Dynamic Multiple-Fault Diagnosis With Imperfect Tests

Sui Ruan, Yunkai Zhou, Feili Yu, *Member, IEEE*, Krishna R. Pattipati, *Fellow, IEEE*, Peter Willett, *Fellow, IEEE*, and Ann Patterson-Hine

***Abstract*—In this paper, we consider a model for the dynamic multiple-fault diagnosis (DMFD) problem arising in online monitoring of complex systems and present a solution. This problem involves real-time inference of the most likely set of faults and their time-evolution based on blocks of unreliable test outcomes over time. In the DMFD problem, there is a finite set of mutually independent fault states, and a finite set of sensors (tests) is used to monitor their status. We model the dependence of test outcomes on the fault states via the traditional D-matrix (fault dictionary). The tests are imperfect in the sense that they can have missed detections, false alarms, or may be available asynchronously. Based on the imperfect observations over time, the problem is to identify the most likely evolution of fault states over time. The DMFD problem is an intractable NP-hard combinatorial optimization problem. Consequently, we decompose the DMFD problem into a series of decoupled subproblems, one for each sample epoch. For a single-epoch MFD, we develop a fast and high-quality deterministic simulated annealing method. Based on the sequential inferences, a local search-and-update scheme is applied to further improve the solution. Finally, we discuss how the method can be extended to dependent faults.**

***Index Terms*—Approximate Bayesian revision, determinisitic simulated annealing, dynamic fault diagnosis, functional HMMMs, hidden Markov models (HMMs), Lagrangian relaxation, multiple faults.**

I. Introduction

A. Motivation

FAULT diagnosis is the process of identifying the failure states of a malfunctioning system by observing their effects at various test points. Fault diagnosis can be roughly categorized as being static or dynamic. In static fault diagnosis, the observed test outcomes are available as a block at one time instant, while in dynamic (online) fault diagnosis, the test outcomes are obtained over time. Both static and dynamic fault diagnoses have a number of applications in engineering and medicine.

Online system-health monitoring and fault diagnosis in complex systems, such as in the space shuttle, aircraft, and satellites, are essential in reducing the likelihood of disasters due to sudden failures, and to improve system availability. In many mission-critical operations, such as the systems aforementioned, it is imperative that malfunctioning systems be quickly diagnosed and reconfigured. With the recent advances in intelligent sensors that provide real-time information on the system state, there is an increasing trend toward transmission of onboard diagnostic results to the ground-based test systems and operators. Such an integrated system-health management solution is expected to reduce maintenance and operations costs by reducing troubleshooting time at the ground station, by reducing "cannot duplicates," and by facilitating condition-based maintenance.

In spite of its importance, there are at least two technical challenges to be overcome. First, the computational burden of onboard processing of test results in complex systems is substantial due to the facts that such systems are composed of large numbers of components and sensors, and that sensors are sampled frequently. Second, due to operator errors, electromagnetic interference, environmental conditions, or aliasing inherent in the signature analysis of onboard tests, the nature of tests may be unreliable (imperfect). Imperfect tests introduce additional elements of uncertainty into the diagnostic process: The "PASS" outcome of a test does not guarantee the integrity of components under test because the test may have missed a fault; on the other hand, a "FAIL" outcome of a test does not mean that one or more of the implicated components are faulty because the test outcome may have been a false alarm. In addition, at any sampling time, the results of all test decisions in a system are not available due to varying sampling rates of the sensors and signal-processing limitations. Thus, the problem is one of determining the fault states of components, given a set of partial and unreliable test outcomes over time. Consequently, diagnostic logic that hedges against this uncertainty in test outcomes is of significant interest to the diagnostic community [2], [11], [14], [15], [18].

B. Previous Work

Previous research on system fault diagnosis has focused on test-sequencing problems for single fault diagnosis [15], [17] and its variants [18], [20], test sequencing for multiple-fault

Manuscript received June 3, 2006; revised September 4, 2007 and July 3, 2008. Current version published October 16, 2009. This work was supported in part by the Office of Naval Research under Contract 00014-00-1-0101 and N00014-06-1-0080 and in part by the NASA Ames Research Center under Contract NAG2-1635. This paper was recommended by Associate Editor H. Pham.

S. Ruan was with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT 06269-1157 USA. She is now with the Department of Revenue Management and Planning, American Airlines, TX 76155 USA.

Y. Zhou is with the Department of Mathematics, Southern Methodist University, Dallas, TX 75275-0156 USA (e-mail: yzhou@smu.edu).

F. Yu, K. R. Pattipati, and P. Willett are with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT 06269-1157 USA (e-mail: yu02001@engr.uconn.edu; krishna@engr.uconn.edu; willett@engr.uconn.edu).

A. Patterson-Hine is with the NASA Ames Research Center, Moffett Field, CA 94035-1000 USA (e-mail: apatterson-hine@mail.arc.nasa.gov).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCA.2009.2025572

diagnosis (MFD) [19], and the "static" MFD [12], [19], [21], [24], [25] of identifying the set of most probable fault states given a set of test outcomes. Dynamic single-fault diagnosis problem was first proposed by Ying *et al.* [23], where it is assumed that, at any time, the system has, at most, one fault state present. This modeling is unrealistic for most of the real-world systems. Another version of dynamic fault diagnosis was studied in [7], where unknown probabilities of sensor error, incompletely populated sensor observations, and multiple faults were allowed, but the faults could only occur or clear once per sampling interval. This assumption is invalid when observations are less frequent or the effects of many faults are removed due to resets, or when a system experiences a major catastrophe resulting in multiple faults appearing during the same sampling interval.

The solution strategies applied in [7] and [23] are also computationally infeasible for general dynamic fault diagnosis problems. In dynamic single-fault framework [23], a hidden Markov modeling (HMM) framework was adopted, and a moving window Viterbi algorithm was used to infer the evolution of fault states. In the multiple-fault case, the state space of hidden Markov model increases exponentially from $(m+1)$ to 2^m, where m is the number of possible fault states. Consequently, the HMM-based method would be viable only for small-sized systems. The solution method proposed in [7] is a multiple-hypothesis-tracking approach, where at each observation epoch, k-best fault-state configurations are stored. At each epoch, all candidate fault sets, derived from the previously identified faults, are listed, based on, at most, one change per-epoch assumption. Then, of all $k(m+1)$ possible candidate sets, each has its score calculated, the candidate set which obtains the highest score is selected as the inference result at the epoch, and the candidates with the k-best scores are updated. The method is equivalent to enumeration in a limited search space; consequently, it is either computationally expensive or far from optimal.

In this paper, we relax the modeling assumptions imposed in [7] and [23] and devise a computationally efficient method for near-optimal solution to the dynamic MFD (DMFD) problem.

C. Scope and Organization of This Paper

This paper is organized as follows. In Section II, we formulate the DMFD problem with imperfect test outcomes. In Section III, a solution is proposed, where the DMFD problem is decomposed into a series of single-epoch MFD problems, coupled with a local-search heuristic for interepoch smoothing to further increase the overall likelihood of correct fault diagnosis. Simulation results are presented in Section IV. In this section, we discuss how the DMFD model can be extended to allow for interfault state dependences and present preliminary results. Finally, this paper concludes with a summary and future research directions in Section V.

II. DMFD Problem Formulation

A. Problem Formulation

The DMFD problem consists of a set of possible fault states in a system and a set of binary-outcome tests that are observed at each sample (observation, decision) epoch. Fault states are assumed to be independent. Each test outcome provides information on a subset of the set of fault states. At each sample epoch, a subset of test outcomes is available. Tests are imperfect in the sense that the "Pass" outcome of a test does not guarantee the integrity of components under test because the test may have missed the faults. A "Fail" outcome of a test does not mean that one or more of the implicated components are faulty because the test outcome may have been a false alarm. Our problem is to determine the time evolution of fault states based on imperfect test outcomes observed over time. Formally, the DMFD problem with imperfect tests, $D = \{S, \kappa, Pa, Pv, T, Pd, Pf\}$, a special case of factorial hidden Markov model [10], consists of the following conditions.

1) $S = \{s_1, \ldots, s_m\}$ is a finite set of m fault states associated with the system.
2) $\kappa = \{0, \ldots, k, \ldots K\}$ is the set of discretized observation epochs. We denote $x_i(k)$ as the status of fault state s_i at epoch k. That is, $x_i(k) = 1$ when s_i is faulty at epoch k, and $x_i(k) = 0$, otherwise. Let $\underline{x}(k) = \{x_1(k), x_2(k), \ldots, x_m(k)\}$ denote the status of all fault states at epoch k. We assume that the initial state $\underline{x}(0)$ is known.
3) Each fault state is modeled as a two-state nonhomogenous Markov chain. For each fault state, e.g., s_i, at each epoch, the fault-appearance probability $Pa_i(k)$ and fault-disappearance probability $Pv_i(k)$ are defined as $Pa_i(k) = \text{Prob}(x_i(k) = 1 | x_i(k-1) = 0)$ and $Pv_i(k) = \text{Prob}(x_i(k) = 0 | x_i(k-1) = 1)$, respectively, as shown in Fig. 1(b).
4) $T = \{t_1, t_2, \ldots, t_n\}$ is a finite set of n available binary-outcome tests, where the integrity of the system can be ascertained. The outcomes of tests are binary, i.e., $O(t_j) \in \{\text{pass}, \text{fail}\}$.
5) The set of fault states S and the set of binary test outcomes T are related by a fault-dictionary matrix $D = \{d_{ij}\}$, where $d_{ij} = 1$ if failure source s_i is detected by test t_j and zero otherwise. For each $d_{ij} = 1$, a pair of detection and false-alarm probabilities, Pd_{ij} and Pf_{ij}, is specified, where Pd_{ij} and Pf_{ij} are the detection and false-alarm probabilities of test t_j, respectively, i.e., $Pd_{ij} = \text{Prob}(O(t_j) = \text{fail} | x_i = 1)$ and $Pf_{ij} = \text{Prob}(O(t_j) = \text{fail} | x_i = 0)$, as shown in Fig. 1(c). For notational convenience, when s_i does not affect the outcome of t_j, i.e., $d_{ij} = 0$, we let the corresponding $Pd_{ij} = Pf_{ij} = 0$.
6) At each observation epoch, k, $k \in \kappa$, test outcomes up to and including epoch k are available, i.e., we let $T^k = \{T(b) = (T_p(b), T_f(b))\}_{b=1}^{k}$, where $T(b)$ is the set of test outcomes at epoch b, with $T_p(b) (\subseteq T)$ and $T_f(b) (\subseteq T)$ as the sets of passed and failed tests at epoch b, respectively, as shown in Fig. 1(a). The tests are imperfect in the sense that outcomes of some tests may not be available, i.e., $(T_p(b) \bigcup T_f(b)) \subset T$. In addition, tests exhibit missed detections and false alarms. In this paper, we make the causal independence assumption: the passed and failed test outcomes are conditionally independent,

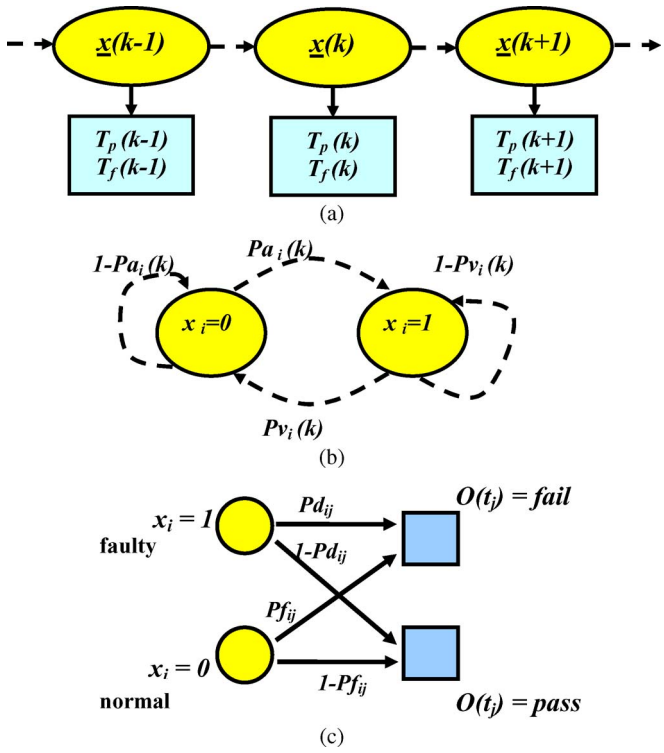


Fig. 1. DMFD modeling illustration. (a) Factorial hidden Markov model of DMFD. (b) Fault appearance and disappearance probabilities of s_i. (c) Detection and false-alarm probabilities of t_j for fault state s_i.

given the status of fault states, and the probabilistic noisy-OR assumption: a test t_j fails at epoch k if it fails on any of its associated fault states at epoch k [21], [26].

The DMFD problem is one of finding, at each decision epoch $k \in \{1, \ldots, K\}$, the most likely fault-state candidates $\underline{x}(k) \in \{0,1\}^m$, i.e., the fault-state evolution over time, $X^K = \{\underline{x}(1), \ldots, \underline{x}(K)\}$, that best explains the observed test-outcome sequence T^K. We formulate this as one of finding the maximum *a posteriori* (MAP) configuration

$$\widehat{X}^K = \arg\max_{X^K} \text{Prob}\left(X^K | T^K, \underline{x}(0)\right). \quad (1)$$

B. Illustrative Example

As an illustrative example, consider the system shown in Table I. The system consists of ten fault states and ten binary tests. For example, fault s_1 can be detected by t_1 and t_8 with $(Pd_{1,1}, Pf_{1,1}) = (0.95, 0.05)$ and $(Pd_{1,8}, Pf_{1,8}) = (0.7, 0)$, respectively. This implies that when fault state s_1 occurs, test t_1 detects it with probability 0.95, and if fault state s_1 does not occur, test t_1 has 0.05 probability of falsely implicating it. Similarly, test t_8 detects s_1 with a probability of 0.7 and has no false alarms. If s_1 is not present at epoch k, then at epoch $k+1$, the probability of having s_1 present is 0.12, while if s_1 is present at epoch k, s_1 has a probability 0.63 of disappearing at epoch $(k+1)$. Observations at each of the epochs from $k = 1$ to $k = 10$ are given in Table I. For example, at $k = 4$, the outcome of t_1 is observed as having failed, while

TABLE I
SIMPLE EXAMPLE SYSTEM

	$(s_i, t_j, Pd_{ij}, Pf_{ij})$	P_a	P_v
s_1	(1, 1, 0.95, 0.05) (1, 8, 0.7, 0)	0.12	0.63
s_2	(2, 5, 0.3, 0) (2, 8, 0.9, 0)	0.22	0.63
s_3	(3, 2, 0.5, 0) (3, 9, 0.6, 0)	0.05	0.63
s_4	(4, 4, 0.8, 0) (4, 5, 0.9, 0)	0.15	0.63
s_5	(5, 7, 0.8, 0) (5, 10, 0.7, 0)	0.05	0.63
s_6	(6, 4, 0.9, 0) (6, 6, 0.7, 0) (6,7, 0.6,0)	0.03	0.63
s_7	(7, 3, 0.9, 0) (7, 5, 0.95, 0)	0.06	0.86
s_8	(8, 9, 0.4, 0) (8, 10, 0.9, 0)	0.12	0.63
s_9	(9, 3, 0.9, 0.05) (9, 4, 0.7, 0)	0.12	0.63
s_{10}	(10, 2, 0.7, 0) (10, 6, 0.9, 0)	0.12	0.63

k	Failed Tests	Passed Tests
1	$\emptyset$	S
2	8	1 2 3 4 5 6 7 9 10
3	1 5 8	2 3 4 6 7 9 10
4	1	2 3 4 5 6 8 9 10
5	2	1 3 4 5 6 7 8 9 10
6	4 5 10	1 2 3 6 7 8 9
7	3 4	1 2 5 6 7 8 9 10
8	3 4 5	1 2 6 7 8 9 10
9	3 5	1 2 4 6 7 8 9 10
10	6	1 3 4 5 7 8 9 10

the outcomes of $\{t_2, t_3, t_4, t_5, t_6, t_8, t_9, t_{10}\}$ are observed as having passed, and the result of t_7 is missing. The task is to identify the evolution of fault states over the ten epochs. A feasible time evolution of fault states for this system is $\widehat{X}^{10} = \{[\emptyset], [2], [1, 2], [1], [3], [4, 8], [9], [7, 9], [7], [10]\}$.

III. PROBLEM SOLUTION

Applying the Bayes rule, the objective function is equivalent to

$$\widehat{X}^K = \arg\max_{X^K} \text{Prob}\left(T^K | X^K, \underline{x}(0)\right) \text{Prob}\left(X^K | \underline{x}(0)\right). \quad (2)$$

With passed and failed test outcomes being conditionally independent, given the status of fault states ("the noisy-OR assumption"), and the Markov property of fault-state evolution, the problem is equivalent to

$$\max_{X^K} \prod_{k=1}^{K} \{\text{Prob}\left(T_p(k) | \underline{x}(k)\right) \cdot \text{Prob}\left(T_f(k) | \underline{x}(k)\right) \cdot \text{Prob}\left(\underline{x}(k) | \underline{x}(k-1)\right)\} \quad (3)$$

where $T_p(k) \subseteq T$ and $T_f(k) \subseteq T$ denote the sets of passed and failed tests at epoch k, respectively.

We define a new function $f_k(\underline{x}(k), \underline{x}(k-1))$ as

$$f_k\left(\underline{x}(k), \underline{x}(k-1)\right) = \ln\{\text{Prob}\left(T_p(k) | \underline{x}(k)\right) \times \text{Prob}\left(T_f(k) | \underline{x}(k)\right) \times \text{Prob}\left(\underline{x}(k) | \underline{x}(k-1)\right)\}. \quad (4)$$

Therefore, the problem is equivalent to

$$\widehat{X}^K = \arg\max_{X^K} \sum_{k=1}^{K} f_k\left(\underline{x}(k), \underline{x}(k-1)\right). \quad (5)$$

Given the fault-state status $\underline{x}(k)$, the outcomes of tests are independent. Consequently

$$\text{Prob}(T_p(k)|\underline{x}(k)) = \prod_{t_j(k)\in T_p(k)} \text{Prob}(O(t_j(k)) = \text{pass}|\underline{x}(k)) \tag{6}$$

$$\text{Prob}(T_f(k)|\underline{x}(k)) = \prod_{t_j(k)\in T_f(k)} \text{Prob}(O(t_j(k)) = \text{fail}|\underline{x}(k)). \tag{7}$$

For test t_j to pass at epoch k, it shall pass on all its associated fault states, so that

$$\text{Prob}(O(t_j(k)) = \text{pass}|\underline{x}(k)) = \prod_{i=1}^{m} \text{Prob}(O(t_j(k)) = \text{pass}|x_i(k)) \tag{8}$$

where

$$\begin{aligned}\text{Prob}(O(t_j(k)) &= \text{pass}|x_i(k)) \\ &= \begin{cases} 1 - Pf_{ij}, & x_i(k) = 0 \\ 1 - Pd_{ij}, & x_i(k) = 1 \end{cases} \\ &= (1 - Pd_{ij})^{x_i(k)}(1 - Pf_{ij})^{1-x_i(k)}, \qquad x_i(k) \in \{0, 1\}.\end{aligned} \tag{9}$$

Evidently

$$\text{Prob}(O(t_j(k)) = \text{fail}|\underline{x}(k)) = 1 - \text{Prob}(O(t_j(k)) = \text{pass}|\underline{x}(k)). \tag{10}$$

In the same vein, the assumption of independent evolution of fault states leads to

$$\text{Prob}(\underline{x}(k)|\underline{x}(k-1)) = \prod_{i=1}^{m} \text{Prob}(x_i(k)|x_i(k-1)) \tag{11}$$

where

$$\text{Prob}(x_i(k)|x_i(k-1)) = \begin{cases} 1 - Pa_i(k), & x_i(k-1) = 0,\ x_i(k) = 0 \\ Pa_i(k), & x_i(k-1) = 0,\ x_i(k) = 1 \\ Pv_i(k), & x_i(k-1) = 1,\ x_i(k) = 0 \\ 1 - Pv_i(k), & x_i(k-1) = 1,\ x_i(k) = 1. \end{cases}$$

We put it in an exponential form for ease of computation as

$$\begin{aligned}\text{Prob}(x_i(k)|x_i(k-1)) &= (1-Pa_i(k))^{(1-x_i(k-1))(1-x_i(k))} Pa_i(k)^{(1-x_i(k-1))x_i(k)} \\ &\quad \times Pv_i(k)^{x_i(k-1)(1-x_i(k))} (1-Pv_i(k))^{x_i(k-1)x_i(k)}, \\ &\quad x_i(k-1),\ x_i(k) \in \{0, 1\}.\end{aligned} \tag{12}$$

From (3)–(12), we obtain

$$\begin{aligned}f_k(\underline{x}(k), \underline{x}(k-1)) &= \sum_{t_j\in T_p(k)} \sum_{i=1}^{m} [x_i(k) \ln(1 - Pd_{ij}) \\ &\qquad + (1 - x_i(k)) \ln(1 - Pf_{ij})] \\ &\quad + \sum_{t_j\in T_f(k)} \ln \left[1 - \prod_{i=1}^{m} (1 - Pd_{ij})^{x_i(k)} \right. \\ &\qquad \left. \times (1 - Pf_{ij})^{(1-x_i(k))}\right] \\ &\quad + \sum_{i=1}^{m} \{(1 - x_i(k-1))(1 - x_i(k)) \ln(1 - Pa_i(k)) \\ &\qquad + (1 - x_i(k-1)) x_i(k) \ln Pa_i(k) \\ &\qquad + x_i(k-1)(1 - x_i(k)) \ln Pv_i(k) \\ &\qquad + x_i(k-1)x_i(k) \ln(1 - Pv_i(k))\}, \\ &\qquad \underline{x}(k),\ \underline{x}(k-1) \in \{0, 1\}^m.\end{aligned} \tag{13}$$

The problem posed in (5) and (13) is an NP-hard combinatorial optimization problem. Indeed, even the single-epoch problem, i.e., $\widehat{\underline{x}}(k) = \arg\max_{\underline{x}(k)} f_k(\underline{x}(k), \widehat{\underline{x}}(k-1))$, is NP-hard [19]. It is generally believed that NP-hard problems cannot be solved to optimality within polynomially bounded computation times [13], [22].

A. Optimal Solution

The DMFD problem is a special case of factorial HMM. Here, factorial HMM is a hidden Markov model with distributed state representation. Consequently, the Viterbi algorithm, a form of dynamic programming [4], [8], can be used to decode the time evolution of fault states. Formally

$$\delta_k(\underline{x}(k)) = \max_{X^{k-1}} \ln\left(\text{Prob}(T^k|X^k)\text{Prob}\left(X^k|\underline{x}(0)\right)\right), \qquad k = 1, \ldots, K. \tag{14}$$

We can obtain the recursion of $\delta_k(\underline{x}(k))$ as

$$\begin{aligned}\delta_{k+1}(\underline{x}(k+1)) &= \max_{X^k} \ln\left(\text{Prob}(T^{k+1}|X^{k+1})\text{Prob}\left(X^{k+1}|\underline{x}(0)\right)\right) \\ &= \max_{X^k} \left\{ \ln(\text{Prob}(T(k+1)|\underline{x}(k+1))) \right. \\ &\qquad + \ln(\text{Prob}(\underline{x}(k+1)|\underline{x}(k))) \\ &\qquad \left. + \ln\left(\text{Prob}(T^k|X^k)\text{Prob}\left(X^k|\underline{x}(0)\right)\right) \right\} \\ &= \max_{\underline{x}(k)} \{f_{k+1}(\underline{x}(k+1), \underline{x}(k)) + \delta_k(\underline{x}(k))\}.\end{aligned} \tag{15}$$

With $\delta_k(\underline{x}(k))$ $(k = 1, \ldots, K)$ available, we can obtain the optimal $\widehat{X}^K = \{\widehat{\underline{x}}(k)\}_{k=1}^K$ by

$$\widehat{\underline{x}}(k) = \arg\max_{\underline{x}(k)} \{\delta_k(\underline{x}(k))\}, \qquad k = K, \ldots, 1. \tag{16}$$

The complexity of the earlier Viterbi algorithm is $O(K2^{2m})$. By exploiting the independence of fault-state evolution, this complexity can be reduced to $O(Km2^{m+1})$, which is still exponential. Evidently, the optimal solution via Viterbi algorithm suffers from the curse of dimensionality and is impractical for large-scale problems. Consequently, there is much interest in approximation algorithms that can find near-optimal solutions with reasonable computation times.

We propose a practical solution, which decomposes the problem into a sequence of subproblems. At epoch k, we solve a single-epoch MFD problem, i.e., $\widehat{\underline{x}}(k) = \arg\max_{\underline{x}(k)} f_k(\underline{x}(k), \widehat{\underline{x}}(k-1))$, where $\widehat{\underline{x}}(k-1)$ is the solution from the previous epoch, i.e., $(k-1)$th epoch. Based on the sequential inferences, a local-search method is applied to further improve the quality of the solution.

B. Single-Epoch MFD Subproblems

For epoch k, the binary-programming problem of $\widehat{\underline{x}}(k) = \arg\max_{\underline{x}(k)} f_k(\underline{x}(k), \widehat{\underline{x}}(k-1))$ is NP-hard [19]. Theoretically, many combinatorial optimization techniques can be applied to solve the single-epoch MFD, such as the branch-and-bound, dynamic programming, genetic algorithms [5], and so on. However, these algorithms either require substantial memory or significant computation times ("curse of dimensionality").

In [19], Shakeri *et al.* presented a Langragian relaxation algorithm (LRA) for the single-epoch MFD problem. The Langrangian relaxation method decomposes the problem into a series of low-order linear-integer-programming problems, by introducing a set of Langrange multipliers. The linear-integer-programming problem is solved by Lagrangian-relaxation-based set-covering algorithms with polynomial complexity. Surrogate-subgradient method is applied to update the Langrange multipliers. This method can obtain a near-optimal solution and also provide an upper bound on the single-epoch objective function. For the DMFD problem, we presented in [18] a strategy for the initialization of Lagrange multipliers at each epoch to speed up the convergence. In that work, the Lagrange multiplers at an epoch are initialized with the optimal multipliers of the same failed tests from the most recent past epochs. With this initialization, the computation times can be reduced significantly, compared to an approach where, at each epoch, the Langrange multipliers are initialized from scratch. In spite of these improvements, as we demonstrate later, the LRA has substantial computational requirements at each epoch and is infeasible for real-world dynamic fault diagnosis problems requiring real-time inference.

In [24], an approximate-belief-revision (ABR) heuristic is proposed for solving the Quick Medical Reference-Decision Theoretic (QMR-DT) problem (a variant of the single-epoch MFD problem). ABR consists of a mean-field approximation and a message-passing scheme. The ABR algorithm seeks to identify a set of propositions that best explains the observed evidence by employing the following two steps: 1) approximate the belief (pseudomarginal posterior probability) of each fault state using the naive mean-field method, and 2) iteratively update the messages transferred between the fault states and the observed test outcomes. This process is repeated until the beliefs converge. The second step corresponds to Gauss–Seidel iteration from optimization theory [3]. The pseudoposterior probabilities estimated by the ABR are coarse; however, the rank order reflects the likely fault states. ABR is computationally efficient as compared to LRA, while it cannot achieve the high diagnostic accuracy of the LRA.

We propose a faster and high-quality deterministic simulated annealing (DSA) algorithm for single-epoch MFD problem, which is inspired by the ABR algorithm and stochastic simulated annealing (SSA). The new algorithm exploits their advantages, viz., the ability of SSA to get out of a local minimum and the computational efficiency of ABR, while overcoming their disadvantages, viz., the diagnostic inaccuracies of ABR and the slow convergence of SSA. In the following, we first introduce the ABR algorithm, then elaborate on the DSA algorithm, and finally state its relation to the SSA algorithm.

1) ABR algorithm: In ABR, each decision variable $x_i(k)$ is relaxed to be a real value in [0, 1], a soft-decision variable, and is interpreted as a pseudoposterior probability, i.e., $x_i(k) \cong \text{Prob}(x_i(k) = 1 | T_p(k), T_f(k), \widehat{x}_i(k-1))$.

Let us define the following operators:

$$\mathbf{0}(\underline{x}(k), i) = \{0, (j = i); x_j(k), (j \neq i)\} \tag{17}$$
$$\mathbf{1}(\underline{x}(k), i) = \{1, (j = i); x_j(k), (j \neq i)\}. \tag{18}$$

Since the expression shown at the bottom of the page, also, from (4), we have $\text{Prob}(T(k), X(k) | \widehat{X}(k-1)) = \exp(f_k(\underline{x}(k), \widehat{\underline{x}}(k-1)))$. ABR essentially updates the estimate of $x_i(k)$, at each iteration l, as follows:

$$\begin{aligned}\widehat{x}_i^{(l+1)}(k) &= \frac{e^{f_k\left(\mathbf{1}\left(\widehat{\underline{x}}^{(l)}(k), i\right), \cdot\right)}}{e^{f_k\left(\mathbf{0}\left(\widehat{\underline{x}}^{(l)}(k), i\right), \cdot\right)} + e^{f_k\left(\mathbf{1}\left(\widehat{\underline{x}}^{(l)}(k), i\right), \cdot\right)}} \\ &= \left[1 + e^{-\left(f_k\left(\mathbf{1}\left(\widehat{\underline{x}}(k), i\right), \cdot\right) - f_k\left(\mathbf{0}\left(\widehat{\underline{x}}(k), i\right)\right), \cdot\right)}\right]^{-1} \\ &= \left[1 + e^{-\Delta_i^{(l)}(k)}\right]^{-1}\end{aligned} \tag{19}$$

$$\text{Prob}\left(x_i(k) = 1 | T(k), x_{j, j \neq i}(k), \widehat{x}_i(k-1)\right) = \frac{\text{Prob}\left(T(k), \mathbf{1}\left(\underline{x}(k), i\right) | \widehat{X}(k-1)\right)}{\text{Prob}\left(T(k), \mathbf{1}\left(\underline{x}(k), i\right) | \widehat{X}(k-1)\right) + \text{Prob}\left(T(k), \mathbf{0}\left(\underline{x}(k), i\right) | \widehat{X}(k-1)\right)}$$

where

$$\Delta_i^{(l)}(k) = f_k\left(\mathbf{1}\left(\underline{\widehat{x}}^{(l)}(k), i\right), \cdot\right) - f_k\left(\mathbf{0}\left(\underline{\widehat{x}}^{(l)}(k), i\right), \cdot\right) \tag{20}$$

and $f_k(\underline{x}(k), \cdot)$ is the simplified notation for $f_k(\underline{x}(k), \underline{\widehat{x}}(k-1))$. The sequence of fault-state belief updates is permuted randomly. Note that the sequence of updating $\widehat{x}_i^{(l+1)}(k)$ is the Gauss–Seidel method, i.e., latest value of $\widehat{x}_j(k)(j \neq i)$ is used instead of using $\widehat{x}_j^{(l)}(k)$ of last batch as in the Jacobi method, and the belief-propagation algorithms [16].

As discussed in [24], the belief-update mechanism with the Gauss–Seidel method facilitates quick convergence of $\underline{x}(k)$, and $\widehat{x}_i(k)$ reflects the belief state of s_i at epoch k. However, $\widehat{x}_i(k)$ is an inaccurate estimate of $\text{Prob}(x_i(k) = 1 | T_p(k), T_f(k), \widehat{x}_i(k-1))$. A simple decision rule is applied on the converged $x_i(k)$, such that if $x_i(k) \geq 0.5$, then $x_i(k) = 1$; otherwise, $x_i(k) = 0$. These introduce significant suboptimality.

2) DSA: To overcome the inaccuracy of ABR algorithm and keep the advantage of the belief-update mechanism of the Gauss–Seidel method employed by the ABR algorithm, we propose a new belief-update method, which may be viewed as a DSA method.

In our DSA algorithm, each decision variable $x_i(k)$, like those in ABR, is relaxed to be a real value in [0, 1]. When converged, the values of $x_i(k)$ are close to being binary, rather than their posterior-probability estimates.

The sequence of fault-state belief updates are permuted randomly. When fault state s_i at epoch k is considered, with the current estimate of $\underline{x}(k)$ as $\underline{\widehat{x}}^{(l)}(k)$ at the iteration index l, the next estimate of $x_i(k)$ is obtained as

$$\widehat{x}_i^{(l+1)}(k) = \left[1 + e^{-\frac{\Delta_i^{(l)}(k)}{c^v}}\right]^{-1} \tag{21}$$

where $\Delta_i^{(l)}(k)$ is defined in (20). Here, c^v $(v = 1, \ldots, V)$ forms a decreasing list, i.e., $c^v \in C = [\beta^{-V_1}, \ldots, \beta^{-1}, 1, \beta, \ldots, \beta^{V_2}]$, where $\beta \in (0, 1)$ is called the cooling rate. Based on a value of c^v, (21) is iterated for each fault state in a randomized sequence, the iteration repeats until $\underline{x}(k)$ converges, and then c^v moves to the next value in the list C. Details of DSA algorithm for the DMFD problem is shown in Fig. 2.

At the beginning of the algorithm, $\mathbf{1}(\underline{\widehat{x}}^{(l)}(k), i)$ or $\mathbf{0}(\underline{\widehat{x}}^{(l)}(k), i)$, whichever has a higher function value f_k, is given a slight preference, i.e., if $\Delta_i^{(l)}(k) > 0$, then $\widehat{x}_i^{(l+1)}(k)$ is assigned a value slightly over 0.5, and when $\Delta_i^{(l)}(k) < 0$, then $\widehat{x}_i^{(l+1)}(k)$ is assigned a value slightly less than 0.5. As c^v decreases, higher function value of $\mathbf{1}(\underline{\widehat{x}}^{(l)}(k), i)$ or $\mathbf{0}(\underline{\widehat{x}}^{(l)}(k), i)$ is given dominant preference, i.e., if $\Delta_i^{(l)}(k) > 0$, $\widehat{x}_i^{(l+1)}(k)$ is assigned a value slightly less than one, when $\Delta_i^{(l)}(k) < 0$, $\widehat{x}_i^{(l+1)}(k)$ is assigned a value slightly larger than zero. Finally, $\widehat{x}_i(k)(\forall i)$ gradually converges to either zero or one. When $C = [1]$, the DSA algorithm is the same as ABR.

Initial temperature, cooling process, and stopping criteria directly influence the convergence speed of the simulated-annealing algorithm. DMFD may be directly used in real-time situations, so the timing requirement of the solution is a constraint imposed on the choice of initial temperature, cooling schedule, and the stopping criteria. For single-epoch MFD, the stopping criteria are as follows.

1) $\underline{x}^{(l)}(k)$ converges.
2) Maximum number of iterations is reached.
3) minimum temperature is reached.

In theory, DSA can asymptotically approach the global optimal solution when C is infinitely long, and β approaches one. Therefore, for high-quality solutions, length of C should be sufficiently long, and β should be sufficiently large [1]. However, to satisfy the real-time requirements, C should be short, and β should be small. Experimental results provided in a later section demonstrate that a moderate size of C, e.g., $|C| = 5$ and $\beta = 0.5$, provide satisfactory results. When $\underline{x}^{(l)}(k)$ converges, the same decision rule as that of ABR is applied.

How does DSA compare with the SSA? In SSA, $x_i(k) \forall i, k$ are hard-decision variables, i.e., binary. A Markov process is used to generate a stochastic search trajectory. We accept $\mathbf{1}(\underline{\widehat{x}}^{(l)}(k), i)$ as the next solution with probability

$$p_i^{(l)}(k) = \left[1 + e^{-\frac{\Delta_i^{(l)}(k)}{c^v}}\right]^{-1} \tag{22}$$

A Deterministic Simulated Annealing Method for Single Epoch MFD

Input: - system model D, Pa, Pv, Pd and Pf
- current epoch k, estimation of previous epoch, $\underline{\widehat{x}}(k-1)$
- test outcomes $T_f(k)$, $T_p(k)$
- $C = [\beta^{-V_1}, ..., 1, ..., \beta^{V_2}]$

Output: - $\underline{\widehat{x}}(k)$

Step 0: $\underline{\widehat{x}}(k) = 0$;

Step 1: For v from 1 to $|C|$
$c^v = C(v)$;
Repeat
For $i \in \{1, ..., m\}$ in a random sequence
- Compute $f_k(\mathbf{1}(\underline{\widehat{x}}^{(l)}(k), i), \cdot)$ and $f_k(\mathbf{0}(\underline{\widehat{x}}^{(l)}(k), i), \cdot)$;
- Update $\widehat{x}_i^{(l+1)}(k) = [1 + e^{-\frac{f(\mathbf{1}(\underline{\widehat{x}}^{(l)}(k), i), \cdot) - f(\mathbf{0}(\underline{\widehat{x}}^{(l)}(k), i), \cdot)}{c^v}}]^{-1}$

end For
Until $\underline{\widehat{x}}(k)$ converges;
end for

Step 2: Discretize $\underline{\widehat{x}}(k)$ by, for $i \in \{1, ..., m\}$, if $\widehat{x}_i(k) \geq 0.5$, $\widehat{x}_i(k) = 1$, $\widehat{x}_i(k) = 0$ otherwise;

Step 3: Output $\underline{\widehat{x}}(k)$.

Fig. 2. DSA method for single-epoch MFD.

i.e., set

$$\underline{x}^{(l+1)}(k) = \begin{cases} \mathbf{1}\left(\widehat{\underline{x}}^{(l)}(k), i\right), & \text{with probability } p_i^{(l)}(k) \\ \mathbf{0}\left(\widehat{\underline{x}}^{(l)}(k), i\right), & \text{with probability } 1 - p_i^{(l)}(k). \end{cases} \quad (23)$$

When $\Delta_i^{(l)}(k) > 0$, i.e., $\mathbf{1}(\widehat{\underline{x}}^{(l)}(k), i)$ has larger objective function, $\widehat{\underline{x}}^{(l+1)}(k)$ is more likely to select $\mathbf{1}(\widehat{\underline{x}}^{(l)}(k), i)$, while if $\Delta_i^{(l)}(k) < 0$, $\widehat{\underline{x}}^{(l+1)}(k)$ is more likely to select $\mathbf{0}(\widehat{\underline{x}}^{(l)}(k), i)$. By doing so, both improvement and deterioration are permitted at each iteration. Initially, for large values of c^v, a large fraction of deterioration is accepted; as c^v decreases, only smaller fractions of deterioration are accepted, and as the value of c^v approaches zero, no deterioration is accepted at all. Asymptotic convergence to the optimal solution is achieved only after an infinite number of transitions. In any finite-time implementation, one must resort to approximations of the asymptotic convergence.

SSA is very slow, partly because of the discrete nature of the search through the space of all configurations, the m-dimensional hypercube. Each trajectory is along a single edge, thereby missing full-gradient information that would be provided by analog-state values in the "interior" of the hypercube [6]. The similarity between DSA and SSA for the DMFD problem is that, at each iteration, the term $[1 + e^{-\Delta_i^{(l)}(k)/c^v}]^{-1}$ is calculated. However, it is used for different purposes in DSA and SSA. In DSA, it is used to update the soft-decision variable $x_i(k)$, while in SSA, it is used as the probability of selecting $\mathbf{1}(\widehat{\underline{x}}^{(l)}(k), i)$. In DSA, the relaxed soft-decision variables, which include the gradient information, facilitate quick convergence of the simulated-annealing methods.

3) Extension to multiepoch MFD: Single-epoch MFD can be extended to incorporate fault states of multiple consecutive epochs rather than only one epoch. Let $Y_k = [\underline{x}(k - H + 1), \ldots, \underline{x}(k-1), \underline{x}(k)]$, where H denotes the number of epochs in multiepoch MFD problem. Then

$$F_k(Y_k) = \sum_{b=k-H+2}^{k} f_b\left(\underline{x}(b), \underline{x}(b-1)\right) + f_{k-H+1}\left(\underline{x}(k-H+1), \widehat{\underline{x}}(k-H)\right). \quad (24)$$

Define $\mathbf{1}(Y_k, b, i)$ and $\mathbf{0}(Y_k, b, i)$ as

$$\mathbf{1}(Y_k, b, i) = \{1, (b' = b, j = i); x_j(b'), \text{otherwise}\}$$
$$\mathbf{0}(Y_k, b, i) = \{0, (b' = b, j = i); x_j(b\prime), \text{otherwise}\}. \quad (25)$$

Therefore, the DSA algorithm is applied on variable Y_k as follows:

$$\widehat{x}_i^{(l+1)}(b) = \widehat{Y}_k^{(l+1)}(b, i) = \left\{1 + e^{-\frac{F_k\left(\mathbf{1}\left(\widehat{Y}_k^{(l)}, b, i\right)\right) - F_k\left(\mathbf{0}\left(\widehat{Y}_k^{(l)}, b, i\right)\right)}{c^v}}\right\}^{-1}. \quad (26)$$

Within the time window $[k - H + 1, \ldots, k]$, iteration as in (26) can be applied back and forth until Y_k has converged.

C. Local Search for Interepoch Smoothing

At each time epoch k, when the solution to a single-epoch MFD problem is obtained, results of a w epoch window, i.e., from time epoch $k - w + 1$ to k, can be further improved by local-search algorithms. Here, w is a user-defined constant, denoting the epoch-window size, for improvement. We propose a consistency-check-and-update (CCU) heuristic as a local-search strategy on the solution space of $\{0, 1\}^{mw}$.

To check the consistency of an estimate of $\widehat{x}_i(k)$ with its neighboring estimates, i.e., $\widehat{x}_i(k-1)$ and $\widehat{x}_i(k+1)$, we obtain $\text{Prob}(x_i(k)|\widehat{x}_i(k-1), \widehat{x}_i(k+1))$ as

$$\text{Prob}\left(x_i(k)|\widehat{x}_i(k-1), \widehat{x}_i(k+1)\right) = \frac{\text{Prob}\left(\widehat{x}_i(k+1)|x_i(k)\right)\text{Prob}\left(x_i(k)|\widehat{x}_i(k-1)\right)}{\sum_{x_i(k)}\text{Prob}\left(\widehat{x}_i(k+1)|x_i(k)\right)\text{Prob}\left(x_i(k)|\widehat{x}_i(k-1)\right)} \quad (27)$$

where $\text{Prob}(\widehat{x}_i(k+1)|x_i(k))$ and $\text{Prob}(x_i(k)|\widehat{x}_i(k-1))$ are as defined in (12).

When $\text{Prob}(x_i(k) = 1|\widehat{x}_i(k-1), \widehat{x}_i(k+1)) > 0.5$ while $\widehat{x}_i(k) = 0$ or $\text{Prob}(x_i(k) = 0|\widehat{x}_i(k-1), \widehat{x}_i(k+1)) < 0.5$ while $\widehat{x}_i(k) = 1$, we consider the estimation $\widehat{x}_i(k)$ to be not consistent with $\widehat{x}_i(k-1)$ and $\widehat{x}_i(k+1)$. For inconsistent estimate $\widehat{x}_i(k)$, we flip the bit i of $\widehat{\underline{x}}(k)$ and keep the change if the flip action increases the overall objective function; otherwise, it remains at the previous estimate.

Beginning with current epoch k, CCU improves the diagnosis $\{\widehat{\underline{x}}(k-w+1), \ldots, \widehat{\underline{x}}(k)\}$ in the following way.

1) Backward move: When considering improvement at epoch b, for each fault state (in a random sequence), i.e., $\widehat{x}_i(b)$, check whether it is consistent with $\widehat{x}_i(b-1)$ and $\widehat{x}_i(b+1)$. If not, flip $\widehat{x}_i(b)$ to check whether it will increase the objective function of $f_{b+1}(\widehat{\underline{x}}(b+1), \underline{x}(b)) + f_b(\underline{x}(b), \widehat{\underline{x}}(b-1))$. If yes, keep the change; continue checking and updating in the epoch decreasing direction until no improvement can be made in any epoch or until the left most of the epoch window, i.e., $k - w + 1$, is reached.
2) Forward move: apply the same checking and updating process along the epoch increasing direction in a way that is similar to the backward move.
3) Move backward and forward inside the epoch window, until no more updates for improvement can be made in either direction.

This smoothing method can effectively identify inconsistent diagnosis and improve diagnostic accuracy. It is beneficial specially when the trajectory of fault-state evolution is stable and/or tests are less reliable, i.e., tests with low detection probability and high rate of false alarms.

IV. Simulation and On-Going Research

A. Randomly Generated Models

To compare the performance of various DMFD algorithms of interest, we constructed models of various sizes, (m, n) ranging from (20, 20) as small-scale models, (100, 150) as medium-scale models, and to (2000, 2500) as large-scale models, where m is the number of fault states and n is the number of tests. These models are reasonably realistic in the sense that detection probabilities of tests are high, i.e., 0.6–0.95, the false-alarm probabilities of tests are low, i.e., 0–0.05, and tests evenly cover the fault states. True fault-state set and, accordingly, the test outcomes of each epoch are generated according to the model parameters listed in Tables III–V). Algorithms LRA, ABR, DSA, DSA with CCU (denoted as DSA$^+$), and LRA with CCU (denoted as LRA$^+$) are applied. The performance metrics for a DMFD algorithm, for example, algorithm A, are defined as follows.

1) Diagnostic accuracy, including correct-isolation rate and false-isolation rate, i.e., $\overline{\text{CI}}(A)$ and $\overline{\text{FI}}(A)$. Let $Y(A,k)$ be the fault-state set at epoch k detected by algorithm A and $R(k)$ be the true fault-state set at epoch k. Correct-isolation and false-isolation rates of algorithm A for epoch k, i.e., $\text{CI}(A,k)$ and $\text{FI}(A,k)$, respectively, are calculated as

$$\text{CI}(A,k) = \frac{|Y(A,k) \cap R(k)|}{|R(k)|} \tag{28}$$

$$\text{FI}(A,k) = \frac{|Y(A,k) \cap \neg R(k)|}{|S| - |R(k)|}. \tag{29}$$

Accordingly, the average correct-isolation and false-isolation rate of algorithm A over all epochs are obtained as follows:

$$\overline{\text{CI}}(A) = \frac{\sum_{k=1}^{K} \text{CI}(A,k)}{K}$$

$$\overline{\text{FI}}(A) = \frac{\sum_{k=1}^{K} \text{FI}(A,k)}{K}. \tag{30}$$

2) Computation time per epoch $\overline{T}(A)$: The average computation time per epoch that algorithm A takes to diagnose.
3) Objective function value per epoch, $\overline{F}(A) : \overline{F}(A) = F(\widehat{X}_A^K)/K$, where $\widehat{X}_A^K$ is the solution of X^K obtained by algorithm A, and $F(X^K) = \sum_{k=1}^{K} f_k(\underline{x}(k), \underline{x}(k-1))$.

Our algorithms were coded in MATLAB, and results are obtained on a 2.26-GHz CPU and 512-MB-RAM PC. To illustrate the performance differences, results of ten Monte Carlo runs for models at each scale are listed in Tables III–V. The average computation time per epoch of each sized model is listed in Table II.

For the ABR and DSA algorithms, the stopping criterion used to verify that $x^{(l)}(k)$ has converged is

$$\alpha = \left| \underline{x}^{(l+1)}(k) - \underline{x}^{(l)}(k) \right| \leq 10^{-4}. \tag{31}$$

The annealing list C in the DSA algorithm is selected as $C = [4, 2, 1, 0.5, 0.25]$.

TABLE II
Average Computation Time Per Epoch (in Seconds)

Scale $(m \times n)$	Small (20×20)	Medium (100×150)	Large (2000×2500)
LRA	0.479	2.4135	92.7
ABR	0.0188	0.1596	25.1
DSA	0.0309	0.2812	72.8
LRA^+	0.480	2.416	95.4
DSA^+	0.0318	0.981	74.8
ABR^2	0.0093	0.0567	3.1
DSA^2	0.0164	0.0945	12.3

The results for small-scale system models, in Tables II and III, show that DSA can achieve similar diagnostic-accuracy and objective-function values as those from LRA, but with much less computational effort; ABR achieves reasonable diagnostic-accuracy and average objective-function values; CCU (e.g., DSA$^+$ and LRA$^+$) can provide performance improvement to their respective original algorithms (DSA and LRA).

From the results in Tables II and IV for medium-scale system models, it is seen that DSA achieves very good diagnostic performance with a fraction of computation time needed by the LRA. ABR has the least computation effort; however, the performance is inferior to LRA and DSA.

From the results in Tables II and V for large-scale system models, we can observe the following: 1) among ABR, LRA, and DSA, DSA consistently achieves the best performance; 2) LRA needs the most computation time while yielding less satisfactory diagnosis results, and the average duality gap for single-epoch MFD is quite high (over 10%); and 3) CCU can further improve the performance of DSA and LRA.

To summarize, among the ABR algorithm, DSA algorithm, and LRA, DSA can provide satisfactory accuracy with moderate computational effort; ABR consumes the least computation effort, while its diagnosis is less accurate. LRA needs considerable computation time, may get stuck at a local optimum with a high-duality gap [3], and the quality of diagnosis is not necessarily superior to DSA. The CCU heuristic (DSA$^+$, LRA$^+$) improves the diagnostic accuracy of the underlying algorithms (DSA, LRA).

B. Comparison With Various Choices of C

To evaluate the effect of cardinality of C and values of β on diagnostic performance, we vary β in [0, 1] and the list length from 1 to 13, respectively, on an arbitrary medium-scale model. Here, $C = [\beta^{(-|C|+1)/2}, \ldots, 1, \ldots, \beta^{(|C|-1)/2}]$.

The average function value and computation times by different settings of C list are shown in Figs. 3 and 4. From the simulation results, we observe that as $\beta \to 1$ and $|C| \to \infty$, the average function value can approach global optimum, but the computation time is substantially longer. When $C = 1$, it is actually ABR. As $\beta \to 0$, the computation time is less, but the algorithm behaves as greedy, which means no guarantee of diagnosis quality. As shown in Fig. 3, the solution quality of $\beta = 0.1$ varies most as compared with other β values in the two cases. When $|C|$ is limited, e.g., $|C| = 5$, β assuming moderate values, e.g., $\beta = 0.5$, there is a tradeoff between solution quality and computation time.

TABLE III
RESULT OF SIMULATION SYSTEM I

Number of Failure Sources:				20		(Pd, Pf):				(0.6 - 0.9, 0-0.05)	
Number of Tests				20		(Pa, Pv):				(0.03-0.06, 0.2-0.6)	
LRA	$\overline{CI}(LRA)$	86.7%	90%	77.3%	82%	92.2%	74.4%	92.9%	79.5%	86.3%	95%
	$\overline{FI}(LRA)$	1.23%	0.85%	2.84%	0.28%	0.61%	2.08%	0.29%	1.53%	0.28%	0.54%
	$\overline{F}(X^K_{LRA})$	-8.13	-5.64	-11.17	-7.32	-8.17	-11.2	**-5.49**	-7.54	**-7.91**	**-6.21**
	Duality Gap	10.1%	12.2%	10.2%	18.1%	12.7%	15.1%	9.5%	13%	2.6%	9.8%
ABR	$\overline{CI}(ABR)$	82%	90%	74.7%	79.6%	92.2%	70.1%	90.42%	78.5%	91.3%	90%
	$\overline{FI}(ABR)$	0.61%	0.56%	1.56%	0.28%	0.61%	1.16%	0.29%	0.63%	0%	0.26%
	$F(X^K_{ABR})$	-12.6	-6.44	-13.45	-7.47	-8.98	-14.71	-5.58	-8.62	-8.61	-7.62
DSA	$\overline{CI}(DSA)$	86.7%	91.7%	77.3%	80.8%	93.2%	80%	92.9%	84.7%	81.3%	95%
	$\overline{FI}(DSA)$	1.23%	0.56%	3.15%	0.59%	0.61%	3.32%	0.29%	1.74%	1.07%	0.54%
	$\overline{F}(X^K_{DSA})$	-8.13	**-5.48**	**-10.6**	**-7.23**	**-8.01**	-10.69	**-5.49**	-7.12	-8.12	**-6.21**
LRA	$\overline{CI}(LRA^+)$	86.8%	90%	77.3%	82.1%	92.2%	75.4%	92.9%	79.5%	86.3%	95%
with	$\overline{FI}(LRA^+)$	0.92%	0.85%	3.13%	0%	0.61%	2.08%	0.29%	1.24%	0.28%	0.54%
CCU	$\overline{F}(X^K_{LRA^+})$	**-8.05**	-5.64	-11.14	-7.32	-8.17	-11.19	**-5.49**	-7.49	**-7.91**	**-6.21**
DSA	$\overline{CI}(DSA^+)$	86.8%	91.7%	77.3%	80.8%	93.2%	78.7%	92.9%	84.6%	81.25%	95%
with	$\overline{FI}(DSA^+)$	0.92%	0.56%	3.15%	0.31%	0.61%	3.32%	0.29%	1.18%	1.07%	0.54%
CCU	$\overline{F}(X^K_{DSA^+})$	**-8.05**	**-5.48**	**-10.6**	**-7.23**	**-8.01**	**-10.67**	**-5.49**	**-7.04**	-8.12	**-6.21**

TABLE IV
RESULT OF SIMULATION SYSTEM II

Number of Failure Sources:				100		(Pd, Pf):				(0.6 - 0.9, 0-0.05)	
Number of Tests:				150		(Pa, Pv):				(0.01-0.02, 0.3-0.4)	
LRA	$\overline{CI}(LRA)$	89.6%	88.1%	85.9%	88.4%	86.7%	89%	87.9%	87.4%	89.1%	85.3%
	$\overline{FI}(LRA)$	0.52%	0.52%	0.66%	0.5%	0.77%	0.5%	0.51%	0.58%	0.58%	0.67%
	$\overline{F}(X^K_{LRA})$	-40.75	-43.09	-42.18	-42.19	-41.18	-43.42	-40.5	-42.34	-42.18	-44.18
	Duality Gap	12.7%	13.4%	13.7%	13.5%	13.5%	12.3%	12.3%	11.7%	12.1%	13.4%
ABR	$\overline{CI}(ABR)$	90.7%	89.7%	86.3%	89.7%	87.7%	91.43%	88.9%	86.9%	89.1%	85.4%
	$\overline{FI}(ABR)$	0.29%	0.33%	0.48%	0.37%	0.46%	0.23%	0.27%	0.3%	0.39%	0.34%
	$\overline{F}(X^K_{ABR})$	-41.38	-45.1	-44.47	-42.64	-43.84	-43.64	-42.85	-45.13	-44.68	-47.62
DSA	$\overline{CI}(DSA)$	91.7%	91.8%	88.5%	91%	89.4%	91.4%	91%	89.6%	92%	88.6%
	$\overline{FI}(DSA)$	0.52%	0.46%	0.68%	0.56%	0.69%	0.45%	0.55%	0.51%	0.6	0.59%
	$\overline{F}(X^K_{DSA})$	-39.75	-42	-41.03	-40.93	-40.23	-42.49	-39.64	-41.42	-41.41	-43.17
LRA	$\overline{CI}(LRA^+)$	89.3%	88.8%	87.2%	88.1%	86.6%	89.1%	88.1%	87.7%	89.5%	84.6%
with	$\overline{FI}(LRA^+)$	0.43%	0.44%	0.56%	0.4%	0.68%	0.35%	0.43%	0.44%	0.44%	0.55%
CCU	$\overline{F}(X^K_{LRA^+})$	-40.48	-42.69	-41.75	-41.85	-40.87	-43.21	-40.31	-42.02	-41.86	-43.87
DSA	$\overline{CI}(DSA^+)$	91.3%	91.3%	89.4%	90.2%	89.5%	91.7%	90.7%	90.8%	92.2%	87.5%
with	$\overline{FI}(DSA^+)$	0.43%	0.38%	0.57%	0.52%	0.56%	0.32%	0.43%	0.41%	0.5%	0.55%
CCU	$\overline{F}(X^K_{DSA^+})$	**-39.66**	**-41.82**	**-40.83**	**-40.82**	**-40.02**	**-42.33**	**-39.49**	**-41.22**	**-41.26**	**-43.02**

TABLE V
RESULT OF SIMULATION SYSTEM III

Number of Failure Sources:				2000		(Pd, Pf):				(0.7 - 0.95, 0-0.05)	
Number of Tests:				2500		(Pa, Pv):				(0.01-0.015, 0-0.6)	
LRA	$\overline{CI}(LRA)$	84.3%	83.8%	84.4%	83.9%	84.5%	84.7%	85.2%	85.9%	85.9%	84.5%
	$\overline{FI}(LRA)$	0.21%	0.21%	0.24%	0.21%	0.21%	0.21%	0.21%	0.21%	0.24%	0.21%
	$\overline{F}(X_{LRA})$	-1181.7	-1188.7	-1184.5	-1189.4	-1191	-1188.2	-1189.8	-1191.1	-1185.2	-1191.5
	Duality Gap	14.73%	14.77%	14.66%	14.66%	14.72%	14.57%	14.62%	14.66%	14.58%	14.53%
ABR	$\overline{CI}(ABR)$	86.04%	86.06%	86.60%	86.17%	86.15%	86.99%	87.12%	87.43%	87.03%	87.30%
	$\overline{FI}(ABR)$	0.11%	0.12%	0.11%	0.10%	0.11%	0.11%	0.11%	0.11%	0.11%	0.11%
	$\overline{F}(X_{ABR})$	-1171.3	-1177.8	-1172.1	-1178.7	-1180.5	-1177.6	-1178.3	-1180.6	-1176.1	-1180.8
DSA	$\overline{CI}(DSA)$	88.68%	88.22%	88.85%	88.98%	89.31%	89.35%	89.66%	89.45%	89.62%	89.62%
	$\overline{FI}(DSA)$	0.16%	0.18%	0.17%	0.15%	0.15%	0.15%	0.17%	0.16%	0.16%	0.17%
	$\overline{F}(X_{DSA})$	-1168.6	-1175.5	-1170.1	-1175.9	-1177.5	-1176.1	-1176.3	-1178.5	-1173.6	-1178.7
DSA	$\overline{CI}(DSA^+)$	87.79%	87.79%	87.76%	87.92%	88.14%	88.18%	88.56%	88.62%	88.84%	88.48%
with	$\overline{FI}(DSA^+)$	0%	0.11%	0.13%	0.12%	0.10%	0.11%	0.12%	0.11%	0.11%	0.12%
CCU	$\overline{F}(X_{DSA^+})$	**-1167.6**	**-1174.5**	**-1169.2**	**-1174.9**	**-1176.7**	**-1175.2**	**-1175.4**	**-1177.7**	**-1172.8**	**-1178**
DSA^2	$\overline{CI}(DSA^2)$	88.1%	87.9%	88.5%	88.6%	89%	89%	89.4%	89.3%	89.1%	89.3%
	$\overline{FI}(DSA^2)$	0.15%	0.17%	0.16%	0.14%	0.14%	0.16%	0.15%	0.15%	0.16%	0.15%
	$\overline{F}(X_{DSA^2})$	-1169	-1175.8	-1170.3	-1176.2	-1177.9	-1176.2	-1176.5	-1178.6	-1174.2	-1178.8

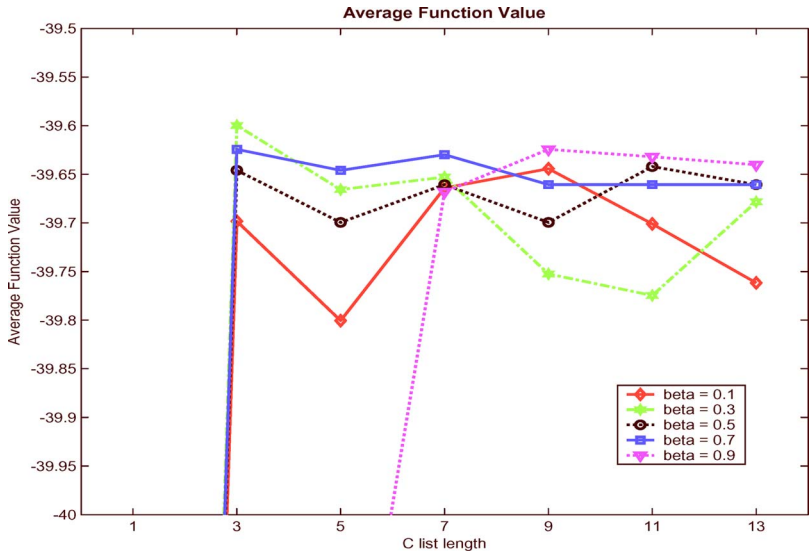


Fig. 3. Effect of the choices of $\underline{C}$ on average function value.

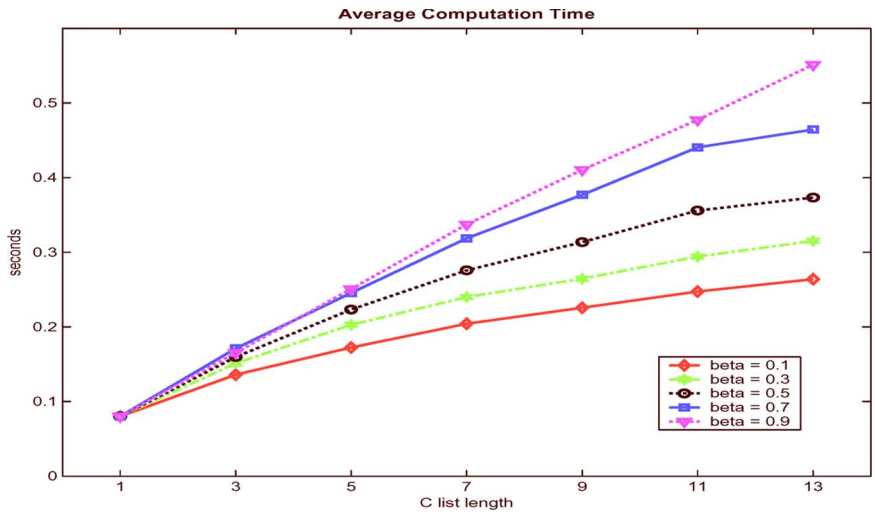


Fig. 4. Effect of the choices of $\underline{C}$ on average computation time.

C. Further Speedup of the DSA (ABR) Algorithm

To expedite the convergence rate of DSA, we experimented with the following stopping criterion instead of the stopping criterion defined in (31):

$$\alpha' = \frac{\left| f_k\left(\underline{x}^{(l+1)}(k)\right) - f_k\left(\underline{x}^{(l)}(k)\right)\right|}{\left| f_k\left(\underline{x}^{(l)}(k)\right)\right|} \leq 0.05. \qquad (32)$$

The computation time for ABR and DSA with this stopping criterion, denoted as ABR^2 and DSA^2, respectively, are listed in Table II. Coarse, but generally acceptable, diagnostic performance can be achieved, e.g., for large-scale models with $(m, n) = (2000, 2500)$, as shown in Table V; we can achieve average $\overline{\text{CI}}(\text{DSA}^2)$ of 88.1% for the ten runs, as compared to average $\overline{\text{CI}}(\text{DSA})$ of 89.2%. Computation time using DSA^2 is much less than that of DSA, e.g., for the same large-scale example; on average, 12.3 s is needed instead of 72.8 s for DSA, as shown in Table II. The reduced inference time may help meet the strict real-time diagnosis requirements. When the DSA algorithm is implemented in a low-level language, e.g., C/C++, dramatic reduction of computation time, by as much as a factor of ten, is expected.

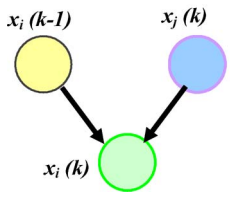
D. Real-World Models

We also apply DSA, with the stopping criterion in (32), on a number of real-world models, whose modeling parameters are listed in Table. VI. The sizes of the models (m, n) varied from (8, 4) to (2080, 1319), where m is the number of fault states and n is the number of tests. These real-world systems are not ideal, i.e., all these systems have fewer number of tests than the number of faults, and some systems cannot have all the fault states covered by tests, e.g., POWERDIST, DOCUMATCH, and LGCU, and spread of test coverage on faults is uneven, e.g., UH60TRANS. The detailed descriptions of these models can be found in [21].

We applied preprocessing where the undetected fault states are screened out of the system before being processed by DSA. Based on the simulation results, as shown in Table VI, one can observe that DSA is an effective algorithm for solving the MAP optimization problem associated with real-time diagnosis as in

TABLE VI
REAL-WORLD MODELS

Model	Acronym	Model Parameters		Performance Metrics	
Cassette Player	CASSET	No. of fault states	8		
		No. of undetected faults	0	$\overline{T}$ (sec)	0.016
		No. of Tests	4	$\overline{CI}$	74.5%
		(Pd, Pf)	(0.9-1, 0)	$\overline{FI}$	3%
		(Pa, Pv)	(0.2, 0.8)		
UH-60 Helicopter Transmission System	UH60TRANS	No. of fault states	160		
		No. of undetected faults	126	$\overline{T}$	0.034
		No. of Tests	51	$\overline{CI}$	68.3%
		(Pd, Pf)	(0.6-1, 0)	$\overline{FI}$	1.9%
		(Pa, Pv)	(0.08, 0.9)		
Power Distribution System	POWERDIST	No. of fault states	106		
		No. of undetected faults	10	$\overline{T}$ (sec)	0.1
		No. of Tests	98	$\overline{CI}$	82%
		(Pd, Pf)	(0.6-1, 0)	$\overline{FI}$	0.68%
		(Pa, Pv)	(0.05, 0.85)		
Document Matching systems	DOCUMATCH	No. of fault states	259		
		No. of undetected faults	2	$\overline{T}$ (sec)	1.11
		No. of Tests	180	$\overline{CI}$	62%
		(Pd, Pf)	(0.6-1, 0)	$\overline{FI}$	1.1%
		(Pa, Pv)	(0.02, 0.95)		
Landing Gear Control Unit	LGCU	No. of fault states	2080		
		No. of undetected faults	0	$\overline{T}$ (sec)	9.9
		No. of Tests	1319	$\overline{CI}$	77%
		(Pd, Pf)	(0.9-1, 0)	$\overline{FI}$	0.36%
		(Pa, Pv)	(0.02, 0.99)		



$x_i(k-1)$	$x_j(k)$	$Prob(x_i(k)=1 \mid x_i(k-1), x_j(k))$
0	0	0.01
0	1	0.3
1	0	0.5
1	1	0.9

Fig. 5. Simple interfault state-dependence illustration.

(3). However, for systems with low inherent testability, MAP optimization-based algorithms, i.e., ABR, DSA, and LRA, do not necessarily achieve satisfactory performance. This points to the need for built-in testability characteristics (optimal sensor placement, fault detection, fault isolation, reduced diagnostic ambiguity) into the system design.

E. Extension to Dependent Faults

A preliminary experiment to extend the DMFD problem formulation to incorporate interfault state dependences is conducted. For this simple interfault state dependence experiment, we assume that, for each fault, there is, at most, one other fault causing the fault, and there is no cyclic relationship among faults. As shown in Fig. 5, fault s_i at time epoch k, i.e., fault state $x_i(k)$ is affected by fault s_j at current epoch, i.e., $x_j(k)$, as well as its status at time $k-1$, i.e., $x_i(k-1)$. It is a special case of coupled hidden Markov model.

The transition probabilities (11) in single-epoch MFD objective function (4) are updated as follows:

$$\text{Prob}\left(\underline{x}(k)|\underline{x}(k-1)\right) = \prod_{i=1}^{m} \text{Prob}\left(x_i(k)|\text{parents}\left(x_i(k)\right)\right). \quad (33)$$

If s_i is affected by one other fault, it has two causal precedents, i.e., $x_i(k-1)$ and $x_j(k)$; therefore, $\text{Prob}\left(x_i(k)|\text{parents}\left(x_i(k)\right)\right) = \text{Prob}\left(x_i(k)|x_i(k-1), x_j(k)\right)$. We apply the same exponential function fitting method on variables $x_i(k)$, $x_i(k-1)$, and $x_j(k)$ as that in (11) and (12) and then apply the DSA algorithm with CCU for its solution. We experimented with this model on small-scale (20×20) and medium-scale (100×150) models; the model parameters and simulation results are listed in Table VII. These models' conditional probabilities are generated by the probabilistic noisy-OR model, i.e., a fault state appears at epoch k if it evolves from its last state or it is affected by another fault state, i.e., $\text{Prob}(x_i(k)=1|x_i(k-1), x_j(k)) = 1 - \text{Prob}(x_i(k)=0|x_i(k-1))\text{Prob}(x_i(k)=0|x_j(k))$. The results also show that DSA with CCU is a promising approach for the dependent-fault cases.

V. SUMMARY

In this paper, we considered the problem of DMFD with unreliable (imperfect) tests. We decomposed the original optimization problem into a series of subproblems corresponding to each epoch. We solved each of the subproblems by a DSA method, which is inspired by its sibling SSA and the ABR heuristic algorithm. To further smooth the "noisy" diagnoses stemming from imperfect test results and increase the accuracy of fault diagnosis, we designed a local-search strategy, denoted as CCU heuristic, which can effectively identify inconsistent diagnoses and improve the diagnostic accuracy. Computation results support that the proposed approach can provide satisfactory solutions to a variety of real-world problems, is scalable, and can meet various real-time requirements.

The limitations of DMFD model are binary test outcome and binary system state assumptions, as well as the general

TABLE VII
PRELIMINARY RESULTS ON THE SIMPLE DEPENDENT-FAULT CASE

Properties		Results	
Number of fault states:	20		
Number of tests:	20	Average $\overline{T}$(sec)	0.031
Number of cause fault states:	5		
Number of affected fault states:	10		
(Pd, Pf):	(0.9-1, 0)	Average $\overline{CI}$	98.17%
(Pa, Pv):	(0.01-0.1, 0-0.6)		
$Pr(x_i(k) = 1 \mid x_j(k-1) = 1)$:	$0.8 - 0.9$		
$Pr(x_i(k) = 0 \mid x_j(k-1) = 0)$:	$0 - 0.2$	Average $\overline{FI}$	4.16%
Number of Simulations:	10		
Number of fault states:	100		
Number of tests:	100	Average $\overline{T}(sec)$	0.291
Number of cause fault states:	10		
Number of affected fault states:	20	Average $\overline{CI}$	96.23%
(Pd, Pf):	(0.9-1, 0)		
(Pa, Pv):	(0.01-0.1, 0-0.8)		
$Pr(x_i(k) = 1 \mid x_j(k-1) = 1)$:	$0.8 - 0.9$	Average $\overline{FI}$	0.34%
$Pr(x_i(k) = 0 \mid x_j(k-1) = 0)$:	$0 - 0.2$		
Number of Simulations:	10		

limitations of factorial hidden Markov models applied to real-world time series [9], [10]. Further research is suggested along two directions. One is to further generalize the DMFD model to allow the fault states to be multivalued (e.g., the level of severity of a fault) and tests to have multiple outcomes (e.g., discrete value reading of sensor outputs). The second direction is to solve general interfault-dependence problem by representing dependences among faults and their evolution over time by a general-coupled HMM.

REFERENCES

[1] E. Aarts and J. K. Lenstra, *Local Search in Combinatorial Optimization*. New York: Wiley, 1997, ser. Wiley-Interscience series in discrete mathematics.

[2] S. V. Amari and H. Pham, "A novel approach for optimal cost-effective design of complex repairable systems," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 37, no. 3, pp. 406–415, May 2007.

[3] D. P. Bertsekas, *Nonlinear Programming*. Belmont, MA: Athena Scientific, 1996.

[4] D. P. Bertsekas, *Dynamic Programming and Optimal Control*. Belmont, MA: Athena Scientific, 2001.

[5] Z. Binglin, Y. Tinghu, and H. Ren, "A genetic algorithm for diagnosis problem solving," in *Proc. Int. Conf. Syst., Man, Cybern.*, 1993, vol. 2, pp. 404–408.

[6] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. New York: Wiley, 2000.

[7] O. Erdinc, C. Raghavendra, and P. Willet, "Real-time diagnosis with sensors of uncertain quality," in *Proc. SPIE Conf.*, Orlando, FL, Apr. 2003, pp. 1490–1499.

[8] G. D. Forney, Jr., "The Viterbi algorithm," *Proc. IEEE*, vol. 61, no. 3, pp. 268–278, Mar. 1973.

[9] Z. Ghahramani, *Learning Dynamic Bayesian Networks*, *Lecture Notes in Computer Science*. Toronto, ON, Canada: Dept. Comput. Sci., Univ. Toronto, 1997.

[10] Z. Ghahramani and M. I. Jordan, "Factorial hidden Markov models," *Mach. Learn.*, vol. 29, no. 2/3, pp. 245–273, Nov. 1997.

[11] M. Krysander and E. Frisk, "Sensor placement for fault diagnosis," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 38, no. 6, pp. 1398–1410, Nov. 2008.

[12] T. Le and C. N. Hadjicostis, "Max-product algorithms for the generalized multiple-fault diagnosis problem," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 6, pp. 1607–1621, Dec. 2007.

[13] G. L. Nemhauser and L. A. Wolsey, *Integer and Combinatorial Optimization*. Hoboken, NJ: Wiley, 1988.

[14] H. M. Paiva, R. K. Galvao, and T. Yoneyama, "A wavelet band-limiting filter approach for fault detection in dynamic systems," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 38, no. 3, pp. 680–687, May 2008.

[15] K. R. Pattipati and M. G. Alexandridis, "Application of heuristic search and information theory to sequential fault diagnosis," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, no. 4, pp. 872–887, Jul./Aug. 1990.

[16] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA: Morgan Kaufmann, 1988.

[17] V. Raghavan, M. Shakeri, and K. Pattipati, "Test sequencing algorithms with unreliable tests," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 29, no. 4, pp. 347–357, Jul. 1999.

[18] S. Ruan, K. R. Pattipati, F. Tu, and A. Patterson-Hine, "On a multimode test sequencing problem," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 3, pp. 1490–1499, Jun. 2004.

[19] M. Shakeri, K. R. Pattipati, V. Raghavan, and A. Patterson-Hine, "Optimal and near-optimal algorithms for multiple fault diagnosis with unreliable tests," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 28, no. 3, pp. 431–440, Aug. 1998.

[20] F. Tu and K. R. Pattipati, "Rollout strategy for sequential fault diagnosis," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 33, no. 1, pp. 86–99, Jan. 2003.

[21] F. Tu, K. R. Pattipati, S. Deb, and V. N. Malepati, "Computationally efficient algorithms for multiple fault diagnosis in large graph-based systems," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 33, no. 1, pp. 73–85, Jan. 2003.

[22] L. A. Wolsey, *Integer Programming*. New York: Wiley, 1998.

[23] J. Ying, T. Kirubarajan, and K. R. Pattipati, "A hidden Markov model-based algorithm for fault diagnosis with partial and imperfect tests," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 30, no. 4, pp. 463–473, Nov. 2000.

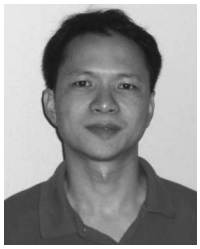
[24] F. Yu, F. Tu, H. Tu, and K. R. Pattipati, "Multiple disease (fault) diagnosis with applications to the QMR-DT problem," in *Proc. Comput. Commun. Control Technol. Int. Conf.*, Austin, TX, 2004, vol. VI, pp. 227–233.

[25] F. Yu, F. Tu, H. Tu, and K. R. Pattipati, "A Lagrangian relaxation algorithm for finding the map configuration in QMR-DT," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 37, no. 5, pp. 746–757, Sep. 2007.

[26] N. L. Zhang and D. Poole, "Exploiting causal independence in Bayesian network inference," *J. Artif. Intell. Res.*, vol. 5, pp. 301–328, 1996.

Sui Ruan received the B.S. and M.S. degrees in electrical engineering from Zhejiang University, Hangzhou, China, in 1996 and 1999, respectively, and the Ph.D. degree from the University of Connecticut, Storrs, in 2007.

From 1999 to 2002, she was with the Intelligent Networking Department, Bell Laboratories, Lucent Technologies, Beijing, China. Since 2007, she has been with the Department of Revenue Management and Planning, American Airlines. Her research interests include organizational design and analysis, and optimization algorithms for fault diagnosis.



Yunkai Zhou received the B.S. and M.S. degrees in computational mathematics from Xi'an Jiaotong University, Xi'an, China, in 1994 and 1997, respectively, and the Ph.D. degree in computational and applied mathematics from Rice University, Houston, TX, in 2002.

Since 2006, he has been an Assistant Professor with the Department of Mathematics, Southern Methodist University, Dallas, TX.

Feili Yu (M'03) received the B.S. degree in optical engineering and the M.S. degree in information and electronic systems from Zhejiang University, Hangzhou, China, in 1994 and 1999, respectively, and the Ph.D. degree from the University of Connecticut, Storrs, in 2007.

He is currently with Independent System Operations (ISO) New England, Springfield, MA. His primary research interests include optimization using evolutionary algorithms, optimal command, and control-architecture modeling and design, fault diagnosis, and electric-power-unit commitment.

Dr. Yu was the recipient of the 2004 Gary F. Wheatley Best Paper Award at the 9th International Command and Control Research and Technology Symposium.

Krishna R. Pattipati (F'95) received the B.Tech. degree (with highest honors) in electrical engineering from the Indian Institute of Technology, Kharagpur, in 1975, and the M.S. and Ph.D. degrees in systems engineering from the University of Connecticut (UCONN), Storrs, in 1977 and 1980, respectively.

From 1980 to 1986, he was with ALPHATECH, Inc., Burlington, MA. Since 1986, he has been with UCONN, where he is a Professor of electrical and computer engineering. He has served as a Consultant to ALPHATECH, Inc., Aptima, Inc., and IBM Research and Development, and is a Cofounder of Qualtech Systems, Inc., a small business specializing in intelligent-diagnostic software tools. He has published over 350 articles, primarily in the application of systems theory and optimization (continuous and discrete) techniques to large-scale systems. His current research interests are in the areas of adaptive organizations for dynamic and uncertain environments, signal processing and diagnosis techniques for complex system monitoring, and multiobject tracking.

Dr. Pattipati has served as the Editor-in-Chief of the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B from 1998 to 2001, Vice President for Technical Activities of the IEEE Systems, Man, and Cybernetics (SMC) Society from 1998 to 1999, and Vice President for Conferences and Meetings of the IEEE SMC Society from 2000 to 2001. He was selected by the IEEE SMC Society as the Outstanding Young Engineer of 1984 and was the recipient of the Centennial Key to the Future Award. He was the corecipient of the Andrew P. Sage Award for the Best SMC Transactions Paper for 1999, the Barry Carlton Award for the Best AES Transactions Paper for 2000, the 2002 and 2008 NASA Space Act Awards for "A Comprehensive Toolset for Model-based Health Monitoring and Diagnosis," the 2003 AAUP Research Excellence Award, and the 2005 School of Engineering Teaching Excellence Award at the UCONN. He was also the recipient of the Best Technical Paper Awards at the 1985, 1990, 1994, 2002, 2004, and 2005 IEEE AUTOTEST Conferences and at the 1997 and 2004 Command and Control Conferences. He was elected a Fellow of the IEEE in 1995 for his contributions to discrete-optimization algorithms for large-scale systems and team decision making.

Peter Willett (F'03) received the B.A.Sc. degree in engineering science from the University of Toronto, Toronto, ON, Canada, in 1982 and the Ph.D. degree from Princeton University, Princeton, NJ, in 1986.

He has been a Faculty Member with the University of Connecticut, Storrs, where he has been a Professor in the Department of Electrical and Computer Engineering since 1998. His primary areas of research are in statistical signal processing, detection, machine learning, data fusion, and tracking. He also has interests in and has published in the areas of change/abnormality detection, optical pattern recognition, communications, and industrial/security condition monitoring.

Dr. Willett is an Editor-in-Chief for the IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS, and until recently, was an Associate Editor for three active journals: IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS (for data fusion and target tracking) and IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PARTS A and B. He is also an Associate Editor for the IEEE AEROSPACE AND ELECTRONIC SYSTEMS MAGAZINE, an Associate Editor for the International Society for Information Fusion (ISIF) *Journal of Advances in Information Fusion*, is a member of the editorial board of the IEEE SIGNAL PROCESSING MAGAZINE and was First Editor of the IEEE AES MAGAZINE's periodic Tutorial Issues. He has been a member of the IEEE AESS Board of Governors since 2003. He was the General Cochair (with Stefano Coraluppi) for the 2006 ISIF/IEEE Fusion Conference in Florence, Italy, and (with Wolfgang Koch) for the 2008 ISIF/IEEE Fusion Conference in Cologne, Germany; the Program Cochair (with Eugene Santos) for the 2003 IEEE Conference on Systems, Man, and Cybernetics in Washington DC; and the Program Cochair (with Pramod Varshney) for the 1999 Fusion Conference in Sunnyvale.

Ann Patterson-Hine received the Ph.D. degree in mechanical engineering from the University of Texas, Austin, in 1988.

Since July 1988, she has been with the NASA Ames Research Center, Moffett Field, CA, where she is a Group Leader of the Research in Intelligent Vehicle Automation Group in the Computational Sciences Division. She has been a Project Leader for advanced-technology demonstrations under the Next Generation Launch Technology Program and many of the program's predecessors, including Reusable Launch Vehicle and Space Launch Initiative Programs. She has participated on the Shuttle Independent Assessment Team and Wire Integrity Pilot Study at Ames. Her research has centered on the use of engineering models for model-based reasoning in advanced monitoring and diagnostic systems.