System-level Prognostics for the National Airspace

Matthew Daigle[1], Shankar Sankararaman[2], and Indranil Roychoudhury[3]

[1] *NASA Ames Research Center, Moffett Field, CA 94035, USA*
matthew.j.daigle@nasa.gov

[2,3] *SGT, Inc., NASA Ames Research Center, Moffett Field, CA 94035, USA*
shankar.sankararaman@nasa.gov
indranil.roychoudhury@nasa.gov

Abstract

In the National Airspace System (NAS), safety is assured through a set of rules, regulations, and procedures to respond to unsafe events. However, safety stands to benefit immensely from the introduction of tools and methodologies from Prognostics and Health Management (PHM). PHM will enable the NAS to stochastically predict unsafe states within the NAS, enabling a proactive preventative response strategy, as opposed to a reactive mitigative one. However, current PHM methods do not directly apply to the NAS for several reasons: they typically apply only at the component level, are implemented in a centralized manner, and are focused only on predicting remaining useful life. In this paper, we extend the model-based prognostics approach to PHM in order to provide a framework that can be applied to the NAS. We offer a system-level approach that supports a distributed implementation, and provide algorithms to predict the probability of an unsafe state, either at a specific time or within a time interval, and to predict the time of an unsafe state. Experimental results in simulation demonstrate the new approach.

1. Introduction

In the National Airspace System (NAS), unsafe situations and events are avoided through an established set of rules, regulations, and procedures, as well as timely actions by pilots, controllers, and other NAS operators (Wells, 2001). When unsafe situations arise, decisions are made *reactively* in order to mitigate the threat and return the system to safety. For example, if two aircraft become too close (known as "loss of separation"), controllers provide instructions to pilots to resolve the conflict. However, both safety and efficiency can be improved significantly if predictive knowledge is used, so that unsafe situations can be avoided *preemptively*, before they even arise.

Thus, the NAS stands to benefit significantly from the tools and methodologies developed within the field of Prognostics and Health Management (PHM). The current set of predictive tools used by the NAS are often ad-hoc, and do not take into account uncertainty, and, if they do, not in a mathematically systematic manner. Predictions are often deterministic, and hence subject to errors and poor decisions based on uninformed confidence in predictions.

However, the methods within PHM cannot be directly applied to prognostics problems in the NAS for two main reasons. First, PHM methods, with some exceptions (Daigle, Bregon, & Roychoudhury, 2012; Khorasgani, Biswas, & Sankararaman, 2016), are focused on component-level methods, and not system-level methods. In the NAS, we want to predict at both the component-level (aircraft, storm systems), and the system-level (the airspace, at multiple levels). In such a large-scale system-level prognostics problem, distributed prognostics approaches become especially critical (Daigle, Bregon, & Roychoudhury, 2014). Second, PHM methods are focused on predicting failure. In the NAS, there are many other kinds of unsafe situations and events that need to be predicted (Roychoudhury et al., 2016). We need to be able to specify many classes of such situations and predict them simultaneously. Further, we are interested in computing not only the time to/of some unsafe situation, but also the probability of such a situation occurring, both at a specific time and within a time interval of interest.

So, in order to apply prognostics to the NAS, we require an extended, more general mathematical formulation of prognostics. Initial work has been presented in (Roychoudhury et al., 2015, 2016; Daigle, Roychoudhury, & Bregon, 2015), in which a centralized system-level prognostics approach focused on the computation of *safety metrics* was developed. In this paper, we further generalize and extend the prognostics approach, so that it is applicable to other domains within PHM. Using this new framework, we define several prognostics problems, and develop approximate algorithms to solve them in a distributed framework. These algorithms are then applied to the NAS in a simulated scenario to demonstrate the new framework.

The paper is organized as follows. Section 2 extends and generalizes the standard model-based prognostics framework, and defines several prognostics problems within this new framework. Section 3 discusses the prediction algorithms and the treatment of uncertainty. Section 4 presents the distributed approach. Section 5 applies the new framework to the prediction of loss of separation and low fuel events to the NAS in simulation. Section 6 concludes the paper.

2. PROBLEM FORMULATION

In prognostics, given a system, we are interested in how the *state* of the system, $\mathbf{x}$, will evolve in time. Specifically, we are interested in determining whether some subset of the state space, $\mathcal{X}$, will be reached in some finite time, and, if so, *when* it will be reached. Often, in PHM, $\mathcal{X}$ specifically represents failure states, and the earliest time at which a failure state is reached is the end of life (EOL). The distinction between nonfailure and failure states is captured using a *threshold function*, e.g., a battery is at EOL when its capacity reaches some prescribed minimum value.

In general, however, there may be many different subsets of the state space that are of interest, and may refer to failure and nonfailure states, safe and unsafe states, different types of unsafe states, etc. Further, the function that classifies a state as belonging to a specific subset is not necessarily a threshold. That is, it may not be as simple as comparing to a single value, but can be any arbitrary function. As such, the name "threshold function" can be misleading.

Generalizing these concepts, given a system with state $\mathbf{x} \in \mathbb{R}^{n_x}$, we define a set of labels $\mathbb{L}$ that apply to the state space. For example, we may have $\mathbb{L} = \{\mathit{nonfailure}, \mathit{failure}\}$, or $\mathbb{L} = \{\mathit{safe}, \mathit{unsafe}\}$. Note that the labeling set can have any number of elements. For each label $l \in \mathbb{L}$, we then introduce a Boolean labeling function, $\mathbf{l}_l : \mathbb{R}^{n_x} \to \{\mathit{true}, \mathit{false}\}$, that maps a given state to true if the label l applies to a given state $\mathbf{x}$, and false otherwise. Using the labeling functions, we can define corresponding subsets of the state space, i.e., for a label l, we define $\mathcal{X}_l = \{\mathbf{x} : \mathbf{l}_l(\mathbf{x}) = \mathit{true}\}$. These labels do not need to be defined such that they partition the state space; many labels may apply to the same state.

For prediction, we are fundamentally interested in when, for some label l, the current state of the system will evolve into some other state that belongs to $\mathcal{X}_l$. Following the model-based prognostics paradigm, we require a state equation:

$$\mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k), \mathbf{v}(k)), \tag{1}$$

where $k \in \mathbb{N}$ is the discrete time variable, $\mathbf{x}(k) \in \mathbb{R}^{n_x}$ is the state vector, $\mathbf{u}(k) \in \mathbb{R}^{n_u}$ is the input vector, $\mathbf{v}(k) \in \mathbb{R}^{n_v}$ is the process noise vector, and $\mathbf{f}$ is the state update function.

The inputs to the prediction problem are the following (Sankararaman, Daigle, & Goebel, 2014):

1. a time of prediction, k_o;
2. a time horizon of prediction, $[k_o, k_h]$;
3. the initial state probability distribution, $p(\mathbf{x}_o(k_o))$;
4. the future input trajectory distribution, $p(\mathbf{U}_{k_o,k_h})$, where $\mathbf{U}_{k_o,k_h} = [\mathbf{u}(k_o), \mathbf{u}(k_o+1), \ldots, \mathbf{u}(k_h)]$; and
5. the future process noise trajectory distribution, $p(\mathbf{V}_{k_o,k_h})$, where $\mathbf{V}_{k_o,k_h} = [\mathbf{v}(k_o), \mathbf{v}(k_o + 1), \ldots, \mathbf{v}(k_h)]$.

These inputs may change for every new time of prediction.

Note that the following notation is adopted. For a vector $\mathbf{a}$, a trajectory of that vector over a time interval $[k_o, k_h]$ is denoted by $\mathbf{A}_{k_o,k_h}$, and $\mathbf{A}_{k_o,k_h}(k) = \mathbf{a}(k)$, for $k \in [k_o, k_h]$.

The core problem of prognostics is to predict the future states of the system within a time interval:

Problem 1. Given a time interval $[k_o, k_h]$, an initial state $p(\mathbf{x}(k_o))$, process noise $p(\mathbf{V}_{k_o,k_h})$, and future inputs $p(\mathbf{U}_{k_o,k_h})$, determine $p(\mathbf{X}_{k_o,k_h})$.

A more specific problem is to compute which of these states belong to $\mathcal{X}_l$ for a given label l, and determine the associated probability of reaching such a state at a particular time:

Problem 2. Given a label l, a time interval $[k_o, k_h]$, an initial state $p(\mathbf{x}(k_o))$, process noise $p(\mathbf{V}_{k_o,k_h})$, and future inputs $p(\mathbf{U}_{k_o,k_h})$, determine the probability that some state $\mathbf{x} \in \mathcal{X}_l$ will be reached at a given time $k \in [k_o, k_h]$.

For example, if $l = \mathit{failure}$, then we would be computing the probability of failure occurring at a particular time.

A related problem is to compute the probability of reaching a state with a given label at *any* time within a given time interval:

Problem 3. Given a label l, a time interval $[k_o, k_h]$, an initial state $p(\mathbf{x}(k_o))$, process noise $p(\mathbf{V}_{k_o,k_h})$, and future inputs $p(\mathbf{U}_{k_o,k_h})$, determine the probability of reaching some state $\mathbf{x} \in \mathcal{X}_l$ for any time $k \in [k_o, k_h]$.

For example, if $l = \mathit{failure}$ and $k_h - k_o$ is one hour, then we would be computing the probability that the system will fail within the next hour.

The final problem is to compute *when* a state with a given

Algorithm 1 `Simulate`

```
1: Inputs: $k_o, k_h, \mathbf{x}(k_o), \mathbf{U}_{k_o,k_h}, \mathbf{V}_{k_o,k_h}$
2: Outputs: $\mathbf{X}_{k_o,k_h}$
3: $\mathbf{X}_{k_o,k_h}(k_o) \leftarrow \mathbf{x}(k_o)$
4: for $k = k_o$ to $k_h - 1$ do
5:     $\mathbf{X}_{k_o,k_h}(k+1) \leftarrow \mathbf{f}(\mathbf{X}_{k_o,k_h}(k), \mathbf{U}_{k_o,k_h}(k), \mathbf{V}_{k_o,k_h}(k))$
6: end for
```

label will be reached. For a label l, we define k_l as follows:

$$k_l(k) = \min\{k' : k' \geq k \textbf{ and } \mathbf{x}(k) \in \mathcal{X}_l\}, \qquad (2)$$

i.e., k_l is the earliest time point at which the system state is assigned the label l. This is a generalization of EOL; if $l = \textit{failure}$, then k_l is EOL, and $k_l - k_o$ is remaining useful life (RUL). The problem of computing k_l is then defined as follows:

Problem 4. Given a label l, a time interval $[k_o, k_h]$, an initial state $p(\mathbf{x}(k_o))$, process noise $p(\mathbf{V}_{k_o,k_h})$, and future inputs $p(\mathbf{U}_{k_o,k_h})$, determine $p(k_l)$.

3. PREDICTION

The solutions to Problems 2 to 4 can be derived given the solution to Problem 1. It is well-known that in many cases, there is no closed-form solution to Problem 1, and so, in general, only approximate algorithms are available (Sankararaman & Goebel, 2013). In this paper, we focus on algorithms based on sampling (Daigle, Saxena, & Goebel, 2012; Robert & Casella, 2004; Glynn & Iglehart, 1989). How the samples are generated is not critical to the presentation of the algorithms, and so we use a generic `GenerateSamples` algorithm in the following presentation, which returns a set of equally weighted samples. With minor modifications, the algorithms can be extended to handle nonequally weighted samples, however to simplify the presentation here we assume equal sample weights.

For sampling-based prediction algorithms, approximate solutions to the problems defined in Section 2 are quite straightforward. Algorithm 1 computes a single state trajectory, $\mathbf{X}_{k_o,k_h}$, given realizations (samples) of the initial state, $\mathbf{x}(k_o)$, the future input trajectory, $\mathbf{U}_{k_o}^{k_h}$, and the future process noise trajectory, $\mathbf{V}_{k_o}^{k_h}$, using the state equation.

Algorithm 2 solves Problem 1 using Algorithm 1. Using `GenerateSamples`, realizations for the different inputs to Algorithm 1 are generated, and Algorithm 1 is called for each. The probability distribution $p(\mathbf{X}_{k_o,k_h})$ is defined by the set of samples of state trajectories that are produced. Here, we denote this set using $\{\mathbf{X}^i_{k_o,k_h}\}_{i=1}^N$, where the i superscript denotes sample i, and N is the total number of samples.

Algorithm 3 offers a solution to Problem 2 given the result of Algorithm 2. It checks all trajectories in the given set of trajectory samples, and counts how many of the trajectories

Algorithm 2 `StatePrediction`

```
1: Inputs: $k_o, k_h, p(\mathbf{x}(k_o)), p(\mathbf{U}_{k_o,k_h}), p(\mathbf{V}_{k_o,k_h})$
2: Outputs: $\{\mathbf{X}^i_{k_o,k_h}\}_{i=1}^N$
3: $\{(\mathbf{x}(k_o)^i, \mathbf{U}^i_{k_o,k_h}, \mathbf{V}^i_{k_o,k_h})\}_{i=1}^N \leftarrow$ GenerateSamples $(N, p(\mathbf{x}(k_o)), p(\mathbf{U}_{k_o,k_h}), p(\mathbf{V}_{k_o,k_h}))$
4: for $i = 1$ to $N$ do
5:     $\mathbf{X}^i_{k_o,k_h} \leftarrow$ Simulate$(k_o, k_h, \mathbf{x}(k_o)^i, \mathbf{U}^i_{k_o,k_h}, \mathbf{V}^i_{k_o,k_h})$
6: end for
```

Algorithm 3 `ProbabilityLAtK`

```
1: Inputs: $k, l, \{\mathbf{X}^i_{k_o,k_h}\}_{i=1}^N$
2: Outputs: $p(\mathbf{x}(k) \in \mathcal{X}_l)$
3: $n_l \leftarrow 0$
4: for all $\mathbf{X}^i_{k_o,k_h} \in \{\mathbf{X}^i_{k_o,k_h}\}_{i=1}^N$ do
5:     if $\mathbf{X}^i_{k_o,k_h}(k) \in \mathcal{X}_l$ then
6:         $n_l \leftarrow n_l + 1$
7:     end if
8: end for
9: $p(\mathbf{x}(k) \in \mathcal{X}_l) \leftarrow n_l/N$
```

have a state that belong to $\mathcal{X}_l$ for time k. The probability that a state at time k will have that label is that number over the total number of samples. Clearly, the higher the value of N, the more accurate the result will be.

Algorithm 4 offers a solution to Problem 3 given the result of Algorithm 2. It is similar to Algorithm 3, but checks only that there is at least one state in each trajectory that belongs to $\mathcal{X}_l$.

Algorithm 5 offers a solution to Problem 4 given the result of Algorithm 2. It is similar to Algorithm 4, but stores instead the time value when the first state in each trajetory that belongs to $\mathcal{X}_l$ is found. The probability distribution is represented by the set of samples of k_l, $\{k_l^i\}_{i=1}^N$.

4. DISTRIBUTED PREDICTION

The larger the system, the larger the state vector, and the higher the computational requirements of the algorithms presented in Section 3. For a large system consisting of many components, the centralized approach does not scale well. To address this issue, one may introduce a distributed approach (Daigle, Bregon, & Roychoudhury, 2012; Daigle et al., 2014).

Algorithm 4 `ProbabilityL`

```
1: Inputs: $k, l, \{\mathbf{X}^i_{k_o,k_h}\}_{i=1}^N$
2: Outputs: $p(\exists k \in [k_o, k_h] \mathbf{X}_{k_o,k_h}(k) \in \mathcal{X}_l)$
3: $n_l \leftarrow 0$
4: for all $\mathbf{X}^i_{k_o,k_h} \in \{\mathbf{X}^i_{k_o,k_h}\}_{i=1}^N$ do
5:     for all $k \in [k_o, k_h]$ do
6:         if $\mathbf{X}^i_{k_o,k_h}(k) \in \mathcal{X}_l$ then
7:             $n_l \leftarrow n_l + 1$
8:             break
9:         end if
10:    end for
11: end for
12: $p(\exists k \in [k_o, k_h], \mathbf{X}_{k_o,k_h}(k) \in \mathcal{X}_l) \leftarrow n_l/N$
```

Algorithm 5 `ProbabilityKl`

```
1: Inputs: $k, l, \{\mathbf{X}^i_{k_o,k_h}\}_{i=1}^N$
2: Outputs: $\{k^i_l\}_{i=1}^N$
3: for all $\mathbf{X}^i_{k_o,k_h} \in \{\mathbf{X}^i_{k_o,k_h}\}_{i=1}^N$ do
4:     for all $k \in [k_o, k_h]$ do
5:         if $\mathbf{X}^i_{k_o,k_h}(k) \in \mathcal{X}_l$ then
6:             $k^i_l \leftarrow k$
7:             break
8:         end if
9:     end for
10:    $k^i_l \leftarrow \infty$
11: end for
```

One way to distribute the prediction task for a model-based approach is by distributing the underlying model, for example, through structural model decomposition (Roychoudhury, Daigle, Bregon, & Pulido, 2013). Structural model decomposition transforms a given system model into a set of *computationally independent* submodels, each with its own state vector and input vector. Thus, we can execute Algorithm 2 for each *local* state vector, and the combination of all the predictions provides the predictions for the *global* state vector.

For the remaining algorithms, whether or not it can be distributed depends on the particular label. Some labels may apply only to local state vectors, in which case the distribution is trivial, given the distributed state trajectory predictions. In other cases, labels may apply to several local state-vectors, i.e., the capture a system-level property. In this case, the state predictions can be done independently, in a distributed fashion, and then combined into a joint state vector to check the labels.

5. CASE STUDY: THE NATIONAL AIRSPACE SYSTEM

In applying prognostics to the NAS, we want to ask the following questions (Roychoudhury et al., 2016):

1. What is the probability of an unsafe state being reached at a given future time (Problem 2)?
2. What is the probability of an unsafe state being reached within a given future time interval (Problem 3)?
3. What kind of unsafe state will be reached (Problem 1)?
4. When will an unsafe state be reached (Problem 4)?

In this section, we apply the prognostics framework and algorithms developed in this paper to the problem of prognostics in the NAS. We first describe how the system is modeled, which includes definition of the state and input vectors, the state equation, the set of labels, and the set of labeling functions. We then demonstrate how the framework can be used to answer the preceding questions using simulation-based experiments.

5.1. Modeling the NAS

The NAS is made up of many interacting components: aircraft, weather systems, pilots, controllers, etc. For the purpose of predicting unsafe states, we consider only the open-loop case, where aircraft are operating independently of each other and controllers are not interfering with their flight paths. If the prognostics algorithms predict an unsafe state with a high probability, then this information would be provided to pilots/controllers or automated systems for decision-making. For the purposes of this paper, and in order to focus our demonstration, we consider only aircraft as the set of components that comprise the NAS.

Because the aircraft are dynamically independent, the prediction problem distributes naturally. For each aircraft, we have an individual model to predict its future trajectory. From a system-level perspective, we can then check various safety properties based on these independently computed trajectories. In practice, one would directly enable aircraft with this predictive capability. Safety properties that apply only to a single aircraft can be computed on that aircraft, and safety properties that apply to sets of aircraft can be assigned to different aircraft, computed at a regional level, or even in the cloud. Such an approach then scales easily as most of the computation is distributed to individual aircraft. Adding an additional aircraft to the system does not significantly increase the computation that must be done at a higher level.

We first describe the models we use for individual aircraft, followed by a description of some salient safety properties, and the labels and labeling functions that define them.

5.1.1. Aircraft Modeling

We use kinematic models of aircraft navigation with simplified dynamics and control, similar to the models developed by others (Bilmoria, Banavar, Chatterji, Sheth, & Grabbe, 2000; Chatterji, Sridhar, & Bilimoria, 1996; Tandale, Wiraatmadja, Menon, & Rios, 2011). In the following description, we present differential equations in continuous time t; for implementation purposes, they are converted to difference equations using a sampling time of 10 s. The aircraft state vector is defined as

$$\mathbf{x}(t) = \begin{bmatrix} V_a(t) \\ V_z(t) \\ \chi_a(t) \\ h(t) \\ \lambda(t) \\ \tau(t) \\ m_{fuel}(t) \end{bmatrix}, \qquad (3)$$

where V_a is the indicated airspeed, V_z is the climb rate, χ_a is the aircraft heading, h is the mean sea level (MSL) altitude, λ is the latitude, τ is the longitude, and m_{fuel} is the fuel mass. Note that in aeronautics, the heading is defined clockwise from the north.

The input vector is defined as

$$\mathbf{u}(t) = \begin{bmatrix} V_a^*(t) \\ V_z^*(t) \\ \chi_a^*(t) \\ V_w(t) \\ \chi_w(t) \end{bmatrix}, \tag{4}$$

where V_a^* is the commanded airspeed, V_z^* is the commanded climb rate, χ_a^* is the commanded aircraft heading, V_w is the wind speed, and χ_w is the wind heading. Here, the commanded inputs are those provided by the pilot (to reach a known flight plan waypoint).

The latitude and longitude dynamics are given by

$$\dot{\lambda} = \frac{V_a \sin \chi_a + W_N}{R_e + h}, \tag{5}$$

$$\dot{\tau} = \frac{V_a \cos \chi_a + W_E}{(R_e + h) \cos \lambda}, \tag{6}$$

where R_e is the MSL radius of the Earth, W_N is the northern component of the wind vector, and W_E is the eastern component of the wind vector:

$$W_N = V_w \cos \chi_w, \tag{7}$$

$$W_E = V_w \sin \chi_w. \tag{8}$$

For the speed and headings, we assume simple dynamics in which the aircraft moves to its commanded values with some inertia:

$$\dot{h} = V_z, \tag{9}$$

$$\dot{V}_z = \frac{1}{J_z}(V_z^* - V_z), \tag{10}$$

$$\dot{V}_a = \frac{1}{J_a}(V_a^* - V_a), \tag{11}$$

$$\dot{\chi}_a = \frac{1}{J_\chi}(\chi_a^* - \chi_a), \tag{12}$$

where the J terms are the inertia parameters, chosen to empirically match available aircraft flight data.

For fuel, we assume that the loss rate is proportional to the airspeed:

$$\dot{m}_{fuel} = -c_{fuel} V_a, \tag{13}$$

where c_{fuel} is the proportionality constant. Fuel loss should be a function of engine output, for which airspeed is the main indicator.

5.1.2. Safety Modeling

In the NAS, there are a number of safety-related conditions and situations to consider, many of which are detailed in (Roychoudhury et al., 2015). In this paper, we consider two types of unsafe states, and define labeling functions for each for a given region of airspace. The first type is *loss of separation*, or *conflict*, for short. While in the airspace, aircraft are required to maintain a certain minimum separation (e.g., defined by a radius of 5 nautical miles and vertical separation of 1000 feet during the en-route portion of a flight), and if two aircraft are closer than these minimum separation both horizontally or vertically, it is considered unsafe. The other type we consider is a *low fuel* situation for an aircraft, which has obvious safety ramifications.

For a region of the airspace with N aircraft, there are then $\binom{N}{2}$ conflict labels. We introduce also a label describing whether any pair of aircraft in the airspace is in conflict. We have N low fuel labels, and introduce also a label describing whether any aircraft in the airspace is in a low fuel state. In addition, we introduce a label describing whether the airspace is unsafe, which is applied if there is a pair of aircraft in conflict or an aircraft in a low fuel state.

5.2. Results

In order to demonstrate the approach, we present results from a simulated scenario consisting of 5 aircraft. The prediction horizon is 10 minutes, and there are 1000 samples generated using Monte Carlo sampling within the prediction algorithm. Initial states and future inputs (assumed to be constant) were drawn from given distributions. Process noise was neglected. All results shown are for a single prediction, from the current time up through 10 minutes into the future.

Air traffic controllers are typically in charge of monitoring a single region of airspace. From this perspective, they are immediately interested in whether any unsafe state is going to be reached at some point in the future. Fig. 1 shows the predicted probability of reaching an unsafe state in the future, for each future time point. At the initial time, there is already a high probability that the NAS is unsafe; it then drops and increases again, reaching 100%. Fig. 2 shows the distribution of the time of the first unsafe state. The distribution is most dense early on, i.e., the system is already unsafe or will be very soon. The predicted probability of reaching an unsafe state at any point in the future time interval is 100%, i.e., all system trajectories will somehow become unsafe.

An operator may then ask, why is the system unsafe? Fig. 3 shows the predicted probability of reaching a conflict state, and Fig. 4 shows the predicted probability of reaching a low fuel state. Fig. 5 shows the distribution of the time of either state. From these results, it is clear that the system is initially unsafe due to a conflict, and then later due to two air-

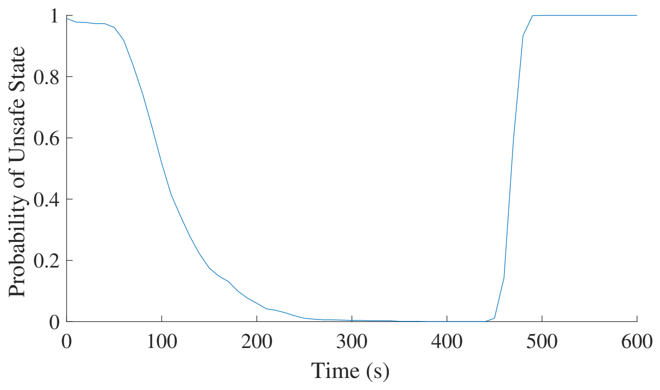

Figure 1. Probability of reaching an unsafe state for all future times.

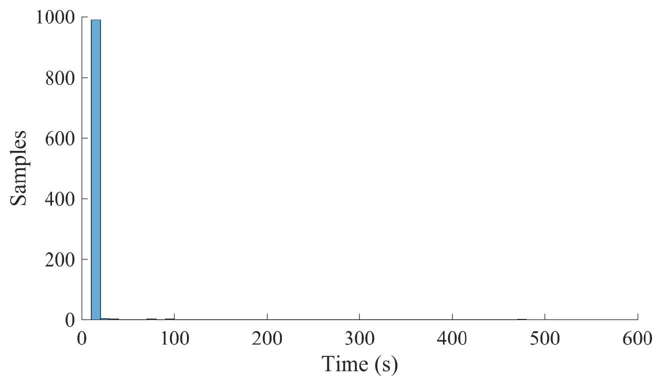

Figure 2. Histogram of time of first unsafe state.

craft entering low fuel states. The latter is the explanation of the 100% probability of reaching an unsafe state within the prediction interval. However, the probability of reaching a conflict state within this interval is also high, at 99.80%.

An operator may then want to determine which aircraft enter a conflict state, and which enter a low fuel state. Fig. 6 shows the predicted probability of reaching a conflict state for each aircraft pair, and Fig. 7 shows the time distribution of first reaching a conflict state. Here, we see that a few pairs of aircraft have a high probability of conflict early on, which then diminishes. A1 and A2 have a 55.30% probability of a conflict within the time interval, A3 and A4 have a 59.20% probability, A1 and A5 a 90.00% probability, and A2 and A5 a 97.80% probability. All other aircraft pairs have a very low probability (less than 2.00%).

Fig. 8 shows the predicted probability of reaching a low fuel state for each aircraft, and Fig. 9 shows the time distribution of first reaching a low fuel state. Here, we see that A1 and A2 will eventually reach a low fuel state; first A2 and then A1.

With this kind of information, it is easy to see how an operator of the NAS can benefit. At least at a high level, operators want to know if something unsafe will happen with a significant probability, within a time horizon in which they have time to act; if so, actions must be taken to prevent the unsafe state from being reached. It is clear also how this kind of information can be useful to an automated decision-maker. These predictions are in the open-loop, i.e., it determines the future safety states assuming no one will intervene when an unsafe state is reached. If unsafe states are reached assuming no invertention, then intervention will be needed to avoid the unsafe states, and that is the problem a decision-making algorithm must solve. A decision-making algorithm can then also use these algorithms to evaluate the quality of different solutions with respect to safety and risk. Each potential decision would be associated with different $p(\mathbf{U}_{k_o,k_h})$, and thus result in different $p(\mathbf{X}_{k_o,k_h})$ with different labels.

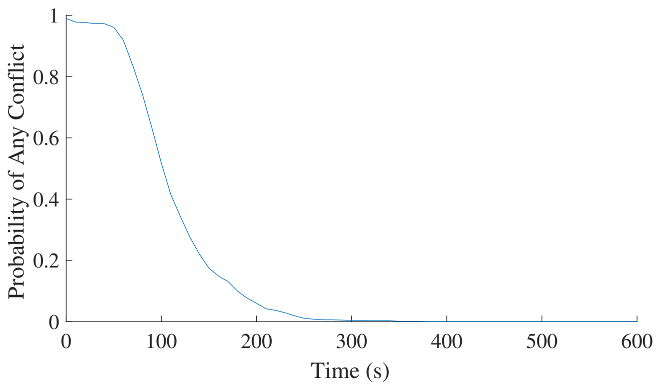

Figure 3. Probability of reaching a conflict state for all future times.

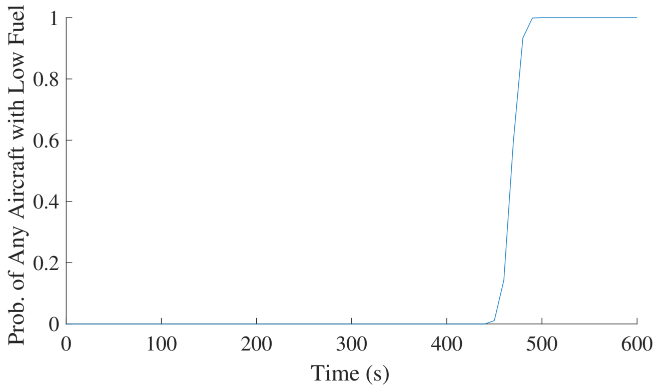

Figure 4. Probability of reaching a low fuel state for all future times.

6. Conclusions

In this paper, we developed a new prognostics framework, extending and generalizing the typical framework for model-based prognostics. Within the new approach, multiple kinds of states can be predicted, the probability of reaching that state computed, and the time to such a state computed. It shows how the main contributions of PHM extend easily to

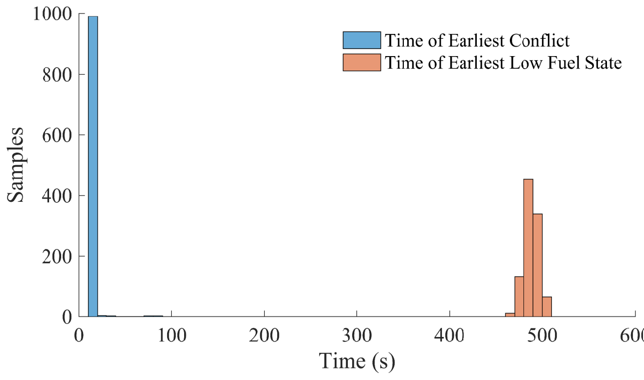

Figure 5. Histogram of time of first conflict and low fuel state.

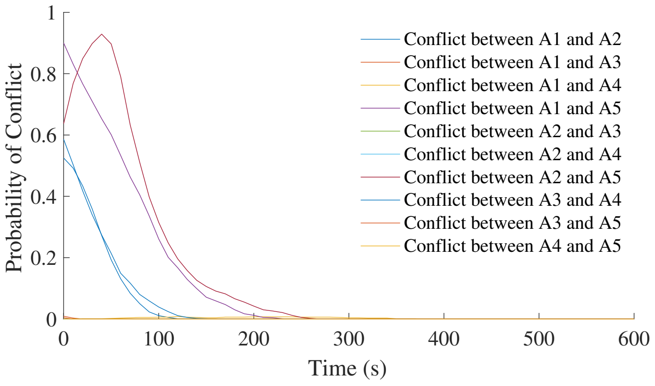

Figure 6. Probability of reaching a conflict state for all future times for each aircraft pair.

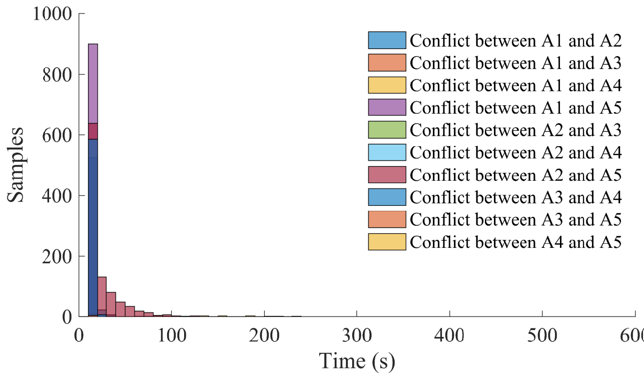

Figure 7. Histogram of time of first conflict state for each aircraft pair.

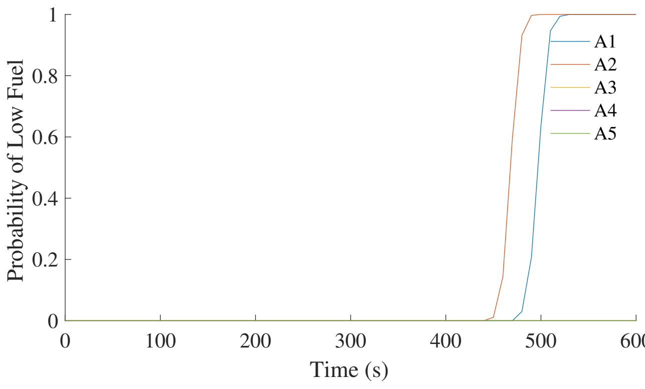

Figure 8. Probability of reaching a low fuel state for all future times for each aircraft.

other domains, and can go beyond simply probability of failure and EOL/RUL prediction. A methodology to distribute the approach was also presented, increasing scalability. The new approach was applied to the prediction of different kinds of unsafe events in the NAS, demonstrating how various kinds of questions about the future states can be addressed within a single framework.

In the future, we will define additional labels for the state space of the NAS and show how these can be predicted within the same framework. For example, predicting a convective weather encounter or a wake vortex encounter, among other things. The investigation of different sampling algorithms is also a topic of future research, as well as determining the extent to which approximate analytical algorithms can be applied. We will also investigate the sensitivity of the various sources of uncertainty to the safety of the NAS.

Additional future work would also involve integrating this prediction framework with a decision-making algorithm. The decision-making algorithm could evaluate, for example, changes to flight plans to meet local flight objectives while maximizing system-level safety. Some initial work in this area has been done in the context of unmanned aerial vehicles (Balaban & Alonso, 2013).

ACKNOWLEDGMENTS

This work was supported by the SMART-NAS project under the Airspace Operations and Safety (AOSP) program within the NASA Aeronautics Research Mission Directorate (ARMD).

REFERENCES

Balaban, E., & Alonso, J. J. (2013). A modeling framework for prognostic decision making and its application to UAV mission planning. In *Annual conference of the prognostics and health management society* (p. 449-460).

Bilmoria, K. D., Banavar, S., Chatterji, G. B., Sheth, K. S., & Grabbe, S. (2000, June). FACET: Future atm concepts evaluation tool. In *3rd USA/EuropeATM R&D Seminar.*

Chatterji, G., Sridhar, B., & Bilimoria, K. (1996, July). En-route flight trajectory prediction for conflict avoidance and traffic management. In *AIAA Guidance, Navigation, and Control and Conference.*

Daigle, M., Bregon, A., & Roychoudhury, I. (2012, Septem-

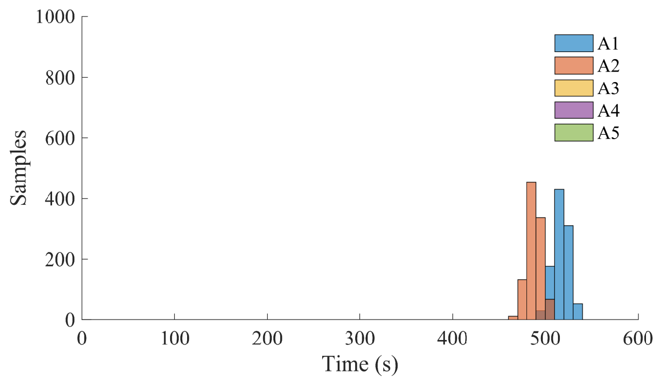

Figure 9. Histogram of time of first low fuel state for each aircraft.

ber). A distributed approach to system-level prognostics. In *Annual conference of the prognostics and health management society 2012* (p. 71-82).

Daigle, M., Bregon, A., & Roychoudhury, I. (2014, June). Distributed prognostics based on structural model decomposition. *IEEE Transactions on Reliability*, *63*(2), 495-510.

Daigle, M., Roychoudhury, I., & Bregon, A. (2015, October). Model-based prognostics of hybrid systems. In *Annual conference of the prognostics and health management society 2015* (p. 57-66).

Daigle, M., Saxena, A., & Goebel, K. (2012). An efficient deterministic approach to model-based prediction uncertainty estimation. In *Annual conference of the prognostics and health management society* (pp. 326–335).

Glynn, P. W., & Iglehart, D. L. (1989). Importance sampling for stochastic simulations. *Management Science*, *35*(11), 1367–1392.

Khorasgani, H., Biswas, G., & Sankararaman, S. (2016, October). Methodologies for system-level remaining useful life prediction. *Reliability Engineering & System Safety*, *154*, 8-18.

Robert, C., & Casella, G. (2004). *Monte carlo statistical methods*. New York: Springer-Verlag.

Roychoudhury, I., Daigle, M., Bregon, A., & Pulido, B. (2013, March). A structural model decomposition framework for systems health management. In *2013 IEEE aerospace conference.*

Roychoudhury, I., Spirkovska, L., Daigle, M., Balaban, E., Sankararaman, S., Kulkarni, C., ... Goebel, K. (2015, November). *Real-time monitoring and prediction of airspace safety* (Tech. Rep. No. NASA/TM-2015-218928). Moffett Field, CA, USA: NASA Ames Research Center.

Roychoudhury, I., Spirkovska, L., Daigle, M., Balaban, E., Sankararaman, S., Kulkarni, C., ... Goebel, K. (2016, January). Predicting real-time safety of the national airspace system. In *AIAA Infotech@Aerospace, AIAA SciTech.*

Sankararaman, S., Daigle, M., & Goebel, K. (2014, June). Uncertainty quantification in remaining useful life prediction using first-order reliability methods. *IEEE Transactions on Reliability*, *63*(2), 603-619.

Sankararaman, S., & Goebel, K. (2013). Why is the remaining useful life prediction uncertain? In *Annual conference of the prognostics and health management society* (p. 337-349).

Tandale, M. D., Wiraatmadja, S., Menon, P. K., & Rios, J. . (2011). High-speed prediction of air traffic for real-time decision support. In *Aiaa guidance navigation and control conference, portland or* (pp. 8–11).

Wells, A. (2001). *Commercial aviation safety*. McGraw Hill Professional.

BIOGRAPHIES

Matthew Daigle received the B.S. degree in Computer Science and Computer and Systems Engineering from Rensselaer Polytechnic Institute, Troy, NY, in 2004, and the M.S. and Ph.D. degrees in Computer Science from Vanderbilt University, Nashville, TN, in 2006 and 2008, respectively. From September 2004 to May 2008, he was a Graduate Research Assistant with the Institute for Software Integrated Systems and Department of Electrical Engineering and Computer Science, Vanderbilt University, Nashville, TN. During the summers of 2006 and 2007, he was an intern with Mission Critical Technologies, Inc., at NASA Ames Research Center. From June 2008 to December 2011, he was an Associate Scientist with the University of California, Santa Cruz, at NASA Ames Research Center. Since January 2012, he has been with NASA Ames Research Center as a Research Computer Scientist. His current research interests include physics-based modeling, model-based diagnosis and prognosis, simulation, and hybrid systems. Dr. Daigle is a member of the Prognostics and Health Management Society and the IEEE.

Shankar Sankararaman received his Bachelors degree in Civil Engineering from the Indian Institute of Technology, Madras in India in 2007 and later, obtained his Ph.D. in Civil Engineering from Vanderbilt University, Nashville, Tennessee, U.S.A. in 2012. His research focuses on the various aspects of uncertainty quantification, integration, and management in different types of aerospace, mechanical, and civil engineering systems. His research interests include probabilistic methods, risk and reliability analysis, Bayesian networks, system health monitoring, diagnosis and prognosis, decision-making under uncertainty, treatment of epistemic uncertainty and multidisciplinary analysis. He is a member of the Non-Deterministic Approaches (NDA) technical committee at the American Institute of Aeronautics, the Probabilistic Methods Technical Committee (PMC) at the American Society of Civil Engineers (ASCE), and the Prognostics and Health Management (PHM) Society. Currently, Shankar is a researcher at NASA Ames Research Center, Moffett Field, CA, where he develops algorithms for system health monitoring, prognostics, decision-making, and uncertainty management.

Indranil Roychoudhury received the B.E. (Hons.) degree in Electrical and Electronics Engineering from Birla Institute of Technology and Science, Pilani, Rajasthan, India in 2004, and the M.S. and Ph.D. degrees in Computer Science from Vanderbilt University, Nashville, Tennessee, USA, in 2006 and 2009, respectively. Since August 2009, he has been with SGT, Inc., at NASA Ames Research Center as a Computer Scientist. His research interests include hybrid systems modeling, model-based diagnostics and prognostics, distributed diagnostics and prognostics, and Bayesian diagnostics of complex physical systems. Dr. Roychoudhury is a Senior Member of the IEEE and a member of the Prognostics and Health Management Society.