Available online at www.sciencedirect.com
ScienceDirect
Control Engineering Practice 15 (2007) 149–162
CONTROL ENGINEERING PRACTICE
www.elsevier.com/locate/conengprac

Hierarchical control of aircraft propulsion systems: Discrete event supervisor approach

Murat Yasar, Asok Ray*

Mechanical and Nuclear Engineering Department, The Pennsylvania State University, 329 Reber Building, University Park, PA 16802, USA

Received 10 December 2004; accepted 25 May 2006
Available online 13 July 2006

Abstract

This paper presents an application of the recently developed theory of language-measure-based discrete event supervisory (DES) control to aircraft propulsion systems. A two-layer hierarchical architecture is proposed to coordinate the operations of a twin-engine propulsion system. The two engines are individually controlled to achieve enhanced performance and reliability, as necessary for fulfilling the mission objectives. Each engine, together with its continuously varying control system, is operated at the lower level under the supervision of a local discrete-event controller for condition monitoring and life extension; the gain-scheduled feedback controller that is used to maintain the specified performance of the turbofan engine is kept unaltered. A global DES controller at the upper level coordinates the local DES controllers for load balancing and health management of the propulsion system.

Keywords: Discrete event supervisory control; Optimal control; Aircraft propulsion systems

1. Introduction

Discrete-event dynamical behavior of physical plants is often modeled as regular languages that can be realized by finite-state automata (Hopcroft, Motwani, & Ullman, 2001). This paper focuses on development of intelligent decision and control algorithms based on the theory of discrete event supervisory (DES) control (Cassandras & Lafortune, 1999; Ramadge & Wonham, 1987) for a twin-engine aircraft propulsion system.

The DES control system is designed to be hierarchically structured in the following sense. The continuously varying control system of each engine interacts with its own local DES controller for health monitoring and intelligent control; and the operational information is abstracted and reported to the propulsion-level DES controller that coordinates the operation of two engines. Furthermore, the propulsion-level supervisory control system allows interactions with exogenous inputs, such as human operators and inputs from other units (e.g., flight control, structural control, energy management, and avionic systems) of the vehicle management system for flexibility of making on-line modifications in the mission objectives. A good feature of the proposed DES control system is that the supervisory control policy can be adaptively updated on-line at both engine and propulsion levels and that the system is tolerant of small anomalies and component faults.

Although the theory of DES control has been developed for almost two decades (Ramadge & Wonham, 1987), only very few applications have been reported in literature. An apparent reason is that, until recently, no quantitative analytical tool was available for design and evaluation of DES controllers. The work reported in this paper makes use of a quantitative measure of regular languages (Ray, 2005; Ray, Phoha, & Phoha, 2005), and is a novel application of hierarchical DES control synthesis for the nonlinear complex dynamical system of twin-engine aircraft propulsion. The real-time implementation of the DES control scheme is challenging because it requires integration of several disciplines (Kumar & Garg, 1995) such as systems theory, computer hardware and software, and domain knowledge of gas turbine engine propulsion.

*Corresponding author. Tel.: +1 814 865 6377; fax: +1 814 863 4848.
E-mail address: axr2@psu.edu (A. Ray).

0967-0661/$ - see front matter © 2006 Elsevier Ltd. All rights reserved.
doi:10.1016/j.conengprac.2006.05.011

The DES control of the propulsion system is validated on a simulation test bed. Thus, feasibility of the DES concept is demonstrated for enhanced operation and control of twin-engine aircraft propulsion in the following areas: (i) real-time decision-making for propulsion control (e.g., load balancing between the engines), (ii) damage reduction (with no significant loss of performance) via life extending control, and (iii) enhanced performance and reliability of the mission.

The terms *controller* and *supervisor* are used interchangeably in this paper. The phrases "Upper level" and "Propulsion level" are synonymous and so are the phrases "Lower level" and "Engine level." The paper is organized in six sections including the present one and an appendix. Section 2 describes the real-time simulation test bed of the twin-engine propulsion system. Section 3 presents the syntheses of the engine-level DES control and propulsion-level DES control systems. Section 4 presents the simulation results and discusses implications of the controller design. Section 5 discusses the theoretical and simulation results of performance evaluation for the propulsion system under DES control. The paper is summarized and concluded in Section 6. The appendix provides supporting information and mathematical background for the control policy synthesis in Sections 3 and 5.

2. Description of the test bed for propulsion system simulation

This section presents implementation of DES control on a real-time simulation test bed of a twin-engine aircraft propulsion system, where the engine model is similar, in complexity and details, to that reported by Diao and Passino (2001) and modular aero propulsion system simulation (MAPSS) model (Parker & Guo, 2002). The objective is to validate the theory of optimal DES control for a real-world nonlinear complex dynamical system such as aircraft propulsion.

A DES controlled propulsion system has been designed and implemented on a simulation test bed that consists of three networked computers using the client/server concept. One of the three computers hosts the propulsion system coordinator for health monitoring of the engines and accordingly making intelligent decisions (e.g., load balancing). The other two computers run separate copies (which may or may not be different depending on the health of the individual engines) of the gas turbine engine simulation model including its continuously varying gain-scheduled feedback control system and a local discrete-event supervisor. The test bed is capable of simulating different dynamics for individual engines due to non-uniform operating conditions. Each of the engine simulators integrates the event-driven discrete dynamics modeled by finite-state automaton as well as time-driven continuous dynamics modeled by ordinary differential equations through continuous-to-discrete and discrete-to-continuous interfaces (Fu, Yasar, & Ray, 2004). Fig. 1 shows the supervisory control architecture of the engine propulsion control system. This software architecture is flexible to adapt deterministic finite state automaton (DFSA) models and controller designs for other complex dynamical systems. Each major function in the simulation program has a modular structure as implemented on the three networked computers of the simulation test bed.

The C++ code of the multi-layer DES control system is superimposed on the existing FORTRAN simulation code of the turbofan engine model and the associated continuously varying engine control system through a C++ program. Specifically, the wrapper program interfaces the major inputs and outputs of the FORTRAN simulation code with the rest of the program in the C++ environment. This approach takes advantage of the available FORTRAN models as individual modules of the integrated C++ program without making any significant changes.

The FORTRAN code of the turbofan engine simulation program, which consists of high-order nonlinear differential and difference equations and supporting algebraic

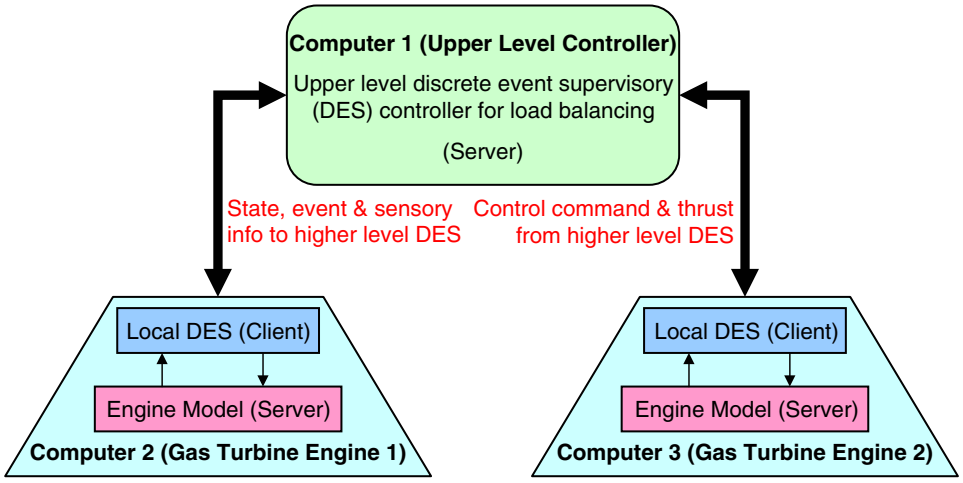

Fig. 1. Supervisory control architecture of the propulsion system.

equations, has been designed for both steady-state and transient operations of a generic jet engine (Diao & Passino, 2001; Parker & Guo, 2002); and different control strategies have been reported in the literature (Diao & Passino, 2001; Fu et al., 2004; Litt, Parker, & Chatterjee, 2003). With the proper inputs such as power lever angle (PLA), and operating condition parameters (e.g., altitude, forward speed, and ambient temperature), the FORTRAN engine model simulates the engine operations from a steady state to transients and to (possibly) other steady states. This simulation code is a stand-alone program with a gain-scheduled feedback controller. The engine simulation model provides various sensor data (e.g., combustion-chamber temperature and high-pressure and low-pressure turbine speeds) together with other critical information (e.g., simulation step size and simulation cycle number), which are collected by the C++ wrapper program and exchanged with the DES controllers through a typical message application protocol interface (API) communication routine. This communication protocol sends and receives message packages through TCP and/or UDP networks. The typical delay in this protocol interface is mainly due to the network communications and is found to be less than 1 ms. Since the engine and flight simulations use integration step sizes in the order of 10–20 ms, the communication delays do not have a major bearing on performance of the proposed control architecture.

Fig. 2 shows the architecture of the engine-level plant and DES controller implementation, which has two replicas, with possible different parameters and initial conditions, in two different computers. Fig. 3 shows the organization of the propulsion-level DES controller together with its own (discrete) event generator, which is implemented on a third computer and makes use of the Message API to communicate with the other two computers.

The DES controller design has two important components that serve as interfaces between the continuously varying control system and the discrete-event supervisory controller—one is event generator and the other is action generator. Event generator receives continuously varying sensor data from the engines. The data along with other information like estimated state and external inputs are used to generate events that, in turn, are inputs to the unsupervised DFSA model of engine operation. The DFSA model is constructed based on the operation scenario; the details are discussed later in Section 3. The DFSA model also serves as a state estimator and provides information on engine states and (both controllable and uncontrollable) events for the discrete-event supervisor to take appropriate actions. Event behavior in the state-based DFSA model is dependent on the state where the event is generated and not on the history or the path of how the state is reached.

The DES controller represents the control policy applied to the DFSA model of engine operation, and it could be a conventional DES controller based on the control specifications (Ramadge & Wonham, 1987) provided by an experienced designer; alternatively, an optimal discrete-event supervisor can be designed by the quantitative method (Fu et al., 2004; Ray, Fu, & Lagoa, 2004). In both cases, the DES controller takes the estimated states as inputs and generates control commands (of controllable event disabling or enabling) as outputs, which are transmitted through a Message API communication routine to the action generator. The primary task of the action generator is to convert the control commands from the supervisor into necessary input functions for the continuously varying plant.

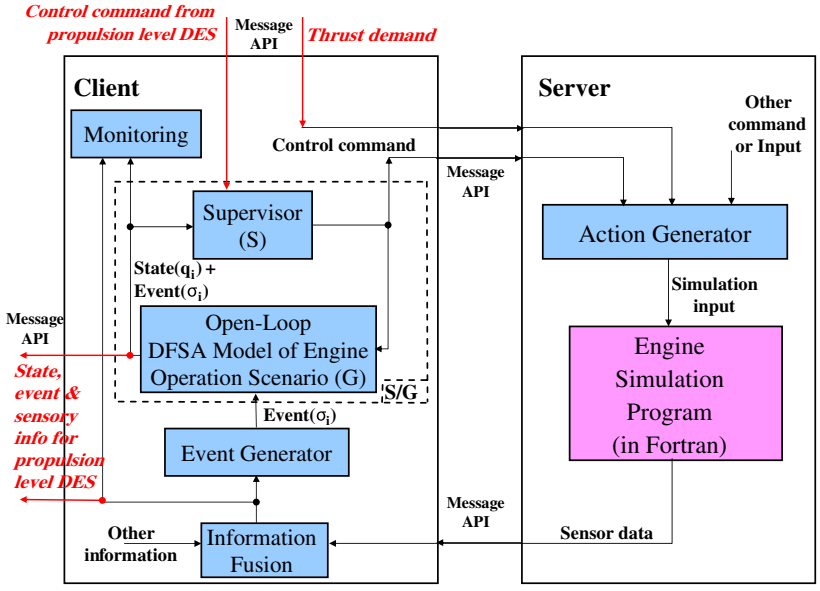

Fig. 2. Engine level plant/DES controller implementation.

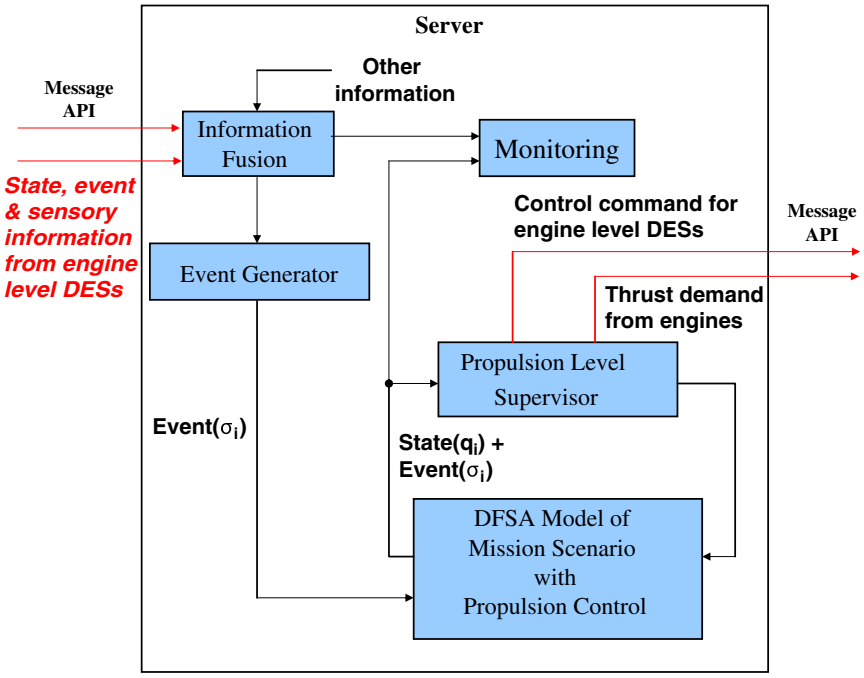

Fig. 3. Propulsion level DES controller implementation.

3. Synthesis of DES controllers

One of the major tasks of the supervisory decision making is fusion of the (possibly) redundant, conflicting, and incomplete information to make timely decisions. Such information can be derived from different types of sensor data (Volponi, Brotherton, Luppold, & Simon, 2003) as well as operational history and the knowledge-base generated from pilot's personal experience. Computer-based advanced analytical techniques are necessary for fusion of the time series data available from multiple sensors and other relevant non-sensor-based information to make specific inferences that could not be achieved through the sole usage of the available sensory information. Improved performance may not result simply from an increased volume of sensor data and engine information unless the ensemble of information is systematically processed in the context of the engine operational conditions and mission objectives (Tolani, Yasar, Ray, & Yang, 2006). In order to achieve the desired performance of a DES controller, it is essential to have an effective event generation algorithm to ensure fusion of the heterogeneous information for: (i) enhanced resolution and reduced ambiguity in decision and control; and (ii) advantageous trade-offs between probability of false alarms and missed detection (Basseville & Nikiforov, 1993).

The unsupervised dynamics of engine operations are modeled as a DFSA, based on postulated scenarios. (Note that the model may change for different mission objectives.) In the present context, the DFSA model assumes that a twin-engine aircraft is carrying out a routine surveillance mission. The mission abortion is allowed at certain states according to the operation scenario. Each

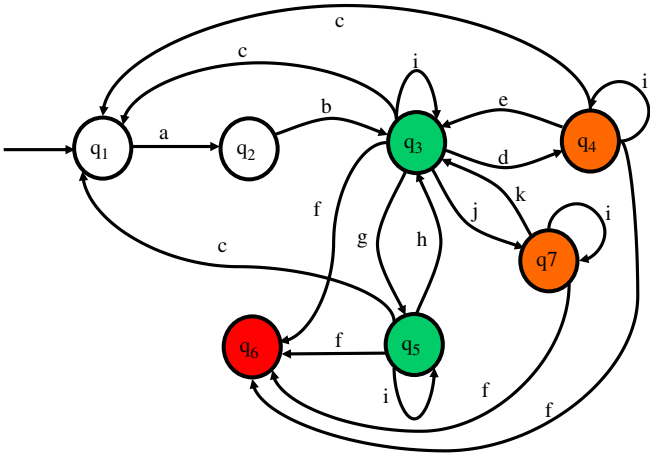

Fig. 4. Unsupervised plant DFSA model at the engine level.

engine of the aircraft is equipped with a continuously varying controller which is supervised by a local engine-level DES controller. The primary objective of the local DES controller is to strike the right balance between the conflicting demands of higher performance from upper level supervisor and limiting structural damage to the engine components. The global propulsion-level DES controller redistributes the load depending on the health of the engine and thrust demand placed by the pilot.

3.1. Engine-level DES control

Fig. 4 presents the DFSA model of the unsupervised engine operations for DES control. Table 1 lists the events, where "C" denotes controllable events and "UC" denotes uncontrollable events; and Table 2 lists the plant states.

Table 1
Event list for the unsupervised engine model

Event	Description	Status
a	Start	C
b	Warm up complete	UC
c	Shut the engine	UC
d	Detection of oscillations	UC
e	Nozzle area reduction	C
f	Engine fails	UC
g	Reduce performance/reduce damage	C
h	Increase performance/increase damage	C
i	Remain in the state	C
j	Reduce performance	C
k	Increase performance	C

Table 2
State list for the unsupervised engine model

State	Description	Status
$\mathbf{q}_1$	Engine start	
$\mathbf{q}_2$	Engine warm up	
$\mathbf{q}_3$	High performance/High damage rate	Marked (good)
$\mathbf{q}_4$	Oscillations	Marked (bad)
$\mathbf{q}_5$	Low performance/Low damage rate	Marked (good)
$\mathbf{q}_6$	Engine inoperable	Marked (bad)
$\mathbf{q}_7$	Low performance/High damage rate	Marked (bad)

The events and states for the DFSA models at the engine level are denoted by lower case letters (e.g., **a** is the start event and $\mathbf{q}_1$ is the engine start state). The engine can operate in two regimes, one is high-performance regime (state $\mathbf{q}_3$), where the damage rate is also high. The other is low-performance regime (state $\mathbf{q}_5$), where the damage rate is low. In the high-performance regime, the engine has a tendency to go to state $\mathbf{q}_4$, where engine variables like combustor temperature have been observed to have oscillatory behavior. Temperature oscillations could be extremely harmful for engine health (DeLaat & Chang, 2003) and must be avoided at all costs. Engine-level controller chooses the regime of operation (state $\mathbf{q}_3$ or $\mathbf{q}_5$) depending on two factors: thrust requirement at the propulsion level and health of the engine, as explained below.

Health of the engine is determined from the damage accumulation that is a function of high-pressure turbine gas inlet temperature and shaft speed; and in addition, damage spikes (i.e., sudden jumps) at random time intervals are introduced to simulate the damage in a real-world engine. The DFSA model of the supervised engine operations is shown in Fig. 5, where the dashed lines indicate those controllable events that are disabled by the optimal control algorithm, described in the Appendix and citations therein. The state $\mathbf{q}_7$ in Fig. 5 becomes unreachable following the disabling action of the supervisor. Therefore, all transitions, originating from the state $\mathbf{q}_7$, are also shown with dashed lines as well as the state itself.

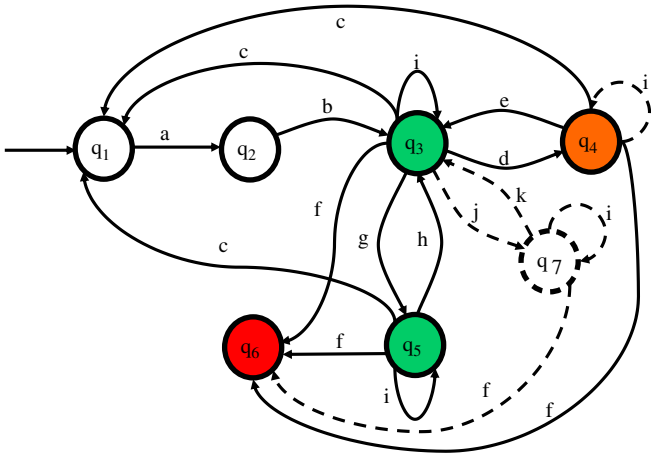

Fig. 5. Supervised plant DFSA model at the engine level.

Table 3
Event list for the propulsion level DFSA model

Event	Description	Status
A	Start engines	C
B	Warm up complete	UC
C	One engine deteriorates	UC
D	Redistribute the load (one engine is unhealthy)	C
E	Both engines deteriorate	UC
F	One good engine fails	UC
G	Both engines fail	UC
H	Increase performance	C
I	Reduce performance	C
J	Request to abort mission	C
K	Request accepted	UC
L	Request rejected (both engines are running)	UC
M	Mission accomplished	UC
N	Turn off engines	UC
O	Redistribute load (both engines are unhealthy)	C
P	Redistribute load (one engine has failed)	C
Q	Request rejected (one engine has failed)	UC
R	One bad engine fails	UC

3.2. Propulsion-level DES control

One of the main tasks of the propulsion-level DES controller is to redistribute the load between two engines depending on the current health condition of each engine and the thrust demand. A DES controller is expected to ensure that this requirement is satisfied, regardless of being optimal or not. The propulsion-level DES controller acts in an advisory role for the mission-related decisions to enhance the mission performance. However, the ultimate decision for mission-related operations is left for the pilot.

Since the propulsion-level supervisor is designed to execute the key decisions at the engine level, the model for operating regimes of the propulsion system include the Cartesian product of the state sets of two (locally supervised) engine models. However, model order reduction via aggregation and deletion of unrealizable states is

Table 4
State list for the propulsion level DFSA model

State	Description	Status
$\mathbf{Q}_1$	Aircraft on ground	
$\mathbf{Q}_2$	Engines warming up	
$\mathbf{Q}_3$	Both engines in high performance operation	
$\mathbf{Q}_4$	One engine in high one engine in low performance	
$\mathbf{Q}_5$	Both engines in low performance operation	
$\mathbf{Q}_6$	One engine stopped one engine in high performance	
$\mathbf{Q}_7$	One engine stopped one engine in low performance	
$\mathbf{Q}_8$	Both engines failed	Marked (bad)
$\mathbf{Q}_9$	Decision for abort mission	Marked (bad)
$\mathbf{Q}_{10}$	Mission successful	Marked (good)
$\mathbf{Q}_{11}$	High damage detected for one engine	
$\mathbf{Q}_{12}$	High damage detected for both engines	

needed because the above Cartesian product may result in a very large number of states. The events and states of the unsupervised DFSA model at the propulsion level are listed in Tables 3 and 4, respectively. Events and states for the DFSA model at the propulsion level are denoted by upper case letters (e.g., **A** is the start event and $\mathbf{Q}_1$ is the aircraft on the ground state). Multiple occurrences of an event have been indicated as single events as seen in three instances (i.e., events **D**, **O** and **P**) in Table 3. The event "request to abort mission" is controllable while the events such as "request accepted", "abort the mission" and "shut down the engine" are uncontrollable events from the supervisor's perspectives.

4. Simulation experiments: results and discussion

Experiments were conducted on the simulation test bed, described in Section 2, to validate the DES control concept. Upon successful implementation of the software modules on the client and server computers, several simulation experiments were performed. The first set of experiments was performed at the engine level to demonstrate the interactions between the DES controller and continuous-time dynamics of the engine. The design specifications of the engine-level supervisor include reduction of the engine component damage and consequently engine life enhancement. Then, the effects of the propulsion-level DES controller that is built upon the engine-level DES controllers are investigated.

Fig. 6 exhibits the predetermined input profile that excites both unsupervised and optimally supervised engine models. Time responses of several outputs (combustor outlet temperature, high-pressure turbine speed, net thrust of the engine, and fuel flow into the main burner) over a period of 12 min were observed. The four plates in each of Figs. 7 and 8 exhibit the response profiles of the above set of plant variables for unsupervised and supervised engine

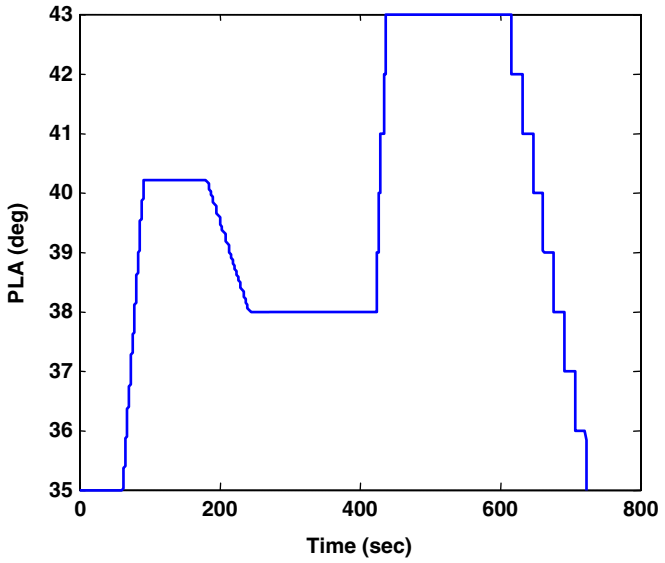

Fig. 6. Power lever angle input.

models, respectively. A comparison of plots in Figs. 7 and 8 indicates that the optimal DES control at the engine level eliminates the high frequency oscillations that are present in the unsupervised plant responses in Fig. 7. The supervisor takes actions immediately upon detection of oscillations by an FFT algorithm. Without making any alterations in the gain-scheduled controller of the engine, the supervisory actions are implemented as a piecewise constant term in the reference input to the (continuous gain-scheduled) controller. In the continuous control system, there are seven summation points for the feedback loop, each providing a reference signal for a specific actuator. It is found that the booster vane angle and nozzle area manipulations have the most significant effects on the response of engine variables (Tolani et al., 2006). In the work reported in this paper, nozzle area is decreased by 20% of the nominal value. Due to supervisor's actions, the potentially sustained oscillations are quenched in less than 30 s after oscillations are observed, and the engine operation is brought to steady state. Thus, sustained oscillations are practically non-existent in the supervised plant responses in Fig. 8. Since high-frequency oscillations of temperature and pressure are the primary sources of fatigue crack damage in the turbine blades, disks, and stationary vanes, the supervisory control becomes very effective for mitigation of structural damage in the engine components. In contrast, in the unsupervised case, the engine health would be adversely affected if the propulsion system is operated in this way to achieve the mission objectives.

The propulsion-level DES controller has three main tasks. The first task is the intelligent decision making and control of the twin-engine aircraft propulsion systems for mission execution; the second task is to improve the overall mission and operational behavior so that engine health can

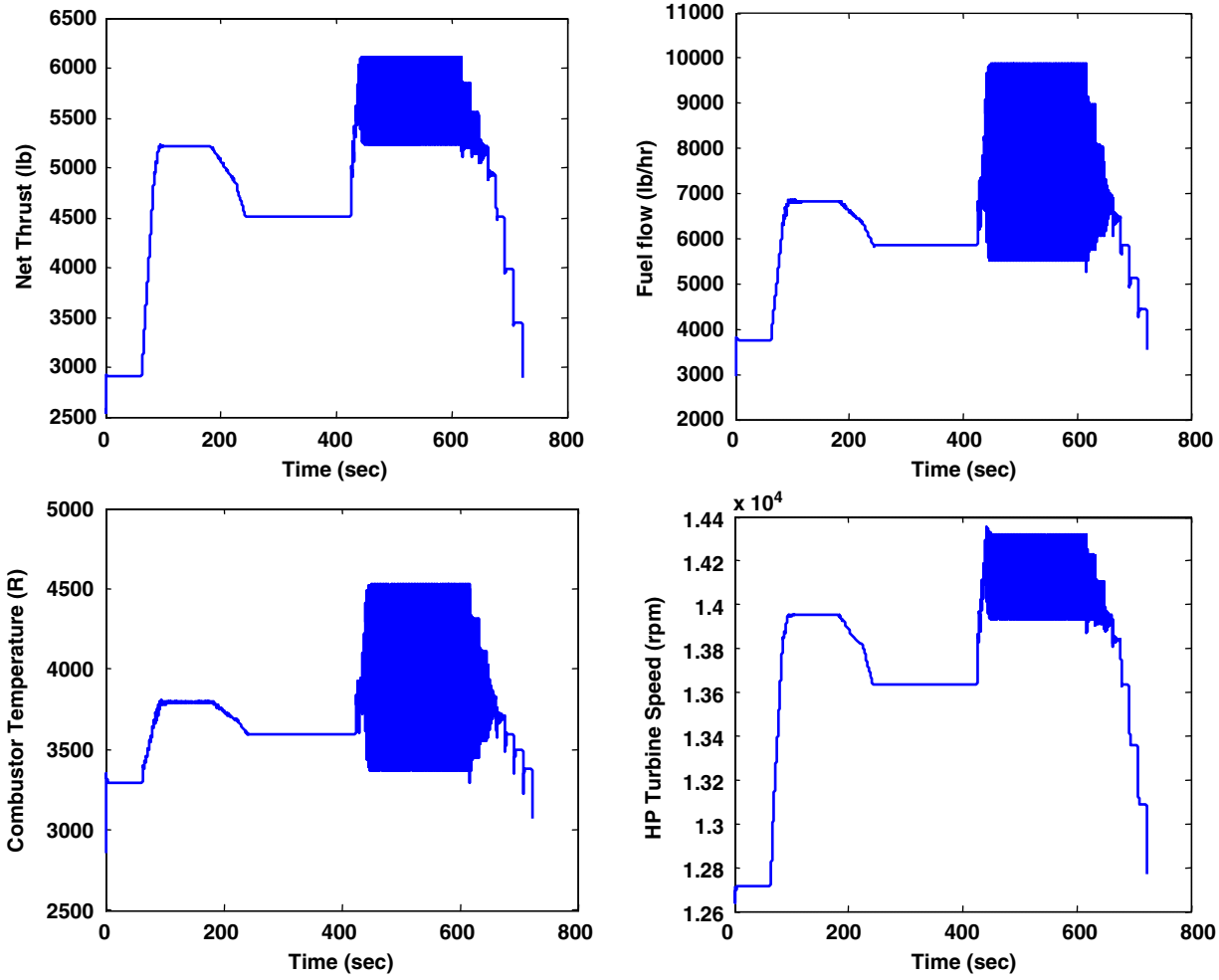

Fig. 7. Simulation output for the unsupervised case.

be enhanced via damage reduction; and the third task is load balancing between two engines so that the propulsion system produces the thrust demanded by the flight control system while attempting to enhance engine life. The issue of load balancing becomes even more important when the health conditions of two engines are significantly different (one can be in "bad condition" and the other in "good condition"). If the situation comes to this point, then the aim of the DES controller is judicious redistribution of the load between two engines such that the "bad engine" carries lower load than the "good engine", subject to the condition that the total thrust output of the engines satisfies the mission and safety requirements. To satisfy these design requirements, the propulsion-level DES controller (Kumar & Garg, 1995; Ramadge & Wonham, 1987), which may or may not be optimal, is designed and implemented over the unsupervised plant model.

Figs. 9 and 10 show the simulated outputs of two engines, where both engines are in "good" health at the start of simulation. The damage increment is a dimensionless quantity denoting the damage accumulation as percentage of the nominal value at each simulation cycle and is attributed to the turbine blades. As the mission progresses, due to an injected fault in Engine 2, the engine deteriorates and starts to run at "bad" health condition after 330 s. After 630 s, another fault is injected in Engine 1 and this engine also deteriorates. Thus, the load distribution of the engines varies in three regions (see Figs. 9 and 10). In the beginning, the load is equally distributed in Region 1. Then the "good engine" (Engine 1) takes the responsibility of producing higher thrust as seen in Region 2 in the plates of Figs. 9 and 10. Later on, both engines are again loaded equally as seen in Region 3 when it is not advisable to impose uneven thrust requirements. The first damage event affects the performance of Engine 2 after approximately 50 s in order to realize the thrust demand of the pilot. At this point, the full thrust demand cannot be satisfied if the load in Engine 2 is reduced. While the thrust produced in Region 2 by each engine is not the same, the most critical impact of the uneven load distribution is formation of excess moment along the yaw axis of the aircraft. However, it might be possible to counteract this situation by adjusting the control surfaces of aircraft (Yasar, Horn, & Ray, 2006).

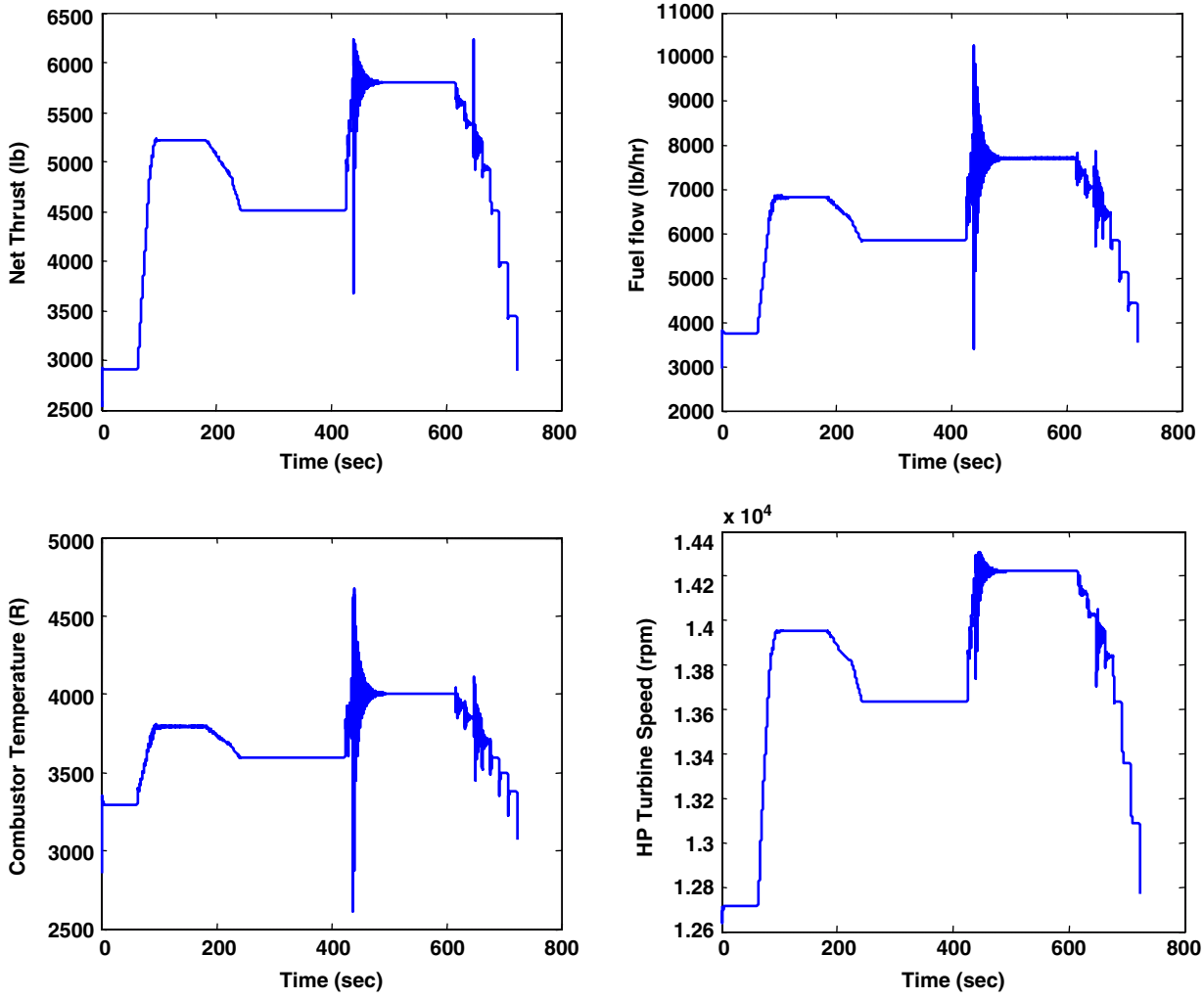

Fig. 8. Simulation output for the optimally supervised case.

5. Evaluations of DES controllers

The language measure, described in Ray (2005) and Ray et al. (2005), has been used to quantitatively evaluate the impact of the propulsion-level DES controller on the overall mission behavior. Given the state weights and the state transition probabilities, the language measure provides a quantitative performance measure of the controllers. Supporting information and pertinent mathematical background are given in the Appendix.

5.1. Identification of state transition probabilities

Quantitative analysis of DES controllers and synthesis of an optimal DES controller require identification of the state transition cost matrix. Similar to continuously varying dynamical systems (CVDS), one must use the techniques of system identification (Ljung, 1999) to evaluate the parameters of the unsupervised DFSA plant model, i.e., the elements $\tilde{\pi}_{ij}^0$ of the event cost matrix $\tilde{\Pi}^0$ (see Definition 2 in the Appendix). As the number of experiments increases, the identified event costs tend to converge within an appropriately specified tolerance. For stationary operation of the engine, since conditional probabilities of the events are assumed to be time-invariant, the identified event costs and their uncertainty bounds can be determined. The probabilities of the events such as deterioration and failure of an engine are triggered by the event generation algorithm, based on the sensory information. Nevertheless, it is not possible to simulate some events, such as acceptance or rejection of the request by the pilot, without a random event generator. The randomization used in triggering this type of events has certainly an effect on the identified event costs, but not directly, since sensor-based events are not induced by this randomization (Ray et al., 2005). As a typical case, Fig. 11 presents identification of event probabilities at state $\mathbf{Q}_5$, where both engines are in low-performance operation.

Fifty simulated missions were performed for both unsupervised plant G and supervised plant S/G to construct the respective state transition probability matrices, Π^0 and Π^S. The different visited states and the triggered events were monitored and plotted to obtain the particular $\tilde{\Pi}$-matrices. After identifying the respective

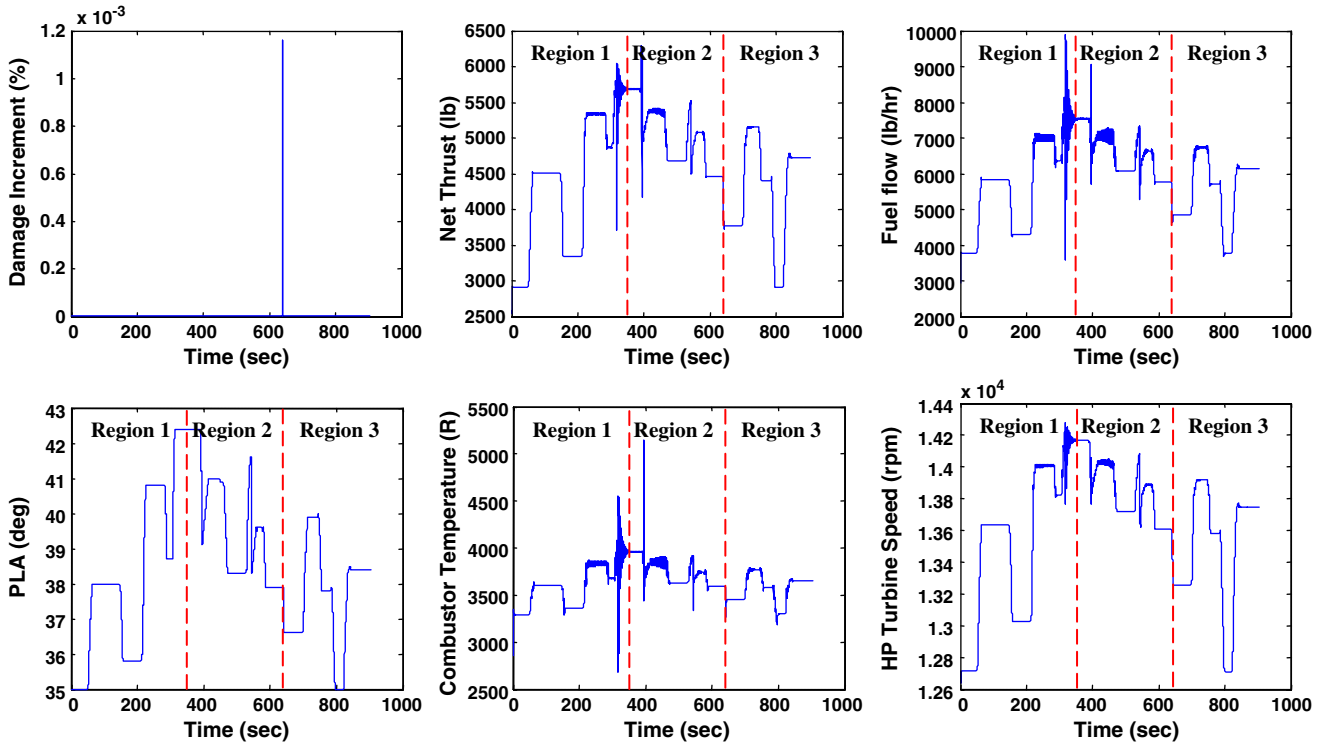

Fig. 9. Effect of conventional propulsion level DES controller on engine 1.

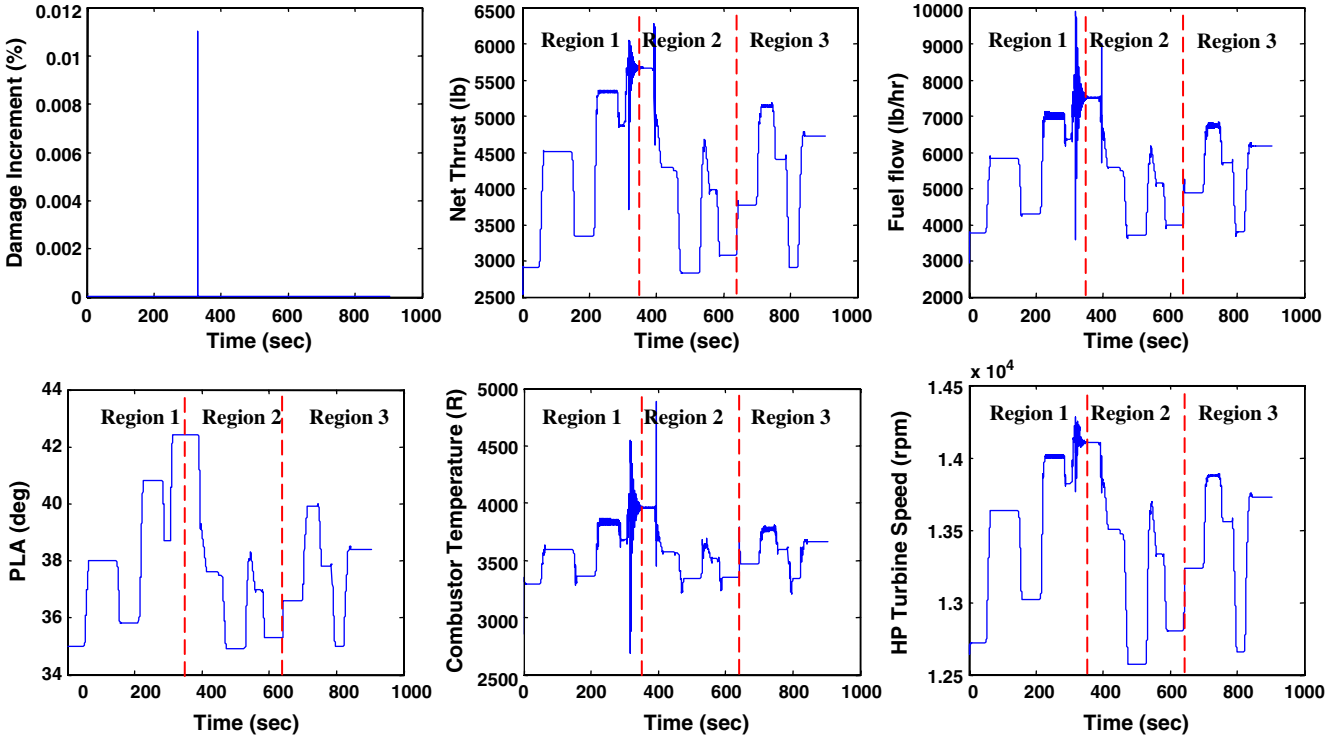

Fig. 10. Effect of conventional propulsion level DES controller on engine 2.

event probability matrices, $\tilde{\Pi}^0$ and $\tilde{\Pi}^S$, of G and S/G, the state transition probability matrices Π^0 and Π^S are computed using Definition 3 in the Appendix. Table 5 lists the Π^0-matrix of the unsupervised DFSA model at the propulsion level.

5.2. Selection of characteristic values for evaluation of DES controllers

Given the state characteristic values and the state transition probabilities, the language measure serves

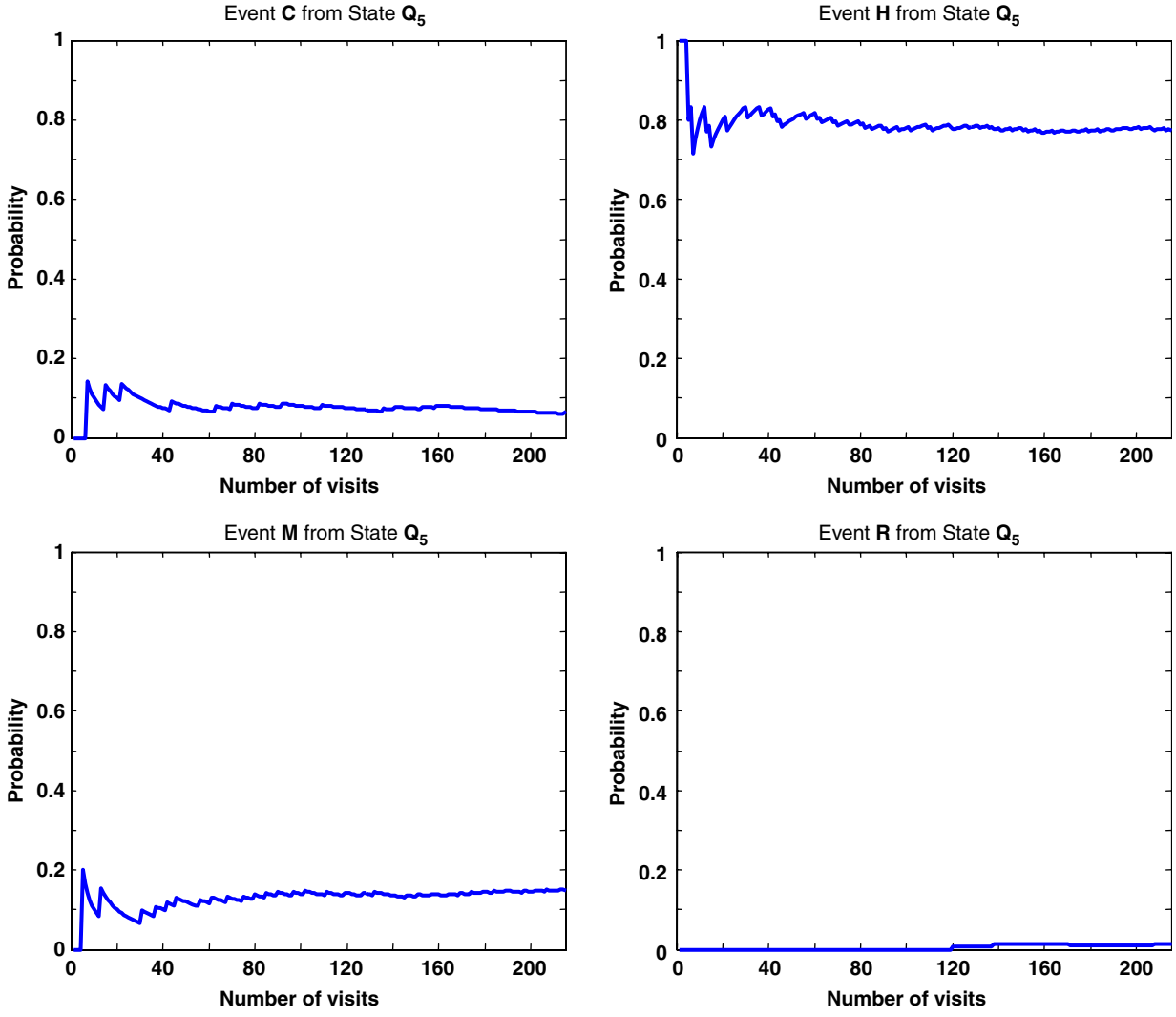

Fig. 11. Convergence of event cost identification.

Table 5
Π matrix of the propulsion level DFSA model

State	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8	Q_9	Q_{10}	Q_{11}	Q_{12}
Q_1	0	1	0	0	0	0	0	0	0	0	0	0
Q_2	0	0	1	0	0	0	0	0	0	0	0	0
Q_3	0	0	0	0	0.899	0.037	0	0	0	0	0.051	0.014
Q_4	0	0	0.500	0	0.500	0	0	0	0	0	0	0
Q_5	0	0	0.772	0	0	0	0.014	0	0	0.149	0.065	0
Q_6	0	0	0	0	0	0	0.250	0.125	0.625	0	0	0
Q_7	0	0	0	0	0	0	0	0	1	0	0	0
Q_8	0	0	0	0	0	0	0	0	0	0	0	0
Q_9	0.415	0	0	0	0	0	0	0	0	0	0.512	0.073
Q_{10}	1	0	0	0	0	0	0	0	0	0	0	0
Q_{11}	0	0	0	0.044	0.370	0	0	0	0.587	0	0	0
Q_{12}	0	0	0	0	0.333	0	0	0	0.667	0	0	0

as a theoretical performance measure for quantitative evaluation of DES controllers. The characteristic values are assigned based on the designer's perception for the importance of terminating on specific marked states. For the propulsion-level DES controller, the weights of the states are selected according to each state's importance (contribution) to the mission management as

$$\bar{\chi} = [0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ -1.00\ \ -0.188\ \ 0.300\ \ 0\ \ 0].$$

The non-marked states have no direct bearing on the mission performance and hence each state (i.e., states $\mathbf{Q}_1$ to $\mathbf{Q}_7$, $\mathbf{Q}_{11}$ and $\mathbf{Q}_{12}$) has been assigned zero weight. The marked states, $\mathbf{Q}_8$, $\mathbf{Q}_9$ and $\mathbf{Q}_{10}$, which do have direct bearing on the mission performance, are assigned non-zero weights as follows:

$\mathbf{Q}_8$: *Both engines failed*, State $\mathbf{Q}_9$: *Mission abort*, State $\mathbf{Q}_{10}$: *Mission successful*. They have relative weights of -1, -0.188 and 0.3, respectively. The bad marked state, $\mathbf{Q}_8$, is assigned the characteristic value of -1.0 because the aircraft will most likely be destroyed (because of both engines being non-functional) if the DFSA terminates on this state. On the other hand, the good marked state, $\mathbf{Q}_{10}$: *Mission successful*, is assigned the characteristic value of $+0.3$ based on its relative importance to the loss of the aircraft. Therefore, by choosing the characteristic values in this way, loosing one aircraft is made equivalent to approximately three successful missions. The other bad marked state, $\mathbf{Q}_9$: *Mission abort*, has also a negative characteristic value which signifies the importance of this state relative to a successful mission and a possible loss of the aircraft. The selection of characteristic value for mission abort is to quantitatively match the theoretical (i.e., language measure-based) performance of the unsupervised plant with its experiment-based performance (which will be introduced in the next section). The corresponding performance measures for the supervised cases must also match with this selection provided that the plant is modeled appropriately and the language measure parameters, Π-matrix and $\bar{\chi}$-vector, are correctly assigned.

Using the relation $\bar{\mu} = [I - \Pi]^{-1}\bar{\chi}$ derived in Ray (2005), the language measures (i.e., the theoretical performance of the propulsion system) are calculated for unsupervised plant model G and DES controlled plant S/G as $\mu_{unsupervised} = 0.1260$ and $\mu_{supervised} = 0.2055$, respectively. It is seen that the DES controller used in the propulsion level has a positive effect on the mission behavior of the propulsion system.

5.3. Optimal DES controller synthesis and evaluation

After identifying the event cost matrix $\tilde{\Pi}^0$ of the unsupervised plant G, the state transition cost matrix Π^0 is constructed (using Definition 3 in the Appendix). State transition cost matrix is the only unknown input for the optimal control algorithm to design the optimal DES controller, since the other necessary parameters, such as the characteristic vector $\bar{\chi}$, are selected by the designer based on the design requirements. Therefore, given the state transition cost matrix Π^0 of the unsupervised plant and the state characteristic vector $\bar{\chi}$, the optimal DES controller can be synthesized as described in the Appendix.

Table 6 lists the iterations of optimal control synthesis for the propulsion-level supervisory control where the first column belongs to the unsupervised plant G. The performance measure of the unsupervised plant is negative at the states $\mathbf{Q}_6$, $\mathbf{Q}_7$, $\mathbf{Q}_8$, $\mathbf{Q}_9$, $\mathbf{Q}_{11}$, and $\mathbf{Q}_{12}$ as indicated by the bold scripts in Table 6. All controllable events leading to these states are disabled and the resulting performance measure at Iteration 1 shows sign change at states $\mathbf{Q}_7$, $\mathbf{Q}_{11}$, and $\mathbf{Q}_{12}$ as indicated by italics in Table 6. All controllable events leading to these states are now re-enabled for further increase in performance at Iteration 2. However, there is no sign change in the performance vector between Iteration 1 and Iteration 2, which immediately shows that the algorithm converged to the optimal solution after this iteration. The synthesis is complete in Iteration 2 (i.e., there is no need to go for the Iteration 3) because there is no sign change; moreover, the performance vector at Iteration 2 shows also no improvement after the previous iteration.

Table 6
Iterations for optimal DES controller synthesis

State	Unsupervised plant	Iteration 1	Iteration 2
$\mathbf{Q}_1$	0.1260	0.2452	0.2452
$\mathbf{Q}_2$	0.1273	0.2477	0.2477
$\mathbf{Q}_3$	0.1286	0.2502	0.2502
$\mathbf{Q}_4$	0.1415	0.2617	0.2617
$\mathbf{Q}_5$	0.1572	0.2785	0.2785
$\mathbf{Q}_6$	**−0.2555**	−0.1238	−0.1238
$\mathbf{Q}_7$	**−0.1510**	*0.0000*	0.0000
$\mathbf{Q}_8$	**−1.0000**	−1.0000	−1.0000
$\mathbf{Q}_9$	**−0.1525**	−0.0353	−0.0353
$\mathbf{Q}_{10}$	0.4248	0.5428	0.5428
$\mathbf{Q}_{11}$	**−0.0250**	*0.1132*	0.1132
$\mathbf{Q}_{12}$	**−0.0488**	*0.0919*	0.0919

Table 7
Language measure and performance for propulsion level supervisors

	Unsupervised	Supervised	Optimal
Language measure ($\boldsymbol{\mu}$)	0.1260	0.2055	0.2452
Performance Index ($\boldsymbol{\nu}$)	0.1276	0.1959	0.2480

The performance of the optimal controller was compared with that of unsupervised plant G and the supervisor S that was designed using the conventional procedure (Cassandras & Lafortune, 1999; Ramadge & Wonham, 1987). Theoretical performance of the supervisors can be associated with the language measure of each supervisor, as described in the Appendix. The language measures of the unsupervised plant and conventional and optimally supervised plants at the propulsion level are listed in Table 7 that also shows a close agreement between the analytically generated language measures and experimentally determined values.

Results of simulation experiments have been used to validate the DES controller performance based on the language-theoretic analysis. The experimental performance index is determined as a function of the relative weights of the visited states. The mission outcomes of the

unsupervised and supervised propulsion systems were recorded during each simulated mission. The numbers of missions ending at both good and bad marked states were multiplied by the respective relative weights of the states. That is, the experimental performance measure is calculated as

$$v = \frac{\sum_{i=1}^{N} n_i \cdot \chi_i}{N},$$

where n_i is the number of missions ending at the ith state, and N is the total number of experiments; and χ_i is the characteristic weight of the ith state. The performance of the unsupervised plant and the other two supervisors, namely, conventionally designed supervisor (Ramadge & Wonham, 1987) and language-based optimal supervisor (Ray et al., 2004), are compared based on the observations of mission execution on the simulation test bed. The experimental evaluations of the performance for different supervisors are presented in Table 7. Both theoretical and experimental (simulation) evaluations of DES controllers provide better mission management under optimal supervision. It is seen that the theoretical performance of the supervisors is in quantitative agreement with the experimental results, presented in Table 7. Optimal DES controller, synthesized using the algorithm described in Appendix and in Ray et al. (2004), has the highest theoretical performance (i.e., highest language measure) among all controllers. The results of the simulation experiments concur with the theoretical measure of the controllers in the sense that optimal supervisor yields the best mission performance.

Remarks on the optimal supervisor design: At the first step of the iterations, the performance measure of the unsupervised plant is negative at states $\mathbf{Q}_6$, $\mathbf{Q}_7$, $\mathbf{Q}_8$, $\mathbf{Q}_9$, $\mathbf{Q}_{11}$, and $\mathbf{Q}_{12}$. Therefore, controllable transitions to these states should be disabled. It is observed that one engine is lost at the states $\mathbf{Q}_6$ and $\mathbf{Q}_7$, implying that the events leading to these states would cause the "one engine failed" condition which is uncontrollable and hence cannot be disabled. Similarly, the only transition to the state $\mathbf{Q}_8$ causes the "both engines failed" condition, which is also uncontrollable; hence, this transition cannot be disabled either. The transitions leading to states $\mathbf{Q}_{11}$ and $\mathbf{Q}_{12}$ are the "deterioration of engines" condition, which is directly related to the damage information received from the engine-level DES control system. Based on this information, the propulsion-level supervisor makes a decision on the health condition of the individual engines. Evidently, this kind of sensory events are uncontrollable, so is the engine deterioration event. The remaining state that should be investigated is the state $\mathbf{Q}_9$, *Mission abort*. The optimal control algorithm disables all mission abort requests, which significantly increases the performance of the mission behavior with increased risk of losing the aircraft. The simulation experiments show that the mission performance at the propulsion level increases under the optimal DES controller, albeit with an increased probability of aircraft loss. Table 7 shows a close agreement between the analytically generated language measures and experimentally determined performance data. It should be noted that, the optimal control policy is likely to change if the elements (i.e., individual state weights) of the $\bar{\boldsymbol{\chi}}$-vector are altered.

6. Summary and conclusions

This paper presents a quantitative approach to analysis and synthesis of hierarchical DES control laws for aircraft propulsion systems. The objectives are:

- Intelligent decision and control of distributed propulsion management systems, where each of the engines has its own local DES control.
- Structural damage reduction and life extension of aircraft engines without any significant loss of the system performance.
- Decision making and mission planning modifications through a high-level DES coordinator.
- Incorporation of optimal control laws for enhanced mission management.
- Extension of this work to other complex dynamic systems such as rotorcrafts and power plants simulation test beds.

A decision and control architecture has been proposed to coordinate the operations of a twin-engine propulsion system. The DES control law has been validated for a twin-engine aircraft propulsion system on a networked simulation test bed. The plant dynamics in the simulation test bed is built upon the model of a generic turbofan gas turbine engine. The software architecture of the simulation test bed is flexible for adaptation to arbitrary DFSA models and a variety of DES control laws, including those that are quantitatively analyzed using a language measure.

Acknowledgments

The authors would like to acknowledge Dr. Devendra Tolani for his support during the work reported in this paper. This work has been supported in part by the US Army Research Laboratory and the US Army Research Office under Grant No. DAAD19-01-1-0646 and by NASA Glenn Research Center under Grant No. NNC04GA49G.

Appendix Language measure and discrete event optimal control

This appendix reviews the previous work on language measure (Ray, 2005; Ray et al., 2005) and optimal control policy (Ray et al., 2004) that is based on this measure with no event disabling penalty. The background information necessary to develop a performance index for the optimal DES control law is introduced. Performance of DES controllers is associated with the language measure through out this paper.

Let the dynamical behavior of a physical plant be modeled as a deterministic finite state automaton (DFSA) $G_i \equiv (Q, \Sigma, \delta, q_i, Q_m)$, where Q is the finite set of states q_j with $|Q| = n$ and $q_i \in Q$ is the initial state; Σ is the (finite) alphabet of events with $|\Sigma| = m$; the function $\delta : Q \times \Sigma \to Q$ represents state transitions and $Q_m \subseteq Q$ is the set of marked states which have some importance (positive or negative) for the DFSA model.

Definition 1. The characteristic state weight function that assigns a signed real weight to states is defined as: $\chi : Q \to [-1,\ 1]$ such that

$$\chi_j \equiv \chi(q_j) \in \begin{cases} [-1,\ 0) & \text{if } q_j \in Q_m^-, \\ \{0\} & \text{if } q_j \notin Q_m, \\ (0,\ 1] & \text{if } q_j \in Q_m^+, \end{cases}$$

where $Q_m^- \subseteq Q$ and $Q_m^+ \subseteq Q$ are the set of negatively and positively marked states respectively. The $(n \times 1)$ characteristic vector is denoted as

$$\bar{\chi} \equiv [\chi_1\ \ \chi_2\ \ \cdots\ \ \chi_n]^{\mathrm{T}}.$$

Definition 2. The event cost is the relative frequency of occurrence of an event given the DFSA state at which the event is generated, and is defined as

- $\pi[\sigma_k | q_j] \equiv \tilde{\pi}_{jk} \in [0,\ 1)$, relative frequency of occurrence of event k at state j;
- $\tilde{\pi}[\sigma_k | q_j] = 0$ if $\delta(q_j, \sigma_k)$ is undefined, i.e., event k is not defined at state j.

The $(n \times m)$ event cost matrix is denoted as $\tilde{\Pi} \equiv [\tilde{\pi}_{ij}]$.

Definition 3. The state transition cost of the DFSA is a function $\pi : Q \times Q \to [0, 1)$ defined as the relative frequency of transition from state j to state k such that $\pi(q_k | q_j) = \sum \tilde{\pi}(\sigma | q_j) \equiv \pi_{jk}$ and the $n \times n$ state transition cost matrix, denoted as Π-matrix, is defined as

$$\Pi = \begin{bmatrix} \pi_{11} & \pi_{12} & \cdots & \pi_{1n} \\ \pi_{21} & \pi_{22} & \cdots & \pi_{2n} \\ \vdots & & \ddots & \vdots \\ \pi_{n1} & \pi_{n2} & \cdots & \pi_{nn} \end{bmatrix}.$$

Note that event costs and state transition costs are very much similar to probabilities and probabilistic interpretation is given in Ray et al. (2005). In this sense, an event cost can be analogous to the probability of an event to occur at a state, and a state transition cost can be analogous to the probability of leaping from one state to another.

Now we define the language measure in terms of the signed state weight function χ and the non-negative state transition cost π.

Definition 4. The signed real measure of the language $L(G_i)$ created by a DFSA G_i, initialized at the state $q_i \in Q$, is defined as

$$\mu_i \equiv \mu(L(G_i)).$$

The $(n \times 1)$ real signed measure vector is denoted as

$$\bar{\mu} \equiv [\mu_1\ \ \mu_2\ \ \cdots\ \ \mu_n]^{\mathrm{T}}.$$

It has been shown in Ray (2005), and Ray et al. (2005) that the measure of the language $L(G_i)$, where $G_i = (Q, \Sigma, \delta, q_i, Q_m)$ can be expressed as $\mu_i = \sum_j \pi_{ij}\,\mu_j + \chi_i$. Equivalently, in vector notation: $\bar{\mu} = \Pi\bar{\mu} + \bar{\chi}$. Therefore, the measure vector $\bar{\mu}$ is uniquely determined as

$$\bar{\mu} = [I - \Pi]^{-1}\bar{\chi}.$$

In the gas turbine engine application, the penalty of disabling controllable events (e.g., redistribution of thrust between two engines, and nozzle area reduction for individual engines) is set to zero as these manipulations do not require any special effort.

The state-based optimal control policy is obtained by selectively disabling controllable events to maximize the measure of the controlled plant language. In each iteration, the optimal control algorithm attempts to disable all controllable events leading to "negatively marked states" and enable all controllable events leading to "positively marked states". It has been also shown in Ray et al. (2004, 2005) that computational complexity of the control synthesis is polynomial in the number of plant states.

The algorithm for synthesis of the optimal control policy is summarized as follows: let G be the DFSA plant model without any constraint of operational specifications. Let the state transition cost matrix of the unsupervised plant be: $\Pi^{plant} \in \Re^{n \times n}$ and the characteristic vector be: $\bar{\chi} \equiv [\chi_1\ \ \chi_2\ \ \cdots\ \ \chi_n]^{\mathrm{T}}$. Starting with iteration index $k = 0$, and $\Pi^0 \equiv \Pi^{plant}$, the control policy is constructed by the following two-step procedure:

Step 1: For every state q_j for which $\mu_j^0 < 0$, disable controllable events leading to q_j. Now, $\Pi^1 = \Pi^0 - \Delta^0$, where $\Delta^0 \geq 0$ is composed of event costs corresponding to all controllable events that have been disabled at $k = 0$.

Step 2: Starting with $k = 1$, if $\mu_j^k \geq 0$, re-enable all controllable events leading to q_j, which were disabled in Step 1. The cost matrix is updated as: $\Pi^{k+1} = \Pi^k + \Delta^k$ for $k \geq 1$, where $\Delta^k \geq 0$ is composed of event costs corresponding to all currently re-enabled controllable events. The iteration is terminated when no controllable event leading to q_j remains disabled for which $\mu_j^k \geq 0$, i.e., if there is no sign change in the measure vector $\bar{\mu}$ between two consecutive iteration steps. At this stage, the optimal value of the performance index, which is the language measure, is $\bar{\mu}^* = [I - \Pi^*]^{-1}\bar{\chi}$.

References

Basseville, M., & Nikiforov, I. V. (1993). *Detection of abrupt changes: Theory and application*. New Jersey: PTR Prentice-Hall.

Cassandras, C. G., & Lafortune, S. (1999). *Introduction to discrete event systems*. Norwell, MA: Kluwer Academic.

DeLaat, J. C., & Chang, C. T. (2003). Active control of high frequency combustion instability in aircraft gas-turbine engines. *International symposium on air breathing engines*, Cleveland, OH, 2003.

Diao, Y., & Passino, K. M. (2001). Stable fault-tolerant adaptive fuzzy/neural control for a turbine engine. *IEEE Transactions on Control Systems Technology*, *9*(3), 494–509.

Fu, J., Yasar, M., & Ray, A. (2004). Optimal discrete event supervisory control of aircraft gas turbine engines. *Proceedings of American Control Conference*, *6*, 5710–5715.

Hopcroft, J. E., Motwani, R., & Ullman, J. D. (2001). *Introduction to automata theory*, *languages*, *and computation* (2nd ed). Reading, MA: Addison-Wesley.

Kumar, R., & Garg, V. K. (1995). *Modeling and control of logical discrete event systems*. Norwell, MA: Kluwer Academic ISBN 0-7923-9538-7.

Litt, J. S., Parker, K. I., & Chatterjee, S. (2003). Adaptive gas turbine engine control for deterioration compensation due to aging. *International symposium on air breathing engines*, Cleveland, OH, 2003.

Ljung, L. (1999). *System identification*: *Theory for the user* (2nd ed). New Jersey: Prentice-Hall.

Parker, K. I., & Guo, T. H. (2002). Development of a turbofan engine simulation in a graphical simulation environment. *JANNAF 26th aero-propulsion subcommittee meeting*, Destin, FL, 2002.

Ramadge, P. J., & Wonham, W. M. (1987). Supervisory control of a class of discrete event processes. *SIAM Journal on Control and Optimization*, *25*(1), 206–230.

Ray, A. (2005). Signed real measure of regular languages for discrete event supervisory control. *International Journal of Control*, *78*(12), 949–967.

Ray, A., Fu, J., & Lagoa, C. (2004). Optimal supervisory control of finite state automata. *International Journal of Control*, *77*(12), 1083–1100.

Ray, A., Phoha, V. V., & Phoha, S. (2005). *Quantitative measure for discrete event supervisory control*. New York: Springer ISBN 0-387-02108-6.

Tolani, D. K., Yasar, M., Ray, A., & Yang, V. (2006). Anomaly detection in aircraft gas turbine engines. *Journal of Aerospace Computing, Information, and Communication*, *3*(2), 44–51.

Volponi, A. J., Brotherton, T., Luppold, R., & Simon, D. L. (2003). Development of an information fusion system for engine diagnostics and health management. *JANNAF 27th airbreathing propulsion subcommittee meeting*, Colorado Springs, CO, 2003.

Yasar, M., Horn, J. F., & Ray, A. (2006). Effects of supervisory decisions on nonlinear aircraft dynamics. *Proceedings of American control conference*, 4154–4159.