from its behavior. When the anomalous behavior is discovered then an IR is generated to report the identified SSC degradation and, if appropriate, a WO is developed and executed to restore SSC condition or performance.
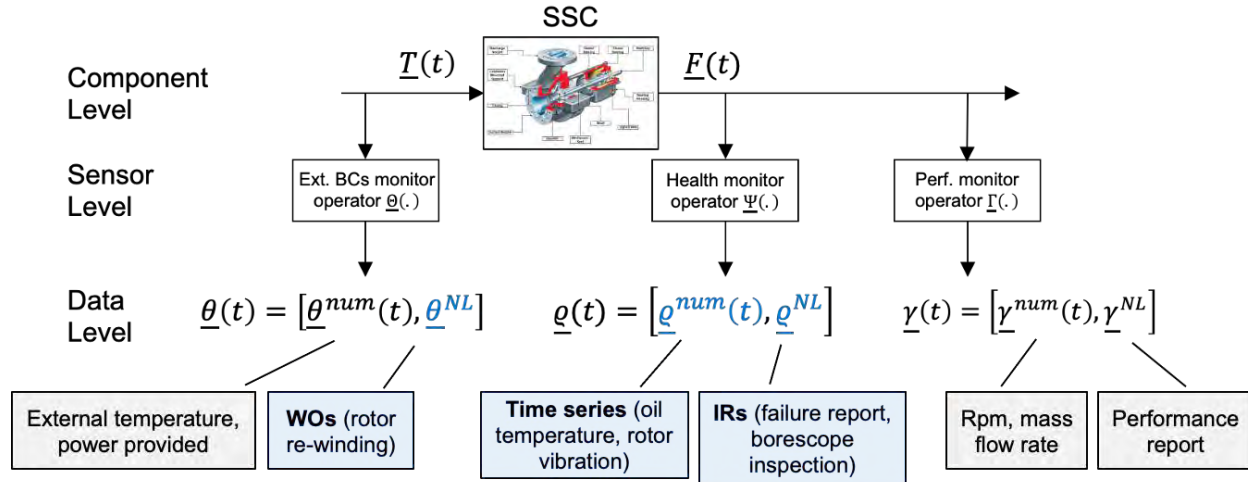
Figure 7. System health program: component representation from a data point of view. Examples are reported for each element in the data level.
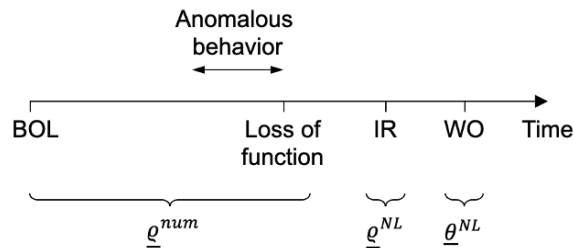
Figure 8. Temporal presentation of SSC health data. Elements of the data level of Figure 7 are reported.

5.2 Analysis of ER Data: A Causal Approach

As indicated in Section 5.1, the goal is to extract information from plant text data (e.g., maintenance reports or IRs). The approach described in this report is not based on the identification of the *correlation* between data elements using machine learning methods as indicated in many literature works [75, 76, 77]. Instead, the goal is to identify and trace the *causal* relationship between events. The proposed approach is based on causal inference [78, 79]. Causal inference differs from classical statistical inference [80, 81] by the fact that it is not based solely on data, but it requires a model which provides insights on the causal relationship between stochastic variables.

These models are typically graphical in nature where representation of the causal relationship between stochastic variables is performed though directed acyclic graphs (DAGs). DAGs consist of nodes which represent stochastic variables and arrows that connect nodes and represent causal relationships between the nodes themselves (see Figure 9).
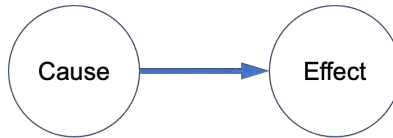
Figure 9. Node representation between cause and effect using DAGs.

In a typical NPP setting, several SSCs are constantly monitored, and relevant events are recorded in the plant M&D center. As an example, specific events (e.g., SSC failure) might be caused by a process that results in SSC deterioration (in condition, performance, or both); in this respect, Figure 9 can be adapted to a NPP setting as shown in Figure 10.
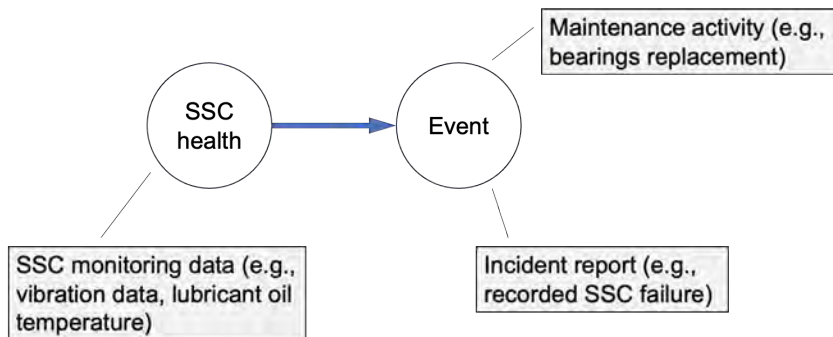
Figure 10. DAG representation between cause and effect in a NPP setting between SSC health (where monitoring condition-based data is available) and recorded event (e.g., SSC failure).

Figure 10 displays a very basic relationship between cause and effect for one single SSC. Note that monitoring data or the recorded event for each DAG node might not be always available from the plant M&D center.

When dealing with complex systems (e.g., an NPP), multiple SSCs are linked together (see Figure 5 - left) to support an emergence function (e.g., electricity production for an NPP). Consequently, the DAG representation in this situation might be very complex. An example is illustrated in Figure 11 which shows a more complex scenario that involves the component cooling water (CCW) pump. The CCW pump is providing cooling to both reactor cooling pumps (RCPs). In this scenario, degradation of the CCW pump causes a partial loss of cooling in the RCP seals.

In this context, a DAG diagram represents the causal relationship between events and SSC health: it recreates the "story" behind observed events and data. We are in fact moving away from current methods that aim to identify correlations between events and data. However, note that the DAG diagram is obviously not available but needs to be created. Our methods are designed to create a DAG diagram based on ER data. Note that to achieve this objective possessing ER data alone is not sufficient, we also need:

1. Models that can provide insights on how SSCs operate and how they are connected to each other (see system engineer perspective indicated in Section 5.1.1)

2. Links between ER data and SSC models (see system engineer perspective indicated in Section 5.1.2)

10

These elements are addressed by SSC OPM models that capture form and functional elements of SSCs, and by data mining methods that capture order, duration and coincidence of events. Our data analytics methods (that are presented in Sections 5.3-5.5) employ OPM models and advanced data mining methods to:

1. Capture information contained in available ER data (text and numeric)

2. Explore causal relationship between ER data elements

3. Exploit the generated relationships for anomaly detection, diagnostic, and prognostic purposes

The initial development of these models and methods started during FY 2021 and will continue during FY 2022.
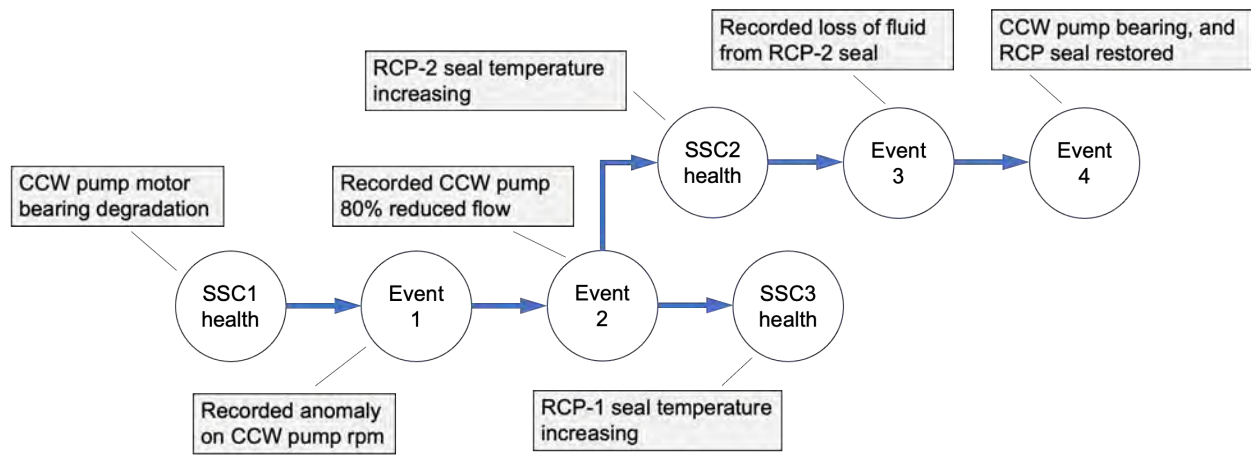
Figure 11. Example of DAG representation of cause and effect in a NPP setting when multiple SSCs are considered.

5.3 Analysis of ER Data

As indicated in Section 5.1.2, equipment reliability data can be of different formats (i.e., numeric and textual). In addition, the events/logs that are recorded in $\underline{\theta}^{NL}(t)$ or $\underline{\gamma}^{NL}(t)$ can be defined over an interval or at a single time instant. These two observations lead to a challenge when we analyze equipment reliability data: identify a common data structure that can be employed to represent numeric and textual data and events defined over time instants and time intervals. The advantage of having a common data structure is that it considerably simplifies the causal representation of events and monitoring data for condition-based monitoring applications.

This challenge has been resolved by representing all elements of $\underline{\theta}(t)$, $\underline{\varrho}(t)$, and $\underline{\gamma}(t)$ (numeric and textual) in symbolic form (i.e., a series of symbols). This approach has the advantage that it simplifies the

integration of numeric data with recorded events to identify patterns and outliers. In more detail, the method is structured in the following four steps:

1. Symbolic conversion of numerical time series. This is performed using the symbolic aggregate approximation (SAX) method [12]. Data preprocessing (e.g., identification of anomalous behavior) may be required depending on the actual situation.

2. Symbolic representation of textual data. This is performed by characterizing events and logs into a tree form using NLP methods [13,14]. A tree form has the advantage that it easily captures the structural relationship among text objects (e.g., nouns, verbs).

3. Combine data from Steps 1 and 2 into a common symbolic data structure. In our case, this is performed by creating a multivariate symbolic time series.

4. Apply model-based causal inference methods on the structure generated in Steps 3 by coupling data analysis methods with component OPM diagrams to infer component health, its FMs, and related maintenance activity that should be performed.

A graphical description of the methods here presented is shown in Figure 12.
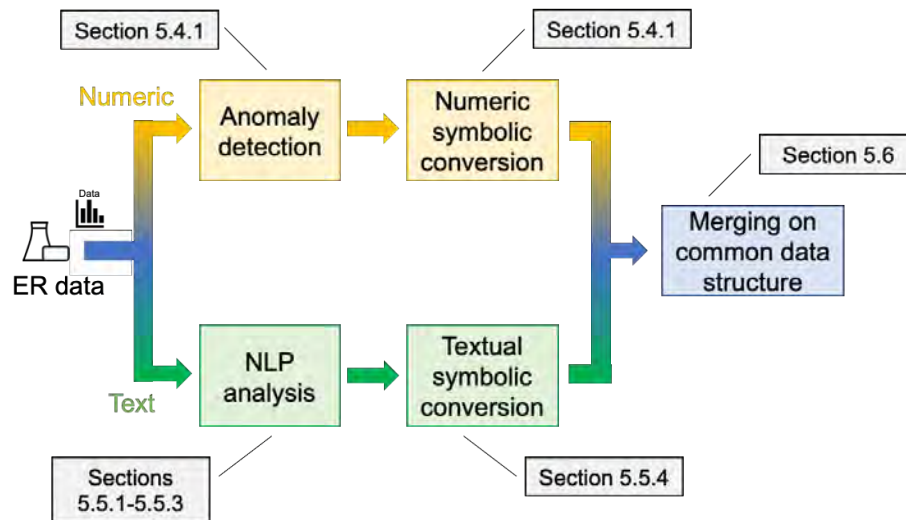
Figure 12. ER data analytics workflow. Numeric and text data are initially analyzed separately and then merged into a common symbolic language.

5.4   Analysis of Numeric Data

5.4.1   Anomaly Detection from Numeric Data

During FY 2021 work, the anomaly detection process has been performed using the auto-associative kernel regression (AAKR) [15] algorithm. This algorithm relies solely on data observed during normal conditions, and it uses such training data to estimate a reconstructed signal based on the evolution of the observed (i.e., real) signal and identify anomalous behaviors if they should occur.

One of the tasks when monitoring complex systems is the validation of the measured data before it is actually used to infer system status. While several methods that employ advanced machine learning methods can be found in the literature [16,17,18,19], we have opted for methods characterized by robustness

and explainability. For the characteristics of this project, the AAKR method [15] fully satisfies the desired requirements.

In brief, the AAKR method employs historic measured data that does not contain anomalous behaviors (i.e., normal conditions) and it validates this data set with currently measured data through a kernel regression. We indicate:

- Ξ^{obs-nc}: the set of N historical observations where each observation contains J data elements of different nature (e.g., pump oil temperature, vibration data)

- Ξ^{obs} as the currently measured data for the considered J data elements.

Based on the observed data (i.e., Ξ^{obs-nc}) and the currently measured data (i.e., Ξ^{obs}), the AAKR algorithm reconstructs through a regression the values of the J data elements (here indicated as Ξ^{rec}) which, under normal conditions, should be very similar to Ξ^{obs}, i.e., $\Xi^{obs} \cong \Xi^{rec}$. The condition $\Xi^{obs} \neq \Xi^{rec}$, indicates anomalous behavior.

The basic version of the AAKR method [15] estimates the expected data Ξ^{rec} as follows:

$$\Xi^{rec} = \frac{\sum_{n=1}^{N} w(n)\, \Xi^{obs-nc}(n)}{\sum_{n=1}^{N} w(n)} \tag{1}$$

where:

$$w(n) = \frac{1}{\sqrt{2\pi h^2}} e^{-\frac{\left(\Xi^{obs} - \Xi^{obs-nc}(n)\right)^2}{2h^2}} \tag{2}$$

The parameter h is determined during the AAKR training process and it sets the regression function (see Equation 2). This method has been developed as a Python class and tested over several datasets. Given the different nature of the J data elements (i.e., different ranges and units), the terms Ξ^{obs-nc} and Ξ^{obs} are properly normalized. This normalization is performed in the "training" step of the AAKR method where the Ξ^{obs-nc} data are provided. We performed a Z-normalization where each of the J data elements are normalized by subtracting its mean ($mean(\Xi^{obs-nc})$) and dividing by its standard deviation ($std(\Xi^{obs-nc})$):

$$\Xi^{obs-nc} = \frac{\Xi^{obs-nc} - mean(\Xi^{obs-nc})}{std(\Xi^{obs-nc})} \tag{3}$$

An example of anomaly detection is represented in Figure 13. The anomaly can be identified when observed signal Ξ^{obs-nc} (blue line) deviates considerably from the reconstructed signal Ξ^{rec} (red line); the anomaly is identified in Figure 13 within the region highlighted in red.

In common industrial settings, not only historic normal conditions but also recorded abnormal conditions $\Xi^{obs-abnc}$ data might be available. The goal is now to also employ the abnormal conditions data $\Xi^{obs-abnc}$ to improve anomaly detection capabilities. This was performed by creating two instances of the AAKR method: one instance trained on Ξ^{obs-nc} and one trained on $\Xi^{obs-abnc}$. The goal of the second instance is to confirm the detection of an anomalous behavior when similar behavior has been recorded in the past.

Within the LWRS program, the development of advanced anomaly detection methods using deep learning methods is currently underway under the Plant Modernization pathway as indicated in [82]. The RIAM project is collaborating with this pathway to integrate both efforts. For more complex situations, where AAKR limitations might be reached, we will integrate the anomaly detection methods based on deep learning developed by the Plant Modernization pathway into the RIAM workflows.
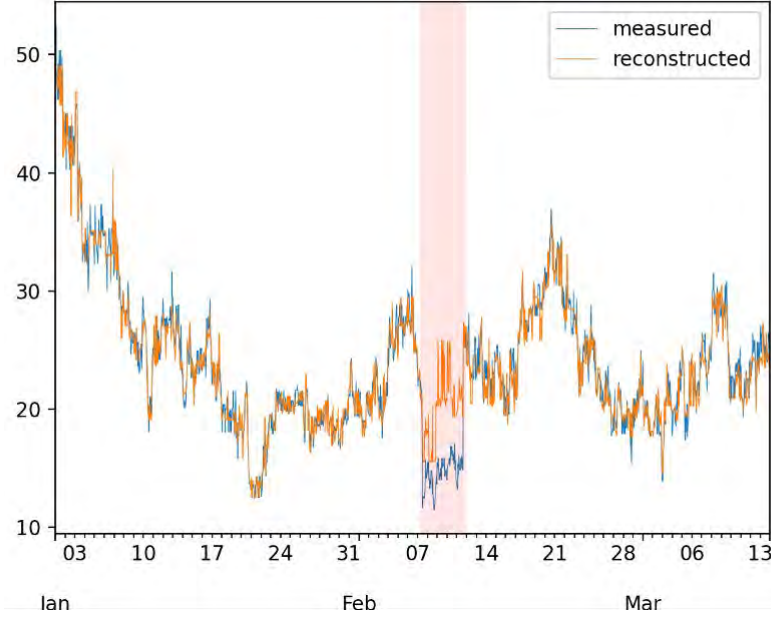
Figure 13. Anomaly detection using the AAKR method. Anomaly is identified when observed (Ξ^{obs-nc}) and reconstructed (Ξ^{rec}) signal differs (see region highlighted in red).

5.4.1.1   AAKR Application Example

We tested the developed method on a dataset provided by the LWRS Plant Modernization pathway. This dataset contains the temporal profile of two variables, $u0$ and $y0$, during both normal and abnormal conditions. This dataset was partitioned in two parts, a training set and a testing set:

- A part designed to train the AAKR model and set the optimal AAKR parameter (e.g., h).

- A part designed to test the performance of trained model.

Regarding the train data, Figure 14 shows the scatter plot of two variables (e.g., $u0$ and $y0$) for normal (green points) and abnormal (red) conditions. Note how normal and abnormal conditions only differ for very large and very small values of these two variables.

Regarding the test data, Figure 15 shows the temporal profile of the observed variable $y0$ (plotted in red) and the reconstructed profile of the same variable using the AAKR method (plotted in green). Figure 15 also indicates using the blue line where anomalous and abnormal behaviors should be detected (label set to 0 for normal and to 1 for abnormal behavior). Note how initially the reconstructed $y0$ (green line) from the AAKR method is not dissimilar for the recorded one (red line). We then proceeded to analyze the temporal behavior of the residual of the AAKR algorithm, i.e., the difference between the observed and the reconstructed $y0$. Figure 16 shows the temporal behavior of the residual (blue line). Given the wide fluctuations of the residuals, we smoothed it using classical kernel density estimation methods (orange line) to capture trends in the residual time series.

Based on the training data, the residual time series under normal conditions was fluctuating in the ± 0.012 interval (red lines in Figure 16) and employed this interval to detect anomalies when actual residual values cross this interval. Note from Figure 16, the predicted anomaly behavior (i.e., when the orange line escapes the ± 0.012 interval) matches with the expected anomaly (when label, green line in Figure 16, transitions from 0.0 to 1.0).

14
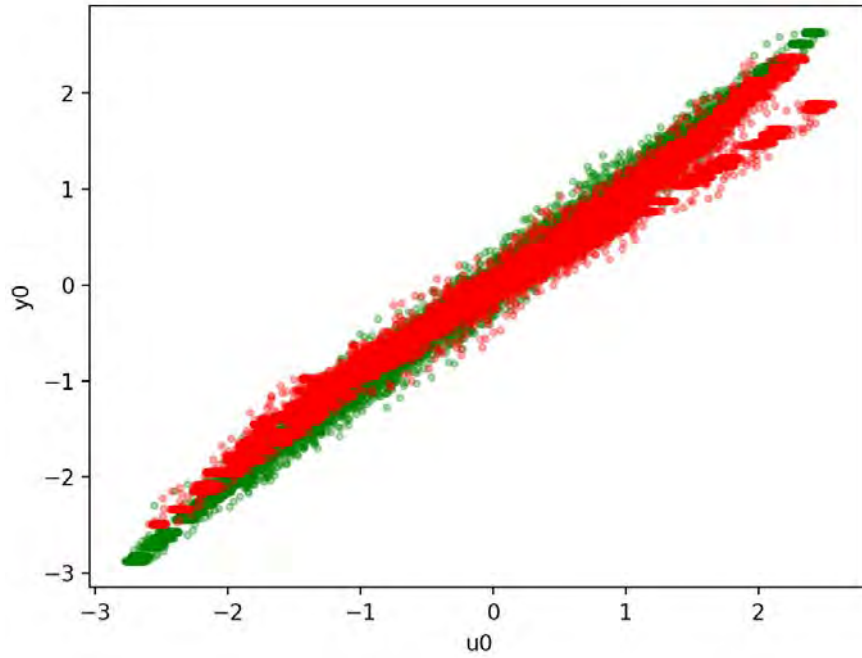
Figure 14. Scatter plot of the considered two variables (i.e., $u0$ and $y0$) for normal (green points) and abnormal (red) conditions.
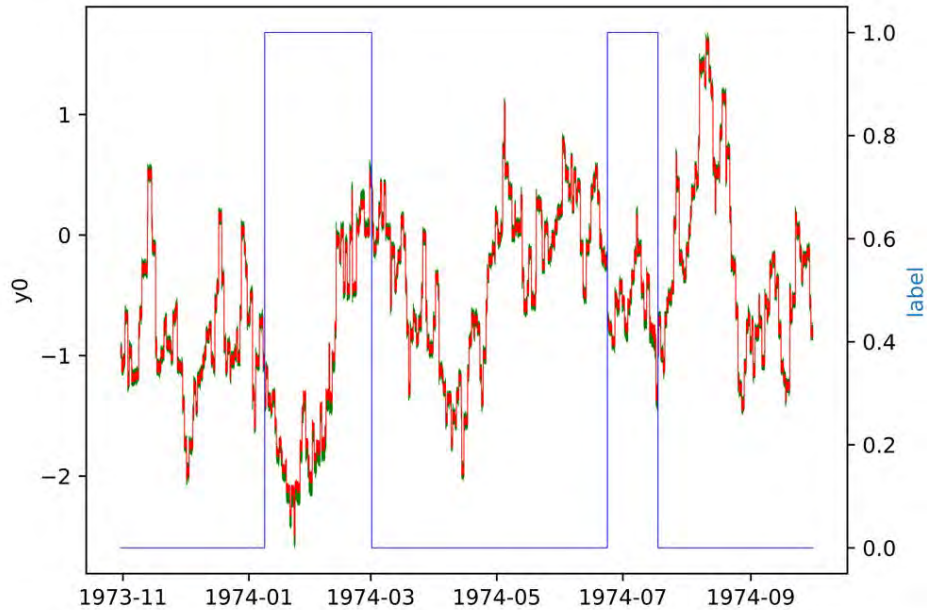
Figure 15. Temporal profile of the variable $y0$ (red line) and the reconstructed profile of $y0$ using the AAKR method (green line). The blue line indicates where anomalous and abnormal behaviors should be detected (label set to 0 for normal and to 1 for abnormal behavior).
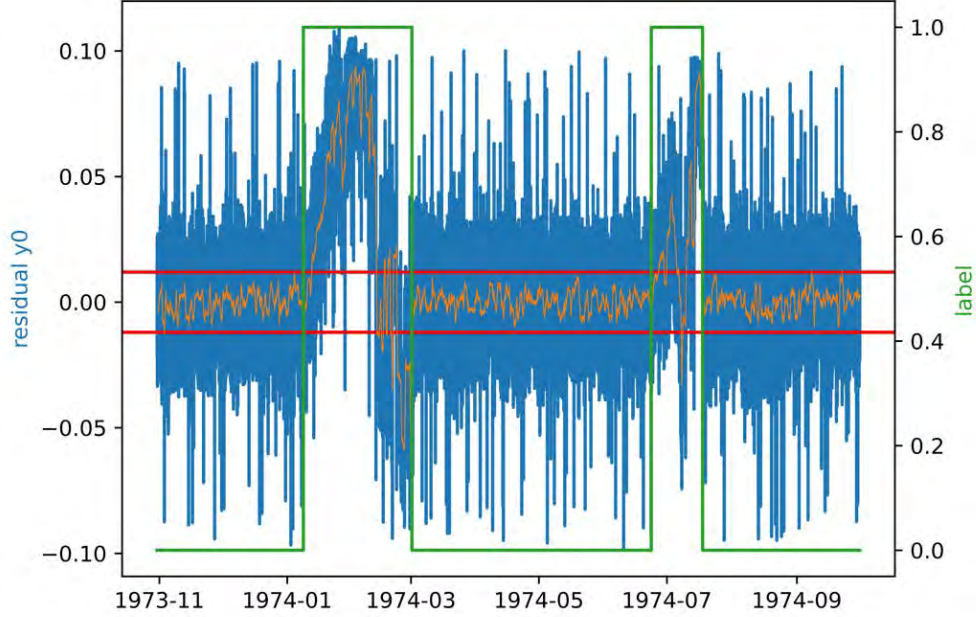
Figure 16. Anomaly detection through the analysis of the residual of the variable $y0$ (blue line). The residual is smoothed (orange line) and employed for anomaly identificaion when it crosses the provided boundaries (red lines).

5.4.2   Symbolic Representation of Numerical Time Series

The next step is to convert the time series into symbolic form using the SAX algorithm [12]. In short, this method performs a symbolic conversion of the original numerical data (i.e., it is converted as an ordered list of symbols [e.g., letters]). SAX is an algorithm that allows the user to represent continuous time-varying data S as a series of n symbols $\bar{\bar{S}} = \bar{\bar{s}}_1, \bar{\bar{s}}_1, \dots, \bar{\bar{s}}_n$ where $\bar{\bar{s}}_i$ is a symbol.

While temporal discretization is performed by partitioning the time axis into n intervals having the same length, the discretization of the numerical time series is typically performed by dividing the range of the desired variable into α equi-probable regions. Each region has a character $\bar{\bar{s}}$ associated to it and the alphabet size has cardinality[3] α. The resulting conversion generates a time series of length n and an alphabet size equal to α. The SAX algorithm consists of the steps (also see Figure 17) described in Algorithm 1. The result is a phrase $\bar{\bar{S}}$: a timely ordered sequence of symbols. An example of discretization for the temporal profile of a scenario taken from [20] is shown in Figure 18.

Typically, data generated by simulations contain the temporal profile of multiple variables; moreover, as also shown in Figure 17, a fixed number (i.e., n) of time intervals having equal length is not optimal to capture rapid changes of S.

The issue of dealing with multiple variables can be solved by:

- Performing Steps 2 and 4 in Algorithm 1 independently for each variable.
- Maintaining the order of symbols for every variable in each time interval.

---

[3] Cardinality of the alphabet refers to the number of characters used.

16

Input: n and α

1: Normalize the data (mean equal to 0 and standard deviation equal to 1)

2: Partition the temporal interval into n equal sized intervals

3: Divide the distribution of $\theta(t)$ into equi-probable regions and assign a symbol to each region (alphabet has cardinality equal to α)

4: Consider the average value $\bar{s}$ of $\theta(t)$ in each interval

5: For each $\bar{s}$ assign its own $\bar{\bar{s}}$ according to the discretization performed in Step 4

6: Generate a phrase $\bar{\bar{S}}$ as a timely ordered sequence of symbols
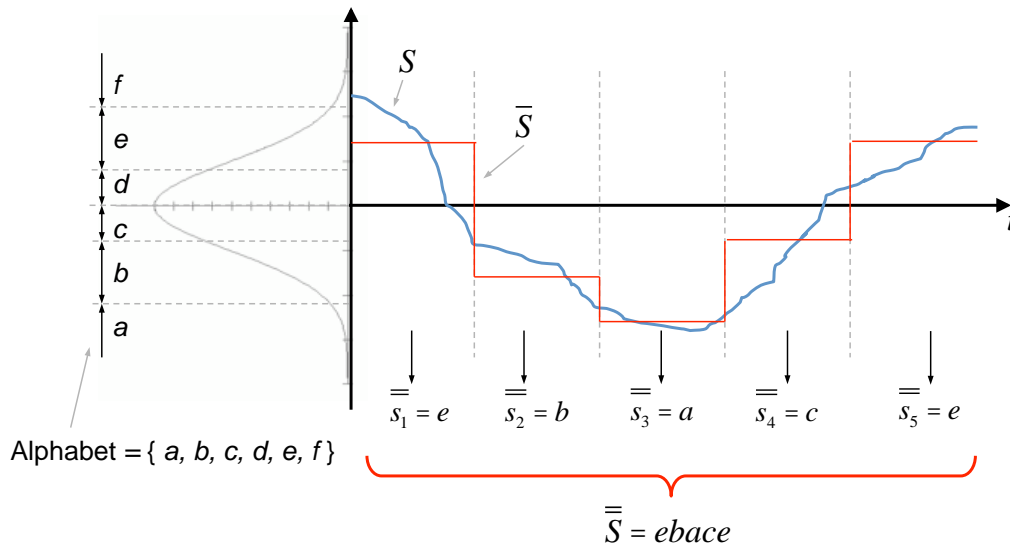
Figure 17. Example of symbolic conversion with a = 6 and n = 5.

Regarding the issue of identifying rapid changes of state variables, a solution is to recursively analyze the rate of change of the covariance matrix computed in that interval (as shown in [20]). The rationale is to choose time intervals such that the rate of change of the covariance matrix eigenvalues is below a fixed threshold (see Algorithm 2). As input, the minimum and a maximum length of the time interval are required to preserve accuracy and avoid extremely long phrases.
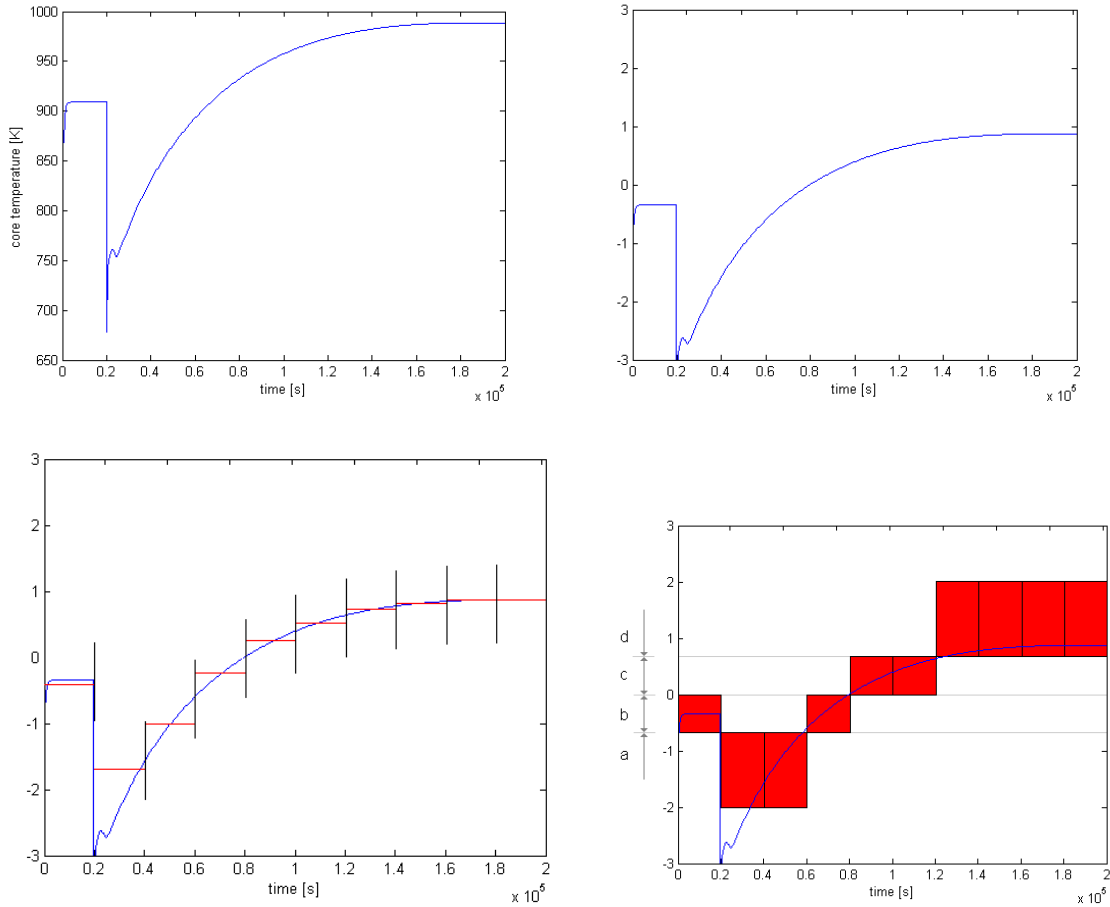
Figure 18. Application of the SAX algorithm [12] for a time series: raw data (top left), data normalized (top right), temporal discretization (bottom left) and symbol sequence generation (bottom right). SAX generates the sequence of symbols: "baabccdddd".

Algorithm 2: Adaptive time discretization.
1. Input: maximum value for the rate of change of the eigenvalues of the covariance matrix, $\dot{\lambda}_{Max}$; minimum and maximum length of the time discretization, i.e., t_{Min} and t_{Max}
2. Divide the time scale into intervals having length t_{Max}
3. Evaluate the covariance matrix of the data points contained in that interval and determine its eigenvalues λ_i
4. Determine the highest eigenvalue relative variation $\dot{\bar{\lambda}}$
5. If $\dot{\bar{\lambda}} > \dot{\lambda}_{Max}$ then split the interval into 2 intervals of equal length
6. Repeat Steps 3 and 4 for all intervals

18

5.5   Analysis of Text Data

Most methods found in the literature [21,22] perform processing of text reports using supervised learning [23] to predict the report nature (e.g., failure, operating). In this report, we are following a different path: the goal is to analyze the sentence structure of logs and reports, organize information in a structured form, and create a structural relationship among text objects (i.e., understand who/what did what, when, why, where). This is being accomplished by employing NLP methods[4] to perform two main tasks, syntactic and semantic analysis tasks.

As a starting point, we are characterizing the content of a generic IR or a maintenance report. Note that a maintenance report content is fairly straightforward since it basically reports component replacement or restoration activity. In addition, it does not really contribute to the causal reasoning since it represents the last node in the DAG scheme.

On the other hand, IRs (along with SSC monitoring data, see Sections 5.1 and 5.4) provide insights on the nodes of the DAG diagram. In other words, an IR describes a portion of a DAG: a node, or the causal relation between two nodes. Thus, two classes of IRs can be defined:

1. Class 1 IR: When the IR reports a DAG node, this node can be either an event (e.g., SSC malfunction) or data regarding the health of the component (e.g., excessive corrosion on pump impeller).

2. Class 2 IR: When the IR reports a causal relation between two nodes, the content of these nodes can be any combination between events and SSC health information linked by a causal relationship.

Note that this classification scheme defined by these two mutually exclusive classes needs to be validated with NPP actual data to measure its validity (i.e., the degree to which the two classes are in actuality mutually exclusive). In the validation process we can measure the percentage of actual NPP IRs that falls in each class and, more importantly, the percentage of IRs that do not fall in either of the two classes. Note that the classification provided above are relevant to IRs related to plant equipment performance. Since IRs can be written on a wide variety of topics (e.g., issues related to programmatic performance, human performance, etc., it is expected that a substantial fraction of IRs would be classified as not being one of the two classes related to plant system and equipment health defined above.

Section 5.5.1 describes the general NLP pipeline to analyze IRs. Then, steps required to extract information from each of the two classes of IRs are described in Sections 5.5.2 and 5.5.3.

5.5.1   IR Analysis Pipeline

The first step in the analysis of text data is to perform syntactic analysis [24,25] of the raw text by employing the rules of formal grammar. It is here assumed that the text is in a digital form (typically in a string form). The syntactic analysis is performed through the following main steps (see Table 2 presented below for a more detailed list of analysis steps):

1. Sentence segmentation and word tokenization: each sentence is translated into a list of string elements

2. Part of speech (POS) tagging: identification of grammatic elements of each string (e.g., nouns, verbs). Here we relied of the POS tags developed in the Penn Treebank project[5] (see Table 3)

---

[4] In this work, we are employing three main `Python` libraries: STANZA (https://stanfordnlp.github.io/stanza/), NLTK (www.nltk.org), and SPACY (https://spacy.io).

[5] Penn Treebank project official website: https://catalog.ldc.upenn.edu/LDC99T42

3. Named entity recognition: classify text entities (e.g., names, dates, events) and identify them (e.g., component ID, event occurrence time)

4. Relation extraction: create a knowledge graph where entities identified in Step 3 are linked together in a graph that reflects the structure of the original sentence

Steps 1 to 8 listed in Table 2 are common in any NLP analysis [13,14]. Our approach deviates from standard NLP method in Steps 9 and 10.

In Step 9 we identify in the text the elements of the SSC OPM model (i.e., operands, forms or functions as indicated in 5.1.1). From each SSC OPM model we can generate a set of OPL textual elements which lists not only all OPM elements but also their relationship. Appendix A presents a detailed OPM model (see Figure 47) for a centrifugal pump and its corresponding OPL elements (see Table 29).

In Step 10 we infer the causal relationship between elements of the IR. These relationships are in the form of *cause* and *consequence*. Here, we exploit the observations reported in the IR by plant system engineers and trace back causal relationship with other IRs using the SSC OPM models.

Table 2. NLP analysis pipeline.

ID	Steps	Data generated	Note
1	Retrieve raw text	Raw text data	
2	Cleaning	Cleaned text data	Process of cleaning raw text data from not text related elements[6]
3	Segment sentence	List of strings	Each sentence is analyzed separately
4	Clean punctuation	List of strings	Punctuation is removed
5	Tokenize sentence	List of lists of strings	Each sentence is split into a set of words
6	Stemming and lemmatization	List of lists of strings	Each word is converted into its own dictionary form or to its stem/root form
7	Part of speech tagging	List of lists of tuples	Process of marking each word as corresponding to a particular part of speech using grammatical rules
8	Entity recognition	List of lists of tuples	Process designed to identify and classify named entities into predefined classes such as: SSC type, systems, locations, time values
9	OPM entity recognition	List of lists of tuples	Process designed to identify OPM elements (functions or forms)
10	Information Extraction	List of lists of tuples	Process of extracting information content from text (see Section 5.5.2 and 5.5.3).

---

[6] For this task we have employed `Beautiful Soup` (https://www.crummy.com/software/BeautifulSoup/bs4/doc/)

Table 3. Alphabetical list of POS tags developed in the Penn Treebank project.

Number	Tag	Description
1	CC	Coordinating conjunction
2	CD	Cardinal number
3	DT	Determiner
4	EX	Existential there
5	FW	Foreign word
6	IN	Preposition or subordinating conjunction
7	JJ	Adjective
8	JJR	Adjective, comparative
9	JJS	Adjective, superlative
10	LS	List item marker
11	MD	Modal
12	NN	Noun, singular or mass
13	NNS	Noun, plural
14	NNP	Proper noun, singular
15	NNPS	Proper noun, plural
16	PDT	Predeterminer
17	POS	Possessive ending
18.	PRP	Personal pronoun
19	PRP$	Possessive pronoun
20	RB	Adverb
21	RBR	Adverb, comparative
22	RBS	Adverb, superlative
23	RP	Particle
24	SYM	Symbol
25	TO	to
26	UH	Interjection
27	VB	Verb, base form
28	VBD	Verb, past tense
29	VBG	Verb, gerund or present participle
30	VBN	Verb, past participle
31	VBP	Verb, non-3rd person singular present
32	VBZ	Verb, 3rd person singular present
33	WDT	Wh-determiner
34	WP	Wh-pronoun
35	WP$	Possessive wh-pronoun
36	WRB	Wh-adverb

A missing task, which will be investigated during FY 2022, is the process of coreference resolution. This process is tasked to find the expressions that refer to the same entity in the text. This is particularly relevant where the text includes several sentences and a reference to an entity is indicated not with its proper name but with a pronoun. An example is provided in Figure 19 where coreference in the second sentence, the pronoun "its", needs to be linked to the entity "pump". Note that this resolution might occur both within and between sentences.
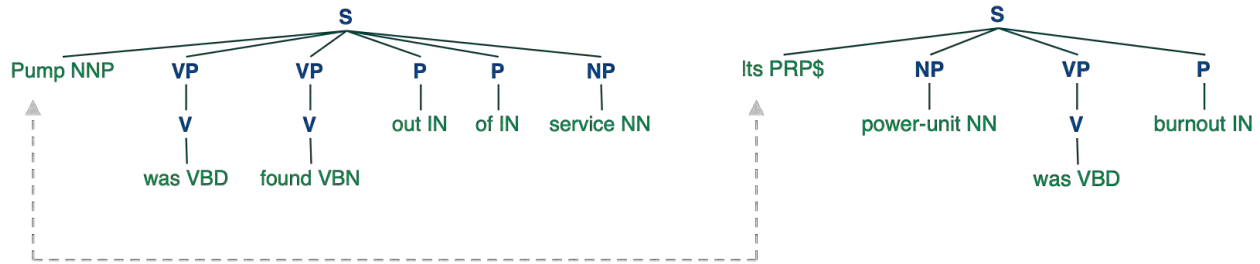
Figure 19. Example of coreference resolution (dashed line) for the text: "Pump was found out of service. Its power-unit was burnout". The pronoun "its" references the noun "pump".

5.5.2   Information Extraction from Class 1 IRs

The methods designed to extract information from IRs that belong to Class 1 has been structured in a similar way to the one presented in [72]. We, in fact, based our methods on a new set of rule templates based on specific trigger words and relations. During FY 2021, our work focused on the development of status nouns and verbs which would indicate degradation of SSC functions or SSC internal elements.

The chosen set of status words includes verbs, adjectives, and nouns obtained again from the WordNet[7] database. For Class 1 IRs, we have identified three categories of status words (negative, anomalous, and positive) and they are shown in Table 4, Table 5, and Table 6. Table 7 provides an initial list of status relations encoded using STANZA.

Table 4. Set of negative status nouns, verbs and adjectives.

Status nouns	Status verbs	Status adjectives
Failure	Fail	Unable
Degradation	Degrade	Ineffective
Breach	Break	Anomalous
Fracture	Decline	
Decline	Go bad	
Decay	Rupture	
Loss	Breach	
	Reduce	
	Increase	
	Decrease	
	Fracture	
	Aggravate	
	Worsen	
	Lose	

Table 5. Set of positive status nouns, verbs and adjectives.

Status nouns	Status verbs	Status adjectives
Operation	Function	Operating
Functioning	Work	Operational
	Operate	Functional
	Run	Usable

Table 6. Set of anomalous status nouns, verbs and adjectives.

Status nouns	Status verbs	Status adjectives
Observation	Find (out)	Unchanged
Detection	Observe	Unaltered
	Detect	Constant
	Determine	Consistent
	Discover	Stable
	Get	Unaffected
	Notice	
	Become	
	Record	
	Register	
	Show	

Table 7. Set of status relations.

Relation
A (noun) "status verb" "status adjective"
A (noun) "status verb" "status verb-ing"
"Status adjective" B (noun) "status verb"
"Status noun" "status verb" prep. B (noun)

These status relations were coded in a Python based code which relies on the Stanford NLP library STANZA. Once the IR has been processed using all steps listed in Table 2, a set of tuples is created from each sentence in the form (SSC, form/function, health status). These tuples are designed to represent in digital form the DAG node as follows:

SSC; subject = 'OPM function/form'; health status = 'ok, 'degraded' or 'anomalous'

As an example of Class 1 IR is provided as follows:

Oil puddle was found in proximity of CCW Pump 1B.

By using the NLP analysis steps 1 through 7 listed in Table 2 using STANZA and NLTK Python libraries, the resulting grammatical structure of the IR is shown in Figure 20. This figure shows the POS tags (see Table 3) represented on top of each word, and the grammatical dependencies[8] between words (represented with arrows).
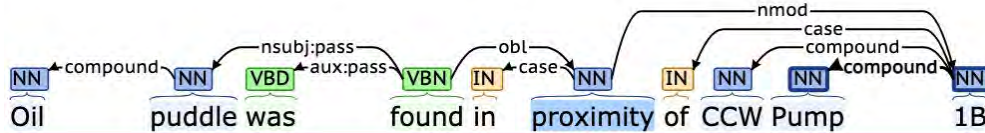
Figure 20. Grammatical decomposition and analysis of the example class 1 IR.

Step 8 in Table 2 is accomplished by looking in the IR for specific SSC tags (i.e., CCW pump 1B). It is here assumed as well that SSC tags are unique and given. Once the SSC has been identified, its OPM

---

[8] Refer to https://downloads.cs.stanford.edu/nlp/software/dependencies_manual.pdf for a complete description of each Stanford dependency.

model (see Appendix A) is employed to identify OPM elements in the sentence that refer to such model (see Step 9 in Table 2). In this case, the word "oil" is linked to the OPM form element "ISO VG100 oil".

Next, Step 10 of Table 2 is performed where the verb "find" is identified (i.e., verb being part of anomalous status, see Table 6). The following tuple is constructed:

(SSC=CCW Pump 1B; subject=ISO VG100 oil; health status=anomalous)

Note that now the OPM model is employed to propagate anomalous behavior contained in the IR to other OPM elements such as:

Motor → rotating → pump → accelerating function

5.5.3 Information Extraction from Class 2 IRs

For the extraction of the causal relationship between elements of a sentence, we identified the works presented in [72, 73, 74] as candidates to effectively perform such task. From our literature overview, this area of research is fairly new but very active. In addition, all developed NLP methods do not guarantee a 100% success rate at analyzing text data and extract information.

As a first attempt, we followed the methodology presented in [72] since it provides robust and explainable analysis results. This method is based on a set of rule templates based on specific trigger words and relations. The chosen set of words includes verbs and nouns obtained from the WordNet database[9] and is shown in Table 8.

Table 8. Set of trigger causal verbs and nouns [72].

Causal nouns	Causal verbs
Result	Cause
Reason	Stimulate
Cause	Make
	Derive
	Trigger
	Result
	Lead
	Increase
	Decrease

Similarly, the chosen set of causal relations has been constructed from common English syntactical rules as indicated in Table 9. These causal relations were coded in a Python based code which relies on the Stanford NLP library STANZA[10]. This library provides a set of algorithms to perform linguistic analysis that can be used to construct fairly complex NLP analysis pipelines.

Once the IR has been processed using all steps listed in Table 2, a set of tuples is created from each sentence in the form (cause=A → consequence=B). These tuples are designed to represent in digital form the DAG node indicated in Figure 9:

(SSC, OPM form/function, health status) → (SSC, OPM form/function, health status)

---

[9] WordNet official website: https://wordnet.princeton.edu/

[10] STANZA official website: https://stanfordnlp.github.io/stanza/

As an example of class 2 IR is provided as follows:

Bearing failure of CCW Pump 1B caused reduced flow.

By using the NLP analysis steps 1 through 7 listed in Table 2 using STANZA and NLTK Python libraries, the resulting grammatical structure of the IR is shown in Figure 21. This figure shows the POS tags (see Table 3) represented on top of each word, and the grammatical dependencies between words (represented with arrows).

Table 9. Set of causal relations [72].

Relation
A (noun) "causal verb" B
A (verb) "causal verb" B
B was "causal verb" A
A is a "causal noun" of B
B was "causal verb" by A (verb)
A "causal verb" in/to/from B
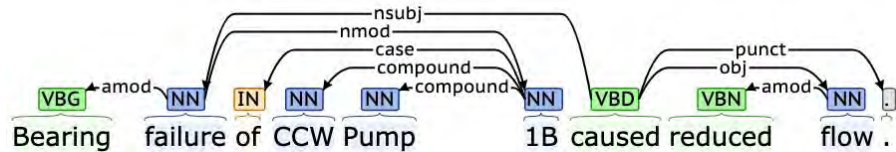
Figure 21. Grammatical decomposition and analysis of the example class 1 IR.

Step 8 in Table 2 is accomplished by looking in the IR for specific SSC tags (i.e., CCW pump 1 B). Again, it is here assumed that SSC tags are unique and given. Once the SSC has been identified, its OPM model (see Appendix A) is employed to identify OPM elements in the sentence that refer to such model (see Step 9 in Table 2). In this case, these OPM elements in the text have been identified: "bearing" and "flow".

Next, Step 10 of Table 2 is performed; here, the causal verb "cause" (see Table 8) is identified which indicates a causal relationship as follows:

(CCW Pump 1B, bearing, degraded) → (CCW Pump 1B, high internal v flow, degraded)

5.5.4 Symbolic Representation of Text Data

At this point, a generic IR has been characterized (see Sections 5.5.2 and 5.5.3). The next step is to capture timing and ordering of events. An example is given in Figure 22; note that this example is directly linked to the DAG diagram represented in Figure 11. Figure 22 shows timing and ordering of events in a graphical form. The goal now is to create a digital structure that allows us to data mine duration, coincidence, and order of events.

In this respect, the time series knowledge representation (TSKR) algorithm [14] offers a flexible way to solve this problem. TSKR is a hierarchical language for expressing temporal knowledge in symbolic data. The term hierarchical refers to the fact that the symbolic conversion is performed in three levels (see

Figure 23) that describes the concept of duration (through tones), coincidence (through cords), and order (through phrases).
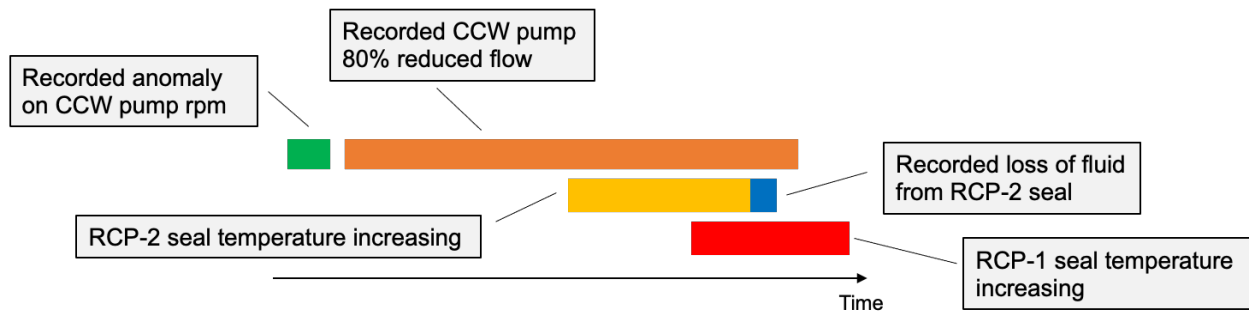
Figure 22. Example of temporal discrete events for the events shown in Figure 11.
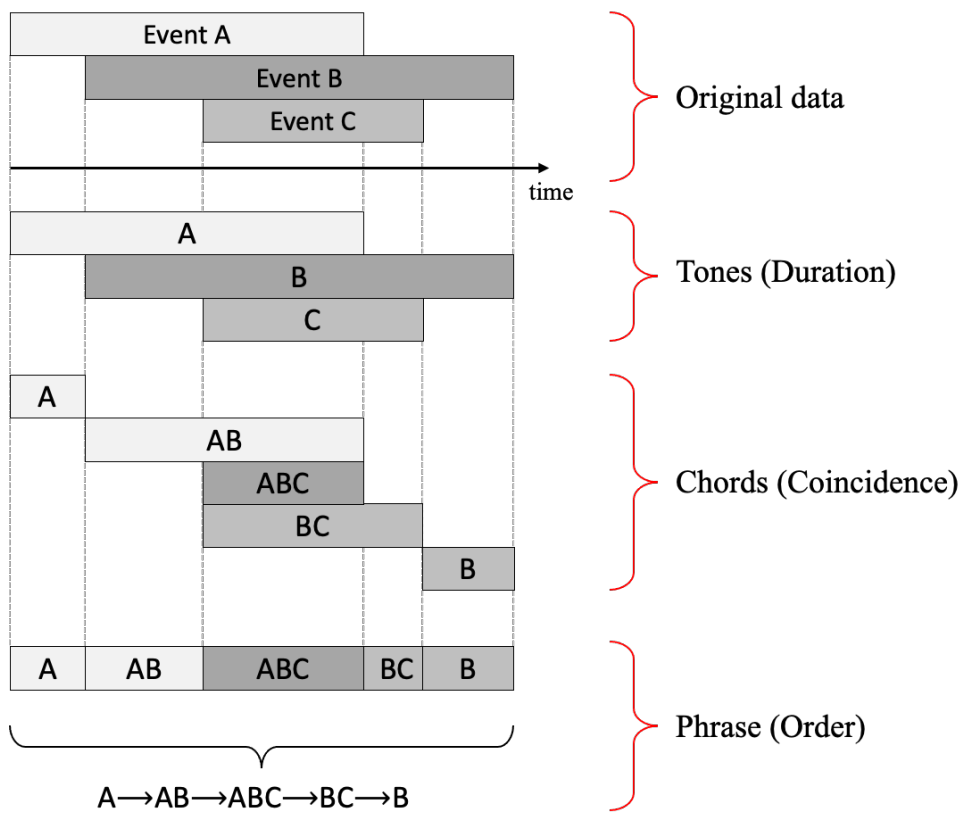
Figure 23. Generation of a phrase from a sequence of tones.

A few definitions are need before presenting the actual algorithm:

- Tone: Tone is the elemental component of time interval analysis. It is described by the triple $\{\sigma, s, e\}$ where σ is the symbol associated to the time interval $[s, e]$
- Chord: A chord pattern describes a time interval where $k > 0$ tones coincide

- Phrase: A phrase is a sequence of $k > 1$ nonoverlapping chords.

The algorithm structure is presented in Algorithm 3 below.

Algorithm 3: TSKR algorithm
1: Mining aspects: given a set of d-dimensional time series, select and label k aspects
2: Mining tones: generate a series of tones from the k aspects generated in Step 1
3: Mining marginally interrupted tones: find and filter small temporal gaps in order to avoid generation of chords having small temporal intervals
4: Mining chords: given k tones, generate a set of chords
5: Mining phrases: generate a symbolic interval ordered sequence representing occurrences of the set of chords determined in Step 4

Note that a set of k not overlapping tones still produces k chords, each of them containing a single tone.

5.6   Construction of Common Data Structure

At this point, the numeric (see Section 5.4) and text data (see Section 5.5) need to be "merged" together in a single time series. This is performed by following the same philosophy behind the TSKR algorithm (see Section 5.5.4), where the recorded events (which are tree structures) are inserted in the corresponding cells of the time series symbolic array. As an example, in Figure 24 we are focusing on a time series where three events are recorded from text data: events E_1 and E_3 are defined over a time instant while E_2 is defined over an interval. The tree structure for each event (e.g., type of event, component ID, date) is then associated with the corresponding cell generated by the SAX algorithm.
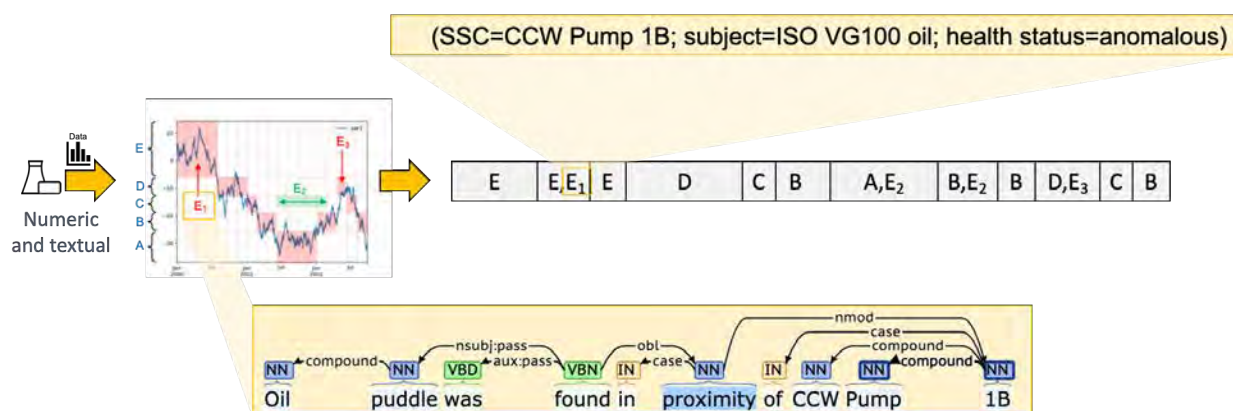
Figure 24. Symbolic conversion of numerical (time series in the plot on the left) and textual data (events E_1, E_2, and E_3) into a single data structure.

5.6.1 Data Analysis Methods

At this point, the data structure shown in Section 5.6 is fairly flexible to analyze by employing standard methods [9] (such as Markov models, decision trees, and suffix trees) but also more advanced symbolic analysis methods [17]. However, the most relevant approach is based on subseries clustering: the symbolic time series (as generated in Section 5.6) can be partitioned into subseries and can identify those subseries that occur frequently (i.e., motif discovery), those that have never been recorded in the past (i.e., anomaly detection), and those that are close to a newly recorded subseries (data forecast). As a final comment, note that the memory requirements for the data structure shown in Section 3 are very small; numeric values occupy more memory than a string or a chart. This guarantees fast performances for diagnosis and prognosis applications.

The core development of analysis methods for symbolic data structures will be performed during FY 2022.

5.7 Evaluation of Data Analytics and Machine Learning to Support SFCP Evaluations

One element of achieving cost reductions at operational NPPs in the United States is for plant owner/operators to adopt approved risk-informed applications that permit NPPs to address operational and maintenance issues more efficiently by consideration of the affected SSC's contribution to plant safety. Several of these risk-informed approaches have been developed by industry and approved for implementation by the U.S. Nuclear Regulatory Commission (NRC). These applications include the following (references indicate approved industry implementation guidance documents):

- Risk-informed Categorization and Treatment of Structures, Systems, and Components (10 CFR 50.69) [26, 27]

- Risk-Informed Technical Specifications Initiative 4B, Risk-Managed Technical Specifications [28]

- Risk-Informed Technical Specifications Initiative 5B, Risk-Informed Method for Control of Surveillance Frequencies [29].

An operating plant may adopt these risk-informed programs by obtaining an amendment to their operating license. Once the amendment is obtained, application of the program is controlled by plant processes and procedures that implement industry guidance specified in application specific documents that have been endorsed by the NRC.

Although each of the risk-informed applications cited above have unique objectives and requirements, they all possess the characteristic that their implementation requires the retrieval and evaluation of historical plant data. Performance of these activities requires significant effort from plant staff to identify, retrieve, and evaluate the relevant data. Due to both the volume and diversity of these data, this effort can be time consuming and resource intensive. Application of computational techniques using data analytics (DA) and machine learning (ML) approaches has the potential to substantially reduce the time and effort necessary to perform these activities, thus providing additional economic benefits from the adoption of various risk-informed initiatives.

The research detailed in this section describes areas where an initial proof of concept to apply DA/ML techniques could be applied to one of the risk-informed applications identified above—Risk-Informed Technical Specifications Initiative 5B. Because the original surveillance frequencies specified in the plant Technical Specifications were conservatively chosen, these frequencies are considered by regulatory authorities to provide an adequate level of plant safety. However, because these task frequencies were chosen to be conservative, subsequent plant operating experience has shown that cost savings could be obtained by extending many of these activities without incurring any degradation in SSC performance or

plant safety. As a result of approval of Risk-Informed Technical Specifications Initiative 5B, many U.S. NPPs have obtained license amendments permitting licensee control of plant surveillance intervals for critical safety-related SSCs. In addition, between 2011 and 2012, the NRC updated the published Standard Technical Specifications for each of the reactor types in use in the U.S. to reflect adoption of this risk-informed initiative (see References [30–34]).

Plants that have adopted a risk-informed approach to control the frequencies of required surveillance activities must implement a surveillance frequency control program (SFCP). Once the plant license amendment is approved, the SFCP is implemented in accordance with industry guidance contained in NEI 04-10 [29]. The plant license amendment that applies this guidance permits licensee control of surveillance test frequencies for required surveillance activities that are specified in the plant technical specifications. In this licensee-controlled process, existing surveillance frequencies are removed from the technical specifications and a paragraph is added to the Administrative Controls section that refers to the NEI 04-10 [29] document for the control of surveillance frequencies. The surveillance test requirements (test methods) themselves are not changed and remain in the plant technical specifications.

The methodology described in NEI 04-10 [29] applies a risk-informed, performance-based approach for the modification of surveillance frequencies as specified in the plant technical specifications. The approach is consistent with the philosophy described in NRC Regulatory Guide 1.174 [35]. In the approach, probabilistic risk assessment (PRA) methods are used to evaluate the risk impacts of proposed revisions to the testing intervals with sensitivity studies used to address the impact of uncertainties. Additionally, a multi-disciplinary integrated plant decision making panel (IDP) performs a comprehensive review of the potential impacts that could result from the proposed revisions that accounts for operating experience (both plant-specific and industry), test history, manufacturer recommendations, applicable codes and standards, and other factors, as well as the risk insights obtained from the PRA evaluations.

As a result of implementation of these programs at many plants, the industry has accumulated significant experience in the execution of this risk-informed application. An important feature of this application is that, once the license amendment has been approved, the plant owner/operator has discretion over which particular SSCs will be evaluated for surveillance frequency extensions and the timing of implementation of those extensions. As a result, NPPs that have implemented this risk-informed program generally have applied it to those SSCs where the application of the process was expected to be straightforward and relatively simple. However, even for these cases, experience indicates that collection and evaluation of the necessary data required to support the desired surveillance frequency extensions is time consuming and resource intensive.

Because of this experience, it is postulated that this application would serve as a useful case to evaluate the use of DA/ML to support utility implementation of risk-informed programs. In the conduct of SFCP analyses, plant data used in the various engineering evaluations and reviews (SSC failure data, instrument calibration data, etc.) typically require manual collection, processing, and evaluation. These activities are labor intensive and potentially could be performed more efficiently using DA/ML techniques. The initial research described in this report is intended to identify which DA/ML techniques would be most applicable (e.g., use of image recognition to characterize instrument calibration "As Found"/"As Left" data from completed surveillance tests that are used to evaluate instrument drift rates) and to develop a future DA/ML application for pilot demonstration at a host NPP.

The process of extending surveillance intervals within the conduct of a plant SFCP requires the evaluation of both the impact of the surveillance extension on plant risk (as determined in the plant PRA) and operational experience related to the specific SSCs that are tested. The first of these evaluations is predominantly a theoretical evaluation that uses the existing plant PRA model to develop insights related to the potential impact on risk due to the postulated surveillance frequency change. The second evaluation consists predominantly of obtaining and analyzing relevant operational history associated with the

particular SSCs that are impacted. Each of these two evaluations are addressed below with potential areas where DA/ML techniques could be applied to obtain more efficient and cost-effective results.

Although reviews of the impact on plant risk to support SFCP determinations can be resource intensive and time consuming, they generally are well defined and relatively straightforward to accomplish with requirements specified in NEI 04-10 [29]. Specific requirements include the following:

- Ensure plant PRA models meet the requirements specified in NRC Regulatory Guide (RG) 1.200 [36] for model scope, completeness, and quality (Step 5 in NEI 04-10). Of particular importance to a plant SFCP are items related to evaluating key modeling assumptions and sources of uncertainty, which serve as inputs to performing appropriate sensitivity evaluations related to the proposed surveillance frequency changes.

- Review plant PRA models to determine whether the SSCs tested in the surveillance are modeled (Step 8 in NEI 04-10).

- If the SSCs tested in the surveillance are not modeled in the PRA, assess the feasibility of modifying the model to include them (Step 9 in NEI 04-10). If such PRA model modifications are feasible, they are implemented (Step 11 in NEI 04-10); if they are not feasible then a determination is made whether a qualitative analysis would be sufficient or, alternatively, if a bounding analysis can be performed to ensure the risk impact in terms of core damage frequency (CDF) and Large Early Release Frequency (LERF) of the proposed frequency change are acceptable (Step 10 in NEI 04-10).

- In addition, review the PRA models to ensure that the total cumulative effect on risk (CDF and LERF) that is accrued by all frequency changes incorporated as part of the SFCP are acceptably small.

- Conduct sensitivity studies to assess the potential impact of uncertainties (Step 10 in NEI 04-10). This process is accomplished by modifying the unavailability terms in the basic events that represent the SSCs addressed in the surveillance activity. Generally, a factor of 3 is considered to be an appropriate value as a 95th percentile upper bound applicable to distributions of reliability for SSCs in operating NPPs.

The results of these evaluations are summarized and presented to the IDP for review and approval of the proposed surveillance frequency change.

A fundamental characteristic of the activities described above is that they all utilize the plant PRA model. A key area where DA/ML techniques could provide cost and resource savings is in the activity that reviews the plant PRA model to determine whether SSCs tested in the surveillance activity are included in the model. This generally requires identification of which basic events (if any) correspond to the specific SSCs and the functions they provide that are tested in the surveillance. For these evaluations, DA/ML could be applied to search the PRA model using a combination of NLP techniques to identify if and where the specific SSCs occur in the model. In addition, since most safety-related SSCs in NPPs are present in more than one train, if SSCs from one train are found, the similar SSCs for these trains also should exist in the model. This is true both for mechanical equipment (e.g., four trains of residual heat removal [RHR] pumps, valves, and piping) and actuation/control instrumentation (e.g., four loops of containment pressure sensing instrumentation to actuate emergency core cooling system [ECCS] and containment isolation functions). As a result, use of clustering or nearest neighbor algorithms may be useful to limit the time and cost of searches (i.e., perform a detailed search through the model for one instance of the SSCs and a more limited search to identify the basic events for the other instances of the related SSCs that perform identical functions for the other trains). Additionally, although not specifically a DA/ML approach, development of computational scripts could be used to automate generation of data (via repeated runs of the PRA model with varied failure rates for the SSCs impacted by the surveillance frequency change) used in the conduct of the required sensitivity studies; thus, potentially further reducing analysis cost and time.

Although the application of DA/ML approaches may provide improved efficiencies in the conduct of SFCP evaluations, a much more fruitful area for use of these techniques could be in obtaining and analyzing the operational history associated with the particular SSCs under consideration. There are two reasons for this. First, tasks related to this portion of a SFCP assessment are much broader in scope than that related to reviews and analyses of plant PRA models. Second, experience with SFCPs at operating plants has shown that the data needed to perform the required evaluations often are contained in multiple locations (e.g., different databases) and often in different formats; this situation is a major reason that performance of SFCP reviews can be resource intensive and time consuming.

As discussed previously, the evaluation of the risk impacts of postulated frequency extensions represents a relatively straightforward application with most of the necessary data and analysis being contained in a single software application (i.e., the plant PRA model contained in the PRA software). Additionally, the analyses performed generally are well defined with straightforward acceptance criteria. In contrast, the engineering evaluations are much broader in scope with less well-defined criteria for acceptance. For these evaluations, a wide range of information is collected and reviewed. This includes:

- Bases for the surveillance frequency that was prescribed in the plant Technical Specifications or described in the plant final safety analysis report (FSAR).

- Commitments made to external agencies regarding the surveillance interval. In addition to commitments made to the NRC, these also can include commitments made to other governmental agencies (at the federal, state, or local level) as well as commitments to other organizations, such as industrial insurance carriers (such as American Nuclear Insurers [ANI] or Nuclear Electric Insurance Limited [NEIL]).

- Previous surveillance history associated with the SSCs for which the surveillance frequencies are being considered for extension as well as surveillance history associated with other similar equipment.

- Operational and maintenance history associated with the SSCs for which the surveillance frequencies are being considered for extension as well as surveillance history associated with other similar equipment used at the plant. This typically would include reviews of work orders as well as any issues or adverse performance trends that have been identified in the plant maintenance rule program or corrective action program (CAP).

- Review of applicable industry information related to performance issues related to the SSCs for which the surveillance frequencies are being considered for extension. This would include information contained in the Institute of Nuclear Power Operations (INPO) operating experience (OPEX) database, NRC-related publications (generic letters [GLs], NUREGs, etc.), and vendor-supplied information.

Once these data are obtained and reviewed, the results typically are summarized to support the engineering evaluations that determine whether the desired surveillance frequency extension is feasible. Experience has shown that the collection and analysis of the data described above can be labor intensive and time consuming. Therefore, at many plants the application of SFCP has been limited to plant surveillances for which it is easily determined that implementation of the frequency extension is cost beneficial.

As is evident from these examples, such reviews utilize a wide range of information that is obtained from a variety of data sources. Once the data are collected, they must be put into a form that facilitates appropriate engineering analyses to support the surveillance frequency change. This represents a second area that has been found to be time consuming and labor intensive. A common example of this occurs in evaluation of operational experience to determine if the occurrence of calibration drift over the proposed extended surveillance interval would result in an unacceptable increased likelihood of the SSC not providing its intended function when required. Such a determination can be made using information obtained from previous surveillances (e.g., calibration "as found"/"as left" records) and work order/Maintenance Rule/CAP data. However, experience indicates that the historical data related to plant

calibration records at many plants have not been digitized and are obtained from paper records (either within a "hard copy" of the surveillance instruction itself or on instrument/loop calibration sheets). As a result, significant time can be spent obtaining the records, translating the relevant data from the "hard copy" into a suitable electronic format (e.g., spreadsheet or data base), and then performing relevant engineering analyses (e.g., calculation of instrument drift rates and uncertainties) to demonstrate that the new proposed task frequency would be acceptable.

Because the operational experience based data that need to be collected and reviewed to support a surveillance frequency extension are broad and diverse, use of DA/ML approaches could substantially reduce the effort required to obtain this information as well as perform an initial review/characterization of it. Potential application areas along with the respective DA/ML techniques are identified below.

- As indicated previously, an essential activity in the conduct of SFCP evaluations is to identify the bases for the specific surveillance activities and their frequencies. NLP techniques could be used to search the relevant sources such as plant Technical Specifications, FSAR, or other design basis documents to identify these bases.

- In the evaluation of plant surveillances, an important activity is to identify uniquely tested components (i.e., a component whose function is being tested in the surveillance being considered for frequency extension as well as any other surveillances which are performed less frequently). To identify these components, use of NLP techniques could be employed to evaluate the procedures in the plant surveillance program to determine (1) which specific test procedures test these components, (2) at what frequency each is tested, and (3) identify the test procedure(s) that test the component most frequently. For components that perform multiple safety-related functions or possess multiple success criteria, NLP also could be used to identify these characteristics at the individual function level (to the extent that this information is provided in the specific surveillance test procedures).

- Similarly, NLP techniques can be combined with structured data in plant work order (and other databases such as Maintenance Rule and CAP) using Bayesian classifiers to identify specific issues related to historical performance of the plant SSCs (equipment failures, instances, or trends of degraded performance, etc.) that are tested in the particular surveillance activity. Then, other DA/ML techniques, such as cluster analysis, could be applied to perform preliminary evaluations of the impact of these issues to plant safety, production, and economics.

- As indicated previously, industry experience has found that historical data related to plant calibration records often have not been digitized and are only found on paper records. In this case, application of image conversion and character recognition algorithms could be used to convert calibration data from "hard copy" to electronic format suitable for performing engineering evaluations.

However, much of the information that can be obtained from external sources (such as NRC, INPO, vendors) is not likely to be available in a format suitable for direct use of DA/ML applications. For example, relevant NRC reports (GLs, NUREGs, etc.) generally are electronically available in Adobe Acrobat (.pdf) format. Although use of various techniques, such as a combination of image conversion and character recognition, could be used to convert these records into a format amenable to use by other DA/ML techniques (such as NLP), it is an open question to what extent such an approach would be cost effective. As a result, these reviews may be more cost effective to perform manually.

The most critical step in the SFCP process is review and approval by the IDP to modify a particular surveillance interval. Per industry SFCP implementation guidance provided in NEI 04-10 (Step 16) [26], the IDP is "*is charged with the task of reviewing the proposed STI for both qualitative considerations and the quantitative results.*" In addition, the IDP develops recommendations related to how the revised surveillance intervals are implemented (e.g., implementation using a phased approach where the frequencies are extended gradually over a specified period of time). The IDP also is responsible for reviewing the cumulative impact of all changes implemented via the SFCP, monitoring the impact of

changes on SSC failure rates, and ensuring appropriate documentation is developed and maintained. Within the context of application of DA/ML techniques to support the SFCP, the IDP should be familiar with the techniques that are used, their potential limitations, and metrics used to evaluate their effectiveness in supporting decision making. In particular, the IDP should be cognizant of some standard performance criteria used for the evaluation of performance of DA/ML algorithms, such as:

- True/False Positives and True/False Negatives

- Precision and Recall

- Sensitivity and Specificity [37].

These metrics measure different but interrelated aspects of the performance of DA/ML applications. Of particular importance is that the IDP understands that DA/ML applications apply statistical processes, and that performance measured against the various indicators represents a tradeoff. For example, although the ideal objective would be to have both no false positive and no false negative occurrences for identification of a particular item, achieving this ideal is not possible using DA/ML approaches. This is due to the characteristics of statistical hypothesis testing where a decrease in the rate of occurrence of a Type 1 error necessarily results in the increase in the rate of occurrence of Type 2 errors (see Herroelen, et al.'s 1998 journal article, pp. 411-416 [38]). Note that the information related to characterization True/False Positives and True/False Negatives of typically is displayed in a 2×2 matrix format called a "confusion matrix" [39].

Based on the discussion above, the following list provides a summary of areas where research could be conducted to determine the extent to which DA/ML techniques would enhance the cost effectiveness of performing SFCP evaluations. These areas include applying the following:

- DA/ML to search plant PRA models to identify where specific SSCs for which frequency extensions are under consideration occur in the model.

- NLP techniques to identify the bases for specific surveillance activities and their frequencies from relevant sources such as plant technical specifications, FSAR, or other design basis documents. Similar techniques also can be applied to identify commitments associated with SSCs for which frequency extensions are under consideration.

- NLP techniques to structured plant data using Bayesian classifiers to evaluate historical performance of plant SSCs.

- Image conversion and character recognition algorithms to convert calibration data from "hard copy" to electronic format suitable for performing engineering evaluations.

In addition, development of overview training for the IDP to provide an understating of DA/ML techniques with particular emphases on the potential benefits and limitations for various ML techniques and standard performance monitoring metrics and what they signify would be needed to support effective deployment of these techniques in this application.

5.8   Evaluation of Effect on PRA Due to Variation of Component Surveillance Frequency

As indicated in Section 5.7, a thorough analysis of ER data can be used to support risk informed applications such as SFCP. This section provides a direct application on how the analyzed ER data can be integrated into existing plant PRA models to support SFCP decisions. Here we are looking at how a change in surveillance frequency for a specific SSC affects the probability of the basic event(s) associated with that SSC.

Surveillance of standby components is performed to gain confidence that a standby plant SSC will operate as necessary in the event of a need. There are two models used to evaluate the probability of failure for a surveillance test, one for the demand of the SSC to start and the other is for the operating during the test. For the purposes of this report, the evaluations will be conducted assuming that the standby SSC is a pump such as are used in NPP emergency core cooling systems.

The binomial model is used to perform a Bayesian inference for a demand-based component and the most common prior distribution is in the form of a Beta distribution. The parameter of interest for the binomial model is the probability of failure (here indicated as p). Nuclear industry performance parameters for centrifugal pumps are expressed as a Beta distribution. The Beta distribution is a conjugate distribution to the binomial distribution which means that the math of the Bayesian inference works out to provide a posterior result of a Beta-Binomial Bayesian update as a closed-form Beta distribution with exact parameters (i.e., α and β for a beta distribution $Beta(\alpha, \beta)$), the parameter of interest for the mean and the moments is p.

The Poisson model is used to perform a Bayesian inference for the failure rate of a running component and the most common prior distribution is in the form of a Gamma distribution. The parameter of interest for the binomial model is the rate of failure (here indicated as λ). Nuclear industry performance parameters for centrifugal pumps are expressed as a Gamma distribution. The Gamma distribution is a conjugate distribution to the Poisson distribution, therefore producing a closed-form Gamma distribution with exact parameters (i.e., α and β for a gamma distribution $Gamma(\alpha, \beta)$) as the Bayesian inference posterior, the parameter of interest for the mean and the moments is λ.

There are pump trains for safety systems such as high-pressure safety injection that are normally on standby and are operated outside of operational need only during surveillance (or in response to a plant initiating signal signifying the onset for a transient or accident condition). Other pump trains are operated continuously such as for service water. For these systems where redundancy exists (e.g., a service water system that has three installed pumps with only two required to be in service to provide full system function) it is common practice at many nuclear power plants to rotate the trains of normally operating pumps from operating continuously to standby, thereby keeping all trains at the same level of run times. All failure rates for pumps are determined empirically through a Bayesian update from operational data. The surveillance frequency of normally standby pumps is typically monthly for turbine driven pumps and quarterly for motor driven pumps. There is not a surveillance frequency for rotated normally operating pump trains because the rotation frequency acts as a surveillance.

If surveillance is performed at regular intervals, the probability of failure on demand is assumed to be consistent because the operating conditions generally are the same for all components. Increasing the intervals introduces the possibility of increasing the probability of failure for those failure modes that occur when the component is idle. The increase in failure probability with increased time to surveillance can be modeled by a distribution that fits the data. One proposed distribution is a logistic-normal distribution where the slope is constant. Another is a Weibull distribution where the slope increases with time as is the case when continuously running components experience wear-out, or end-of-life. The proper distribution needs to be fit from experimental data. A logistic-normal distribution is used here as an example. A fleet of 70 standby pumps that are normally surveilled at a 30-day frequency is run through a testing program where the frequency is increased by ten days for three surveillance periods. The resulting data for 40, 50, and 60 days is used to predict the failure probability for these surveillance periods and also 70, 80, and 90 day surveillance periods. The posterior distributions for periods with data is a numerical result from a Bayesian update using a logistic link function as a prior with uninformed parameters. This allows the data to drive the posterior. The resulting logistic link function parameters are then used to predict failure probabilities for the ensuing surveillance periods, thus eliminating the need for running further testing program surveillance periods.

Assume that a testing program was run on an ensemble 75 identical turbine driven centrifugal pumps that are normally in standby and are currently surveilled at 30-day intervals. Historical data is placed in the 30-day data and the new data are populated from the test for 40, 50, and 60 days as is shown in Table 10.

Table 10. Component failure to start: events recorded for different surveillance periods.

Surveillance period (days)	Failures	Demands
30	146	26557
40	1	75
50	1	75
60	2	75

Here we employ the logistic link function $ln\left(\frac{p[i]}{1-p[i]}\right) = a + bi$, where $p[i]$ represents the probability for the i-th surveillance period, and parameters for intercept and slope (i.e., a and b) are the linear parameters for the logistic linear function.

The Bayesian inference formula for this update is then the following:

$$w(p[i]) = \frac{\binom{n}{x}p[i]^x(1-p[i])^{n-x} * ln\left(\frac{p[i]}{1-p[i]}\right)}{\int \binom{n}{x}p[i]^x(1-p[i])^{n-x} * ln\left(\frac{p[i]}{1-p[i]}\right) dp} \tag{4}$$

where:

- n is the number of pumps in the study

- x is the number of failures in the i-th surveillance period

- $w(p[i])$ is the posterior probability

- $\binom{n}{x}p[i]^x(1-p[i])^{n-x}$ is the aleatory model likelihood

- $ln\left(\frac{p[i]}{1-p[i]}\right)$ is the prior

Simplified, the Bayesian formula is as follows:

$$Posterior(\Theta|data) = \frac{Likelihood(data|\Theta) * Prior(\Theta)}{\int Likelihood * Prior \, d\Theta} \tag{5}$$

where:

- Θ is the parameter of interest, in our case p

- *Likelihood* = Binomial distribution

- *Prior* = logistic linear function

- *data* = the results of the 30, 40, 50, and 60 day surveillance tests

A Bayesian inference was run using a Markov Chain Monte Carlo (MCMC) program to produce the posterior results and, in this respect, the posterior distribution for the logistic link function parameters a and b are described by the moments listed in Table 11. We then proceed to propagate the posterior distributions of a and b to the component probability of failure corresponding for different surveillance

intervals as indicated in Table 12. The box plot of the distributions of the component probability of failure to start for each surveillance period (see Figure 25) shows the growth in uncertainty for the predicted reliability data for 70, 80, and 90 days.

Table 11. Moments of the posterior distribution for the logistic link function parameters a and b.

Logistic link function parameter	Mean	5th percentile	Median	95th percentile
a	-5.68	-6.06	-5.69	-5.24
b	0.48	0.08	0.48	0.81

Table 12. Moments of the posterior distribution for the component probability of failure to start for different surveillance intervals.

Surveillance (days)	Mean	5th percentile	Median	95th percentile
30	5.50E-3	4.78 E-3	5.49E-3	6.27E-3
40	9.00E-3	5.90E-3	8.94E-3	1.23E-2
50	1.54E-2	6.43E-3	1.46E-2	2.70E-2
60	2.69E-2	6.97E-3	2.36E-2	5.83E-2
70	4.79E-2	7.55E-3	3.80E-2	1.22E-1
80	8.36E-2	8.17E-3	6.08E-2	2.38E-1
90	1.38E-1	8.85E-3	9.58E-2	4.13E-1
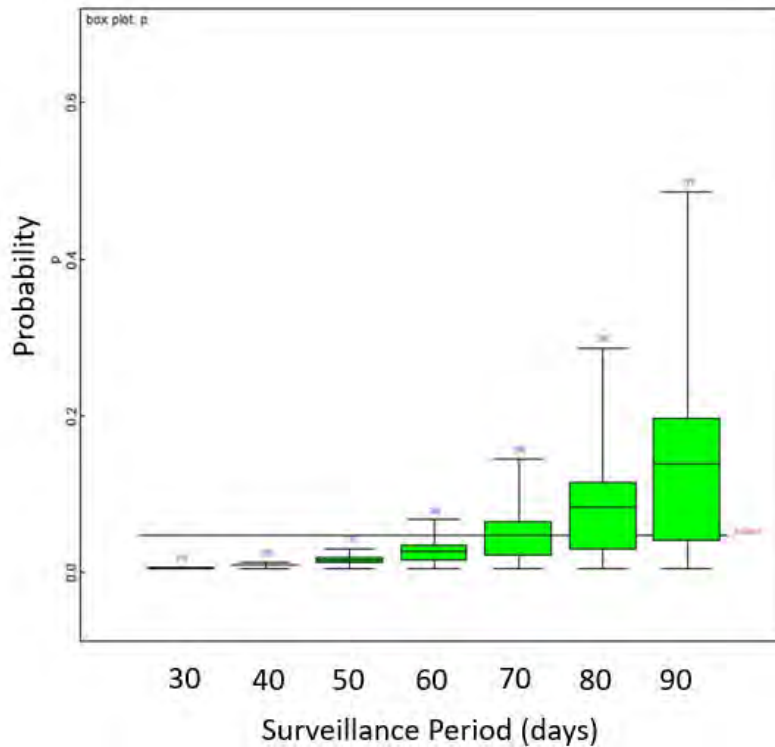
Figure 25. Box plots of the posterior distribution for the component probability of failure to start for different surveillance intervals.

There are similar increased failure mechanisms for running pumps immediately after they are started from a standby condition. The first hour of a pump starting up and running from standby is considered a loading hour where the pump experiences an initial increase in failure rate. This increased failure rate from a normally operating pump can be attributed to failure modes that occur when the component is idle. After the first hour, the pump can be considered to follow the failure rate for continuously running pumps. An increase in failure rate is logically assumed to increase further with the longer proposed surveillance time. This increased λ can be modeled using a loglinear distribution for a constant slope or a Weibull distribution if the slope increases with time. The proper distribution needs to be fit from experimental data. A Loglinear example is provided here.

Assume that the same testing program runs each of the 75 turbine driven centrifugal pumps that are normally in standby and are currently surveilled at 30-day intervals for four hours, for a total of 300 hours for the pump population. Historical data are placed in the 30-day data and the new data are populated from the tests for 40, 50, and 60 day surveillance intervals.

Table 13. Component failure to run: events recorded for different surveillance periods.

Surveillance Period (days)	Failures	Hours
30	10	1922
40	2	300
50	2	300
60	3	300

Here we employ a loglinear link function $ln(\lambda[i]) = a + bi$, where $\lambda[i]$ is the failure rate for the ith surveillance period and parameters a and b are the linear parameters for the loglinear function.

Using the loglinear link function model, the math is similar to above, only that the Poisson distribution is used for the likelihood and the loglinear link function is used as the prior. A Bayesian inference was run using an MCMC program to produce the posterior results below.

The posterior distribution for the loglinear link function parameters a and b are described by the moments listed in Table 11.

Table 14. Moments of the posterior distribution for the logistic link function parameters a and b.

Loglinear link function Parameter	Mean	5th percentile	Median	95th percentile
a	-5.48	-6.26	-5.461	-4.73
b	0.175	-0.19	0.18	0.51

We then proceed to propagate the posterior distributions of a and b to the component probability of failure corresponding for different surveillance intervals as indicated in Table 15. The box plot of the distributions of the component probability of failure to start for each surveillance period (see Figure 26) shows the growth in uncertainty for the predicted reliability data for 70, 80, and 90 days.

Table 15. Moments of the posterior distribution for the component probability of failure to run for different surveillance intervals.

Surveillance (days)	Mean	5th percentile	Median	95th percentile
30	5.22E-3	2.94 E-3	5.06E-3	8.06E-3
40	6.13E-3	3.86E-3	6.01E-3	8.80E-3
50	7.50E-3	3.84E-3	7.26E-3	1.20E-2
60	9.57E-3	3.37E-3	8.76E-3	1.85E-2
70	1.27E-2	2.87E-3	1.05E-2	2.98E-2
80	1.75E-2	2.41E-3	1.27E-2	4.86E-2
90	2.50E-2	2.02E-3	1.52E-2	8.00E-2
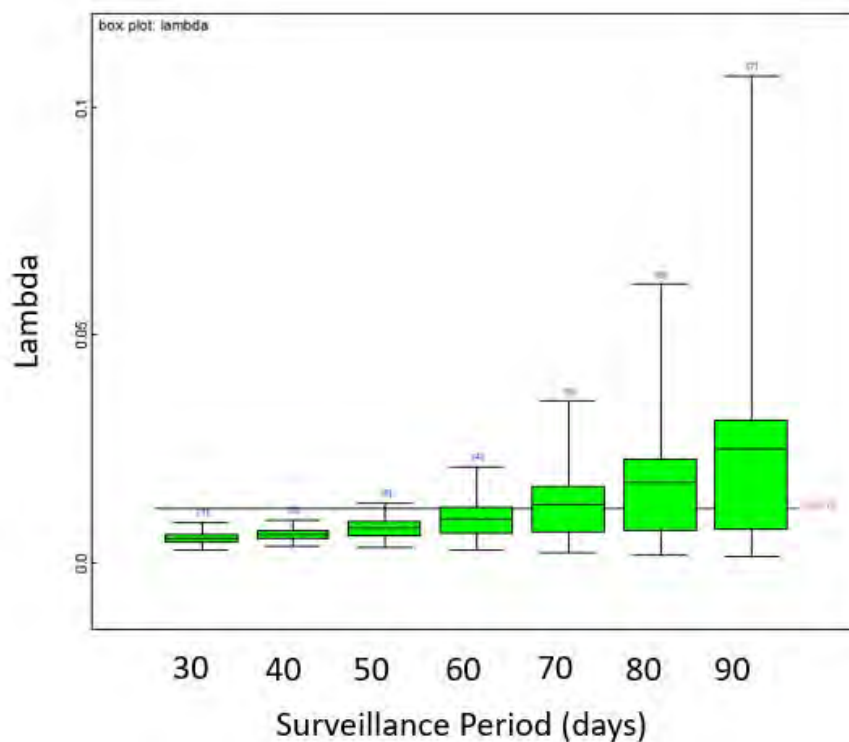
Figure 26. Box plots of the posterior distribution for the component probability of failure to run for different surveillance intervals.

6. RELIABILITY MODELING

Current reliability models are based on Boolean logic structures [40] (e.g., fault trees [FTs]), which describe the deterministic functional relationship between SSCs and human interventions. Each basic event in a reliability model represents a specific elemental occurrence (failure of a component, failure to perform an action by the plant operators, recovery of a safety system, etc.), and a probability value is associated with each basic event, which represents the probability that the basic event can occur. However, maintenance and surveillance operations are typically not explicitly integrated into a PRA structure. Therefore, a probability value associated with a basic event is a representation of the past operational experience for associated component, and it does not incorporate information about the SSC's present

health status (i.e., from diagnostic and condition-based data) and health projections (when available from prognostic data) on anticipated changes in SSC's condition and performance in the near future.

A possible alternate path can start by redefining the word "reliability" to encompass a broader meaning that better reflects the needs of a system health and asset management decision making process. Rather than focusing on how likely an event is to occur (in probabilistic terms), we think in terms of how far this event is from occurring. This new interpretation of risk transforms the concept from one that focuses on the probability of occurrence to one that focuses on assessing how far away (or close) an SSC is to an unacceptable level of performance or failure. This transformation has the advantage that it provides a direct link between the SSC health evaluation process and standard plant processes used to manage plant performance (e.g., the plant maintenance and budgeting processes). The transformation also places the question into a form that is more familiar and readily understandable to plant system engineers and decision makers. When dealing with condition-based data (actual and past/archived data), margin $\widetilde{M}$ is defined here as the distance between SSC conditions (e.g., oil temperature, vibration spectrum) that lead to failure and the current observed SSC condition (see Figure 27).
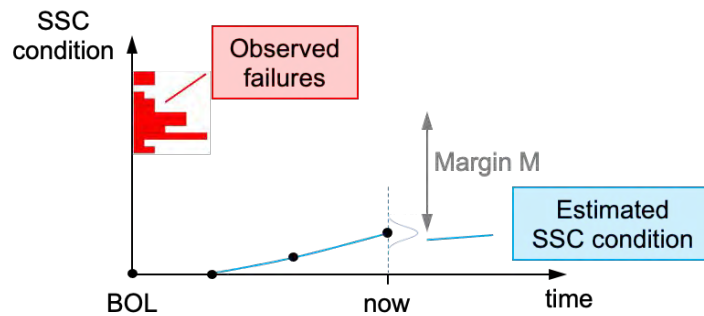
Figure 27. Margin in a condition-based maintenance context: evolution of an SSC condition as a function of time and margin definition.

Consider two components (A and B). The $\widetilde{M}$ for both components can be visualized in a 2-dimensional space, as shown in Figure 28. Starting with brand-new components (i.e., $\widetilde{M}_A, \widetilde{M}_B = 1$), aging degradation that affects both can be represented by the blue line in Figure 28, which parametrically represents the combination of the normalized margins $(\widetilde{M}_A(t), \widetilde{M}_B(t))$ as a point in time t. Note that if no maintenance (whether preventive or corrective) was ever performed on either component, this path would move from the coordinates (1,1), components A and B at the beginning of life, to the coordinates (0,0) where both components had failed. We can identify these regions in Figure 28: the occurrence of both events where $\widetilde{M}_A = 0$ and $\widetilde{M}_B = 0$ and the occurrence of either event when $\widetilde{M}_A = 0$ or $\widetilde{M}_B = 0$. Now we can calculate the $\widetilde{M}$ for the events listed above. This is accomplished by following the definition of margin: by measuring the distance between the actual condition of components A and B and $\widetilde{M}$ conditions identified by the event under consideration (e.g., the occurrence of both or either events):

$$\widetilde{M}(A\ AND\ B) = dist[(\widetilde{M}_A, \widetilde{M}_B), (0,0)]$$
$$\widetilde{M}(A\ OR\ B) = min(\widetilde{M}_A, \widetilde{M}_B)$$

(6)

The function $dist[X, Y]$ is designed to calculate the Euclidean distance between points X and Y.

Hence, exact solutions can be obtained extremely fast. More precisely, reliability calculations using $\widetilde{M}$-based data can be performed by completing these four steps:

1. Construct the FT; at this point, an FT contains only deterministic information about the architecture of the system under consideration (i.e., it simply models how the basic events are related to each other from a functional perspective).

2. Generate the minimal cut sets (MCSs) from the FT; as also indicated in Step 1, an MCS still represents the minimal combinations of basic events (BEs) that lead to the top event (TE).

3. Assign $\widetilde{M}$ to each basic event.

4. Calculate the $\widetilde{M}$ of the union of the MCSs.

As part of system reliability modeling, it is always important to determine the importance of each basic event. In a PRA setting, this is performed by relying on risk importance measures, such as Birnbaum or Fussell-Vesely. Given the different nature of $\widetilde{M}$, it is possible to perform a risk importance ranking by relying on a classical sensitivity measure (derivative based) for each basic event BE defined as: $S_{BE} = \frac{\partial \widetilde{M}(TE)}{\partial \widetilde{M}(BE)}$. In other words, S_{BE} indicates how a small variation of $\widetilde{M}(BE)$ directly affects $\widetilde{M}(TE)$.
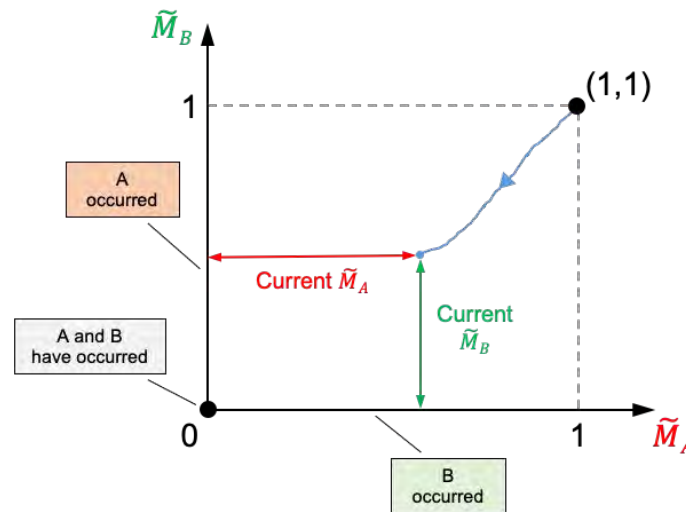
Figure 28. Graphical representation of event occurrences based on a margin framework.

6.1   Margin Models

A margin model described here is intended to be the front-end computational model which provides a numeric margin value given the set of historic and current ER data (e.g., generated by plant condition-based (CB) or plant health management [PHM] program). Without loss of generality, the format of CB or PHM data that are gathered at a specific time instant can be:

- Static: point value measurement of the SSC parameter of interest (e.g., pump shaft temperature)

- Time dependent: time series measurement of the SSC parameter of interest (e.g., vibration measurement).

The initial set of models that have been developed focus mainly on the first class (i.e., static data); however, time-dependent data can be translated into a series of static data points (e.g., through fast Fourier decomposition or statistical moment decomposition).

The definition of margin for a generic component cannot be generalized since it depends on the component class under consideration and the type of CB or PHM data that can be gathered (see Table 16).

Table 16. Example of CB/PHM data for centrifugal pumps.

CB/PHM data	Data format
Bearing temperature	Numeric scalar
Shaft vibration data	Spectrum data
NPSH available	Numeric scalar
Visual inspection	Images, sound recordings

6.1.1   Margin Calculation from Static Data

This section presents the mathematical description behind the calculation of margin provided actual data and data representing failing conditions. We are using the following notation for these two datasets:

- Actual data: R data points $\boldsymbol{y}_r \subseteq \mathbb{R}^N (r = 1, \dots, R) \rightarrow \boldsymbol{Y} = [\boldsymbol{y}_1; \dots; \boldsymbol{y}_R]$ is a matrix of dimension $R \times N$

- Failure data: Q data points $\boldsymbol{x}_q \subseteq \mathbb{R}^N (q = 1, \dots, Q) \rightarrow \boldsymbol{X} = [\boldsymbol{x}_1; \dots; \boldsymbol{x}_Q]$ is a matrix of dimension $Q \times N$.

Here we are assuming that we are dealing multiple data points for both failure and actual data (i.e., multiple measurement performed for the same SSC parameter of interest).

The goal now is to determine the SSC margin $M(\boldsymbol{Y}|\boldsymbol{X})$ given actual data $\boldsymbol{Y}$ and failure data $\boldsymbol{X}$: $M(\boldsymbol{Y}|\boldsymbol{X}) = \langle \boldsymbol{X} - \boldsymbol{Y} \rangle$. The actual algorithm is described in detail below.

Algorithm 4. Margin calculation for static data given actual data Y and failure data X.

1. Determine a matrix $\Delta = [d_{q,r}]$ $(R \times Q)$
    a. $d_{q,r} = dist(\boldsymbol{x}_q - \boldsymbol{y}_r)$
    b. Each element $d_{q,r}$ represents the <u>distance</u> between failed data point and actual data point (Euclidean distance)
    c. If $d_{q,r} < 0$, then actual data point $\boldsymbol{y}_r$ has passed the failed data point $\boldsymbol{y}_r$, in other words margin = 0
2. Based on 1.c, define $\tilde{\Delta} = [\tilde{d}_{q,r}]$ such that
$$\tilde{d}_{q,r} = \begin{cases} 0 & \text{if } d_{q,r} < 0 \\ d_{q,r} & \text{if } d_{q,r} \geq 0 \end{cases}$$
    a. Each element $\tilde{d}_{q,r}$ represents now the <u>margin</u> the between failed data point and actual data point (compare to 1.b)
3. Determine margin between $M(\boldsymbol{Y}) = mean(\tilde{\Delta})$

The margin calculation shown above has been coded inside SR²ML as part of the `MarginModel`. The code snippet shown in Figure 29 provides an example of definition of such model within the RAVEN input file where the margin is defined in a 2-dimensional space where failure times (e.g., failTime) are recorded for different values of operational temperature conditions (e.g., failTemp). Given actual lifetime and current

temperature operational conditions (e.g., actualTime and actualTemp, respectively), the model produced the margin value according to Algorithm 1.

```xml
<Models>
 <ExternalModel name="PointSetMargin" subType="SR2ML.MarginModel">
  <variables>actualTime,actualTemp,marginPS1</variables>
  <MarginModel type="PointSetMarginModel">
   <failedDataFileID>failureData.csv</failedDataFileID>
   <marginID>marginPS1</marginID>
     <map var='failTime'>actualTime</map>
     <map var='failTemp'>actualTemp</map>
  </MarginModel>
 </ExternalModel>
</Models>
```

Figure 29. Margin model input file.

6.1.2   Margin Analysis Code

The actual implementation of a computational engine that can perform margin-based reliability analysis has been completed during FY 2021. The code is based on the MarginSolver model available in SR^{2}ML. This model requires:

- The set of minimal cut sets which can be generated by existing PRA codes such as CAFTA [41], RISK SPECTRUM [42], or SAPHIRE [43]

- The margin associated to each basic event.

This model generates in output the margin value associated to the union of the provided minimal cut sets. An example of margin-based solver input structure is indicated in Figure 30.

```xml
<Models>
 <ExternalModel name="MCSmodel"subType="SR2ML.MCSSolver">
  <variables>statA,statB,statC,statD,statE,TOP</variables>
  <solver type='margin'>
   <metric>2</metric>
  </solver>
  <topEventID>TOP</topEventID>
  <map var='statA'>A</map>
  <map var='statB'>B</map>
  <map var='statC'>C</map>
  <map var='statD'>D</map>
  <map var='statE'>E</map>
 </ExternalModel>
</Models>
```

Figure 30. Margin-based solver input file.

The component margin models (see Section 6.1.1) can be directly linked to the MarginSolver model as part of a multi-entity model using the RAVEN EnsembleModel capability.

6.1.3   Cut Sets vs. Path Sets for Margin Analysis

In the context of margin-based reliability modeling, rather than considering the system cut sets, it would be more suited to rely on the concept of path sets [44]. From a reliability standpoint, these two concepts are

strongly related to each other. In a system composed by an arbitrary set of elements, it is possible to define these terms as follows:

- *Cut set*: it represents a subset of components of a system that, when failed, cause the overall system to fail

- *Path set*: it represents a subset of components of a system that, when functioning, guarantee the overall system to function.

In other terms, while cut sets focus on ways that the system can fail, path sets focus on ways that the system can operate correctly (i.e., success paths).

Assuming the system is composed by a set of N components where each component i is characterized by the state variable s_i as follows [45]:

$$s_i = \begin{cases} 1 & \text{if the component is operating} \\ 0 & \text{if the component has failed} \end{cases} \tag{7}$$

The system state vector $\boldsymbol{s}$ is here defined as:

$$\boldsymbol{s} = (s_1, s_2, \dots, s_N) \tag{8}$$

It is now possible to define the system state $\Phi(\boldsymbol{s})$ as:

$$\Phi(\boldsymbol{s}) = (s_1, s_2, \dots, s_N) \tag{9}$$

where:

$$\Phi(\boldsymbol{s}) = \begin{cases} 1 & \text{if the system is operating} \\ 0 & \text{if the system has failed} \end{cases} \tag{10}$$

From a reliability modeling perspective, $\Phi(\boldsymbol{s})$ is typically constructed by employing fault trees. By employing Boolean logic operations, the minimal cut sets can be obtained. Note the $\Phi(\boldsymbol{s})$ describes from a functional perspective the component dependencies at the system level. Now, the generation of minimal path sets can be constructed from $\Phi(\boldsymbol{s})$ by solving NOT[$\Phi(\boldsymbol{s})$].

The concept of margin is designed to measure the health of an SSC; hence, it focuses on the operability aspect of such SSC. When focusing on continuously operating systems (e.g., secondary side of NPPs), it is relevant, from a decision making point of view, to identify the ways that guarantee the system to work reliably. Hence, path sets coupled with margin-based calculations are more suitable for analyses under these conditions.

6.2 Plant Models: VERT

VERT is a generation risk assessment (GRA) software application available in the GitLab repository focused on simplifying, while improving, the process of evaluating NPP electricity generation (see Figure 31). VERT uses GRA methods to examine how component reliability and degradation impact the ability for an NPP to generate electricity (see Figure 32 for a tree representation of the VERT repository structure). Whereas PRA uses fault trees to focus on scenarios impacting plant nuclear safety, GRA uses fault trees to focus on scenarios impacting plant electricity generation, including temporary plant power output derating (see Table 17 for a summary of VERT targeted use cases). Combining GRA with a price model allows for direct economic analyses.

VERT models require in input health information of SSCs (either probabilistic or margin data [see Section 6.1]) and they determine the risk associated with loss of power generation. From here it is then possible to identify the highest contributors (i.e., SSC) to power generation risk.
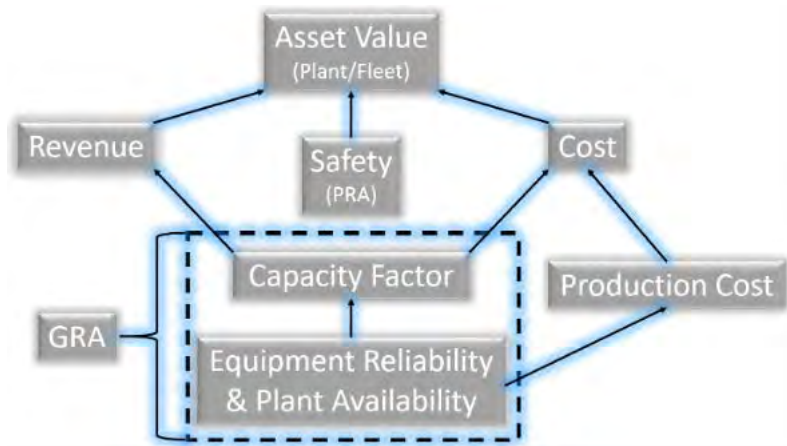
Figure 31. GRA role in power plant asset management.
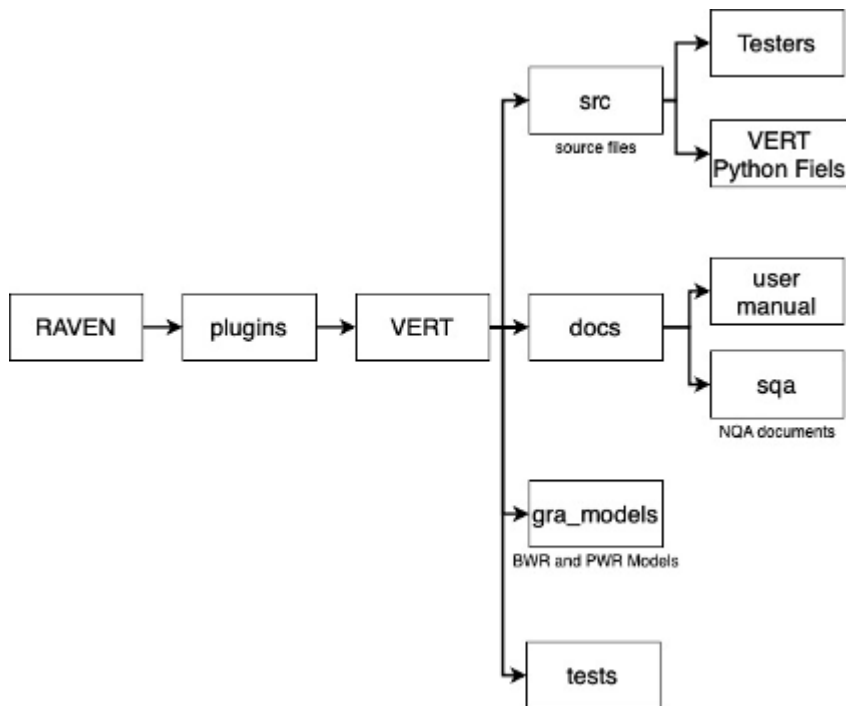
Figure 32. Folder tree of the VERT repository.

VERT couples the INL-developed SAPHIRE fault tree evaluation software and RAVEN, which provides a framework for parametric variability along with automated results production and analysis (see Figure 33). Using component reliability data and degradation models, VERT can identify SSCs that

significantly contribute to plant electricity generation loss over time (see Figure 34). The identified high-risk equipment can then be targeted for reliability and condition improvements. Similarly, VERT can be used to identify components that are not significant contributors to a reduction in electricity generation. This equipment may be subject to over-conservative reliability and condition improvement activities (e.g., preventive maintenance actions). Insights from VERT support the optimization of maintenance, inspection, and other electricity generation improvement strategies. Note that the functionality and structure of VERT have been provided in a previous report (see [5]). Table 18 provides a basic summary of the VERT repository in terms of development language, license, and GitHub repository. The VERT regression tests are in the final stages of approval to ensure quality performance of the application.
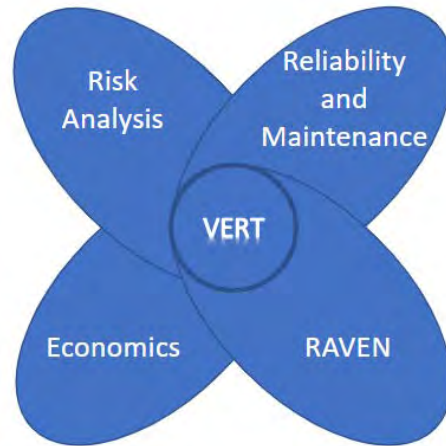
Figure 33. VERT conceptual schematic.
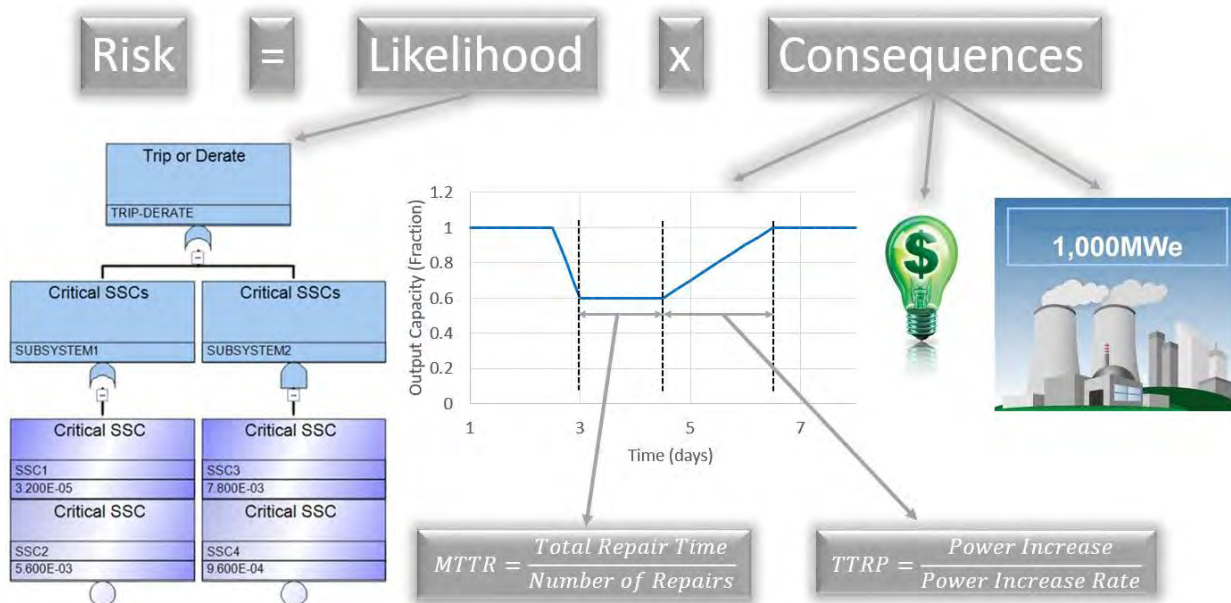
Figure 34. VERT risk methodology diagram.

VERT was used to perform an analysis on U.S. NPPs using plant operation data from 1980-2020. The analysis results confirmed that NPP operational performance has been improving since 1980. The results also identified NPP equipment that impacts plant performance, both positively and negatively. A description of the analysis and its results were presented at the 28th International Conference on Nuclear Engineering (ICONE) and published in the conference proceedings. A detailed report on the need for a generation risk assessment tool, and how VERT meets that need, will be presented at and published in the proceedings of the American Nuclear Society's 2021 International Topical Meeting on Probabilistic Safety Assessment to be held in November 2021.

A case study based on maintenance optimization strategies using VERT is currently being pursued. VERT will be utilized to provide crucial information to the processes involved with the International Atomic Energy Agency (IAEA) maintenance optimization program for NPPs. Many entities including the Electric Power Research Institute (EPRI) and operating utilities regularly implement these strategies for the support of nuclear power generation. The time and labor requirements for procedures deemed important for maintenance optimization can be reduced using VERT. For example, classification is essential for grouping components in certain maintenance strategy regimes. The classification process requires information on production losses if the components were unavailable. VERT automatically quantifies the estimated production losses of components due to failure and maintenance unavailability. The VERT case study will demonstrate this capability among others to reduce costs and increase effectiveness of maintenance optimization strategies.

Table 17. List of VERT models and methods and their corresponding use case.

Model/Method	Use Case
GUI for GRA model construction	Automated creation of PWR/BWR GRA models and interface with SAPHIRE and RAVEN
Generic LWR GRA models	Reference GRA models for generic PWR and BWR for benchmarking purposes
Generic LWR PRA models	Reference PRA models for generic PWR and BWR for benchmarking purposes

Table 18. Summary of VERT architecture.

Development language	Python
Dependencies	RAVEN, SAPHIRE
Repository site	https://hpcgitlab.inl.gov/mandd/vert
License	NDA agreement (to be released open-source soon)
Supported operating system	Windows

Two generic SAPHIRE GRA models have been developed for use with VERT. The models represent a generic boiling and pressurized water reactor (BWR and PWR respectively) NPP simplified at the super-component level. The super-component approach groups components within a system to represent a single basic event. For example, all main feedwater piping is grouped into the same super-component. The models include all SSCs that are required for power generation in LWRs. The fault tree structures were developed to represent NPP layouts. Using common PRA methodologies, the generic models include the SSCs present

in the reactor, steam turbine, generator, and balance of plant systems of NPPs. These main systems are then broken down into their constituent sub-systems (see Figure 35 for balance of plant sub-systems) and then into the super-components. The GRA models have 134 fault trees, 1535 basic events, and 171 gates. Table 19 provides a listing of the system fault trees and linkage for the VERT generic GRA models. The BWR and PWR fault trees are structured differently according to their representative plant design types.
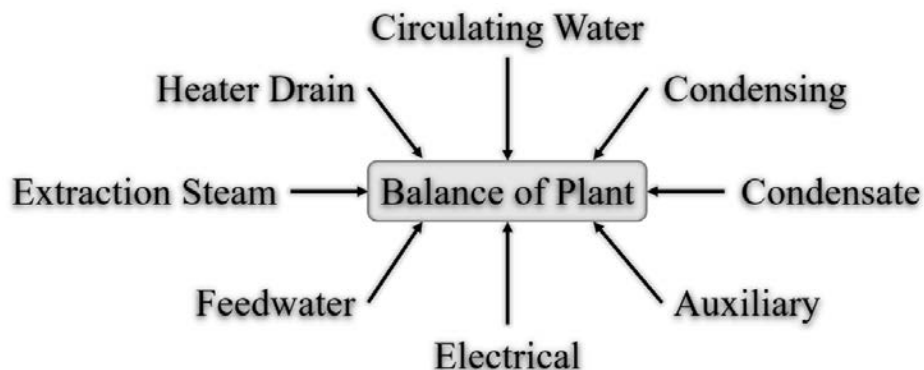
Figure 35. Balance of plant sub-systems in generic LWR GRA models.

Table 19. Summary of VERT fault trees for generic GRA models.

Fault tree ID	System	Description	Link to fault trees
2010-2090	Reactor	Core and Fuel	Input to reactor fault tree and top event to cause codes 2010-2090
2010-2999	Reactor	Failure of Nuclear Reactor	Input to system fault tree and top event to all other reactor fault trees
2110-2160	Reactor	Control Rods and Drives Failures	Input to reactor fault tree and top event to cause codes 2110-2160
2170-2199	Reactor	Reactor Vessel and Internals Failures	Input to reactor fault tree and top event to cause codes 2170-2199
2200-2399	Reactor	Reactor Coolant System Failures	Input to reactor fault tree and top event to cause codes 2200-2399
2400-2599	Reactor	Steam Generators and Steam System Failures	Input to reactor fault tree and top event to cause codes 2400-2599
2600-2649	Reactor	Core Cooling/Safety Injection Failures	Input to reactor fault tree and top event to cause codes 2600-2649
2650-2699	Reactor	Electrical Safety Systems Failures	Input to reactor fault tree and top event to cause codes 2650-2699
2700-2799	Reactor	Containment System	Input to reactor fault tree and top event to cause codes 2700-2799
2805-2819	Reactor	Chemical and Volume Control/Reactor Water Cleanup	Input to reactor fault tree and top event to cause codes 2805-2819

Fault tree ID	System	Description	Link to fault trees
2820-2839	Reactor	Nuclear Cooling Water Systems	Input to reactor fault tree and top event to cause codes 22820-2839
2840-2890	Reactor	Auxiliary Systems (Reactor)	Input to reactor fault tree and top event to cause codes 2840-2890
2900-2999	Reactor	Miscellaneous (Reactor)	Input to reactor fault tree and top event to cause codes 2900-2999
3110-3199	Balance of Plant	Condensing System	Input to balance of plant fault tree and top event to cause codes 3110-3199
3110-3999	Balance of Plant	Balance of Plant Failures	Input to system fault tree and top event to all other balance of plant fault trees
3210-3299	Balance of Plant	Circulating Water Systems	Input to balance of plant fault tree and top event to cause codes 3210-3299
3310-3399	Balance of Plant	Condensate System	Input to balance of plant fault tree and top event to cause codes 3310-3399
3401-3499	Balance of Plant	Feedwater System	Input to balance of plant fault tree and top event to cause codes 3401-3499
3501-3509	Balance of Plant	Heater Drain Systems	Input to balance of plant fault tree and top event to cause codes 3501-3509
3520-3529	Balance of Plant	Extraction Steam	Input to balance of plant fault tree and top event to cause codes 3520-3529
3600-3689	Balance of Plant	Electrical	Input to balance of plant fault tree and top event to cause codes 3600-3689
3810-3899	Balance of Plant	Auxiliary Systems	Input to balance of plant fault tree and top event to cause codes 3810-3899
3950-3999	Balance of Plant	Miscellaneous (Balance of Plant)	Input to balance of plant fault tree and top event to cause codes 3950-3999
4000-4099	Steam Turbine	High-Pressure Turbine	Input to steam turbine fault tree and top event to cause codes 4000-4099
4000-4499	Steam Turbine	Steam Turbine Failures	Input to system fault tree and top event to all other steam turbine fault trees
4100-4199	Steam Turbine	Intermediate Pressure Turbine	Input to steam turbine fault tree and top event to cause codes 4100-4199
4200-4250	Steam Turbine	Low Pressure Turbine	Input to steam turbine fault tree and top event to cause codes 4200-4250
4260-4269	Steam Turbine	Valves	Input to steam turbine fault tree and top event to cause codes 4260-4269
4270-4279	Steam Turbine	Piping	Input to steam turbine fault tree and top event to cause codes 4270-4279
4280-4289	Steam Turbine	Lube Oil	Input to steam turbine fault tree and top event to cause codes 4280-4289
4290-4309	Steam Turbine	Controls	Input to steam turbine fault tree and top event to cause codes 4290-4309

Fault tree ID	System	Description	Link to fault trees
4400-4499	Steam Turbine	Miscellaneous (Steam Turbine)	Input to steam turbine fault tree and top event to cause codes 4400-4499
4500-4580	Generator	Generator	Input to generator fault tree and top event to cause codes 4500-4580
4500-4899	Generator	Generator Failures	Input to system fault tree and top event to all other generator fault trees
4600-4609	Generator	Exciter	Input to generator fault tree and top event to cause codes 4600-4609
4610-4650	Generator	Cooling System	Input to generator fault tree and top event to cause codes 4610-4650
4700-4750	Generator	Controls	Input to generator fault tree and top event to cause codes 4700-4750
4800-4899	Generator	Miscellaneous (Generator)	Input to generator fault tree and top event to cause codes 4800-4899
SYSTEM	Entire System	BWR or PWR Power Plant Failures	Top event for the GRA model

7. PLANT RESOURCES MANAGEMENT

The last step in an integrated plant system and asset health management program is to manage plant resources based on the system health analysis indicated in Sections 4 and 5.

7.1 Project/Option Selection

The FMs with higher S_{BE} (see Section 6) are the ones selected as candidates to be subject to MAs (see Figure 6). A list of possible options to address each failure mode is available where costs (i.e., procurement costs for a new or refurbished component) and benefits (i.e., increased margin for loss of production) are readily available or can be numerically determined. Given the candidate MAs and their options, we can now identify the best set of activities and options that give "the most bang for the buck."

This is accomplished by identifying the Pareto frontier [46] out of all the possible MAs and options. Let us assume that a decision can be taken from a set of options by considering the utility and cost of each option. Using a graphical representation (see Figure 36) it is possible to plot each option as a point in a 2-dimensional space, cost versus utility[11]:

- *Cost*. This axis represents the cost associated with each option ranging from 0 (e.g., cheapest option) to a maximum value C_{max} (e.g., the most expensive option)

- *Utility*. This axis represents the added value (or the performance) associated with each option ranging from 0 (i.e., lowest performance option) to a maximum value U_{max} (i.e., option with highest performance).

---

[11] As indicated earlier, the number of attributes considered in complex settings can be $N > 2$. Thus, in such cases, the space would be N-dimensional.
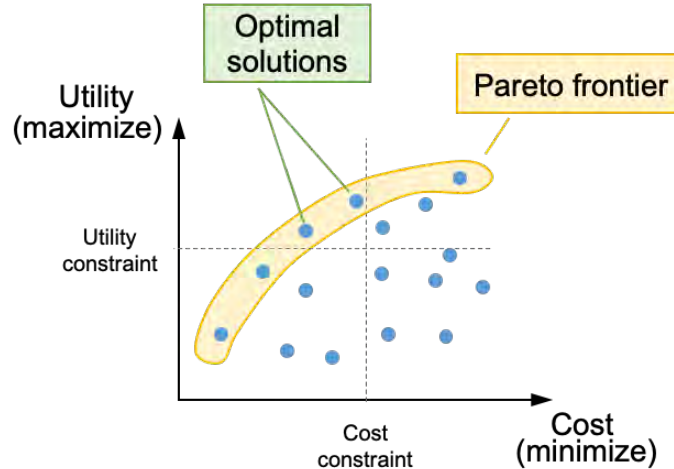
Figure 36. Pareto frontier obtained from a set of options plotted in a cost vs. utility space and imposition of cost and utility constraints (right).

Once the complete set of options has been generated and the utility and cost values have been determined for each option, the next step is to the determine the Pareto optimal frontier, which is fundamentally an envelope of options that dominates (in terms of both utility and cost) the set of remaining options (see Figure 36).

Multi-objective optimization is widely used to design plant/system configuration by balancing cost and performances. An application of the developed tools is indicated in Section 7.1.1 to determine the optimal sensor configuration applied to a generic SSC.

7.2 Long-Term Decisions: Project Scheduling Given Budget Constraints

The method described in Section 7.1 does not explicitly take into consideration project actuation scheduling but instead focuses on the optimal subset of projects that provide higher value through a multi-objective optimization lens. In practical settings, project scheduling is done in phases (e.g., monthly, quarterly) wherein each phase's budget is allocated, and the goal now is to choose an optimal project actuation schedule that minimizes costs and satisfies budget constraints [47]. Due to limited resources, we can only select a subset from a list of several candidate capital projects. Our goal is to maximize overall NPV associated with the selected subset. In doing so, we must respect resource limits and capture key structural and stochastic dependencies of the system, although here we start with the simpler deterministic case, ignoring randomness. The notation and formulation are as follows:

Indices and sets:

$i \in I$	candidate projects
$j \in J_i$	options for selecting project i (e.g., initiate project i in year t or $t + 2$ and in a standard (three year) or in an expedited (two year) manner)
$(i', j') \in IJ_{ij}$	piggybacking situations, i.e., option j' for project i' can be selected only if option j is selected for project i
$k \in K$	types of resources, e.g., capital funds, O&M funds, labor-hours, time during outage
$t \in T$	time periods (years)

50

Data:

a_{ij} reward (revenue less financial cost) of selecting project i via option j

b_{kt} available budget for a resource of type k in year t

c_{ijkt} consumption of resource of type k in year t if project i is performed via option j

Decision variables:

$$x_{ij} = \begin{cases} 1 & \text{if project } i \text{ is selected} \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

Optimization model formulation:

$$\max_x \sum_{i \in I, j \in J_i} a_{ij}\, x_{ij} \tag{12}$$

$$\text{s.t.} \quad \sum_{j \in J_i} x_{ij} = 1, i \in I \tag{13}$$

$$\sum_{i \in I, j \in J_i} c_{ijkt}\, x_{ij} \leq b_{kt}, \quad k \in K, t \in T \tag{14}$$

$$x_{i'j'} \leq x_{ij}, \quad (i', j') \in IJ_{ij}\,, j \in J_i, i \in I \tag{15}$$

$$x_{ij} \in \{0,1\}, j \in J_i\,, i \in I \tag{16}$$

The decision variables, x_{ij}, indicate whether we choose to do project i by means j. The set of available options, $j \in J_i$, can explicitly include the "do-nothing" option, and the first constraint ensures that we choose exactly one option from the available set for each project, including the possibility of selecting the do-nothing option. The second structural constraint ensures that the budget of each resource k is respected in each year t. The third structural constraint captures piggybacking situations in which option j' for project i' (which may have cheaper costs) may be selected only if project-option pair (i, j) is also selected. The objective function includes the NPV for each project-option pair, a_{ij}, and the correct NPV is selected by the 0-1 decision variable, x_{ij}.

7.3   Short-Term Decisions: Maintenance Activity Scheduling Given Personnel Constraints

NPPs have personnel trained and tasked to perform certain maintenance jobs. A plant manager is responsible for all aspects of safe and reliable station operation. A plant's maintenance department includes mechanical, electrical, and instrumentation and control technicians. A plant's operations department operates the reactor and other plant controls. Within the context of maintenance, staff from operations authorize work to commence, align systems for work, and provide for worker and system safety during maintenance by tagging components (e.g., breakers, valves, and dampers) to prevent their operation as well as to isolate them from other systems (e.g., electrical power and pressurized fluid systems). Operators are frequently the first to identify deficiencies in equipment. Because operators best understand system interactions and the relative importance of each piece of equipment, they have a strong voice in prioritizing such deficiencies for work. Staff from scheduling coordinate work efforts and interface with different organizations at the plant. Generally, scheduling occurs in several phases. A long-range schedule, which provides general opportunities for maintenance on pre-specified systems, is used for rough planning.

Several weeks before work starts, a finer-grain job schedule is developed to allow parts to be delivered, tags to be prepared, and scaffolds and other support work to be organized. Finally, a detailed schedule, which includes work sequences to coordinate the jobs with other station jobs, is developed. Schedulers are sometimes separated into two teams for online and outage scheduling. The work control group administers much of the maintenance process by controlling the flow of documentation, from problem identification to work completion.

An important part of the efficient functioning of the described groups is having an optimal schedule for executing jobs. We consider a relatively general problem of job scheduling involving multiple modes of operation, multiple resources, and alternative objectives, all in the context of scheduling maintenance or surveillance activities during normal operation, or scheduling activities during an outage. We will refer to all such activities as "jobs."

7.3.1 Full Version of the Model Formulation

We consider a project that consists of a set of J jobs (or tasks). Due to technological requirements, precedence relations among some of the jobs enforce that job $j = 2, 3, \dots, J$ may not be started before all its predecessors, denoted by $\mathcal{P}_j$, are finished. Here, $j = 1$ indexes an artificial job with zero duration, which precedes all jobs that can start at time zero, and $j = J$ indexes an artificial final job, again with zero duration, which represents the end of the project. (For work on plant SSCs, one could consider these to represent j = 0 as the point in time at which Operations authorizes work to comments while j = J could be considered as the point in time where Operations declares the SSC as being available for service.) The individual job, j, may be executed in one of $m = 1, \dots, M_j$ modes, and the solution to the optimization model recommends which mode should be selected. The jobs may not be preempted, and a mode, once selected, may not change. Executing job j takes d_{jm} time periods and is supported by sets, R and N, of renewable and nonrenewable resources. Consider a horizon with an upper bound, $\bar{T}$, on the project's makespan (i.e., the time at which the final job is completed). We assume K_r^ρ units of renewable resource $r \in R$, are available in each time period $t = 1, 2, \dots, \bar{T}$; an example of a renewable resource is a crew's availability for up to 40 hours per week. The overall capacity of the nonrenewable resource $r \in N$, is given by K_r^v; an example of a nonrenewable resource may be a budget that spans the planning period.

Job j requires k_{jmr}^ρ units of the renewable resource, $r \in R$, for each period of the job's duration (i.e., for time periods when the job is in process). For a nonrenewable resource r, k_{jmr}^v units of the resource are consumed when job j is done in mode m.

The objective is to find a schedule which minimizes the project's makespan while respecting the constraints imposed by the precedence relations and the limited resource availability. (Note that the makespan of a project is defined as the length of time that elapses from the start of work to its completion.) The following model is used for the optimization analysis.

Indices and parameters:

$j = 1, 2, \dots, J$ jobs, with $j = 1$ and $j = J$ denoting artificial jobs

M_j number of modes in which activity j can be performed

$r \in R(N)$ set of renewable (nonrenewable) resources

d_{jm} duration of job j being performed in mode m; integer number of time periods

$K_r^\rho \geq 0$ number of units of renewable resource r available in period t

$$k_{jmr}^{\rho} \geq 0 \quad \text{number of units of renewable resource } r \text{ consumed by job } j \text{ each period while in process}$$

$$K_r^{\upsilon} \geq 0 \quad \text{number of units of non renewable resource } r \text{ available in period } t$$

$$k_{jmr}^{\upsilon} \geq 0 \quad \text{number of units of non renewable resource } r \text{ consumed by job } j \text{ while in process}$$

$$\mathcal{P}_j \quad \text{set of immediate predecessors of job } j$$

$ES_j(EF_j)$ earliest start time (finish time) of job j; if not otherwise specified it calculated by using minimal activity durations and neglecting resource usage (consumption)

$LS_j(LF_j)$ latest start time (finish time) of job j; if not otherwise specified it calculated by using minimal activity durations and neglecting resource usage (consumption) and taking into account the upper bound $\bar{T}$ on the project's duration

$t = EF_j, \dots, LF_j$ time periods, with $\bar{T}$ as an upper bound on the project's makespan

Variables:

Binary decision variables $x_{jmt}, j = 1, \dots, J; m = 1, \dots, M_j, t = EF_j, \dots, LF_j$

$$x_{jmt} = \begin{cases} 1, \text{ if job } j \text{ } completes \text{ in period } t \text{ under mode } m \\ 0, \text{ otherwise} \end{cases}$$

Model:

$$\min \sum_{m=1}^{M_J} \sum_{t=EF_J}^{LF_J} (t-1)x_{Jmt} \tag{17}$$

$$\text{s.t.} \quad \sum_{m=1}^{M_j} \sum_{t=EF_j}^{LF_j} x_{jmt} = 1, j = 1, \dots, J \tag{18}$$

$$\sum_{m=1}^{M_h} \sum_{t=EF_h}^{LF_h} t\, x_{hmt} \leq \sum_{m=1}^{M_j} \sum_{t=EF_j}^{LF_j} (t - d_{jm})x_{jmt}, j = 2, \dots, J, h \in \mathcal{P}_j \tag{19}$$

$$\sum_{j=1}^{J} \sum_{m=1}^{M_j} k_{jmr}^{\rho} \sum_{q=max\{t,EF_j\}}^{min\{t+d_{jm}-1,LF_j\}} x_{jmq} \leq K_r^{\rho}, r \in R, t = 1, \dots, \bar{T} \tag{20}$$

$$\sum_{j=1}^{J} \sum_{m=1}^{M_j} k_{jmr}^{\upsilon} \sum_{t=EF_j}^{LF_j} x_{jmt} \leq K_r^{\upsilon}, r \in N \tag{21}$$

$$x_{jmt} \in \{0,1\}, j = 1, \dots, J; m = 1, \dots, M_j, t = EF_j, \dots, LF_j \tag{22}$$

Note that the objective function in Eq. 17 uses job J to define the project's makespan because this job is the artificial final job with zero duration, which denotes completion of the project. Constraint in Eq. 18 ensures that each job is done exactly once (i.e., in exactly one mode and with exactly one completion time within its time window $[EF_j, LF_j]$). Constraint in Eq. 19 is the precedence relation; the left-hand side of the constraint denotes the time at which the predecessor job finishes, and the right-hand side of the constraint is the start time of the successor job. Constraint in Eq. 20 ensures that the per-period availabilities of the renewable resources are not exceeded. Constraint in Eq. 21 secures feasibility with respect to consumable (nonrenewable) resources; such a constraint only makes sense because there are multiple modes for each job. When $M_j = 1, j = 1, \dots, J$ and $|N| = 0$, the multimode problem degenerates to the single-mode resource-constrained project scheduling problem. Even this simplified variant is an NP-hard problem (i.e., the current consensus suggests that it is unlikely that it can be solved in polynomial time).

The above model, with one mode only, is used by McKendall et al. [48] to solve a nuclear plant's outage scheduling problem with eight maintenance jobs (two artificial) and nine toolboxes. This toy-sized problem is used to simply illustrate ideas. The objective used by McKendall et al. is to schedule maintenance activities such that outage duration (i.e., the project makespan) is minimized. In their example, they identify the most constraining resources as: polar crane, toolboxes, skilled workers, and space. In the considered example, toolboxes and workspaces turn out to be the limiting resources. Each job (activity) requires one unit of space, and three total workspaces are available. Table 20 shows additional data for the problem. Early start/finish time (EST, EFT) and late start/finish time (LST, LFT) values along with most total successors (MTS) are computed using the critical path method (CPM). Figure 37 provides another means of visualizing the problem.

Table 20. A simple outage scheduling problem.

Job	Duration	Immediate Predecessors	Toolboxes	EST	EFT	LST	LFT	MTS
1	0	None	None	1	1	15	15	7
2	2	1	1,5,8	1	3	17	19	3
3	2	1	1,4,6	1	3	17	19	3
4	8	1	2,6,9	1	9	15	23	2
5	4	2,3	6,7	3	7	22	26	1
6	7	2,3	1,2,8	3	10	19	26	1
7	3	4	3,5,9	9	12	23	26	1
8	0	5,6,7	None	12	12	26	26	0
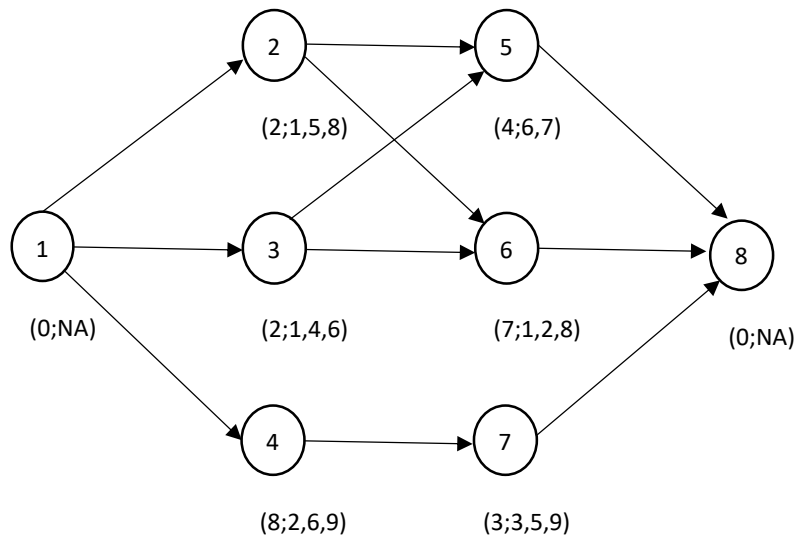
Figure 37. Graphical visualization of the eight-job project from Table 20.

Each node corresponds to a job, with Jobs 1 and 8 being artificial. Each node has a label that corresponds to: (duration; list of up to three toolboxes). The precedence relationships are specified by directed edges in the acyclic graph; for example, Jobs 2 and 3 must be completed prior to Job 5 starting. With respect to precedence relations, Jobs 2 and 3 would start in the first time period, but both require

Toolbox 1; hence, they must be processed in a series. The toolboxes represent nine renewable resources, and a tenth renewable resource is space, which allows at most three jobs to be processed simultaneously.
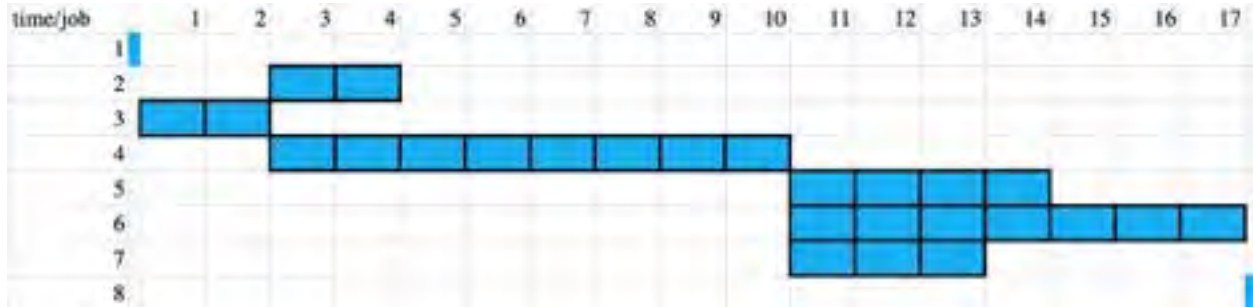
Figure 38. Optimal schedule of the eight-job project from Table 20 and Figure 37.

Each column corresponds to a time period; Jobs 1 and 8 have zero duration (even if they are depicted by slivers); hence, Job 3, which has duration 2, is performed during periods 1 and 2. Job 3 use shares a toolbox with each of Jobs 2 and 4, and hence they both start in period 3. The eight-period Job 4 shares a toolbox with Jobs 5, 6, and 7, and hence they start when Job 4 completes. The makespan is a total of 17 time periods. The space constraint of having just three workspaces is not limiting. Moreover, we see that even if we only had two workspaces, we could delay the start of Job 7 until period 15 and still obtain the same makespan.

7.3.2   Simple Version of the Model Formulation

If we do not include nonrenewable resources in the formulation and for simplicity suppress the earliest and latest start times, we obtain a simpler mathematical formulation:

Indices and parameters:

$t = 1,2 \dots, \overline{T}$ time periods, where $\overline{T}$ is an upper bound on the project's makespan

$j = 1,2, \dots, J$ jobs, with $j = 1$ and $j = J$ denoting artificial jobs

$r \in R$ set of renewable resources

d_j duration of job j

$K_r^{\rho} \geq 0$ number of units of renewable resource r available in period t

$k_{jr}^{\rho} \geq 0$ number of units of renewable resource r consumed by job j while in process

$\mathcal{P}_j$ set of immediate predecessors of job j

Variables:

Binary decision variables $x_{jt}, j = 1, \dots, J; t = 1, \dots, \overline{T}$

$$x_{jt} = \begin{cases} 1, \text{if job } j \text{ } completes \text{ in period } t \\ 0, \text{otherwise} \end{cases}$$

Model:

$$\min \sum_{t=1}^{\bar{T}} (t-1)x_{Jt} \tag{23}$$

$$s.t. \quad \sum_{t=1}^{\bar{T}} x_{jt} = 1, j = 1, \dots, J \tag{24}$$

$$\sum_{t' \le t} x_{jt'} \le \sum_{t' \le t-d_j} x_{it'}, j = 2, \dots, J; t = 1, \dots, \bar{T}; \ i \in \mathcal{P}_j \tag{25}$$

$$\sum_{j=1}^{J} \sum_{t'=t}^{t+d_j-1} k_{jr}^{\rho} x_{jt'} \le K_r^{\rho}, r \in R; t = 1, \dots, \bar{T} \tag{26}$$

$$\in \{0,1\}, j = 1, \dots, J; t = 1, \dots, \bar{T} \tag{27}$$

Note that we again use a J subscript on x_{Jt} in the objective function (see Eq. (23)), which indicates the completion time of the *final* (artificial) job. If that completion were to occur in the first time period, then the makespan would be zero since the artificial job has zero duration, which explains the "$t-1$" coefficient in the objective function. Constraint of Eq. (24) again simply indicates that each job j must be completed in *some* time period, $t = 1, \dots, \bar{T}$. Constraint of Eq. (25) indicates that we can complete job j in period t' only if all of its predecessors, $i \in \mathcal{P}_j$, were completed in an earlier time period, accounting for the duration of job j (i.e., accounting for the offset d_j). We depict this constraint in a different form than that of Eq. (19). Constraint of Eq. (26) is the analog of Eq. (20) and ensures that the available resources of each type in time period t are respected, and because x_{jt} denotes the period in which job j ends, we must look to future time periods to see which jobs are currently active. We note that the same formulation would allow for time-dynamic availability of resources, K_{rt}^{ρ}, and/or time-dynamic consumption of resources, k_{jrt}^{ρ}, by simply appending a time index to these parameters, informed by appropriate data. We further note that the superscript ρ on these parameters allows us to distinguish these renewable resources (e.g., a crew is available 40 hours per week) from nonrenewable resources (e.g., total budget), which may be consumed over the project's duration.

Our formulation is largely based on the formulation given in Alcaraz and Maroto [49], with a modification of the precedence constraint. Our variant of the formulation from Alcaraz and Maroto [49] is not new (see [50]), but it can have a tighter linear programming relaxation. As a result, even though it has more constraints of this type (by a factor of $\bar{T}$) it can lead to shorter solution times when solving the integer program. Many papers have investigated the resource-constrained project scheduling problem (RCPSP), including [51–56]. Early work dates back to the late 1950s; see [57] and references therein.

7.3.3 Illustration with Single-Mode Example from the Project Scheduling Library

Kolisch and Sprecher [49] describe a project scheduling library, which is available at http://www.om-db.wi.tum.de/psplib/main.html and contains a large number of problem instances of multiple varieties. It has examples for the single- and multiple-mode problems. The single-mode part of the library can be found at: http://www.om-db.wi.tum.de/psplib/getdata_sm.html. The multimode part of the library can be found at: http://www.om-db.wi.tum.de/psplib/getdata_mm.html.

File j601_1sm.txt has the original data posted on the library website. The file describes a problem with 62 jobs ($J = 62$; two jobs are artificial), 1 mode ($M_j = 1$) and 4 renewable resources, and 0 nonrenewable resources. The time horizon is 329 ($\bar{T} = 329$), and this is computed as the sum of the processing times (i.e., the worst-case makespan if the precedence and/or resource constraints are such that the projects have to be processed serially).

File resource_j601_1sm.txt lists the four renewable resources and their numerical values, see Table 21.

Table 21. List of the considered renewable resources and their numerical values.

Resource	Value
r1	13
r2	11
r3	12
r4	13

As shown in Table 21, the first resource has a value of 13, the second 11, the third 12, and the fourth has 13 units of resource, all of which are available in each time period. File precedence_j601_1sm.txt has a table with precedence relations. Table 22 below shows the first three rows.

Table 22. List of data required for each job.

Job No.	Number of modes	Number of successors	Successors
1	1	3	2, 3, 4
2	1	3	5, 10, 15
3	1	3	7, 14, 29
…	…	…	…

Job 1 has 3 successors, Jobs 2, 3, and 4. Job 2 has 3 successors, Jobs 5, 10, and 15. File duration_j601_1sm.txt has a table with job durations and how much each job consumes from the four resources.

The optimal makespan for this example is 77. An optimal solution is given in the Table 23. The first column indicates the decision variables from model (2) that take value 1 in an optimal solution. (We do not show those taking value zero.) For example, the duration of Job 2 is 8. Job 2 (labeled in the second column) is completed at the beginning of the nineth period (fourth column), i.e., it starts in period 1 (third column), and is in process for periods 1–8); hence, $x_{29} = x(2,9) = 1$, denotes the completion time of Job 2. Job 2 has 3 successors: Jobs 5, 10, and 15. Note that in the optimal solution, Job 5 starts at period 11, Job 10 in period 12, and Job 15 in period 28. All start after Job 2 was completed in period 9. Job 3 has successors 7, 14, and 19. Job 3 starts in period 1 and has a duration of 1, so that a successor can start as early as period 2, which is when Job 14 starts; Jobs 7 and 19 start later. The start times can be delayed from the earliest available because of resource limitations.

Table 23. Optimal solution for the 60 jobs problem.

	Job *j*	Start time	End time
x(1,1)	1	1	1
x(2,9)	2	1	9
x(3,2)	3	1	2
x(4,11)	4	1	11
x(5,17)	5	11	17
x(6,22)	6	17	22
x(7,61)	7	53	61
x(8,20)	8	11	20
x(9,21)	9	20	21
x(10,21)	10	12	21
x(11,34)	11	26	34
x(12,24)	12	21	24
x(13,27)	13	21	27
x(14,t4)	14	2	4
x(15,33)	15	28	33
x(16,49)	16	48	49
x(17,53)	17	50	53
x(18,37)	18	27	37
x(19,21)	19	12	21
x(20,22)	20	21	22
x(21,34)	21	31	34
x(22,50)	22	44	50
x(23,64)	23	61	64
x(24,54)	24	51	54
x(25,53)	25	46	53
x(26,48)	26	42	48
x(27,34)	27	24	34
x(28,46)	28	37	46
x(29,30)	29	22	30
x(30,38)	30	34	38
x(31,53)	31	50	53
x(32,56)	32	53	56
x(33,46)	33	40	46
x(34,11)	34	10	11
x(35,31)	35	22	31
x(36,42)	36	33	42
x(37,64)	37	63	64
x(38,33)	38	31	33
x(39,52)	39	48	52
x(40,68)	40	59	68
x(41,41)	41	31	41
x(42,50)	42	42	50
x(43,57)	43	53	57
x(44,37)	44	34	37
x(45,40)	45	34	40
x(46,59)	46	53	59

	Job j	Start time	End time
x(47,53)	47	46	53
x(48,57)	48	54	57
x(49,68)	49	66	68
x(50,48)	50	38	48
x(51,68)	51	64	68
x(52,55)	52	53	55
x(53,69)	53	68	69
x(54,59)	54	55	59
x(55,69)	55	59	69
x(56,77)	56	69	77
x(57,75)	57	69	75
x(58,75)	58	65	75
x(59,78)	59	75	78

7.3.4   Objective Functions

In Section 7.3.2, we introduced a problem that minimizes the makespan (i.e., the time by which all projects are completed). Other objectives may be appropriate for various scheduling environments. Pinedo [58] introduces a wide range of objectives for single, parallel, flow shop, etc., situations, and this textbook also considers both deterministic and stochastic versions of the underlying problem.

The simplest modification of the "minimize makespan" goal is to minimize the weighted sum of completion times across all jobs. Here, the weight, ω_j, of job j can be seen as a measure of importance. It may represent a holding cost per unit time or lost revenue as we await completion of job j. In this way, we minimize $\sum_{j=1}^{J} \sum_{t=1}^{\bar{T}} \omega_j (t-1) x_{jt}$, which reduces to minimizing makespan if $\omega_J = 1$ and the weight of all other jobs is zero. If we simply have a single resource that can only process one job per time period (i.e., a single-machine scheduling problem), then the problem can be solved by simply ordering the jobs in decreasing order of the ratio, $\frac{\omega_j}{d_j}$, where d_j is again the duration of job j. Problems with multiple resources are more challenging.

7.3.5   Heuristic Solution Approaches

If we are in the most resource-constrained setting of just one unit of resource per time period, which all jobs require, the problem reduces to sorting, which solves the problem under the objective of minimizing the weighted sum of completion times. If we have unlimited resources, i.e., no resource constraints, then the project-scheduling problem without resource constraints can be solved efficiently, if we seek to minimize makespan, via the critical path method. That problem reduces to finding the longest path in an acyclic network (see Figure 37) via very efficient (polynomial time) algorithms. The problem becomes NP-hard when nontrivial resource constraints are introduced. Being an NP-hard problem, its exact solution can be obtained for modest-sized problems that are not highly resource-constrained. In Sections I and II we illustrated an exact solution for two problems. For larger problems heuristics are the main alternative. Classical metaheuristics include genetic algorithms (GAs), tabu search, and simulated annealing.

A review paper by Kolisch and Hartmann [59] tests many heuristics algorithms on the RCPSP and identifies the best-performing approaches from the following literature: [60-65]. The best algorithms are not general-purpose genetic algorithms (GA)s but procedures where a GA has been modified or extended to exploit special structure. The above cited papers integrate ideas including path relinking, forward-backward improvement, self-adapting mechanisms, non-standard crossover techniques, and other

metaheuristics. Notably, the best-performing approach is not a meta-heuristic but a sampling method for forward-backward improvement presented in Tormos and Lova [64]. Kolisch and Hartmann [59] note that the approach in [64] leads to excellent solutions after computing schedules for problems with 1,000 jobs but its performance deteriorates when dealing with larger problems. In contrast, metaheuristics yield better solutions on large-scale problems. The reason behind this improvement is the fact that they take advantage of learning over time.

First, the initial population is generated, considering the representation of the solutions employed. Then, the schedule associated with each individual in the population is built and its objective function value is computed (e.g., each individual solution's makespan). After that the above steps are repeated until the terminating condition (execution time, number of schedules) is reached. In the selection process, each individual creates several copies depending on its fitness, such that the individuals with highest fitness create more copies. Then the individuals are mated at random, and each pair undergoes crossing to produce offspring. Finally, some individuals of the population mutate, and the population is evaluated again. The number of schedules computed in each generation is not fixed and depends on several factors (population size, crossover probability, mutation probability). The modifications of the GAs in the cited papers introduce changes in the steps described in the illustration chart in combination with other techniques published in the literature (forward-backward improvement, self-adapting mechanisms, non-standard crossover techniques, and other metaheuristics). Most of the critical developments for applying GAs in our setting can be found in:

- Alcaraz and Maroto [66]–changes in the crossover list, use of forward-backward scheduling.

- Debels et al. [61]–crossover-like operator that the authors indicate follows a rough analogy to electromagnetism and essentially consists of linear combination of solutions; forward-backward scheduling.

- Hartmann [62]–self-adapting GA.

- Kochetov and Stolyar [63]–evolutionary algorithm that combines GA, path relinking and tabu search.

- Valls et al. [65]–hybrid GA. Activity list based GA with forward-backward improvement.

- Tormos and Lova [64]–forward-backward improvement to schedules computed by sampling.

7.4   Job Scheduling Implementation in LOGOS

LOGOS was initially developed to optimize capital budgeting/SSC replacements to support risk-informed decisions in NPP operations under the RISA-RIAM project. In the past two years, the capabilities of LOGOS have been significantly enhanced. First, LOGOS capabilities were extended to handle risks under uncertainty during the asset management analysis via a two-stage stochastic optimization scheme to provide priority lists to decision makers in support of risk-informed decisions [4]. Second, both conditional value-at-risk (CVaR) optimization and distributionally robust optimization schemes have been implemented to provide more robust risk-informed asset management [7]. In FY 2021, the development of LOGOS focused on the RCPSP. In the following sections, we provide the detailed descriptions about how to use LOGOS to solve RCPSP problems.

7.4.1   Components of LOGOS Used for RCPSP problem

In LOGOS, eXtensible Markup Language (XML) format is used to create the input file. The main input blocks for RCPSP problem are as follows:

- `<Logos>`: the root node containing the entire input, all the subsequent blocks fit inside this block.

- `<Sets>`: specifies a collection of data, possibly including numeric data as well as symbolic data that is typically used to specify valid indices for indexed components.

- `<Parameters>`: specifies a collection of parameters, which are used to formulate constraints and objectives in the RCPSP model. A parameter can denote a single value, an array of values, or a multi-dimensional array of values.

- `<Settings>`: specifies the calculation settings (i.e., options for optimization solvers, options for constraints, and working directory.)

7.4.2 Sets Input Block for RCPSP

This section contains information regarding the XML nodes used to define the `<Sets>` of the RCPSP model. Figure 39 shows an example of `<Sets>` block, and the `<Sets>` node accepts the following sub-nodes:

- `<tasks>`: comma/space-separated string, specifies the valid indices for tasks/jobs.

- `<resources>`: comma/space-separated string, specifies the indices for renewable resources

- `<predecessors>`: comma/space-separated string, specifies the indices for preceding tasks

- `<successors>`: comma/space-separated string, specifies indices for successors. This sub-node accepts the following attribute:

  o index, string, specifies the index dependence. Valid index is "predecessors".

```xml
<Sets>
  <tasks>
    1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12
  </tasks>

  <resources>
    r1 r2 r3 r4
  </resources>

  <predecessors>
    1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11
  </predecessors>

  <successors index='predecessors'>
    s1 s2 s3 s4;
    s1;
    s1;
    s1;
    s1;
    s1;
    s1;
    s1;
    s1;
    s1;
    s1
  </successors>
</Sets>
```

Figure 39. Example XML for `<Sets>` input block.

7.4.3   Parameters Input Block for RCPSP

This section contains information regarding XML nodes used to define the `<Parameters>` block of RCPSP optimization model. Figure 40 shows an example of `<Parameters>` block, and the `<Parameters>` node accepts the following sub-nodes:

- `<available_resources>`: comma/space-separated string, specifies the available renewable resources. This sub-node accepts the following attribute:
  - index, string, specifies the index of this parameter. Keywords should be predefined in `<Sets>`. Valid keyword is "resources".
- `<task_resource_consumption>`: comma/space-separated string, specifies the resource consumption for each task. This sub-node accepts the following attribute:
  - index, string, specifies the indices of this parameter. Keywords should be predefined in `<Sets>`. Valid keyword is "tasks, resources".

```
<Parameters>
  <available_resources index='resources'>
    13 13 13 12
  </available_resources>
  <task_resource_consumption index='tasks, resources'>
     0   0        0        0
    10   0        0        0
     0   7        0        0
     0   9        0        0
     0   4        0        0
     0   0        0        6
    10   0        0        0
     0   0        6        0
     0   0        0        8
     0   6        0        0
     0   0        0        5
     0   0        0        0
  </task_resource_consumption>

  <task_duration index="tasks">
    0 8 1 10 6 5 8 9 1 9 8 0
  </task_duration>

  <task_successors index='successors' type='str'>
    2 3 4 9
    5
    7
    8
    6
    10
    11
    10
    12
    9
    12
  </task_successors>
</Parameters>
```

Figure 40. Example XML for `<Settings>` input block.

- `<task_duration>`: comma/space-separated string, specifies the duration for each task. This sub-node accepts the following attribute:

- o index, string, specifies the indices of this parameter. Keywords should be predefined in `<Sets>`. Valid keyword is "tasks".
- `<task_successors>`: comma/space-separated string, specifies the successors for each predecessor. This sub-node accepts the following attribute:
  - o index, string, specifies the indices of this parameter. Keywords should be predefined in `<Sets>`. Valid keyword is "successors".
  - o type, string, specifies the text type of the provided node. Valid keywords are "int", "float" or "str".

7.4.4   Settings Input Block for RCPSP

This section contains information regarding the XML nodes used to define the `<Settings>` of RCPSP model. Figure 41 shows an example of `<Settings>` block, and the `<Settings>` node accepts the following sub-nodes:
- `<problem_type>`: string, specifies the type of optimization problem, i.e., "rcpsp".
- `<solver>`: string, represents available solvers including open-source software "cbc" from https://github.com/coin-or/Cbc.git, and "glpk" from https://www.gnu.org/software/glpk/. The users can also use proprietary software "CPLEX" from https://www.ibm.com/products/ilog-cplex-optimization-studio or "Gurobi" from https://www.gurobi.com.
- `<sense>`: string, specifies "minimize" or "maximize" for the RCPSP problem
- `<makespan_upperbound>`: integer, specifies upper bound of the makespan.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Logos>
   ...
   <Settings>
      <makespan_upperbound>65</makespan_upperbound>
      <solver>cbc</solver>
      <sense>minimize</sense>
      <problem_type>rcpsp</problem_type>
   </Settings>
   ...
</Logos>
```

Figure 41. Example XML for `<Settings>` input block.

7.5   Multi-Objective Optimization

In a risk-informed context, decision making is performed by balancing O&M costs and system reliability. Sections 5.1 through 5.3 have focused on several optimization methods designed to minimize/maximize a single-objective function (e.g., O&M costs), while imposing constraints on the other objective function (e.g., system reliability).

The goal of multi-objective optimization is to find optimal conditions under which multiple objective functions are maximized/minimized. While in single-objective optimization methods a single solution that minimize/maximize the objective function can be determined, multi-objective optimization methods determine a set of candidate solutions: the Pareto frontier.

The Pareto frontier is defined as the subset of choices the dominates the remaining set of choices (i.e., all objective functions have better or identical value). As an example, Figure 42 shows an example of multi-objective optimization in a 2-dimensional space (i.e., utility and cost). Each point represents a candidate option, while the set of options that are part of the Pareto frontier [46] (i.e., the optimal solutions) are encircled in the yellow cloud.

If objective constraints are present, then they can be imposed on the obtained Pareto frontier to filter only the options on the Pareto frontier that satisfy them.
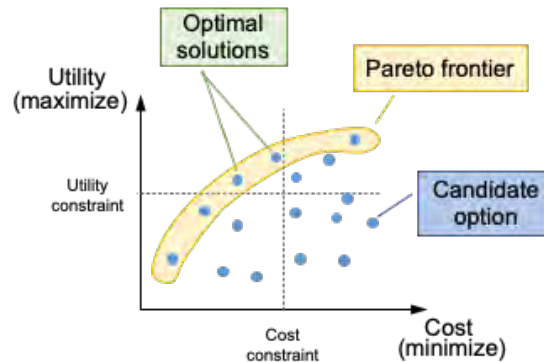

Figure 42. Graphical representation of the Pareto frontier (encircled in the yellow cloud) in a 2-dimensional space. The two objective functions are: utility (that it is desired to be minimized), and cost (that it is desired to be minimized).

7.5.1   Multi-Dimensional Pareto Frontier

During FY 2020, a RAVEN post-processor has been developed to determine the Pareto frontier from a 2-dimensional data set (i.e., two objective functions). During FY 2021, the concept of Pareto frontier post-processor has been extended to manage any number of objective functions (i.e., greater than two objective functions). As an example, the possible set of objective functions in a risk-informed context are:

- System O&M costs

- System availability

- System reliability

- System failure consequence.

This kind of analysis requires a model Ψ (or a set of models) that determines the value of M objective functions $y_1, \dots, y_M$ given a set of N input variables $x_1, \dots, x_N$ (also known as decision functions):

$$Y = \Psi(X) \quad \text{where: } Y = [y_1, \dots, y_M] \text{ and } X = [x_1, \dots, x_N] \tag{28}$$

The set of input variables X are sampled and the corresponding output variables Y are obtained from the model Ψ. The Pareto frontier is then obtained from the generated set of Y as indicated in the workflow shown in Algorithm 5. This method is very effective when the input variables are discrete in nature and the possible combination of values for the input variables are limited.

Algorithm 5. Pareto Frontier based multi-objective optimization.
1. Generate all possible combinations of values for the input variables $x_1, \dots, x_N$
2. Generate the objective functions $y_1, \dots, y_M$ for each combination of the input variables generate in Step 1
3. Determine the Pareto frontier from the data set generated in Step 2

The workflow described in Algorithm 5 can be coded using RIAM risk analytic platform. In particular, RAVEN can be employed to:

1. Sample the model response $Y = \Psi(X)$ using the RAVEN Grid sampler (see Steps 1 and 2 in Algorithm 5)

2. Determine the Pareto frontier using the RAVEN ParetoFrontier post-processor (see Step 3 in Algorithm 5)

This workflow is particularly usefull when:

- The set of input variables $x_1, \ldots, x_N$ are discrete in nature

- All possible combinations of X are limited in number

- The evaluation model response is not time consuming.

In other terms, the last two conditions can be summarized as: the computational cost to evaluate all possible combinations of X is within a reasonable time frame.

7.5.2   Optimization Methods Applied to Sensor Configurations

The application of multi-objective optimization methods is here applied to determine the optimal monitoring configuration for a single SSC. The goal is to balance monitoring installation costs and monitoring capabilities. We are focusing on a centrifugal pump characterized by a set of failure modes indicated in Table 24. For each failure mode j, Table 24 reports the annual probability of occurrence p; Table 24 lists also the consequences when each failure mode is not detected (in terms of unplanned replacement costs C^{unpl}) and when the failure mode is detected (in terms of planned replacement costs C^{pl}).

The considered sensors are listed in Table 25. For each sensor i, Table 25 lists its operational cost, and the failure mode that it can detect.

Table 24. Failure modes considered for a generic centrifugal pump.

j	Failure mode	Cost of unplanned replacement $C^{unpl}[j]$	Cost of planned replacement $C^{pl}[j]$	Probability of occurrence over one year of operation $p[j]$
1	Motor bearing failure	2.5M	60K	0.01
2	Pump bearing failure	1.5M	20K	0.02
3	Pump coupler failure	1.8M	70K	0.05
4	Pump cooling failure	2.0M	100K	0.08
5	Pump seal failure	1.3M	50K	0.09
6	Stator degradation	6.0M	1.0M	0.01

Ideally, all sensors could be chosen: such monitoring configuration would be able to detect all failure modes and avoid unplanned replacement costs; however, such monitoring configuration is characterized by high monitoring expenses. The opposite choice is characterized by a monitoring configuration where no sensor is chosen: null monitoring expenses but high risk of unplanned replacement costs. Between these

very different monitoring configurations, there are several others characterized that can be chosen from (to be precise there are $2^6 = 64$ possible configuration sensors).

Table 25. Sensors that can be chosen to monitor pump degradation.

i	Sensor	Cost $C^{sens}[i]$	Detected failure modes $FM[i]$
1	Motor shaft vibration sensor	40K	1
2	Pump shaft vibration sensor	35K	2
3	Pump oil temperature	20K	4
4	External infrared	70K	3
5	Leak detector relay	20K	5
6	Current signature analysis	35K	6

The goal is to identify the sensor configurations that balance monitoring capabilities and monitoring costs simultaneously. This class of problem can be framed as a multi-objective optimization problem where we can define the two objective functions as follows:

- Component monitoring cost (installation and O&M costs) (see Table 25)

$$Monitoring\ cost = \sum_{i=1}^{6} C^{sens}[i]\, d_i \qquad (29)$$

where d_i for is the decision variable for each sensor $i = 1, ...,6$ d_i is 1 if the sensor is chosen, 0 otherwise

- Monitoring performance, through value of information VoI [84] (see Table 24)

$$VoI = \sum_{i=1}^{6}(EV_i^{wo} - EV_i^{w})\, d_i \qquad (30)$$

where:

  o Expected value with sensor: $EV_i^{w} = p[FM_i]\, C^{pl}[FM[i]]$
  o Expected value without sensor: $EV_i^{wo} = p[FM_i]\, C^{unpl}[FM[i]]$
  o $EV_i^{wo} - EV_i^{w} = p[FM_i]\big(C^{unpl}[FM[i]] - C^{pl}[FM[i]]\big)$

Using the RAVEN code [85] we were able to generate the value of the two objective functions listed above for each of the 64 possible configurations.

The next step has been the determination of the Pareto frontier where RAVEN was again employed using the method indicated in Section 7.1. The set of monitoring configurations belonging to the Pareto frontier is reported in Table 26. Figure 43 graphically shows the 64 possible monitoring configurations in the 2-dimensional objective function space (i.e., monitoring costs and VoI). In the same figure, the configurations belonging to the Pareto frontier are highlighted in red and the configurations is reported as a 6-dimensional array $[d_1, ..., d_6]$. In Figure 43 note that the above-mentioned configurations where no sensors and all sensors are chosen are located in the bottom left and top right corners of the figure respectively.

Table 26. Monitoring configurations belonging to the Pareto frontier.

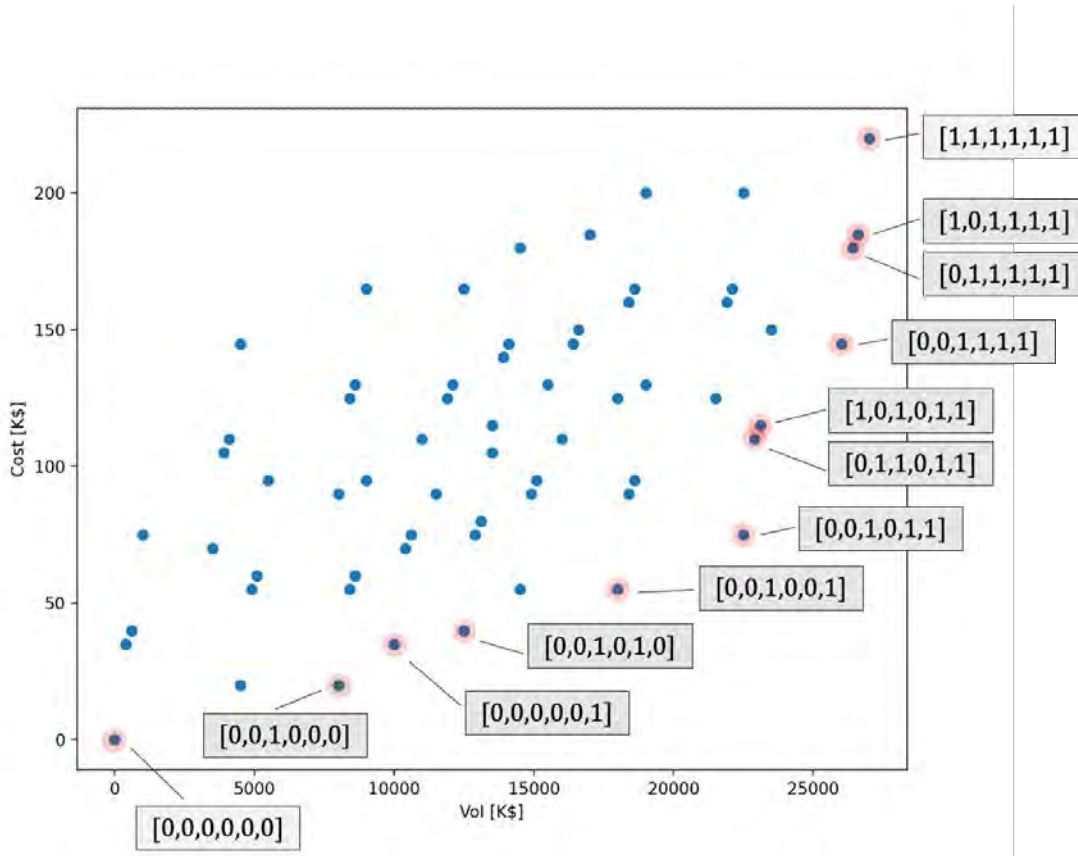| Monitoring configuration						Objective functions	
d_1	d_2	d_3	d_4	d_5	d_6	VoI	cost
0	0	0	0	0	0	0	0
0	0	0	0	0	1	10000	35
0	0	1	0	0	0	8000	20
0	0	1	0	0	1	18000	55
0	0	1	0	1	0	12500	40
0	0	1	0	1	1	22500	75
0	0	1	1	1	1	26000	145
0	1	1	0	1	1	22900	110
0	1	1	1	1	1	26400	180
1	0	1	0	1	1	23100	115
1	0	1	1	1	1	26600	185
1	1	1	1	1	1	27000	220

Figure 43. Scatter plot of the 64 possible monitoring configurations in the objective function space (monitoring costs and VoI). The configurations belonging to the Pareto frontier are highlighted in red and the configurations is reported as a 6-dimensional array $[d_1, \dots, d_6]$.

7.5.3 Multi-Objective Model Optimization

When the input variables are continuous in nature or the possible combination of discrete values for the input variables are very large, then the method presented in Section 5.4.1 cannot be directly applied. A solution to this limitation is to employ evolutionary methods. These methods are based on the GAs developed during FY 2020; during FY 2021 these methods have been initially extended to perform multi-objective optimization.

For the scope of this project, we have chosen the most flexible multi-objective evolutionary algorithms: the nondominated sorting genetic algorithm II (NSGA-II) [83]. The basic structure of the algorithm (see Figure 44) is not too dissimilar to the structure of classical GA methods. The most relevant difference is the sorting step where current population and newly generated children are chosen to determine the population at the next iteration.

In classical GA methods (i.e., single objective), the population at the next iteration is determined by choosing the elements of the current population and the newly generated children that possess with highest fitness. These methods act on a population of sampled points $(x, F(x))$ (rather than focusing on a one-sample-at-a-time mindset) and they iteratively combine pairs of points to generate a new generation of points with higher quality. An initial population of N elements is initially generated (e.g., by Monte-Carlo sampling) and evaluated. Each element $(x, F(x))$ of the population has the input coordinates x encoded into a discrete form, a genotype, while the $F(x)$ term is encoded into a fitness value $\tilde{f}$. The genotype form of x (here indicated as $\tilde{x}$) is called a data structure and it can be of several forms depending on the application. In this report we focus on arrays of discrete values. More advanced data structures can be matrices, tree structures or graph structures.

The main operators that are being employed by GAs are the following:

- Crossover. The encodings of two chromosomes are mixed to generate two new encodings

- Mutation. The encoding of a chromosome is altered by randomly changing the value of a single element of the chromosome

- Replacement. The population of chromosomes is updated by removing chromosomes with low-fitness or high-generational age value and keeping chromosomes with high-fitness or low-generational age.

The main structure of a GA optimization algorithm is described in Algorithm 5 and is also graphically shown in Figure 44.

In a multi-objective context, the concept of fitness does not exist strictly speaking, and the concept of Pareto frontier is now employed. The parent selection of is performed by iteratively determine the Pareto frontier out the ensemble current population plus newly generated children. This iteration process is described in Algorithm 6 and graphically represented in Figure 45.

Algorithm 5. Basic structure of evolutionary algorithm.
1. Create initial population of N samples $(x, F(x))_n$, $n = 1, \dots, N$, e.g., through uniform Monte-Carlo sampling
2. Perform a genotype representation according to the problem under investigation
3. Calculate fitness of each chromosome: $(x, F(x))_n \rightarrow (\tilde{x}, \tilde{f})_n$
4. Reproduction: create the new generation of offspring from current population:

a. Perform parent selection from the population based on their fitness

b. Perform crossover: creation of child population mixing chromosome structure of parents

5. Perform random mutation on the generated offspring

6. Evaluate offspring (i.e., determine $F(x)$) and calculate their fitness $\tilde{f}$
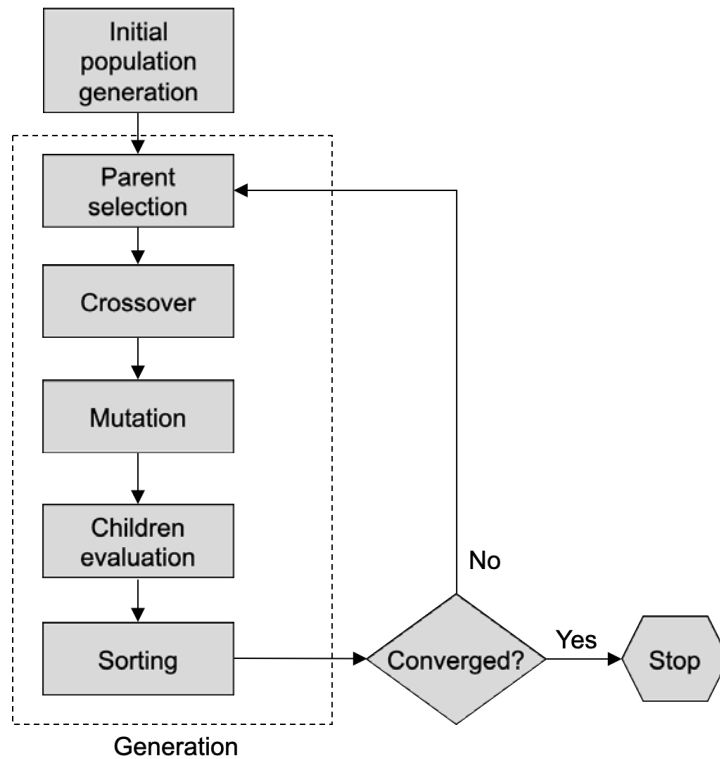
7. Return to Step 4 until convergence is met.

Figure 44. The main structure of a GA optimization algorithm.

| Algorithm 6. Rank based sorting. |
|---|
| 1. Determine the Pareto frontier (frontier 1) out the ensemble current population plus newly generated children: associate rank 1 to the points belonging to this Pareto frontier |
| 2. Remove elements contained in the frontier 1 from the ensemble current population plus newly generated children |
| 3. Determine the Pareto frontier (frontier n) from elements remaining from Step 2 |
| 4. Repeat Steps 2 and 3 until all points are associated to one Pareto frontier |
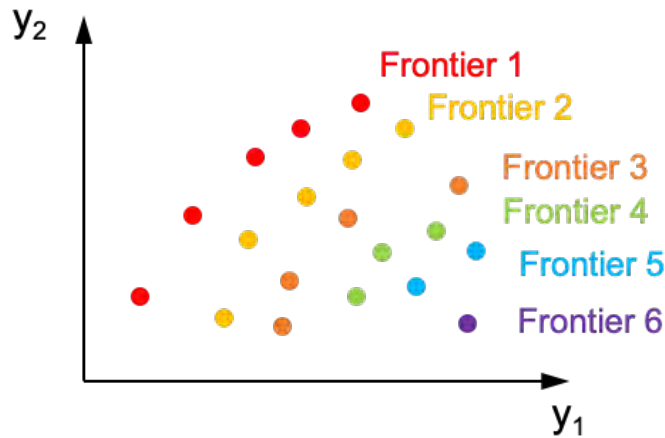
Figure 45. Ranking of elements of a population based on the iterative construction of the Pareto frontier.

The selection of the population for the new generation is not based on the fitness (as currently performed by the single-objective version) but on the rank (i.e., see Figure 46 and Algorithm 5). The new population is created by progressively choosing points with increasing rank (i.e., starting with point belonging to Pareto frontier 1). A sorting based on the crowding distance is performed for the elements belonging to the last frontier that is required to populate the population of the next generation. This workflow is graphically shown in Figure 46.
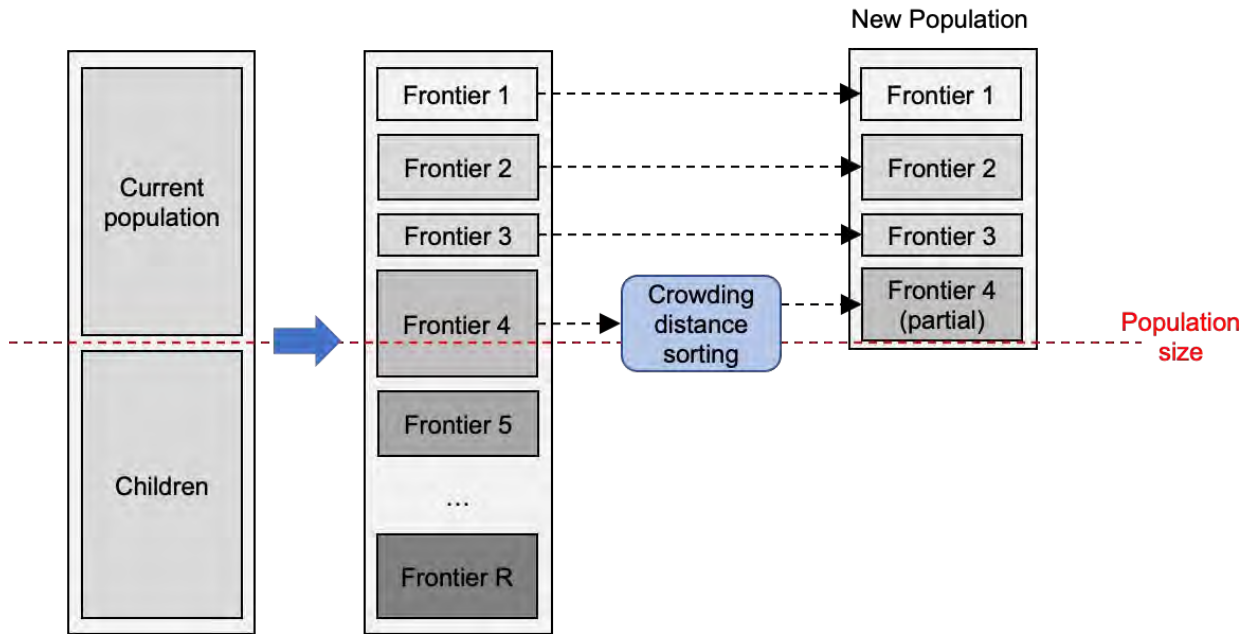

Figure 46. Population and children sorting to determine the population of the next generation [83].

During FY 2021, we have tested under several conditions the single objective version of the GA optimization methods in RAVEN, and they are now a part of the official RAVEN release. Regarding the multi-objective GA optimization method, we have completed several computational elements that will be finalized and tested during FY 2022.

8. CONCLUSIONS

The main goal of the RIAM project is to create a direct link between ER data and decision making; given this premises, this report summarizes the complete set of use cases that the RIAM project is focusing on. This set has been partitioned into three main areas: analysis of ER data, reliability modeling, and plant resources optimization. In the past decade, the nuclear industry has been addressing deficiencies in these three areas with various degrees of success. In this respect, the RIAM project has chosen few critical points in these three areas and developed methods and tools to overcome these critical points with innovative methods.

This report focuses not only on the development but, more important, also on the practical application of the RIAM toolkit for these specific critical points. In the area of analysis of ER data, we have proposed effective methods to integrate SSC monitoring data and IRs to assess SSC health. Such methods rely on NLP methods to extract information from IR reports and then they perform diagnosis analysis using causal inference methods by employing OPM models for each SSC. In the area of reliability modeling, the RIAM project has identified a very effective method to integrate current and past ER data (both IRs and SSC monitoring data) into reliability models that are familiar to plant system engineers. Rather than probability-based calculations, we here have presented innovative margin-based calculations. This modeling approach has the advantage that it can provide a real-time assessment of plant system/health and identify the SSCs that negatively affecting reliability. Lastly, in the area of asset management, we have developed optimization methods designed to optimize plant resources (money, time, and plant staff). These methods balance both plant reliability and operational costs; thus, they provide a more quantitative and complete knowledge to decision makers in terms of project prioritization, scheduling maintenance activities, and SSC lifecycle management.

During FY 2022, we are planning to continue development and testing of NLP methods to analyze ER data. In addition, we will integrate data analysis tools, plant reliability models, and optimization methods to manage plant assets and resources in a single and consistent workflow. This activity will create a complete analysis platform testbed for plant operations.

REFERENCES

[1]   R. Szilard, H. Zhang, S. Hess, J. Gaertner, D. Mandelli, S. Prescott, M. Farmer, "RISA Industry Use Case Analysis," Idaho National Laboratory Technical Report, INL/EXT-18-51012 (2018).

[2]   U.S. Department of Energy, "Light Water Reactor Sustainability Program Integrated Program Plan," Idaho National Laboratory Technical Report, INL/EXT-11-23452 (2020).

[3]   C. Wang, D. Mandelli, M. Abdo, A. Alfonsi, J. Cogliati, P. Talbot, S. Lawrence, C. Smith, D. Morton, I. Popova, S. Hess, C. Pope, J. Miller, S. Ercanbrack, "Development and Release of the Methods and Tools for Risk-Informed Asset Management," Idaho National Laboratory Technical Report, INL/EXT-21-63255 (2021).

[4]   D. Mandelli, C. Wang, S. StGermain, C. Smith, D. Morton, I. Popova, S. Hess, "Combined Data Analytics and Risk Analysis Tool for Long Term Capital SSC Refurbishment and Replacement," Idaho National Laboratory Technical Report, INL-EXT-19-55819 (2019).

[5]   D. Mandelli, Z. Ma, R. Youngblood, S. St Germain, C. Smith, P. Talbot, S. Hess, D. Dube, C. Pope, J. Miller, M. Robbins, D. Das, M. Azarian, J. Coble, "Plant Integral Risk-informed System Health Program," Idaho National Laboratory Technical Report, INL-EXT-19-55808 (2019).

[6]   D. Mandelli, C. Wang, J. Cogliati, C. Smith, S. Hess, R. Sugrue, C. Pope, J. Miller, S. Ercanbrack, D. Cole, J. Yurko, "Integration of Data Analytics with Plant System Health Program," Idaho National Laboratory Technical Report, INL/EXT-20-59928 (2020).

[7]   D. Mandelli, C. Wang, M. Abdo, A. Alfonsi, P. Talbot, J. Cogliati, C. Smith, D. Morton, I. Popova, S. Hess, "Development and Application of a Risk Analysis Toolkit for Plant Resources Optimization," Idaho National Laboratory Technical Report, INL-EXT-20-59942 (2020).

[8]   J. Borky, T. Bradley, *Effective Model-Based Systems Engineering*, Springer ed. (2018).

[9]   D. Dori, E. Crawley, *Object-Process Methodology: A Holistic Systems Paradigm*, Springer ed. (2002).

[10]   S. William, *The Object Primer: Agile Model Driven Development with UML 2*, Cambridge University Press (2004).

[11]   S. Friedenthal, A. Moore, R. Steiner, *A Practical Guide to SysML: The Systems Modeling Language*, Morgan Kaufmann (2008).

[12]   J. Lin, E. Keogh, S. Lonardi, B. Chiu, "A Symbolic Representation of Time Series, with Implications for Streaming Algorithms," *8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pp. 2–11 (2003).

[13]   S. Bird, E. Klein, E. Loper, *Natural Language Processing with Python*, O'Reilly Media (2009).

[14]   H. Lane, H. Hapke, C. Howard, *Natural Language Processing in Action: Understanding, analyzing, and generating text with Python*, Manning Publications (2019).

[15]   P. Baraldi, F. Di Maio, P. Turati, and E. Zio, "Robust Signal Reconstruction for Condition Monitoring of Industrial Components via a Modified Auto Associative Kernel Regression Method," *Mechanical Systems and Signal Processing*, **60**, pp.29–44 (2015).

[16]   V. Chandola, A. Banerjee, V. Kumar, "Anomaly Detection: A survey," *ACM Computing Surveys*, **41**, no. 3, pp. 1–58 (2009).

[17]   V. J. Hodge, J. Austin, "A Survey of Outlier Detection Methodologies," *Artificial Intelligence Review*, **22**, no. 2, pp. 85–126 (2004).

[18]   A. Zimek, P. Filzmoser, "There and Back Again: Outlier Detection Between Statistical Reasoning and Data Mining Algorithms," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, **8**, no. 6 (2018).

[19]   E. M. Knorr, R. T. Ng, V. Tucakov, "Distance-Based Outliers: Algorithms and Applications," *The VLDB Journal the International Journal on Very Large Data Bases*, **8**, no. 3, pp. 237–253 (2000).

[20]   D. Mandelli, C. Smith, A. Yilmaz, T. Aldemir, "Mining nuclear transient data through symbolic conversion," in *ANS PSA 2013 International Topical Meeting on Probabilistic Safety Assessment and Analysis*, Columbia, SC, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2013).

[21]   C. D. Manning, P. Raghavan H. Schütze, *Introduction to Information Retrieval*, Cambridge University Press (2008).

[22]   C. D. Manning, H. Schütze, *Foundations of Statistical Natural Language Processing*, The MIT Press (1999).

[23] M. Mohri, A. Rostamizadeh, A. Talwalkar, *Foundations of Machine Learning*, The MIT Press (2012).

[24] R. Sipos, D. Fradkin, F. Moerchen, and Z. Wang, "Log-Based Predictive Maintenance," *KDD 14: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1867–1876 (2014).

[25] M. Zhou, N. Duan, S. Liu, and H.-Y. Shum, "Progress in Neural NLP: Modeling, Learning, and Reasoning," *Engineering*, **6**, no. 3, pp. 275–290 (2020).

[26] Nuclear Energy Institute (NEI), "NEI 00-04 10CFR 50.69 SSC Categorization Guideline," Washington, DC (2005).

[27] United States Nuclear Regulatory Commission, "Regulatory Guide 1.201 - Guidelines for Categorizing Structures, Systems, and Components in Nuclear Power Plants According to their Safety Significance - Revision 1," Washington, DC (2006).

[28] Nuclear Energy Institute (NEI), "NEI 06-09 Risk-Informed Technical Specifications Initiative 4b – Risk-Managed Technical Specifications (RMTS) Industry Guidance Document," Washington, DC (2006).

[29] Nuclear Energy Institute (NEI), "NEI 04-10 Risk-Informed Technical Specifications Initiative 5b – Risk-Informed Method for Control of Surveillance Frequencies Industry Guidance Document," Washington, DC (2006).

[30] United States Nuclear Regulatory Commission, "Standard Technical Specifications: Babcock and Wilcox Plants (Revision 4) - NUREG 1430," Washington, DC (2012).

[31] United States Nuclear Regulatory Commission, "Standard Technical Specifications: Westinghouse Plants (Revision 4) - NUREG 1431," Washington, DC (2012).

[32] United States Nuclear Regulatory Commission, "Standard Technical Specifications: Combustion Engineering Plants (Revision 4); NUREG 1432," Washington, DC (2012).

[33] United States Nuclear Regulatory Commission, "Standard Technical Specifications: General Electric BWR/4 Plants (Revision 4); NUREG 1433," Washington, DC (2012).

[34] United States Nuclear Regulatory Commission, "Standard Technical Specifications: General Electric BWR/6 Plants (Revision 4); NUREG 1434," Washington, DC (2012).

[35] United States Nuclear Regulatory Commission, "Regulatory Guide 1.174: An Approach for Using Probabilistic Risk Assessment in Risk-Informed Decisions in Plant-Specific Changes to the Licensing Basis - Revision 3," Washington, DC (2018).

[36] United States Nuclear Regulatory Commission, "Regulatory Guide 1.200: "Acceptability of Probabilistic Risk Assessment Results for Risk-Informed Activities - Revision 3," Washington, DC (2020).

[37] Miroslav Kubat, *An Introduction to Machine Learning (2nd Ed.)*, Springer International Publishing (2017).

[38] D. Wackerly, W. Mendenhall, R. Schaeffer, *Mathematical Statistics with Applications (5th Edition)*, Duxbury Press (1996).

[39] S. Shalev-Shwartz, S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*, Cambridge University Press (2014).

[40] J. Lee, N. J. McCormick, *Risk and Safety Analysis of Nuclear Systems*, Wiley edition (2011).

[41] Electric Power Research Institute (EPRI), "Computer Aided Fault Tree Analysis System (CAFTA), Version 6.0b," EPRI Report 3002004316 (2014).

[42] Official website: https://www.lr.org/en/riskspectrum/ .

[43] C. L. Smith, S.T. Wood, D. O'Neal, "Systems Analysis Programs for Hands-On Integrated Reliability, Evaluations (SAPHIRE) Version 8," NUREG/CR-7039, vol. 3 (2011).

[44] R. Youngblood, "Risk Significance and Safety Significance," *Reliability Engineering & System Safety*, **73**, no. 2, pp. 121–136 (2001).

[45] M. Rausand and A. Høyland, *System Reliability Theory; Models, Statistical Methods and Applications*, Wiley Series in Probability and Statistics (2004).

[46] R. K. Sarin, *Multi-Attribute Utility Theory, Encyclopedia of Operations Research and Management Science*, Springer, Boston, MA (2013).

[47] A. Koc, D. P. Morton, E. Popova, S. M. Hess, E. Kee, D. Richards, "Prioritizing Project Selection," *The Engineering Economist*, **54**, no. 4, pp. 267–297 (2009).

[48] A. R. McKendall, J. S. Noble, C. M. Klein, "Scheduling Maintenance Activities During Planned Outages at Nuclear Power Plants," *International Journal of Industrial Engineering - Theory Applications Practice*, **15**, no. 1, pp. 53–61 (2008).

[49] R. Kolisch, A. Sprecher, "PSPLIB - A Project Scheduling Problem Library: OR Software - ORSEP Operations Research Software Exchange Program," *European Journal of Operational Research*, **96**, no. 1, pp. 205–216 (1997). doi: 10.1016/S0377-2217(96)00170-1.

[50] G. Muñoz, D. Espinoza, M. Goycoolea, E. Moreno, M. Queyranne, O. R. Letelier, "A Study of The Bienstock–Zuckerberg Algorithm: Applications in Mining and Resource Constrained Project Scheduling," *Computational Optimization and Applications*, **69**, no. 2, Springer US (2018).

[51] P. Brucker, S. Knust, "A Linear Programming and Constraint Propagation-Based Lower Bound for the RCPSP," *European Journal of Operational Research*, **127**, no. 2, pp. 355–362, (2000). doi: 10.1016/S0377-2217(99)00489-0.

[52] R. Klein, A. Scholl, "Computing Lower Bounds by Destructive Improvement: An Application To Resource-Constrained Project Scheduling," *European Journal of Operational Research*, **112**, no. 2, pp. 322–346 (1999). doi: 10.1016/S0377-2217(97)00442-6.

[53] G. Zhu, J. F. Bard, G. Yu, "A Branch-And-Cut Procedure for The Multimode Resource-Constrained Project-Scheduling Problem," *INFORMS Journal on Computing*, **18**, no. 3, pp. 377–390 (2006). doi: 10.1287/ijoc.1040.0121.

[54] W. Herroelen, B. De Reyck, E. Demeulemeester, "Resource-Constrained Project Scheduling: A Survey of Recent Developments," *Computers & Operations Research*, **25**, no. 4, pp. 279–302 (1998). doi: 10.1016/S0305-0548(97)00055-5.

[55] R. Kolisch, R. Padman, "An Integrated Survey of Deterministic Project Scheduling," *Omega*, **29**, no. 3, pp. 249–272 (2001). doi: 10.1016/S0305-0483(00)00046-3.

[56] G. Muñoz, D. Espinoza, M. Goycoolea, E. Moreno, M. Queyranne, O. R. Letelier, "A Study of The Bienstock–Zuckerberg Algorithm: Applications in Mining and Resource Constrained Project Scheduling," *Computational Optimization and Applications*, **69**, no. 2, Springer US (2018). doi: 10.1007/s10589-017-9946-1.

[57] A. Pritsker, L. Watters, P. Wolfe, "Multiproject Scheduling with Limited Resources-a Zero- One Programming Approach," *Management Science*, **16**, no. 1, pp. 93–108 (1969). doi: 10.1287/mnsc.16.1.93.

[58] M. L. Pinedo, *Scheduling Theory, Algorithms, and Systems*, Elsevier (2012).

[59] R. Kolisch, S. Hartmann, "Experimental Investigation of Heuristics for Resource-Constrained Project Scheduling: An Update," *European Journal of Operational Research*, **174**, no. 1, pp. 23–37 (2006). doi: 10.1016/J.EJOR.2005.01.065.

[60] J. Alcaraz, C. Maroto, "A Robust Genetic Algorithm for Resource Allocation in Project Scheduling," *Annals of Operations Research*, **102**, pp. 83–109 (2001).

[61] D. Debels, B. De Reyck, R. Leus, M. Vanhoucke, "A Hybrid Scatter Search/Electromagnetism Meta-Heuristic for Project Scheduling," *European Journal of Operational Research*, **169**, no. 2, pp. 638–653 (2006). doi: 10.1016/j.ejor.2004.08.020.

[62] S. Hartmann, "A Self-Adapting Genetic Algorithm for Project Scheduling Under Resource Constraints," *Naval Research Logistic*, **49**, no. 5, pp. 433–448 (2002). doi: 10.1002/nav.10029.

[63] Y. A. Kochetov, A. Stolyar, "Evolutionary Local Search with Variable Neighborhood for the Resource Evolutionary Local Search Constrained Project Scheduling Problem," in *Proceedings of the 3rd International Workshop of Computer Science and Information Technologies*, no. 1, (2003).

[64] P. Tormos, A. Lova, "A Competitive Heuristic Solution Technique for Resource-Constrained Project Scheduling," *Annals of Operations Research*, **102**, no. 1, pp. 65–81 (2001). doi: 10.1023/A:1010997814183.

[65] V. Valls, F. Ballestín, S. Quintanilla, "A Hybrid Genetic Algorithm for the Resource-Constrained Project Scheduling Problem," *European Journal of Operational Research*, **185**, no. 2, pp. 495–508, (2008). doi: 10.1016/j.ejor.2006.12.033.

[66] J. Alcaraz, C. Maroto, "A Robust Genetic Algorithm for Resource Allocation in Project Scheduling," *Annals of Operations Research*, **102**, no. 1, pp. 83–109, 2001, doi: 10.1023/A:1010949931021.

[67] S. Marsland, *Machine Learning: An Algorithmic Perspective*, Chapman & Hall/CRC Machine Learning & Pattern Recognition (2009).

[68] T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer ed. (2009).

[69] F. Morchen, "Time Series Knowledge Mining," PhD Thesis, Philipps-University Marburg, Germany (2006).

[70] F. Morchen, "Algorithms for Time Series Knowledge Mining," *KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 668–673 (2006).

[71] C. A. Mattson, A. Messac, "Pareto Frontier Based Concept Selection Under Uncertainty, with Visualization," *Optimization and Engineering*, **6**, pp. 85–115 (2005).

[72] S, Doan, E. W. Yang, S. S. Tilak, P. W. Li, D. S. Zisook, M.Torii, "Extracting Health-Related Causality From Twitter Messages Using Natural Language Processing," *BMC Medical Informatics and Decision Making*, **19**, pp. 71–84 (2019).

[73] J. Yang, S. Caren Han, J. Poon, "A Survey on Extraction of Causal Relations from Natural Language Text," *Computing Research Repository* (2021). [https://arxiv.org/abs/2101.06426]

[74] A. Akkasi, M.-F. Moens, "Causal Relationship Extraction from Biomedical Text Using Deep Neural Models: A Comprehensive Survey," *Journal of Biomedical Informatics*, **119** (2021).

[75] A. Le Glaz, Y. Haralambous, D. Kim-Dufor, P. Lenca, R. Billot, T. Ryan, J. Marsh, J. DeVylder, M. Walter, S. Berrouiguet, C. Lemey, "Machine Learning and Natural Language Processing in Mental Health: Systematic Review," *Journal of Medical Internet Research*, **23**, no. 5 (2021).

[76] T. Young, H. Devamanyu, S. Poria, E. Cambria, "Recent Trends in Deep Learning Based Natural Language Processing," *IEEE Computational Intelligence Magazine*, **13**, no. 3, pp. 55–75 (2018).

[77] Dan Otter, J. R. Medina, J. Kalita, "A Survey of the Usages of Deep Learning for Natural Language Processing," *IEEE Transactions on Neural Networks and Learning Systems*, **30**, no. 2 (2020).

[78] J. Pearl, "Causal Inference in Statistics: An Overview," *Statistics Surveys*, **3**, pp. 96–146 (2009).

[79] D. Lopez-Paz, K. Muandet, B. Schölkopf, I. Tolstikhin, "Towards a Learning Theory of Cause-Effect Inference," *ICML 15, Proceedings of the 32nd International Conference on International Conference on Machine Learning*, **37**, pp. 1452–1461 (2015).

[80] G. Casella, R. Berger, *Statistical Inference*, Cengage Learning, 2nd edition (2001).

[81] A. Gelman, J. Carlin, H. Stern, D. Dunson, A. Vehtari, D. Rubin, *Bayesian Data Analysis*, Chapman & Hall/CRC Texts in Statistical Science (2013).

[82] J. Farber, A. Al Rashdan, H. Abdel-Khalik, Y. Li, M. Abdo, D. Mandelli, D. Huang, "Process Anomaly Detection for Sparsely Labeled Events in Nuclear Power Plants," Idaho National Laboratory Technical Report, INL-EXT-21-01409 (2021).

[83] K. Deb, S. Agrawal, A. Pratap, "A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II," *PPSN 2000: International Conference on Parallel Problem Solving from Nature*, pp. 849-858 (2000).

[84] D. Zonta, B. Glisic, S. Adriaenssens, "Value of Information: Impact of Monitoring on Decision-Making," *Structural Control and Health Monitoring*, **21**, pp. 1043–1056 (2014).

[85] A. Alfonsi, C. Rabiti, D. Mandelli, J. Cogliati, C. Wang, P. W. Talbot, D. P. Maljovec, C. Smith, "RAVEN Theory Manual," Idaho National Laboratory Technical Report, INL/EXT-16-38178 (2020).

APPENDIX A: Centrifugal Pump OPM Model

This section provides an example of OPM modeling. We apply such modeling methodology to a generic centrifugal pump since it is a fairly common component in existing NPPs. The main components that compose a centrifugal pump are as follows:

- Pump:
    - Pump casing: which is mainly designed to: 1) act as pressure containment vessel, and 2) to convert kinetic energy (i.e., velocity) into high pressure of the fluid
    - Pump shaft: designed to support a smooth rotation of the impeller
    - Pump shaft seals: mechanical rings designed to prevent any leakage of fluid
    - Pump bearings: designed to reduce rotation friction of the pump rotating shaft
    - Impeller: directly mounted on the pump shaft, it increases the kinetic energy of the flow
- Motor:
    - Motor casing: designed to contain all components and to dissipate the generated heat
    - Rotor: source of rotation movement of the shaft the pump
    - Stator: designed to convert electric energy into rotation of the rotor
    - Motor shaft: designed to support a smooth rotation of the rotor
    - Motor bearings: designed to reduce rotation friction of the motor rotating shaft
    - Pump coupling: it drives the rotation from the motor shaft to the pump shaft

OPM modeling is performed by creating diagrams that are designed to capture the form and functional aspects of a system/component. These diagrams consist of elements and links among these elements. While the complete syntax of a OPM diagrams can be found in [9], for the scope of this example, the elements and links here employed are shown in Table 27.

Table 28 provides a more detailed description on how parts of an OPM diagram can be converted into a text form (i.e., object process language [OPL]).

A complete description of the OPM methodology[12] can be found in [9].

Given the syntax and semantic description of the basic OPM elements briefly provided in Table 27 and Table 28, we have developed the first tier of the OPM model for a generic centrifugal pump. This is shown in Figure 47 while the corresponding OPL structure is listed in Table 29.

---

[12] Additional information can be found at the following links:
- OPM official website: https://www.opcloud.tech/
- Wikipedia article: https://en.wikipedia.org/wiki/Object_Process_Methodology

Table 27. Common OPM elements and links.

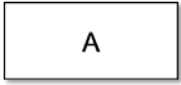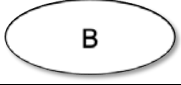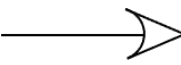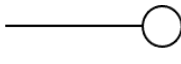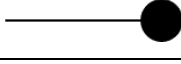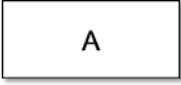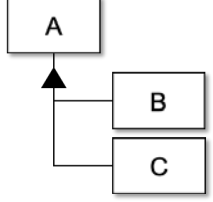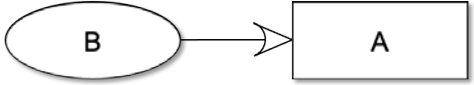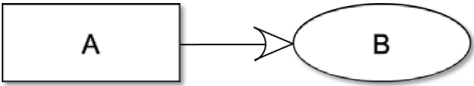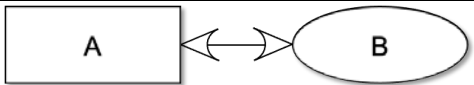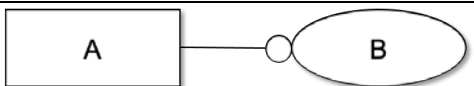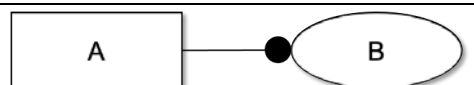| Element symbol	Type
A	Object element
B	Process element
(arrow with filled triangle)	Aggregation link
(arrow with open chevron)	Consumption/result/effect link
(line with open circle)	Instrument link
(line with filled circle)	Agent link

Table 28. Translation of OPM diagrams into OPL.

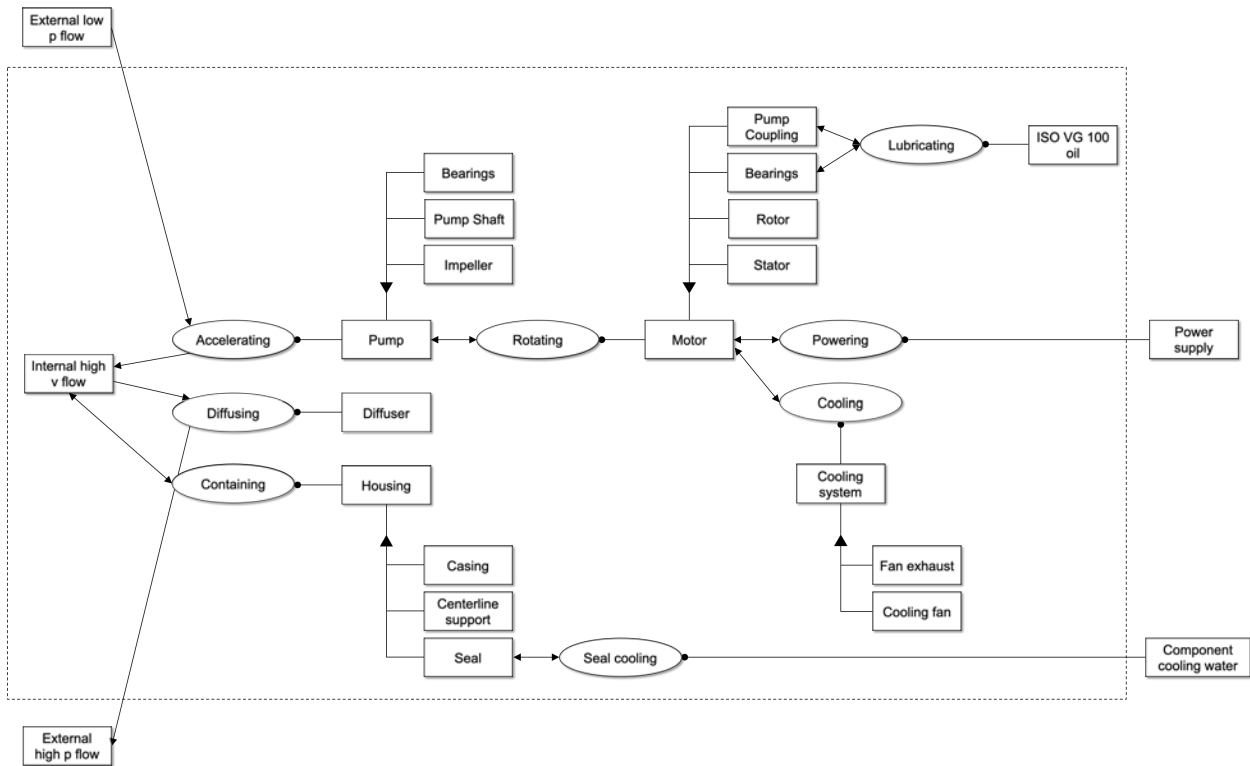| OPM diagram	Name	OPL
A	Object	A is physical
B	Process	B is physical
A, B, C	Aggregation link	A consists of B and C
B → A	Result link	B yields A
A → B	Consumption link	B consumes A
A ↔ B	Effect link	B affects A
A —○ B	Instrument link	B requires A
A —● B	Agent link	B handles A

Figure 47. OPM model for a generic centrifugal pump.

Table 29. List of OPL elements derived from the pump OPM diagram show in Figure 47. Elements in green are form entities while elements in blue are functional entities.

ID	OPL element
1	External low p flow is environmental and physical.
2	Internal high v flow is physical.
3	External high p flow is environmental and physical.
4	Pump is physical.
5	Pump consists of Bearings, Pump Shaft, and Impeller.
6	Bearings is physical.
7	Pump Shaft is physical.
8	Impeller is physical.
9	Diffuser is physical.
10	Housing is physical.
11	Housing consists of Casing, Centerline support, and Seal.
12	Casing is physical.
13	Centerline support is physical.
14	Seal is physical.
15	Motor is physical.
16	Motor consists of Pump coupling, Bearings, Rotor, and Stator.
17	Pump coupling is physical.
18	Bearings is physical.
19	Rotor is physical.
20	Stator is physical.

21	Power Supply is environmental and physical.
22	Cooling System is physical.
23	Cooling System consists of Lubrication pump and Cooling fan.
24	Lubrication pump is physical.
25	Cooling fan is physical.
26	Accelerating is physical.
27	Accelerating requires Pump.
28	Accelerating consumes External low p flow.
29	Accelerating yields Internal high v flow.
30	Diffusing is physical.
31	Diffusing requires Diffuser.
32	Diffusing consumes Internal high v flow.
33	Diffusing yields External high p flow.
34	Containing is physical.
35	Containing requires Housing.
36	Containing affects Internal high v flow.
37	Driving is physical.
38	Driving requires Motor.
39	Driving affects Pump.
40	Powering is physical.
41	Powering requires Power Supply.
42	Powering affects Motor.
43	Cooling is physical.
44	Cooling requires Cooling System.
45	Cooling affects Motor.
46	Seal cooling affects Seal.
47	Component cooling water is environmental and physical.
48	Seal cooling requires Component cooling water.
49	Lubricating affects Pump coupling.
50	Lubricating affects Bearings.
51	Seal cooling requires ISO VG 100 oil.
52	Lubricating is physical.
53	Seal cooling is physical.
54	ISO VG 100 oil is physical.