To appear in the Sixteenth International Workshop on Principles of Diagnosis, 2005

Diagnosis as Approximate Belief State Enumeration for Probabilistic Concurrent Constraint Automata[*]

Oliver B. Martin and **Brian C. Williams**
{omartin, williams}@mit.edu
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA, 02139

Michel D. Ingham
michel.ingham@jpl.nasa.gov
Jet Propulsion Laboratory
Pasadena, CA 91109

Abstract

As autonomous spacecraft and other robotic systems grow increasingly complex, there is a pressing need for capabilities that more accurately monitor and diagnose system state while maintaining reactivity. Mode estimation addresses this problem by reasoning over declarative models of the physical plant, represented as a factored variant of Hidden Markov Models (HMMs), called *Probabilistic Concurrent Constraint Automata (PCCA)*. Previous mode estimation approaches track a set of most likely PCCA state trajectories, enumerating them in order of trajectory probability. Although *Best-First Trajectory Enumeration (BFTE)* is efficient, ignoring the additional trajectories that lead to the same state can considerably underestimate the true state probability and result in misdiagnosis. This paper introduces an innovative belief approximation technique, called *Best-First Belief State Enumeration (BFBSE)*, that addresses this limitation by computing estimate probabilities *directly* from the HMM belief state update equations. Theoretical and preliminary empirical results show that BFBSE considerably increases estimator accuracy, requires less memory, and has no increase in computation time when enumerating a moderate number of estimates for the approximate belief state of subsystem sized models.

1 Introduction

The purpose of estimation is to determine the current state of the system. An estimator infers the current state by reasoning over a model of the system dynamics along with the commands that have been executed and the resulting sensory observations. In many embedded systems, this knowledge of the current state is then used by a controller to drive the system state towards a specific target or goal. The ability for a system to accurately and reliably deduce its current state can dictate whether it is able to achieve its objectives. This is particularly important for highly complex robotic space exploration systems that operate in uncertain environments. Furthermore, deep space communication delays and severely constrained onboard computing capabilities present tremendous challenges to traditional methods of estimation in support of robust autonomous spacecraft operations.

Previous work in model-based monitoring and fault diagnosis, including GDE/Sherlock [de Kleer and Williams, 1987; 1989], GDE+ [Struss and Dressler, 1989], Livingstone [Williams and Nayak, 1996; Kurien and Nayak, 2000], and Titan Mode Estimation [Williams *et al.*, 2003], have made significant advances towards meeting these challenging performance requirements. All of these capabilities achieve reactivity, while maintaining reliability, by framing mode estimation as a best-first shortest-path problem, which can be efficiently solved using a variant of the Viterbi algorithm [Forney, 1978]. This approach is known as *Best-First Trajectory Enumeration (BFTE)* and works quite well when trying to determine the "most likely explanation" to a sequence of observations. Livingstone was successfully flight validated on the NASA Deep Space One probe as part of the Remote Agent Experiment in 1999 [Williams and Nayak, 1996]. Unfortunately, approximating the current state by the most likely trajectory can significantly underestimate the true state probability and result in misdiagnosis.

This paper introduces a novel mode estimation technique called *Best-First Belief State Enumeration (BFBSE)* that approximates the belief state by generating the set of most likely estimates and achieves greater accuracy by computing the estimate probabilities *directly* from the Hidden Markov Model (HMM) Belief State Update equations instead of approximating them by their trajectory probability. This contribution significantly increases the accuracy of the estimator while using less memory and, under certain conditions, less computational time; providing an enabling technology for increasingly complex space missions of the future.

We begin by introducing our modeling formalism and the complete HMM estimation problem, noting some practical limitations. We address these limitations by presenting BFTE and BFBSE as instances of an optimal constraint satisfaction problem. In conclusion, we support our claims of improved estimator accuracy and performance through theoretical and preliminary empirical comparisons between the approaches.

2 PCCA Plant Model

As in previous work, we model the physical plant as a factored Hidden Markov Model that is compactly encoded as *Probabilistic Concurrent Constraint Automata (PCCA)*

[*]This research was funded in part by the Jet Propulsion Laboratory, through the JPL Director's Research and Development Fund.

[Williams *et al.*, 2003]. The PCCA represent a set of concurrently operating components that are interconnected and interact with their surrounding environment. Each automaton has a set of possible discrete modes with conditional probabilistic transitions, which capture both nominal and faulty behavior. These modes are only partially observable, due to a limited number of sensors, but are inherently constrained by the system properties that define each mode. In this section we review the formal definition of the PCCA plant model and provide an illustrative example.

2.1 PCCA Formalism

Consider the probabilistic constraint automaton for component "a" defined by the tuple $\mathcal{A}_a = \langle \Pi_a, \mathbb{M}_a, \mathbb{T}_a, \mathbf{P}_{\mathbb{T}_a}, \mathbf{P}_{\Theta_a} \rangle$:

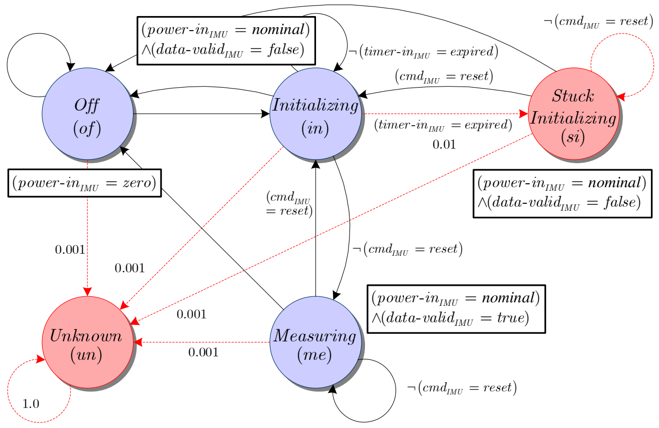
1. $\Pi_a = \Pi_a^m \cup \Pi_a^r$ is a finite set of discrete variables for component "a", where each variable $\pi_a \in \Pi_a$ ranges over a finite domain $\mathbb{D}(\pi_a)$. Π_a^m is a singleton set containing *mode* variable $\{x_a\} = \Pi_a^m$ whose domain $\mathbb{D}(x_a)$ is the finite set of discrete modes in $\mathcal{A}_a$. *Attribute* variables Π_a^r include inputs, outputs, and any other variables used to define the behavior of the component. Σ_a is the complete set of all possible full assignments over Π_a and the state space of the component $\Sigma_a^{x_a} = \Sigma_a \Downarrow_{x_a}$ is the projection of Σ_a onto mode variable x_a.
2. $\mathbb{M}_a : \Sigma_a^{x_a} \rightarrow \mathbb{C}(\Pi_a^r)$ maps each mode assignment $(x_a = v_a) \in \Sigma_a^{x_a}$ to a finite domain constraint $c_a(x_a = v_a) \in \mathbb{C}(\Pi_a^r)$, where $\mathbb{C}(\Pi_a^r)$ is the set of finite domain constraints over Π_a^r. These constraints are known as *modal constraints* and are typically encoded in the propositional form $\lambda \triangleq \textbf{True} \mid \textbf{False} \mid (u = y) \mid \neg\lambda_1 \mid \lambda_1 \wedge \lambda_2 \mid \lambda_1 \vee \lambda_2$, where $y \in \mathbb{D}(u)$. If the current mode is $(x_a^t = v_a)$ at time-step t, then the assignments to each attribute variable $r_a^t \in \Pi_a^r$ at time-step t must be consistent with $c_a(x_a = v_a)$. These constraints capture the physical behavior of the mode.
3. $\mathbb{T}_a : \Sigma_a^{x_a} \times \mathbb{C}(\Pi_a^r) \rightarrow \Sigma_a^{x_a}$ is a set of transition functions. The set of finite domain constraints $\mathbb{C}(\Pi_a^r)$ are also known as the *transition guards*, encoded in the propositional form λ. Given a current mode assignment $(x_a = v_a) \in \Sigma_a^{x_a}$ and guard $g_a \in \mathbb{C}(\Pi_a^r)$ entailed at time-step t, each transition function $\tau_a(x_a = v_a, g_a) \in \mathbb{T}_a(x_a = v_a, g_a)$ specifies a target mode assignment $(x_a = v_a') \in \Sigma_a^{x_a}$ that the automaton could transition into at time-step $t+1$. $\mathbb{T}_a = \mathbb{T}_a^n \cup \mathbb{T}_a^f$ captures both nominal and faulty behavior.
4. $\mathbf{P}_{\mathbb{T}_a} : \mathbb{T}_a(x_a = v_a, g_a) \rightarrow \Re[0,1]$ is a transition probability distribution. For each mode variable assignment in $\Sigma_a^{x_a}$ and guard g_a^t, there is a probability distribution across all transitions into target modes defined by the set of transition functions $\mathbb{T}_a(x_a = v_a, g_a)$.

The entire system plant $\mathcal{P}$ is modeled by a composition of concurrently operating constraint automata. Each automaton is interconnected to both its environment and other automata through constraints on shared variables. Formally, the PCCA plant model is defined by the tuple $\mathcal{P} = \langle \mathcal{A}, \Pi, \mathbb{Q} \rangle$:

1. $\mathcal{A} = \{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n\}$ is the finite set of constraint automata that represent the n components of the plant.
2. $\Pi = \bigcup_{a=1..n} \Pi_a$ is the set of all plant variables. The variables Π are partitioned into a finite set of *mode* variables $\Pi^m = \bigcup_{a=1..n} \Pi_a^m$, *control* variables $\Pi^c \subseteq \bigcup_{a=1..n} \Pi_a^r$, *observation* variables $\Pi^o \subseteq \bigcup_{a=1..n} \Pi_a^r$, and *dependent* variables $\Pi^d \subseteq \bigcup_{a=1..n} \Pi_a^r$. Σ^c, Σ^o, and Σ^d are the sets of full assignments over Π^c, Π^o, and Π^d.
3. $\mathbb{Q} \subset \mathbb{C}(\Pi)$ is a set of finite domain constraints that capture the interconnections between plant components.

2.2 Example

An Inertial Measurement Unit (IMU) is a standard sensor package that is used to provide spacecraft and other robotic systems with translational and angular motion measurements. When power is supplied to this IMU by closing its associated Power Switch (PS), it enters *Initializing* mode and eventually transitions to *Measuring* mode, in which it produces valid data. The IMU has a fault mode where it becomes *Stuck Initializing*; recovery from this fault requires sending a reset command to the IMU. This model also defines a highly unlikely *Unknown* mode, which captures all other unexpected behavior. Our PCCA plant model $\mathcal{P}$ is composed of the constraint automata for the IMU ($\mathcal{A}_{imu}$) and PS ($\mathcal{A}_{ps}$). Graphical representations of each are shown in Figures 1 and 2, respectively, with a formal description of $\mathcal{A}_{imu}$ provided below:



$(power\text{-}in_{IMU} = nominal)$ $\wedge(data\text{-}valid_{IMU} = false)$
$\neg(timer\text{-}in_{IMU} = expired)$
$\neg(cmd_{IMU} = reset)$
$(cmd_{IMU} = reset)$
Off (of)
Initializing (in)
Stuck Initializing (si)
$(timer\text{-}in_{IMU} = expired)$
0.01
$(power\text{-}in_{IMU} = zero)$
$(cmd_{IMU} = reset)$
$(power\text{-}in_{IMU} = nominal)$ $\wedge(data\text{-}valid_{IMU} = false)$
0.001
0.001
$\neg(cmd_{IMU} = reset)$
0.001
$(power\text{-}in_{IMU} = nominal)$ $\wedge(data\text{-}valid_{IMU} = true)$
Unknown (un)
Measuring (me)
0.001
1.0
$\neg(cmd_{IMU} = reset)$

Figure 1: IMU constraint automaton, $\mathcal{A}_{imu}$.

1. $\Pi_{imu} = \{x_{imu}, \mu_{imu}^{cmd}, o_{imu}^{dv}, d_{imu}^{pi}, d_{imu}^{ti}\}$ where $\{x_{imu}\} = \Pi_{imu}^m$ resides in 1 of 5 discrete modes $\mathbb{D}(x_{imu}) = \{$*of, in, me, si, un*$\}$. $\Pi_{imu}^r = \{\mu_{imu}^{cmd}, o_{imu}^{dv}, d_{imu}^{pi}, d_{imu}^{pi}\}$ where μ_{imu}^{cmd} is used to reset the IMU, o_{imu}^{dv} is an observation of the data validity, d_{imu}^{pi} is the power-in, and d_{imu}^{ti} is a time expiration variable. $\Sigma_{imu} = \Sigma_{imu}^m \times \Sigma_{imu}^r$ is the set of all full assignments over Π_{imu} with $5 \cdot (2 \cdot 2 \cdot 2 \cdot 2) = 80$ elements.
2. $\mathbb{M}_{imu}$ includes the constraints encapsulated by rectangles in Figure 1. For example, $\mathbb{M}_{imu}(x_{imu} = me) = (d_{imu}^{pi} = nominal \wedge o_{imu}^{dv} = true)$.
3. Transitions are indicated by the arrows and labels in Figure 1. $\mathbb{T}_{imu}(x_{imu} = si, \mu_{imu}^{cmd} = reset)$ is a set of transition functions $\{\tau^{n_1}, \tau^{n_2}, \tau^{f_1}\}$ where $(x_{imu} = in)$, $(x_{imu} = of)$, and $(x_{imu} = un)$ are the target modes.
4. For the set of transitions described above in item 3, the probability distribution across target modes $\{$*in, of, un*$\}$ at time-step $t{+}1$ is $\mathbf{P}_{\mathbb{T}_{imu}}(x_{imu} = si, \mu_{imu}^{cmd} = reset) = \{0.4995, 0.4995, 0.001\}$.

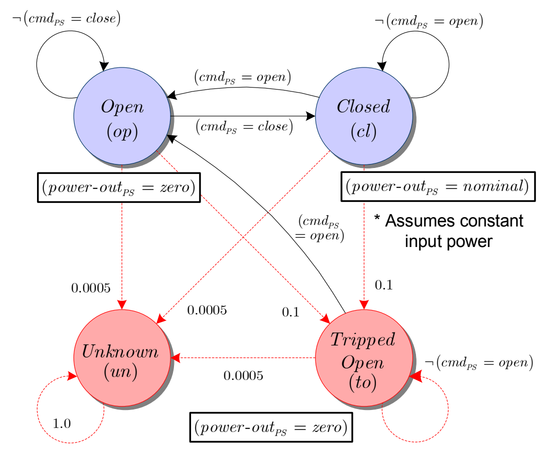


Figure 2: Power Switch constraint automaton, $\mathcal{A}_{ps}$.

The PCCA plant model for this simple IMU/PS example is composed of 2 components and the interconnections between them. The PCCA plant $\mathcal{P}$ is defined as follows:

1. $\mathcal{A} = \{\mathcal{A}_{imu}, \mathcal{A}_{ps}\}$.
2. $\Pi = \Pi_{imu} \cup \Pi_{ps}$ is the set of all variables.
3. $\mathbb{Q} = \{d^{po}_{ps} = d^{pi}_{imu}\}$ contains only one interconnection constraint. For this example, the $power\text{-}out$ (po) of the PS is connected to the $power\text{-}in$ (pi) of the IMU.

3 Estimation of PCCA

This section reviews exact belief state update for PCCAs, while noting practical limitations due to state space explosion and assumptions that make online estimation tractable. The task of estimation is to calculate a belief state of the system in real-time while maintaining accuracy and reliability. A belief state is a probability distribution over the states of a system, which represents the likelihood of the system being in any single state given a history of past commands and observations. For PCCA, a state s_i is defined as a full assignment to mode variables $s_i \in \Sigma^m$ and a belief state $B = \langle S, p \rangle$ is a finite set of estimates that cover all consistent states $S \subseteq \Sigma^m$. Each estimate consists of a state $s_i \in S$ and its posterior probability $p(s_i) \in p$.

3.1 Belief State Update

The Markov property states that the future of a system is conditionally independent of its past, given its current belief state. This property allows an estimator to iteratively compute the next complete belief state B^{t+1} at time-step $t+1$ by only considering the current belief state B^t and commands μ^t at time-step t, along with the resulting observations o^{t+1}. The belief state is then computed using the standard HMM belief state update equations [Baum and Petrie, 1966]:

$$\begin{aligned}&\mathbf{P}(s_j^{t+1}|o^{<0,t>}, \mu^{<0,t>}) = \\ &\quad \textstyle\sum_{s_i^t \in S^t} (\mathbf{P}(s_j^{t+1}|s_i^t, \mu^t)\mathbf{P}(s_i^t|o^{<0,t>}, \mu^{<0,t-1>}))\end{aligned} \tag{1}$$

$$\begin{aligned}&\mathbf{P}(s_j^{t+1}|o^{<0,t+1>}, \mu^{<0,t>}) = \\ &\quad \frac{\mathbf{P}(s_j^{t+1}|o^{<0,t>}, \mu^{<0,t>}) \cdot \mathbf{P}(o^{t+1}|s_j^{t+1})}{\sum_{s_i^{t+1} \in S^{t+1}} \mathbf{P}(s_i^{t+1}|o^{<0,t>}, \mu^{<0,t>})\mathbf{P}(o^{t+1}|s_i^{t+1})}\end{aligned} \tag{2}$$

Equation 1 represents the *a priori* probability of being in the next state s_j^{t+1} at time-step $t+1$, given all the observations $o^{<0,t>}$ and commands $\mu^{<0,t>}$ between time-step 0 and t. $\mathbf{P}(s_i^t|o^{<0,t>}, \mu^{<0,t-1>}) \in p^t$ is the probability that the system was in state s_i at time-step t and $\mathbf{P}(s_j^{t+1}|s_i^t, \mu^t)$ is the state transition probability. Equation 1 propagates the system dynamics into the future before considering new observations. Once all the *a priori* estimates are generated, Equation 2 then updates these estimates by adjusting the probabilities based on new observations o^{t+1} using the Total Probability Theorem and Bayes' Rule to calculate the *a posteriori* probabilities p^{t+1} across all states in S^{t+1}. Detailed definitions of state transition and observation probabilities for PCCA can be found in [Williams *et al.*, 2003].

3.2 Approximation to Exponential Belief State

For systems modeled as PCCA, there is a finite number of estimates in the belief state, though the size of the belief state is exponential in the number of components. The precise size of the belief state for n components is $\prod_{a=1..n} |\mathbb{D}(x_a)|$, where $|\mathbb{D}(x_a)|$ is the number of modes in $\mathcal{A}_a$. For a full-scale spacecraft propulsion subsystem, the size of the belief state is roughly 3.5^{80} (80 mode variables with an average domain size of 3.5) [Williams and Nayak, 1996]. To mitigate this belief state space explosion, previous work in Livingstone and Titan have made the assumption that the true state of the system is captured within only a few of the most likely estimates. This assumption is based on the key insight that, although the full belief state has an exponential size, the bulk of the probability density is concentrated in only a handful of most likely estimates. This is due to the drastically decreasing likelihood of simultaneous multiple point failures.

By leveraging this approximation, the estimation problem is simplified from updating the full belief state B to enumerating the k best estimates of an approximate belief state $\tilde{B}$. To avoid extraneous computation, preserve reactivity of the estimation process, and enable it to be employed for the purposes of real-time control, this enumeration is performed in best-first order. This is done efficiently by framing estimation as an optimal constraint satisfaction problem (OCSP).

3.3 Optimal Constraint Satisfaction Problem

Estimation can be viewed as a problem of constraint optimization where each possible target state in the approximate belief state must be consistent with modal constraints $\mathbb{M}$, component interconnections $\mathbb{Q}$, and observations o^{t+1}.

Definition 1 *An OCSP $\langle \boldsymbol{x}, f, C \rangle$ is a problem of the form "arg max $f(\boldsymbol{x})$ subject to $C(\boldsymbol{x})$," where $\boldsymbol{x}$ is a vector of decision variables, $C(\boldsymbol{x})$ is a set of state constraints, and $f(\boldsymbol{x})$ is a multi-attribute utility function.*

Solving an OCSP consists of generating a prefix to the sequence of feasible solutions, ordered by decreasing value of f. A feasible solution assigns to each variable in $\mathbf{x}$ a value from its domain such that $C(\mathbf{x})$ is satisfied.

OPSAT is an OCSP solver that was implemented using *Conflict-directed A** [Ragno, 2002] to efficiently find solutions to the OCSP in best-first order by interleaving candidate generation and test. OPSAT generates each leading candidate $\mathbf{x}$ using A* search and then tests the candidate for consistency against $C(\mathbf{x})$. If it proves inconsistent, OPSAT summarizes the inconsistency (called a *conflict*) and uses this summary to jump over other leading candidates that are similarly inconsistent.

4 Estimation as an OCSP

By solving the estimation problem as an OCSP using OPSAT, each estimate in an approximate belief state can be enumerated in best-first order. Both the BFTE and BFBSE algorithms take advantage of OPSAT to reduce the exponential number of possible states to the k most likely. The decision variables $\mathbf{x}$ are the set of mode variables Π^m and the state constraints $C(\mathbf{x})$ restrict mode variable assignments $(x_a = v'_a)$ to those that are consistent with observations o^{t+1}, modal constraints $\mathbb{M}_a(x_a = v'_a)$, and component interconnections $\mathbb{Q}$. The primary difference between the two algorithms resides in the utility function $f(\mathbf{x})$ specification.

4.1 *Best-First Trajectory Enumeration*

BFTE has a utility function that is equivalent to that of the Viterbi algorithm [Forney, 1978]. This approach essentially unfolds the belief state transitions into a branching tree structure (Figure 3) and estimates are enumerated by finding the single-step shortest-path in the tree.

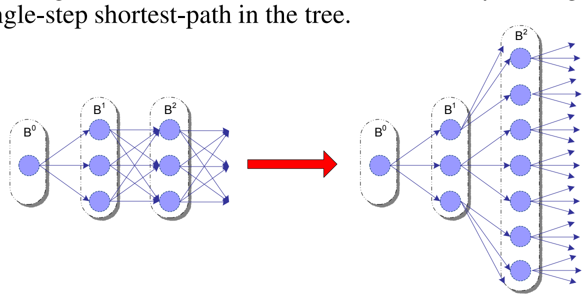


Figure 3: Evolution of the belief state, represented as a *trellis diagram* (left), can be decomposed into a *branching tree*.

Each arrow on the right side of Figure 3 still represents a state transition but the belief state encoding is no longer compact since there can be duplicate states at each time-step in the branching tree. Furthermore, the probability associated with each estimate, as per the HMM belief state update equations, has been split across all the duplicate nodes within the same time-step such that the probability tied to each individual node in the tree is actually a lower bound on the true state probability. This tree representation invalidates the BFTE claim of true best-first state enumeration.

Given the plant model $\mathcal{P}$, the current approximate belief state $\tilde{B}^t$, commands μ^t, and resulting observations o^{t+1}, BFTE generates the estimates in the next belief state $\tilde{B}^{t+1}$ in best-first order according to the state trajectory probability $\mathbf{P}(s_j^{t+1} \mid s_i^t, \mu^t) \cdot p^t(s_i)$. It is important to note that the update step of the HMM belief state update equations is implicit in both BFTE and BSBSE algorithms[1]. Pseudo code for the BFTE algorithm is shown in Figure 4.

[1]Since each solution to an OCSP must satisfy constraints $C(\mathbf{x})$, the update step is implicitly computed such that $\mathbf{P}(o^{t+1}|s_j^{t+1})$ is 1 when observations are consistent with $C(\mathbf{x})$ and 0 otherwise. This is not a proper probability distribution but it avoids state space complexities and requires no additional computation.

```
BFTEalg $(\mathcal{P}, \tilde{B}^t, \mu^t, o^{t+1}) \to \tilde{B}^{t+1}$ ::
  1. For each $s_i^t \in \tilde{S}^t$, setup OCSP$_\mathbf{i}$ $\langle \mathbf{x}, f, C \rangle$:
       • The vector $\mathbf{x}$ includes a decision variable $\mathbf{x}_a$ for each component of the
         plant, whose domain $\mathbb{D}(\mathbf{x}_a)$ is the set of reachable target modes. The
         target mode for each transition $(x_a = v'_a) = \tau_a(x_a = v_a, g_a)$ is
         reachable when the source $(x_a = v_a) \in s_i^t$ and guard $g_a$ are satisfied
         by $C_M^t \wedge s_i^t \wedge \mu^t$, where $C_M^t = \mathbb{Q} \wedge \bigwedge_{(x_a=v_a)\in s_i^t} \mathbb{M}_a(x_a = v_a)$.
       • The utility function $f(\mathbf{x})$ is the prior trajectory probability of next
         state $\mathbf{x}$, that is, $f(\mathbf{x}) = \mathbf{P}(\mathbf{x} \mid s_i^t, \mu^t) \cdot p^t(s_i)$, where $\mathbf{P}(\mathbf{x} \mid$
         $s_i^t, \mu^t) = \prod_{(x_a=v'_a)\in\mathbf{x}} \mathbf{P}(x_a = v'_a \mid x_a = v_a, s_i^t, \mu^t)$ and
         $p^t(s_i)$ is the posterior probability for state $s_i^t$.
       • $C(\mathbf{x})$ encodes the constraint that $\mathbf{x} \wedge C_{M_\mathbf{x}} \wedge o^{t+1}$ must be consistent,
         where $C_{M_\mathbf{x}} = \mathbb{Q} \wedge \bigwedge_{(x_a=v'_a)\in\mathbf{x}} \mathbb{M}_a(x_a = v'_a)$.
  2. Compute the $k$ most likely solutions $\tilde{S}^{t+1} = \{\mathbf{x^1}, \ldots, \mathbf{x^k}\}$ to
     $\bigwedge_{\mathbf{i}=1..k}$OCSP$_\mathbf{i}\langle \mathbf{x}, f, C \rangle$ by incrementally comparing the next-best solution
     from each of the $k$ instances of OPSAT.
  3. Extract the posterior probabilities $p^{t+1} = \{f(\mathbf{x^1}), \ldots, f(\mathbf{x^k})\}$ for each
     solution $\mathbf{x^i}$ in $\tilde{S}^{t+1}$.
  4. Return the $k$ most likely trajectories contained by $\tilde{B}^{t+1} = \langle \tilde{S}^{t+1}, p^{t+1} \rangle$.
```

Figure 4: *Best-First Trajectory Enumeration* algorithm.

As an example of BFTE, consider the simple scenario presented in Figure 5. Assuming that the IMU and PS are in one of the two depicted states at time t, and there are no commands issued nor observations received, the leading four estimates are displayed for time-step $t+1$ according to the HMM belief state update equations. The numbers on the arrows represent the individual component transition probabilities. For pedagogical clarity, the true *a priori* state probabilities are shown on the right for time-step $t+1$ and should be updated with the observation probabilities to complete the belief state update. Modes that are mutually inconsistent, such as $(x_{imu} = in \wedge x_{ps} = op)$, are determined to be conflicting by OPSAT and are not returned.

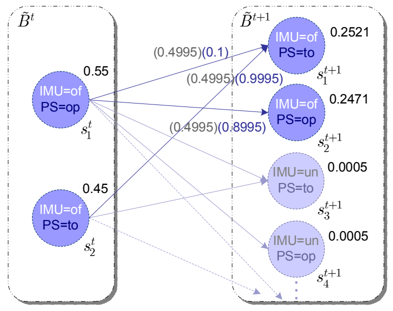


Figure 5: Simple IMU/PS belief state update example.

Table 1: *Best-First Trajectory Enumeration* results

solution #	source state	target state	probability	% error
1	s_1^t	s_2^{t+1}	0.2471	0.0
2	s_2^t	s_1^{t+1}	0.2247	10.9
3	s_1^t	s_1^{t+1}	0.0275	89.1
4	s_1^t	s_4^{t+1}	0.000495	0.0

Table 1 shows the 4 most likely trajectories returned by BFTE for the scenario in Figure 5, with their source state, target state, *a priori* state trajectory probability, and percent error when compared to the true *a priori* probabilities. It is important to notice that since BFTE splits the two trajectories leading to s_1^{t+1}, this state is no longer evaluated as the most likely estimate. Incorrectly estimating the likelihood of states in this fashion can have a major affect on the action a control system will take to achieve a goal.

4.2 *Best-First Belief State Enumeration*

Using BFBSE to generate estimates for the IMU/PS scenario would entirely remove the problem exhibited in Table 1. Instead of approximating the belief state by searching over most likely trajectories, BFBSE uses the HMM belief state propagation equation *directly* as its utility function. Although this technique still approximates the belief state by enumerating only the k best estimates, the estimate representation is more compact and little accuracy is lost. Pseudo code for BFBSE is presented in Figure 6.

```
BFBSEalg $(\mathcal{P}, \tilde{B}^t, \mu^t, o^{t+1}) \rightarrow \tilde{B}^{t+1}$ ::
  1. Setup the OCSP $\langle \mathbf{x}, f, C \rangle$:
      • The vector $\mathbf{x}$ includes a decision variable $\mathbf{x}_a$ for each component of the plant, whose domain $\mathbb{D}(\mathbf{x}_a)$ is the set of modes that are reachable from any current state $s_i^t \in \tilde{S}^t$. For all $s_i^t \in \tilde{S}^t$, the target mode for each transition $(x_a = v'_a) = \tau_a(x_a = v_a, g_a)$ whose source $(x_a = v_a) \in s_i^t$ and guard $g_a$ are satisfied by $C_M^t \wedge s_i^t \wedge \mu^t$ is considered reachable such that $v'_a \in \mathbb{D}(\mathbf{x}_a)$. $C_M^t = \mathbb{Q} \wedge \bigwedge_{(x_a = v_a) \in s_i^t} \mathbb{M}_a(x_a = v_a)$.
      • The utility function $f(\mathbf{x})$ is the prior probability of next state $\mathbf{x}$, that is, $f(\mathbf{x}) = \sum_{s_i^t \in \tilde{S}^t} \mathbf{P}(\mathbf{x} \mid s_i^t, \mu^t) \cdot p^t(s_i)$, where $\mathbf{P}(\mathbf{x} \mid s_i^t, \mu^t) = \prod_{(x_a = v'_a) \in \mathbf{x}} \mathbf{P}(x_a = v'_a \mid x_a = v_a, s_i^t, \mu^t)$ and $p^t(s_i)$ is the posterior probability for state $s_i^t$.
      • $C(\mathbf{x})$ encodes the constraint that $\mathbf{x} \wedge C_{M_\mathbf{x}} \wedge o^{t+1}$ must be consistent, where $C_{M_\mathbf{x}} = \mathbb{Q} \wedge \bigwedge_{(x_a = v'_a) \in \mathbf{x}} \mathbb{M}_a(x_a = v'_a)$.
  2. Compute the $k$ most likely solutions $\tilde{S}^{t+1} = \{\mathbf{x^1}, \ldots, \mathbf{x^k}\}$ to OCSP$\langle \mathbf{x}, f, C \rangle$ using OPSAT.
  3. Extract the posterior probabilities $p^{t+1} = \{f(\mathbf{x^1}), \ldots, f(\mathbf{x^k})\}$ for each solution $\mathbf{x^i}$ in $\tilde{S}^{t+1}$.
  4. Return the $k$ most likely state estimates contained by $\tilde{B}^{t+1} = \langle \tilde{S}^{t+1}, p^{t+1} \rangle$.
```

Figure 6: *Best-First Belief State Enumeration* algorithm.

When applying this algorithm to the IMU/PS scenario, the increase in estimate probability accuracy is readily apparent. Since the HMM belief state propagation equation is directly used as the objective function, and there is complete knowledge of the initial belief state in this scenario, the estimate probabilities match the true *a priori* belief state probabilities shown in Figure 5. Our algorithm is innovative in its ability to achieve higher accuracy while using less space and roughly the same amount of time as discussed in the next section.

4.3 BFTE and BFBSE Complexities

Both estimation techniques have been presented as instances of an OCSP that can be solved efficiently using the Conflict-directed A* algorithm employed in OPSAT. Since the OCSP constraints are identical in BFTE and BFBSE, both techniques will identify the same conflicts for a given candidate solution. This allows us to compare the algorithms by evaluating their space and time complexities based on the A* candidate search portion of OPSAT.

There are two fundamental reasons why the performance of these estimate enumeration techniques differ: BFBSE generates estimates using only one instance of OPSAT, instead of k instances as performed in BFTE; and each node generated by BFBSE requires k times more arithmetic computations than BFTE since we are summing over k incoming state transitions. To more clearly understand the complexity analysis, recall that the best case time and space for A* is roughly $n \cdot b$ and the worst case time and space is b^n, where b is the branching factor and n is the depth of the tree. For BFTE and BFBSE, b is the average number of reachable modes per component $|\mathbb{D}(x_a)|$ and n is the number of components in the model $|\Pi^m|$. Table 2 shows the complexities for BFTE and BFBSE as an augmented form of A* search.

Table 2: Space and time complexity for BFTE and BFBSE

	Best Case		Worst Case	
	Space	Time	Space	Time
BFTE	$k \cdot n$	$k \cdot n \cdot (n + C)$	$k \cdot b^n$	$k \cdot b^n \cdot (n + C)$
BFBSE	$n \cdot b$	$n \cdot b \cdot (n \cdot k + C)$	b^n	$b^n \cdot (n \cdot k + C)$

Notice that this time complexity considers the time it takes to create each node in addition to the number of nodes visited. This quantity (enclosed by parentheses) consists of the time to evaluate the objective function plus a constant C for other data manipulating operations. We also note that the complexities for BFTE are multiplied by k because of the k instances of OPSAT; however, BFTE avoids expanding many of the search tree's fringe nodes by exploiting its *mutually preferential independent* [Ragno, 2002] objective function, such that $b = 1$ in the best case.

This analysis shows that whenever $k > b$, the space required by BFBSE is always less than BFTE. Conversely, for large values of n, BFTE is faster in the best case by a factor of b but both are equally fast in the worst case. In the following section, we will see that, for practical problems, b is small and C dominates the objective function term unless the model is very large. For subsystem or modest size system models, BFBSE is more accurate, uses less memory, and requires less computation time.

5 Experimental Results

The following empirical comparison between BFTE and BFBSE (implemented in C++) was conducted using a Mars Entry Decent and Landing (EDL) model [Ingham, 2003] that is roughly the size of a spacecraft subsystem. The model has a total of 42 variables, including 10 mode variables with an average domain size of 4.4. The results were gathered using a 1.7GHz Intel Pentium III processor with 512MB of RAM.

The single-step estimation scenario in Figure 5 illustrated the likelihood-ordering limitations of BFTE and the additional accuracy gained using BFBSE. By considering estimation over many cycles (Figure 7), the loss of belief state probability density for BFTE becomes readily apparent, highlighting the amount of belief state knowledge lost over time. The reduction in probability density is exponential in the number of estimation cycles for both algorithms, but this reduction is dramatically less for BFBSE.

The space and time performance results are shown in Figures 8 and 9, respectively. For a varying size initial belief state, space is measured by the maximum number of nodes placed in the A* priority queue, while estimation time is measured in milliseconds. Each estimation technique has two sets of data points: the solid lines represent the space and time required to generate the *single* best estimate from the k states in

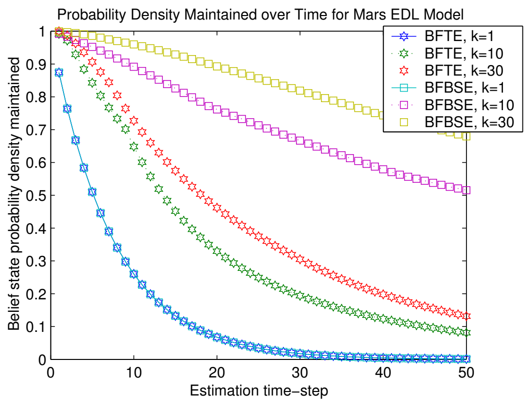


Figure 7: Probability density maintained over time.

the initial belief state (extracting best case behavior) and the dotted lines are the space and time required to generate the k most likely estimates (simulating average case performance for the estimator in a real application).

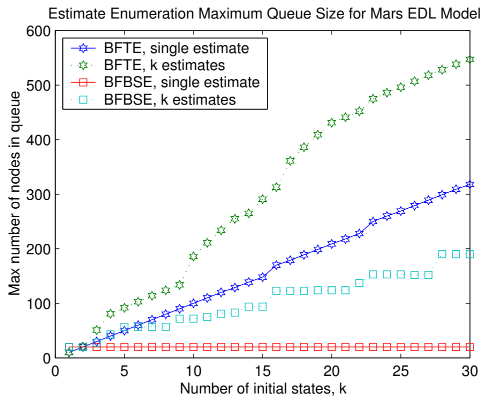


Figure 8: Maximum queue size for BFTE and BFBSE.

The space and time performance results show good alignment with the complexity analysis. The single-estimate results for BFBSE in Figure 8 reveals constant queue size as predicted in the best case complexity analysis. Similarly, we see linear queue growth for BFTE. Comparing the k estimate trends show that queue size is significantly less for BFBSE. The time results in Figure 9 also follow our predicted expectations. It is interesting to note that BFTE only outperforms BFBSE in both space and time when, as expected, $k < b$ where $b \approx 3$ for most real models. Since it is advantageous to set $k > 3$ (recall Figure 7), BFBSE will outperform BFTE in space and time for moderate size models and sufficiently sized belief states (e.g., $k \approx 10$).

6 Conclusion

This paper presented BFBSE as an approximate monitoring and diagnosis technique for PCCA that dramatically increases estimate accuracy by directly using the HMM Belief State Update equations. Our complexity analysis and empirical data shows that BFBSE outperforms BFTE, requiring less memory and less computational time for moderately sized approximate belief states and subsystem sized models.

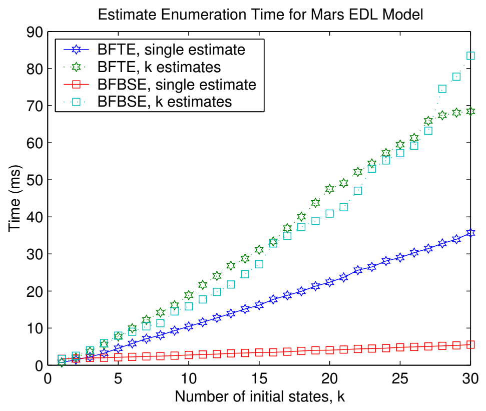


Figure 9: Time required for BFTE and BFBSE.

Further improvements to BFBSE should result from our current work to correctly update the prior state probabilities with a valid observation probability distribution. Efficient use of the observation probabilities in the OCSP objective function should also lead to better diagnostic discrimination, pruning lower probability states earlier in the search.

References

[Baum and Petrie, 1966] L. Baum and T. Petrie. Statistical inference for probabilistic functions of finite-state Markov chains. *Annals of Mathematical Statistics*, 37:1554–1563, 1966.

[de Kleer and Williams, 1987] J. de Kleer and B. C. Williams. Diagnosing multiple faults. *Artificial Intelligence*, 32(1):97–130, 1987.

[de Kleer and Williams, 1989] J. de Kleer and B. C. Williams. Diagnosis with behavioral modes. In *Proceedings of IJCAI-89*, 1989.

[Forney, 1978] G. D. Forney, Jr. The Viterbi algorithm. In *Proceedings of the IEEE*, pages 267–278, 1978.

[Ingham, 2003] M. Ingham. *Timed Model-based Programming: Executable Specifications for Robust Mission-Critical Sequences*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 2003.

[Kurien and Nayak, 2000] J. Kurien and P. Nayak. Back to the future for consistency-based trajectory tracking. In *Proceedings of AAAI-00*, pages 370–377, 2000.

[Ragno, 2002] R. Ragno. Solving optimal satisfiability problems through Clause-directed A*. Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, 2002.

[Struss and Dressler, 1989] P. Struss and O. Dressler. "Physical Negation" - Integrating fault models into the General Diagnostic Engine. In *Proceedings of IJCAI-89*, pages 1318–1323, 1989.

[Williams and Nayak, 1996] B. C. Williams and P. Nayak. A model-based approach to reactive self-configuring systems. In *Proceedings of AAAI-96*, pages 971–978, Portland, OR, August 4-8, 1996.

[Williams *et al.*, 2003] B. C. Williams, M. D. Ingham, S. H. Chung, and P. H. Elliott. Model-based programming of intelligent embedded systems and robotic space explorers. *Proceedings of the IEEE*, 91(1):212–237, January 2003.