Soft Computing Applications to Prognostics and Health Management (PHM): Leveraging Field Data and Domain Knowledge

Piero P. Bonissone and Naresh Iyer

General Electric Global Research, One Research Circle, K1 4A10A,
Niskayuna, NY 12309, USA
E-mail: bonissone@crd.ge.com

Abstract

Assets Prognostics and Health Management (PHM) is a promising application area for Soft Computing (SC). To better understand PHM requirements, we introduce a decision-making framework in which we analyze PHM decisional tasks. This framework is the cross product of the decision's *time horizon* and the *domain knowledge* used by SC models. Within such a framework, we analyze the progression from simple to annotated lexicon, morphology, syntax, semantics, and pragmatics. We use this metaphor to monitor the leverage of domain knowledge in SC to perform anomaly detection, anomaly identification, failure mode analysis (diagnostics), estimation of remaining useful life (prognostics), on-board control, and off board logistics actions. We illustrate a case study in anomaly detection, which is solved by the construction and fusion of an ensemble of diverse detectors, each of which is based on different SC technologies.

1 Introduction to PHM

Within the broad spectrum of Soft Computing (SC) applications, we will focus on Prognostics & Health Management (PHM) for assets such as locomotives, medical scanners, aircraft engines, etc. The main goal of PHM is to maintain these assets' operational performance over time, improving their utilization while minimizing their maintenance cost. This tradeoff is typical of long-term service agreements offered by OEM's to their valued customers.

1.2 PHM Decisional Tasks

After performing the usual preparation tasks, such as 1) *sensor validation*, and 2) input *data pre-processing*, we focus on PHM's five principal decisional tasks: 3) *anomaly detection and identification*; 4) *diagnostics*; 5) *prognostics*; 6) *fault accommodation*; and 7) *logistics decisions*. This is illustrated in Figure 1.

*Anomaly detectio*n leverages unsupervised learning techniques, such as clustering. Its goal is to extract the underlying structural information from the data, define normal structures and identify departures from such *normal* structures. *Diagnostics* leverages supervised learning techniques, such as classification. Its goal is to extract potential signatures from the data, which could be used to recognize different failure modes. *Prognostics* produces estimates of Remaining Useful Life (RUL). Its goal is to maintain and forecast the asset health index. Originally, this index reflects the expected deterioration under normal operating conditions. Later the index is modified by the occurrence of an anomaly/failure, reflecting faster RUL reductions.

All these functions are interpretations of the system's state. These interpretations lead to an *on-board control* action and an *off-board logistics*, repair and planning action. On-board control actions are usually focused on maintaining performance or safety margins, and are performed in real-time. Off-board maintenance/repair actions cover more complex offline decisions. They require a decision support system (DSS) performing multi-objective optimizations, exploring Pareto frontiers of corrective actions, and combining them with preference aggregations to generate the best decision tradeoffs.

Anomaly detection leverages unsupervised learning techniques, such as clustering. Its goal is to extract the underlying structural information from the data, define normal structures and identify departures from such normal structures. Diagnostics leverages supervised learning techniques, such as classification. Its goal is to extract potential signatures from the data, which could be used to recognize different failure modes. Prognostics produces estimates of Remaining Useful Life (RUL). Its goal is to maintain and forecast the asset health index. Originally, this index reflects the expected deterioration under normal operating conditions. Later the index is modified by the occurrence of an anomaly/failure, reflecting faster RUL reductions. All these functions are interpretations of the system's state. These interpretations lead to an on-board control action and an off-board logistics, repair and planning action. On-board control actions are usually focused on maintaining performance or safety margins, and

are performed in real-time. Off-board maintenance/repair actions cover more complex offline decisions. They require a decision support system (DSS) performing multi-objective optimizations, exploring Pareto frontiers of corrective actions, and combining them with preference aggregations to generate the best decision tradeoffs.

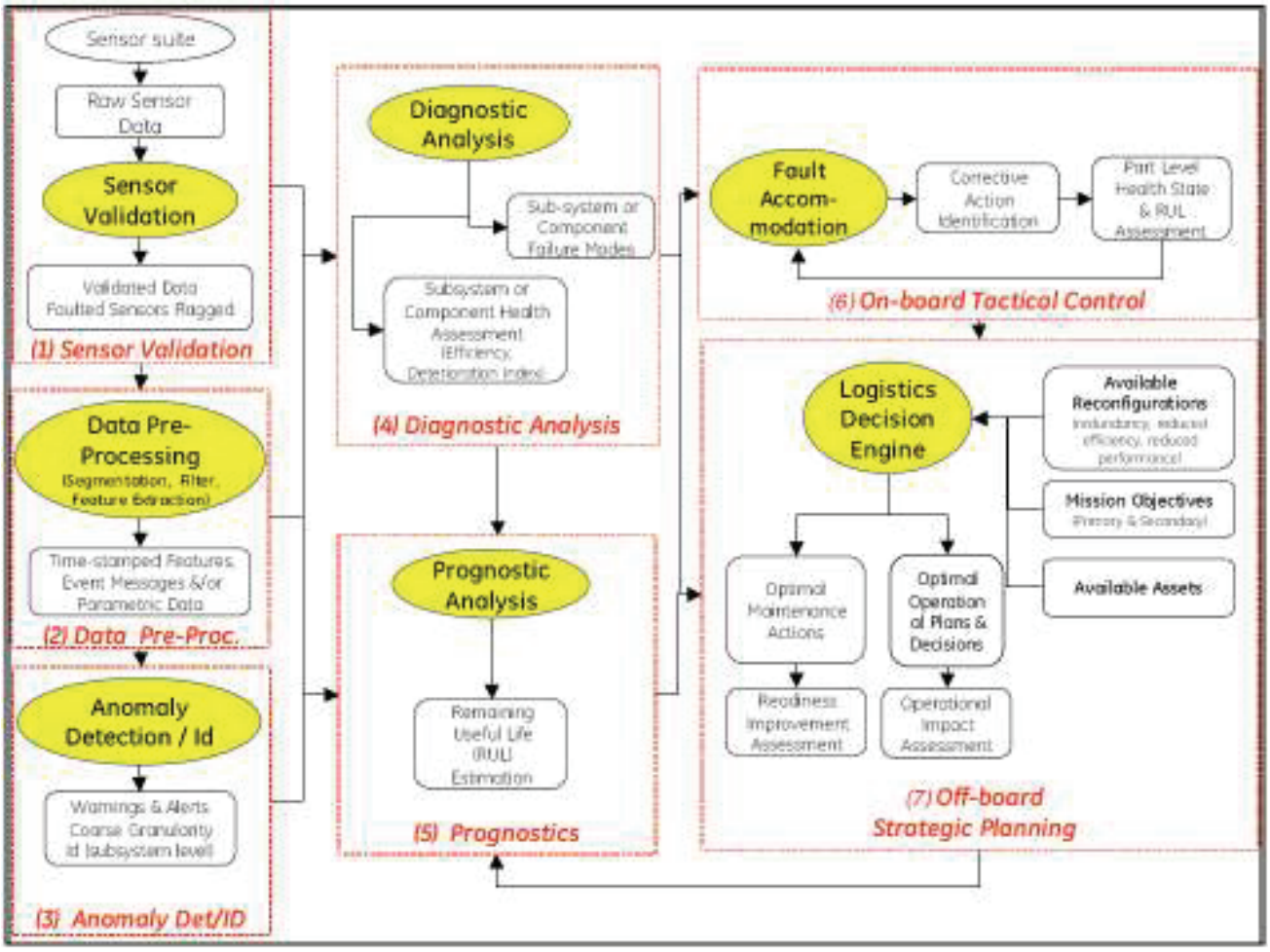

Figure 1. Functional Description of PHM Architecture

2 Decision Framework

The decision horizon of the applications defines the requirements for data quality and size, domain knowledge, problem complexity, and crispness of performance evaluation needed to automate the decision-making process. This process increases in scope and complexity as the decision horizon increases. To cope with such variety of requirements, we propose a temporal segmentation along the decision time horizon, as illustrated in Figure 2.

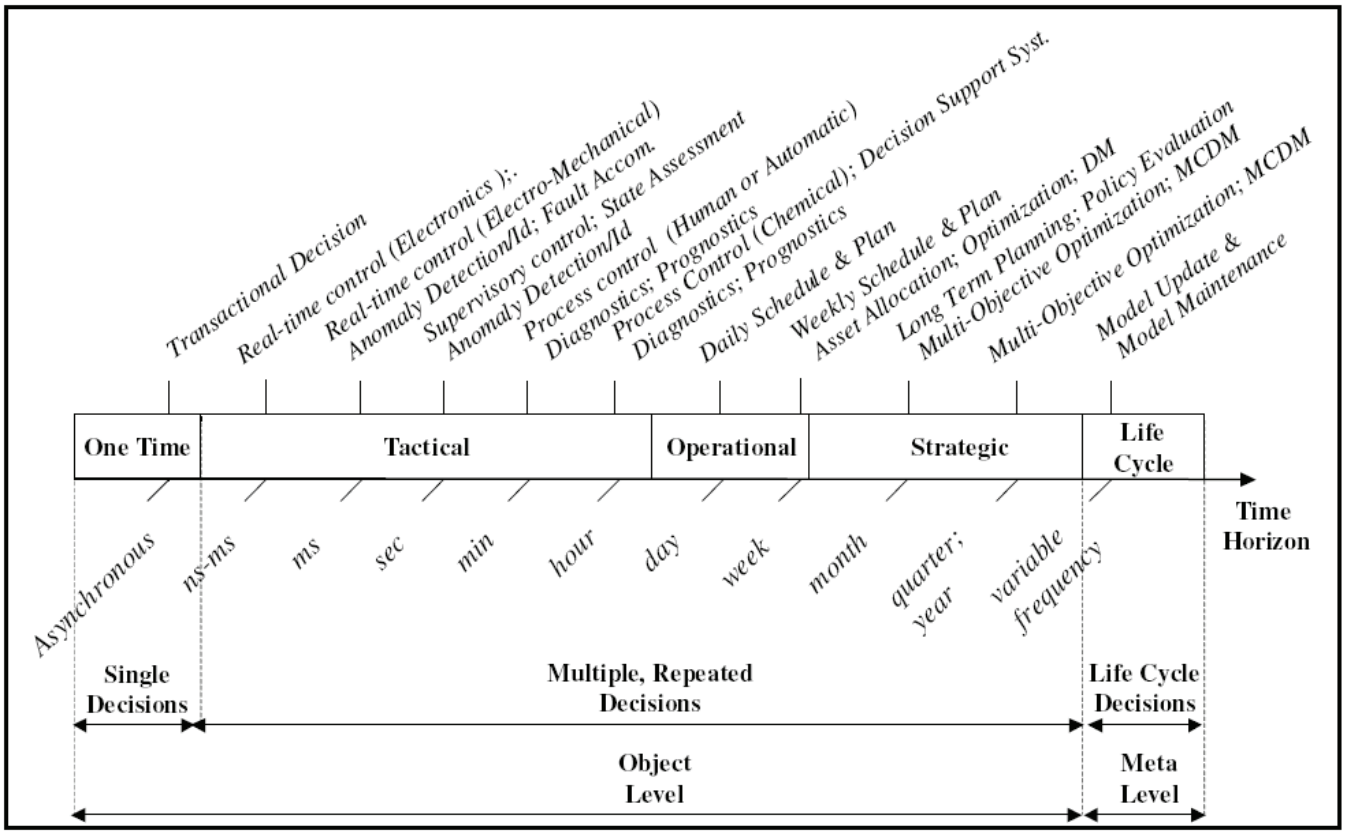

Figure 2. Temporal Segmentation

2.1 Temporal Segmentation

We can identify the following segments:

1) Single Decisions. These are situations in which data are collected once and decisions are made only once, on the basis of such information. For instance, this is typical of transactional applications, such as insurance underwriting.

2) Multiple, Repeated Decisions. These are situations in which data are collected synchronously and decisions must be taken, multiple times, usually within a similar time frame. Depending on their frequency, we could distinguish among three types of decisions: *tactical*, *operational*, and *strategic*.

Tactical decisions range from microseconds (for real-time control of electronics) to hours (for process control of chemical plants). The time scale of the underlying system dynamics determines the required decision frequency.

Operational decisions on the other hand can be taken daily or weekly (e.g. production schedule), while *strategic* decisions (e.g., company-wide maintenance have a horizon ranging from months to quarters.

3) Life Cycle Decisions. Finally we have a different type of temporal requirements to support the life cycle of these models. Unlike the previous decisions, which are made at the object-level, these decisions belong to the meta-level, as they try to assess and maintain the validity of the object-level models. These lifecycle issues have been discussed before by the author [1-3], [4] and due to space limitations, will not be part of this discussion.

In references [5-6] we first introduced a framework based on time and knowledge, to analyze decision-making problems addressed by Soft Computing techniques. Having analyzed the temporal horizon, which is the first dimension in this framework, we now analyze and categorize the domain knowledge used for such decisions.

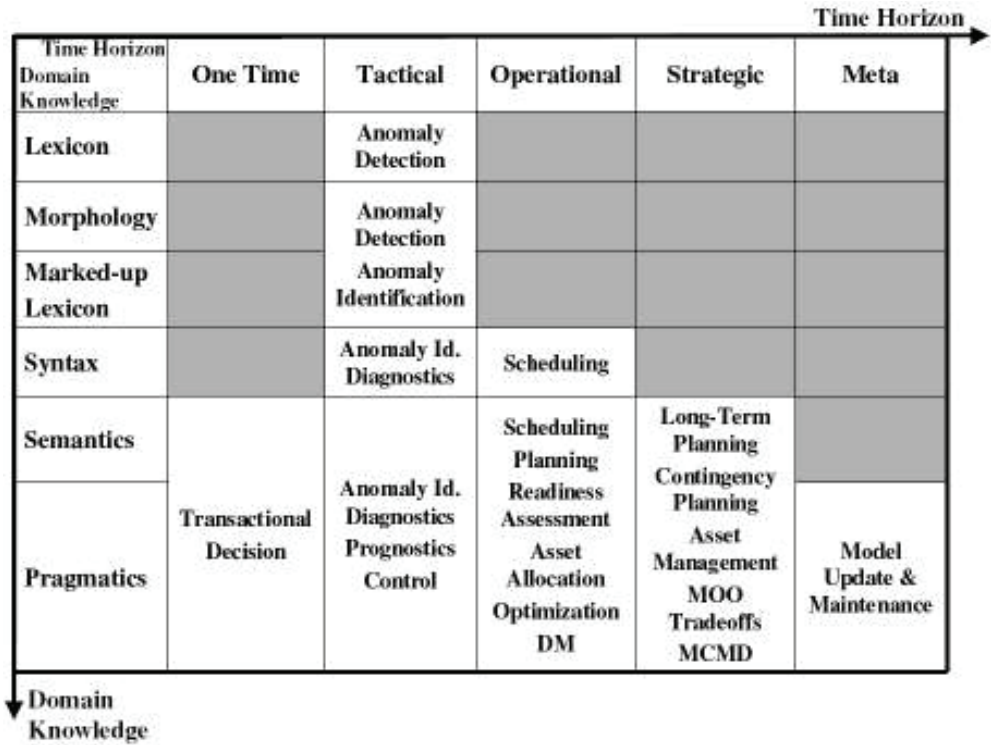

Figure 3. The Time x Knowledge Framework

2.2 Knowledge Segmentation: A Linguistic Metaphor

While it is relatively easy to measure and segment time, the same cannot be said when it comes to measuring, characterizing, and segmenting domain knowledge. The SC models used to address these PHM decisional tasks are based on the integration of domain knowledge with field data. To this end, we propose a linguistics metaphor for characterizing the type of domain knowledge exploited by a SC system. The correspondence between linguistic and PHM concepts is illustrated in Figure 4.

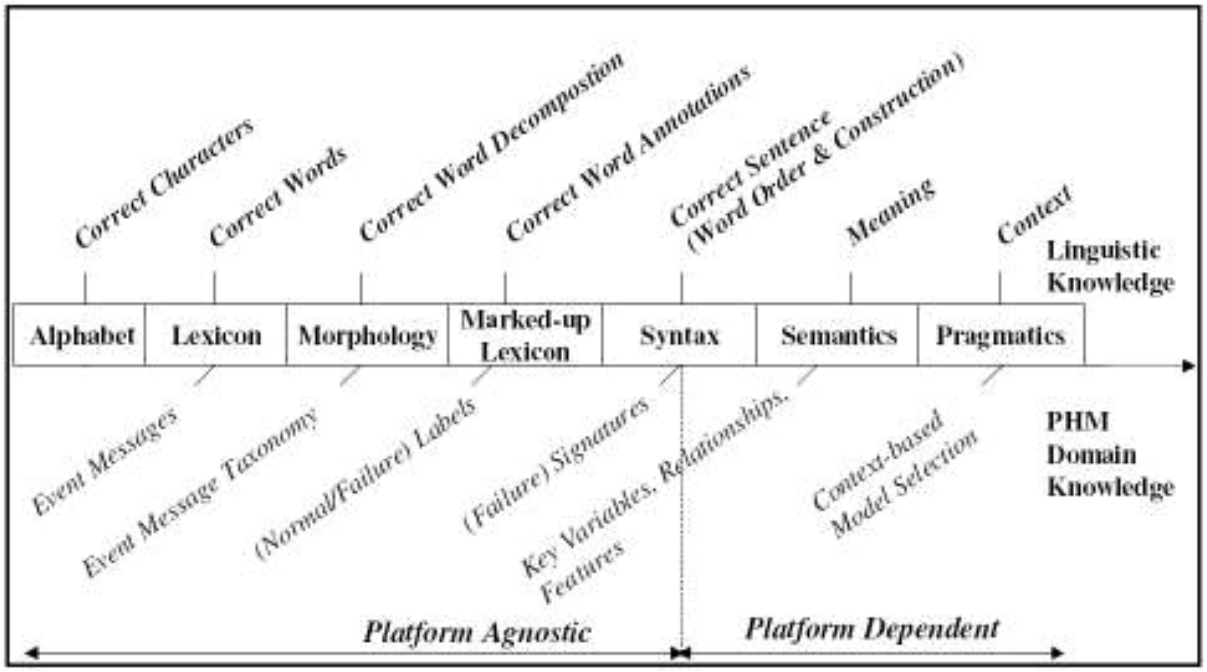

Figure 4. Knowledge Segmentation

In formal languages, the lowest level of information is the alphabet, representing the set of legal characters that compose each word in the language. The next level, which is our starting point in the metaphor, is the lexicon, i.e., the set of terminals in the language. This set represents the collection of all possible legal words that could Linguistic Knowledge (Failure) Signatures Key Variables, Relationships, Features Correct Characters Event Messages Event Message Taxonomy (Normal/Failure) Labels Alphabet Lexicon Syntax Semantics Correct Words Correct Word Decompostion Correct Sentence (Word Order & Construction) Correct Word Annotations Meaning Context Context-based Model Selection Morphology Pragmatics Marked-up Lexicon PHM Domain Knowledge Platform Agnostic Platform Dependent appear in a sentence of the language. In PHM, the lexical counter-part is the set of possible (normal or abnormal) event codes that could be recorded in the asset log. Without any additional knowledge about their relationships, order, meaning, etc., they are simply words recorded over time. This is the case of analyzing event logs without a data dictionary. Let us now introduce the concept of morphology, which describes the structures of words. Among these structures, the most common is that of word stem or root. The PHM counterpart of this concept is the taxonomy of event codes, which might indicate that all codes sharing the same initial symbol combinations refer to components belonging to the same system.

As we extend our metaphor, we observe that an annotated or marked-up lexicon provide us with a coarse classification of the words. Similarly, by labeling certain event codes or event logs (e.g. normal, type-1 abnormal, etc.) we produce the references required to label different types of anomalies. This can be used for training classifiers, such as neural networks, fuzzy-based classifiers, etc. As we increase the problem complexity, we need to leverage deeper domain knowledge.

The next level in our linguistic analogy is syntax, which defines the correct composition of language constituents (regardless of their meaning). Temporal information, either ordinal or cardinal, is important to define and identify signatures corresponding to different failure modes. The temporal pattern defining the presence or absence of events (or defining particular patterns over the values of parametric data) represents the core of the diagnostic analysis.

Further extending our metaphor, we introduce the concept of semantics, which decomposes the meaning of text into the meanings of its components and their relationships. This decomposition property is similar to that of system analysis. The PHM counterpart is the rich engineering knowledge, usually express by analytic, first-principles based models. Using this knowledge we identify key state variables, leverage their functional dependency to verify the correctness of other variables, extract the most informative features to create more compact representations, etc.

We conclude our metaphor with pragmatics, which use external, contextual knowledge to fully understand the meaning of our communication. While all prior levels dealt with information contained in the message itself (object-

level), pragmatics requires higher-level knowledge (meta-level) to provide the contextual information needed to disambiguate a sentence, correctly interpret its meaning, etc. In John Searle's speech acts [7], the speaker's communicative intentions, the social distance between the speaker and the listener, and other factors are leveraged to refine the meaning. The PHM counterpart of pragmatics is also at the meta-level. It consists in leveraging contextual information (such as operational regimes, environmental conditions, health deterioration) to determine the degree of applicability of local models and to select the best (or the best mixture). The selection of the most appropriate SC techniques, in conjunction with "sibling" disciplines, such as AI, Statistics, and Information Theory, depends on the type of available domain knowledge. Table I depicts the most useful SC approaches for different knowledge types, labeled according to the proposed linguistic metaphor. Note that only for tactical horizon applications we can develop models that are based on relatively shallow knowledge. In such cases, it is common to construct an ensemble of such models, ensuring their diversity (errors' uncorrelation) [8] and performing a fusion to increase the output's reliability. We also observe from the literature in Multi Classifier Systems [9] that model fusion is usually applied to tasks with small complexity outputs, such as classes or predicted values.

Table 1: Soft Computing Techniques & Domain Knowledge

SC/Stat/AI Techniques	Domain Knowledge
Self-Organizing Feature Maps, Kolmogorov complexity, 1-class SVM, NN, unsupervised ML techniques, fuzzy clustering, non-parametric statistics	Lexicon Morphology
Supervised ML techniques, NN, fuzzy classifiers, CART, Random Forest, MARS, BBN	Marked-up Lexicon
Automated kernel splitting, grammatical inference, evol. algorithms (EA)	Syntax
Feature extraction/selection, fuzzy models, 1st principle based simulations, temporal reasoners, case-based reasoners, planners, EA	Semantics
Model selection/mixing, EA, MOEA, fuzzy models for fusion, preference aggregation and tradeoffs	Pragmatics

As the time horizon increases, deeper domain knowledge is required to create the models, while the models outputs are usually complex (schedules, plans) and less suitable for fusion. Furthermore, the performance metrics associated with these tasks become less precise and more qualitative in nature. This characteristic is very suitable for the use of fuzzy system as possible fitness evaluator for the evolutionary algorithms that might be used to explore these models.

3 Case Study in SC applied to PHM: Anomaly Detection

Anomaly detection (AD) provides the focusing mechanism in PHM. Its goal is to generate reliable, early warning about the subset of assets that is deviating from its normal behavior. As mentioned in section 1, this is achieved by extracting underlying structural information from the data, defining normal structures, and identifying departures from such structures. When real-valued sensor data is available, this detection is performed by unsupervised learning models, such as clustering in feature spaces [10-11]. By combining statistics and engineering knowledge, we have developed clustering and classification techniques (such as neural networks, support vector machines, random forest, etc) to identify the possible outliers (novelties) that are likely to be the anomalies. These classifiers are anchored in the semantics of the problem domain, leveraging key state variables and functional relationships derived from causality or correlation. The feature spaces used by these models are usually defined by functions of numerical variables that characterize the asset or process in question.

3.1 Leveraging symbolic data

However, there are other sources of information that we want to tap to build our models. Many complex assets in today's world are designed to generate logs of non-parametric data like events or messages as they are operating – computers, software, automobiles, complex rotating machines like locomotives, aircraft and turbines, all record data-streams of various kinds during their operation. Examples include event-codes generated by built-in-test (BIT) systems that monitor operational capability of the unit at regular time-intervals; other examples include state information related to the unit's response to control actions initiated by a human or automatic controller on various subsystems of the unit. We call such information generated by a unit as non-parametric to distinguish it from continuous and higher-frequency data that is captured by sensors to record variables like operating temperature, pressure, and vibration.

In our approach, we will rely only on symbolic data, such as error or event messages and we will propose a solution at the lexical level - by comparing the presence/absence of event messages that characterize each unit's behavior - without resorting to semantics - by extracting features based on domain knowledge.

3.2 Problem Formulation

Each operational cycle of a unit can be represented in terms of an event-log, in which the unit records time-stamped event or fault messages. Each event could be a routine event (e.g. take-off for an aircraft) or an abnormal event (e.g., a mismatch in speeds of two coupled-wheels of a locomotive). For an operational cycle, the event log has a chronological list of all the events that occurred during the cycle. This event-log characterizes the behavior of that specific unit during that operational cycle (e.g. event-log generated by an aircraft during one or more successive flights). Without loss of generality, we will use the example of an aircraft to refer to a unit and we consider one operational cycle as a flight of the aircraft.

As shown in Figure 5, multiple flights of an aircraft can be represented in the form of a matrix containing the counts of the corresponding event messages generated by each flight.

EVENT MESSAGES	Flight # 1	Flight # 2	. . .	Flight # n
Event message 1	m(1, 1)	m(1, 2)		m(1, n)
Event message 2	m(2, 1)	m(2, 2)		m(2, n)
Event message 3				
. . .				
Event message f	m(f, 1)	m(f, 2)		m(f, n)
DURATION	d(1)	d(2)		d(n)

Figure 3. Matrix M representing the history of flights of the same aircraft. Each column shows the count of event messages for each flight

For the purposes of monitoring a fleet of units, like turbines or aircraft, the more precise goal is to provide an early warning of anomalous behavior among units in a fleet that are operating in a dynamic environment. Such early detection is necessary to be optimal in actions like pre-allocating additional maintenance resources, preparing for operational downtime, reducing secondary damage, or improving safety, and in general reduce O&S costs. We describe the use of a few anomaly detection techniques to assess changing behavior of aircraft operation based on analyzing event-logs over time. In all such techniques we compute pair-wise dissimilarities between two flights of the same aircraft, to determine the degree to which one flight's behavior differs from another based strictly on the content of their event logs.

3.3 Proposed SC Solution

If N is the total number of flights of the aircraft being considered, we generate an $N \times N$ similarity matrix indicating the pair-wise distances between all flights with respect to each other. We make use of various high dimensional visualization techniques to view this square similarity matrix so that the relative inter-flight relative similarities maybe visualized simultaneously allowing us to define regions of normal and abnormal flight behavior.

Our method requires the following steps:

1) Represent each flight by a string.

2) Create a dissimilarity matrix, containing pairwise distances between strings (flights).

3) Project the dissimilarity matrix onto a lower dimensional space, minimizing projection distortion.

4) Cluster the projected points, identifying the clusters representing normal operations and the ones representing anomalies.

To illustrate this concept, let us consider an aircraft flying according to its operational schedule. As described earlier, each aircraft has an event-log, in which the aircraft computer systems record time-stamped event messages. At the end of the flight, the event log has a chronological list of all the events that occurred during the flight. This event-log characterizes the behavior of that aircraft during that flight. We can represent each aircraft flight as an object in an event space in which the aircraft's behavior is compared to previous flights.

Step 1. We create a matrix M, of dimension $[f+1, n]$, where f is the number of different messages recorded over all flights, and n is the total number of flights. Each column represents the count of different messages recorded during that flight. At the end of each column we also record the flight duration. Figure 5 shows an example of Matrix M.

The entry $m(i,j)$ represents the number of occurrences of fault code "i", during flight "j". We normalize the counts by the flight duration, $d(j)$, obtaining a ratio of count per hour flight. This will allow us to compare message logs collected over flights of different durations. We will refer to this normalized count as:

$$\hat{m}(i,j) = \frac{m(i,j)}{d(j)} \qquad (1)$$

Finally we change these counts into frequencies, so that we have a density for each flight. Thus, the normalized entry $freq(i,j)$ is computed as:

$$freq(i,j) = \frac{\hat{m}(i,j)}{\sum_{i=1}^{f} \hat{m}(i,j)} \qquad (2)$$

At this point, we can now represent each flight *"j"* by a string x_j containing f frequencies:

$$\boldsymbol{x_j} = [freq(1,j), freq(2,j), \ldots freq(f,j)] \qquad (3)$$

Step 2. For each pair of flights (x_i, x_j) we compute similarity using multiple similarity metrics, resulting in several measures from which anomalous flights might be detected. In this paper we illustrate the Normalized Compression Distance (NCD), which is a surrogate for the Normalized Information Distance, based on Kolmogorov complexity:

$$NCD(x_i, x_j) = \frac{C(x_i, x_j) - \min\{C(x_i), C(x_j)\}}{\max\{C(x_i), C(x_j)\}} \qquad (4)$$

where *C(x)* denotes the length of the compressed string "*x*", using a standard compressor. Suppose that *C(x)* < *C(y)*. Then, the metric *NCD(x,y)* captures the improvement due to compressing string "*y*" using string "*x*" as the previously compressed database (numerator), with compressing string "*y*" from scratch (denominator). This distance has been used to create static classification, affinity groups in music (showing musical similarities/differences of various composers), linguistic taxonomies (showing the hierarchical grouping of many natural languages), and biological taxonomies (showing the hierarchical grouping of animals based on DNA similarities) [12-14]. In our application we use it to track the behavior of a fleet of aircrafts, whose state changes over time, and to update their similarity-based clustering. The dissimilarity matrix $D(x_i, x_j) = [NCD(x_i,x_j)]$ is a n by n matrix. Each entry is a normalized distance value in the interval [0,1]. When $i=j$, $NCD(x_i, x_i) = 0$. Also, $NCD(x_i, x_i) = NCD(x_j, x_i)$, so matrix *D* is symmetric, with 0 diagonal. With the NCD we build a square matrix *D* as mentioned earlier representing all the inter-flight similarities (or conversely distances) for the set of flights considered.

In another paper [15] we illustrate other distances which we also used to created the dissimilarity matrix D. and, in the process, an ensemble of diverse detectors:

- Self-organizing feature maps (SOFM) [16]
- neural network models, which perform two essential functions: vector quantization and vector projection.
- Random forest-based dissimilarity measures [17]
- a classification method that applies bagging [18] to a variation of classification trees [19] for supervised or unsupervised [20].
- Histogram matching measures, such as Jeffrey-Kullback-Leibler Divergence, and Chi-Square.

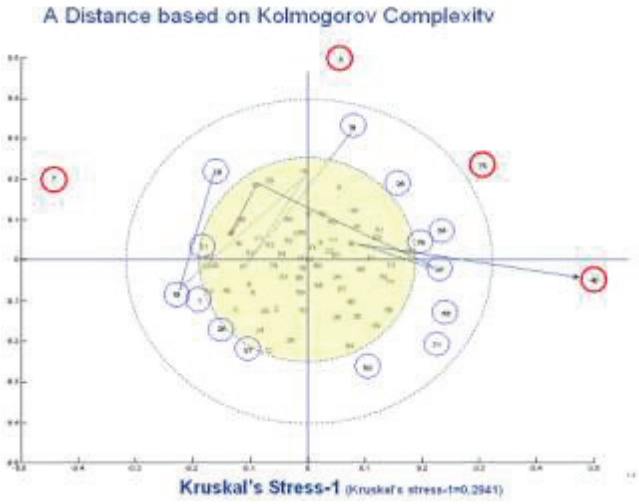

Figure 6: 2D Projection of Inter-flight Distances

Step 3. Next we visualize the content of the matrix *D* by projecting it onto a 2- dimensional space. There are many ways to implement this projection, for instance by using Multi Dimensional Scaling (MDS). MDS is a well-known statistical technique for visualizing multi-dimensional data. This technique tries to minimize the overall distortion (minimizing Kruskal's Stress-1 criterion) between distances in the original and the lower-dimensional space. Figure 3 shows the use of the MDS approach to perform this projection.

Step 4. From Figure 3, it is possible to see that the region of normality is centered on the origin, and that larger distances from the origin represent flights that are quite different from normal operations, and therefore are possible anomalies. By continuously monitoring and tracking new flights, computing the NCD with previous flights, and projecting them onto this 2-dimensional map, we can classify those flights that are potential anomalies.

4 Conclusions and Future Work

We have introduced assets Prognostics and Health Management as a promising application area for a variety of Soft Computing techniques. We have proposed a framework, based on decision-horizon and domain knowledge, to analyze many PHM-related tasks. Depending on the depth of domain knowledge leveraged in the construction of the SC models we have ranged from platform-agnostics to platform specific models. The former, such as lexical-level anomaly detectors, are week models that are applicable across multiple assets. The latter, such as first-principle based models, are powerful models that are limited in their applicability scope.

We have illustrated an anomaly detector based on the Normalized Compression Distance (NCD). By using file compressors that rely on different compression schemes, we have built an ensemble of six different detectors, with enough diversity to make them suitable for fusion. In reference [15], we provide a detailed description of such fusion.

This paper has focused on anomaly detection models based on the weakest kind of knowledge (syntactical and taxonomic) and the simplest kind of data (symbolic event codes) To implement more demanding PHM functions, we will need to augment our input with time-sampled sensor data and resort to other techniques – listed in Table I – that will leverage a deeper domain knowledge.

References

[1] Bonissone, P.: The life cycle of a fuzzy knowledge-based classifier. Proc. North American Fuzzy Information Processing Society, Chicago, IL, USA (2003) 488-494

[2] Bonissone, P.: Development and Maintenance of Fuzzy Models in Financial Applications. In Lopez-Diaz, Gil, Grzegorzewski, Hyrniewicz, Lawry (Eds.): Soft Methodology and Random Information Systems, Springer (2004)
[3] Bonissone, P., Varma, A., Aggour, K.: An Evolutionary Process for Designing and Maintaining a Fuzzy Instance-based Model (FIM). Proc. First Workshop on Genetic Fuzzy Systems (GFS'05), Granada, Spain (2005)
[4] Patterson, A., Bonissone, P., Pavese, M.: Six Sigma Quality Applied Throughout the Lifecycle of and Automated Decision System. Journal Quality and Reliability Engineering International, 21, (2005) 275-292
[5] Bonissone, P.: Knowledge and Time: A Framework for Soft Computing Applications in Prognostics and Health Management (PHM). Proc. IPMU 2006, Paris, France (2006)
[6] Bonissone, P., Goebel, K., Iyer. N.: Knowledge and Time: Selected Case Studies in Prognostics and Health Management (PHM), Proc. IPMU 2006, Paris, France (2006)
[7] Searle, J. R.: Speech Acts. An Essay in the Philosophy of Language. Cambridge (1969).
[8] Kuncheva, L., Whitaker, C.J.: Measures of Diversity in Classifier Ensembles. Machine Learning, 51 (2003) 181–207
[9] Roli, F., Giacinto, G., Vernazza, G.: Methods for Designing Multiple Classifier Systems. MCS 2001, LNCS 2096, (2001) 78-87
[10] Markou, M., Singh, S.: Novelty detection: A review - Part 1: Statistical approaches. Signal Processing, vol. 83, no. 12, (2003) 2481-2497
[11] Markou, M., Singh, S.: Novelty detection: A review - Part 2: Neural network based approaches. Signal Processing, vol. 83, no. 12, (2003) 2499-2521
[12] Cilibrasi, R., Vitany, P.: Clustering by Compression. IEEE Transactions on Information Theory, Vol. 51(4), (2005) 1523-1545
[13] Li, M., Chen, X., Ma, B., Vitany, P.: The Similarity Metric. IEEE Transactions on Information Theory, Vol. 50(12), (2004)
[14] Cilibrasi, R., Vitany, P., de Wolf, R.: Algorithmic Clustering of Music Based on String Compression. Computer Music Journal, 28:4, (2004) 49–67
[15] Varma, A., Bonissone, P., Yan, W., Eklund, N., Goebel, K., Yier, N., Bonissone, S.: Anomaly Detection using Non-Parametric information. Proceedings of GT2007 ASME Turbo Expo 2007: Power for Land, Sea and Air, Montreal, Canada (2007)
[16] Kohonen, T., Oja, E., Simula, O., Visa, A., Kangas, J.: Engineering applications of the self-organizing map", Proceedings of the IEEE, vol. 84, no. 10, (1996) 1358-1384
[17] Breiman, L. Random forests. Machine Learning, 45(1), (2001) 5–32
[18] Breiman, L. Bagging predictors. Machine Learning, 24(2), (1996) 123–140
[19] Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth, Belmont, CA, (1984) 20.Shi, T., Horvath, S.: Unsupervised Learning with Random Forest Predictors. Journal of Computational and Graphical Statistics. Vol. 15, no. 1, (2006) 118-138(21)