2008 Conference on Decision and Control (submitted)
http://www.cds.caltech.edu/~murray/papers/bm08-cdc.html

Failure Probability of Verifiable Goal-based Control Programs due to State Estimation Uncertainty

Julia M. B. Braman and Richard M. Murray

Abstract— **Fault tolerance and safety verification of control systems that have state estimation uncertainty are essential for the success of autonomous robotic systems. A software control architecture called Mission Data System, developed at the Jet Propulsion Laboratory, uses goal networks as the control program for autonomous systems. Certain types of goal networks can be converted into linear hybrid systems and verified for safety using existing symbolic model checking software. A process for calculating the probability of failure of some verifiable goal networks due to state estimation uncertainty is presented. Extensions of this procedure to include other types of uncertainties are discussed, and example problems are presented to illustrate these procedures.**

I. Introduction

Autonomous robotic missions generally have complex, fault tolerant control systems. There are several ways to incorporate the necessary fault tolerance in a control architecture. One way is to create a flexible control system that can reconfigure itself in the presence of faults. However, if the control system cannot be verified for safety, even in the presence of state variable estimation uncertainty, the added complexity of the reconfigurability of a system could reduce the system's effective fault tolerance.

One particularly useful way to model a fault tolerant control system is as a hybrid system. The control of hybrid systems has been well researched [1]. When the continuous dynamics of these systems are sufficiently simple, it is possible to verify that the execution of the hybrid control system will not fall into an unsafe regime [2]. There are several software packages available that can be used for this analysis, including HyTech [3], UPPAAL [4], and PHAVer [5], all of which are symbolic model checkers. PHAVer in particular is able to exactly verify linear hybrid systems with piecewise constant bounds on continuous state derivatives. Safety verification for fault tolerant hybrid control systems ensures that the occurrence of certain faults will not cause the system to reach an unsafe state.

However, these verification software packages cannot verify linear hybrid systems that have uncertainty in the estimated state variables that are involved in mode transition logic. For autonomous systems, none of the state variables used in the control system are known perfectly, and these uncertainties can affect the safety of the system. Stochastic hybrid systems include uncertainty in the transitions of the hybrid automaton as probabilistic transition conditions. Many papers have been written on the verification of stochastic hybrid systems. Prajna et al [6] use barrier certificates to bound the upper limit of the probability of failure of the stochastic hybrid system; Kwiatkowska et al [7] discuss a probabilistic symbolic model checking software called PRISM; and Amin et al [8] describe stochastic reachability and maximal probabilistic safe set computations for discrete time stochastic hybrid systems. However, purely probabilistic transition conditions do not model estimation uncertainty in the constrained state variables well; deterministic transitions with probabilistic components may be a better model.

In this paper, systems specified using Mission Data System (MDS), a goal-based control architecture developed at the Jet Propulsion Laboratory, are analyzed. The structure of this paper is as follows. Section II summarizes important concepts of MDS that pertain to this work and describes the goal network conversion and verification procedure. Section III summarizes the previous work on verifying goal-based control programs in the presence of estimation uncertainty, including the problem set up and the uniform completion case [9]. Sections IV and V describe the major contributions of this paper, the extension of the failure probability calculation to non-uniform completion goal networks, goal networks with completion time uncertainty, and goal networks with location uncertainty. Section VI has some examples of the failure probability calculation for verifiable goal networks, and Section VII concludes the paper.

II. Background Information

A. State Analysis and Mission Data System

State Analysis is a systems engineering methodology that focuses on a state-based approach to the design of a system [10]. Models of state effects in the system to be controlled are used for such things as the estimation of state variables, control of the system, planning, and goal scheduling. State variables are representations of states or properties of the system that are controlled or that affect a controlled state. Examples of state variables could include the position of a robot, the temperature of the environment, the health of a sensor, or the position of a switch.

Goals and goal elaborations are created based on the models. Goals are specific statements of intent used to control a system by constraining a state variable in time. Goals are elaborated from a parent goal based on the intent and type of goal, the state models, and several intuitive rules, as described in [10]. A core concept of State Analysis is that the language used to design the control system should be nearly the same as the language used to implement the

J. Braman and R. Murray are with the Dept. of Mech. Eng., California Institute of Technology, Pasadena, CA 91125, USA braman@caltech.edu

control system. Therefore, the software architecture, MDS, is closely related to State Analysis.

Goal networks replace command sequences as the control input to the system. A goal network consists of a set of goals with their associated starting and ending time points and temporal constraints. A goal may cause other constraints to be elaborated on the same state variable and/or on other causally related state variables. The goals in the goal network and their elaborations are scheduled by the scheduler software component so that there are no conflicts in time, goal order or intent. Each scheduled goal is then achieved by the estimator or controller of the state variable constrained.

Elaboration allows MDS to handle tasks more flexibly than control architectures based on command sequences. One example is fault tolerance. Re-elaboration of failed goals is an option if there are physical redundancies in the system, many ways to accomplish the same task, or degraded modes of operation that are acceptable for a task. The elaboration class for a goal can include several pre-defined tactics. These tactics are simply different ways to accomplish the intent of the goal, and tactics may be logically chosen by the elaborator based on programmer-defined conditions. This capability allows for many common types and combinations of faults to be accommodated automatically by the control system [11].

B. Goal Network Conversion and Verification Procedure

Hybrid system analysis tools can be used to verify the safe behavior of a hybrid system; therefore, a procedure to convert goal networks into hybrid systems is an important tool for goal network verification. A process for converting certain types of goal networks is described in [12] and summarized in Fig. 1. These goal networks can have several state variables and several layers of goal elaborations, however time points must be well-ordered, which means that the time points must fire in the order they are listed in the elaboration.

Each state variable in the goal network is labeled as either controllable, uncontrollable, or dependent. A controllable state variable (CSV) is directly associated with a command class. An uncontrollable state variable (USV) is not associated with a command class in any way. A dependent state variable (DSV) has model dependencies on at least one controllable state variable. Different hybrid automata are created from goals on and states of these different types of state variables.

An outline of the conversion procedure for the goals on CSVs and DSVs is as follows:

1) Create elaboration and transition logic tables for each goal that elaborates any constraints on CSVs and for each CSV and DSV, respectively.
2) Place all goals on CSVs and DSVs between consecutive time points into groups.
3) In each group, create locations (modes) by combining branch goals (goals that are not ancestors of other goals in the group) with all parent and sibling goals (goals in the same tactic or other root goals). Label each location with the dynamical update equations for all CSVs and continuous DSVs constrained in the location. Create Success and Safing locations.
4) Create transitions between locations and groups using the elaboration and transition logic tables found above. Elaboration logic controls transitions into groups and failure transitions between locations in a group, and transition logic controls the transitions to the next group or to the Success location.
5) Add exit and failure transitions based on time to locations in groups that have time constraints. Add entry actions that reset the time variable to zero when transitions into these groups are taken.
6) Remove unnecessary locations, groups, and transitions.

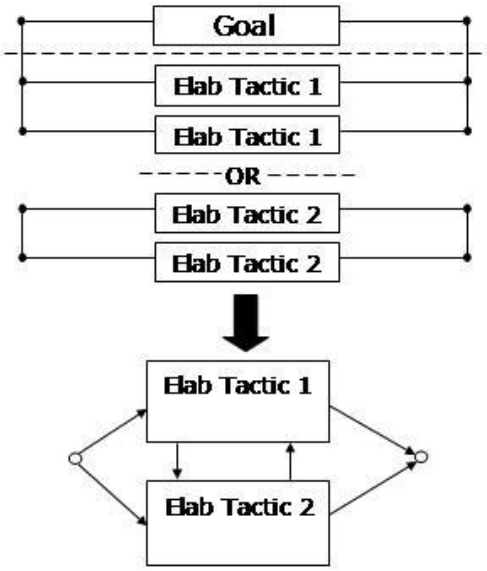

Fig. 1. Representation of a goal network to hybrid system conversion. The goals are represented by rectangles between circular time points, and elaboration is depicted by a dashed line, with child goals below the line. In the hybrid automaton, one group is shown where the circles represent group connectors and rectangles represent locations.

For each each USV, a separate hybrid automaton is created by making locations from the discrete states or discrete sets of states that the variable can take. The transition conditions are stochastic rates or are based on the state models. For safety verification, the hybrid automata are converted into PHAVer code and the appropriate transitions are synchronized between the automata. The unsafe (or incorrect) set is determined and conditions that would cause the hybrid automata to enter the unsafe set are searched for using the verification software. If no such conditions are found, the goal network is said to be verified.

III. Failure Probability Calculation for the Uniform Completion Case

In the previous section, the safety verification of the goal network could be completed assuming that all state variables are known exactly. For certain classes of verifiable goal networks, if the uncontrollable state variables that drive the

transitions into and between locations in a group have some bounded estimation uncertainty, the probability of reaching the unsafe set due to this uncertainty can be calculated. These uncertain state variables must have stochastic transitions between their discrete states (or discrete sets of states) that depend only on the previous state and can be modeled by a stationary Markov process. The probability that the estimated value of an uncertain state variable is the same as or different from the actual value is derived from the estimation uncertainty, and only depends on the actual value of the state variable.

In previous work [9], the failure probability calculation for the uniform completion case was derived. In the uniform completion case, the minimum execution time of a group is also the maximum execution time; in other words, all locations in the group contribute an equal amount to the completion of the goals or to the time constraint placed on the group. Therefore, a uniform completion time c_k would require exactly c_k execution time steps before the group would be exited normally. The given information for this problem includes the completion time c_k for each group, g_k, where $k = 1, ..., N$ and N is the total number of groups in a goal network's hybrid automaton conversion. Also known is the number of state variables that are uncertain, n, and the number of discrete states or sets of states that each uncertain state variable can take, m_i, where $i = 1, ..., n$. Let j_i (k_i) represent the jth (kth) possible value of the ith uncertain state variable, where $j_i = 1, ..., m_i, \forall i = 1, ..., n$ (likewise for k_i). Let v_{ei} represent the estimated value of the ith uncertain state variable and let v_{ai} represent the actual value of each uncertain state variable. Stationary Markov transition probabilities are given for each uncertain state variable from each discrete state to each discrete state, $P(v_{ai} = j_i | v_{ai,p} = k_i)$, where $v_{ai,p}$ is the previous actual value of the ith uncertain state variable. Finally, the probabilities of estimating each state variable to be each discrete state given the actual value, $P(v_{ei} = j_i | v_{ai} = k_i)$, which are calculated from the estimation uncertainty for the state variable, are given.

Let S be the set of all possible combinations of actual and estimated values that each uncertain state variable can take; in Fig. 2, S is the overall closed set. S has $\prod_{i=1}^{n} m_i^2$ elements, and each element takes the form $v_{a1}v_{e1}v_{a2}v_{e2}...v_{an}v_{en}$. For each group there are sets $\Omega_k \subset S$, where the elements of Ω_k cause the automaton to enter the unsafe set from g_k and are called "unsafe" elements. Since the goal network was verified, entrance into this set is always due to estimation uncertainty, and the probability of entering this set is the failure probability. Entrance into the unsafe set for each group is dictated by the actual values of the uncertain state variables, while the transition conditions within the group are dependent on the estimated value of the uncertain state variable. There also are sets $F_k \subset S$ in which each element causes the automaton to leave g_k and enter the Safing location without also entering the unsafe set. A set

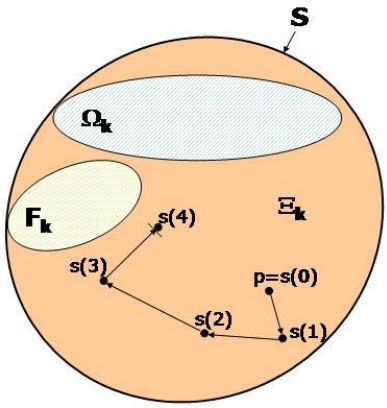

Fig. 2. A representation of sets of uncertain state variable elements. Set S is the sum of the set of unsafe elements, Ω_k, the set of "Safing" elements, F_k, and the set of non-failing elements, Ξ_k. A representative path starting at $s(0) = p$ that the automaton takes with completion time $c_k = 5$ is shown.

F_k may be empty. Finally, there are sets

$$\Xi_k \equiv S - (\Omega_k + F_k) \tag{1}$$

to which the remainder of elements in S belong. Elements in Ξ_k, called "non-failing" elements, allow operation to continue in the group or allow a successful transfer out of the group to the next one. These sets are represented graphically in Fig. 2.

The necessary parameters for the failure probability calculation are found in the following way. The probability of failing due to the initial condition is the sum of the probability of each element of Ω_k occurring, $a_k = \sum_{u=1}^{u_k} P(s(0) = x_u)$. Let u_k be the number of elements in set Ω_k and let x_u represent the uth element. So,

$$a_k = \sum_{u=1}^{u_k} \prod_{i=1}^{n} P(v_{ai} = j_{i,au}) P(v_{ei} = j_{i,eu} | v_{ai} = j_{i,au}), \tag{2}$$

where $j_{i,au}$ is the uth actual value of the ith uncertain state variable in set Ω_k and $j_{i,eu}$ is the uth estimated value of the ith state variable.

The vector W_k contains the probability of starting in each non-failing initial condition. Let q_k be the number of elements in Ξ_k and $W_k(q)$ be the qth element of W_k, where $q = 1, ..., q_k$ and $W_k(q) = P(s(0) = x_q)$.

$$W_k(q) = \prod_{i=1}^{n} P(v_{ai} = j_{i,aq}) P(v_{ei} = j_{i,eq} | v_{ai} = j_{i,aq}) \tag{3}$$

For each group, there is a $q_k \times q_k$ matrix, Q_k, whose elements are the probability of making a transition from element $q \in \Xi_k$ to element $q' \in \Xi_k$, where $q, q' = 1, ..., q_k$, and $Q_k(q, q') = P(s(r+1) = x_{q'} | s(r) = x_q)$.

$$Q_k(q, q') = \prod_{i=1}^{n} (P(v_{ai} = j_{i,aq'} | v_{ai,p} = j_{i,aq}) \times P(v_{ei} = j_{i,eq'} | v_{ai} = j_{i,aq'})), \tag{4}$$

where $v_{ai,p}$ is the previous actual value of the ith uncertain state variable.

For each group, there exists a $q_k \times 1$ vector, $W_{u,k}$, whose elements are the sum of probabilities of the transitions from the qth element in Ξ_k to each element in Ω_k, which is the transition from each non-failure state to any failure state, or $W_{u,k}(q) = \sum_{u=1}^{u_k} P(s(r+1) = x_u | s(r) = x_q)$.

$$W_{u,k}(q) = \sum_{u=1}^{u_k} \prod_{i=1}^{n} (P(v_{ai} = j_{i,au} | v_{ai,p} = j_{i,aq}) \times P(v_{ei} = j_{i,eu} | v_{ai} = j_{i,au})) \quad (5)$$

For both the uniform completion case and the non-uniform completion case described in the next section, the objective of finding the failure probability is to discover the effect that the estimation uncertainty has on the execution of otherwise correct goal networks. After entering a group in a goal network execution, there are only three ways to exit:

1) The completion time is reached, and the execution moves normally into the next group.
2) A transition to the "Safing" set, $s(r) \in F_k$, is taken before the completion time is reached.
3) A transition to the unsafe set, $s(r) \in \Omega_k$, is taken before the completion time is reached.

Only execution paths that fall under the third category are failure paths, and because only verifiable goal networks are considered, these paths are only possible due to the estimation uncertainty of the state variables that control transitions into the unsafe set. The definition of being verifiable is that there are no paths that reach the unsafe set when there is no estimation uncertainty.

The failure probability for each group, $W_s(k)$, is calculated by summing the path probabilities of all possible execution paths into the unsafe set. For $c_k = 1$, the only way to reach the unsafe set is to start in it, the probability of which is represented by $W_s(k) = a_k$. For $c_k = 2$, the probability of starting in the non-failure initial conditions and then making the transition to the unsafe set is added to the probability of starting in the unsafe set, and so on. The equation for the failure probability for $c_k \in [2, \infty)$, where $k = 1, ..., N$, is

$$W_s(k) = a_k + W_k \cdot \left[\sum_{x=0}^{c_k - 2} Q_k^x W_{u,k} \right]. \quad (6)$$

IV. Non-Uniform Completion Failure Probability

In the non-uniform completion case, the minimum execution time of a group is not the same as the maximum execution time. Sets of locations contribute a different amount to the completion of goals or time constraints on the group. An example of this is a group with a goal constraining a robot to cover some distance, however, different locations in this group constrain the maximum speed of the robot to different values. The set of locations with the maximum speed limit constraint have a contribution to the goal completion of 1; that is, this set of locations dictates the minimum execution time of the group. All other sets of locations with lower speed limit constraints have contribution values that are less than one. Paths that exit the group normally due to goal completion have location contribution value sums that are equal to or exceed the completion time.

Unlike the uniform completion case, the normal completion paths have different lengths and location contribution value patterns in the non-uniform completion case. Therefore, the number of ways to reach group completion, and likewise the unsafe set, is much larger in this case. The failure probability is calculated in the same way for this problem as for the uniform completion case, by summing the path probabilities of each failure path. This section will describe the algorithm to find all the failure paths and how the path probabilities are calculated.

A. Failure Path Algorithm

To find all the possible failure paths, the algorithm must be given a vector that contains the contribution values corresponding to each subset of locations in the group. The contribution value corresponding to the fastest subset of locations will always be 1 due to the definition of completion time; for the non-uniform completion case, there will always be at least one other subset of locations with a contribution value, $c_v \in [0, 1)$ and let n_g be the number of contribution values. The algorithm to find all failure paths is shown in Fig. 3 and can be summarized as follows:

1) Add a failure path that consists of one normal execution step for each subset of locations with $c_v < c_k$ into set s_1.
2) For the next n_g steps, create a set of paths s_{i+1} (where $i = 1, ..., n_g$) by appending a transition into the subset of locations with the ith contribution value to each path in the previous i sets whose path sum plus the ith contribution value is less than c_k.
3) Until the previous n_g sets are empty, create new sets of paths by appending a transition into the subset of locations with the ith contribution value (where $i = 1, ..., n_g$, looping through the values until complete) to all the paths in the previous n_g sets whose path sum plus the ith contribution value is less than c_k. If the last subset of locations transitioned into in a path has a contribution value of zero, another transition into that same subset cannot be added to that path due to how the path probabilities are calculated.

B. Path Probability Calculation

The same basic matrices that were used in the uniform completion case can be modified to apply to the non-uniform completion case. The initial failure probability, a_k, stays the same. However, the vector of probabilities of initially starting in the non-failing set, W_k, must be broken into vectors that describe starting in non-failing elements associated with each of the subsets of locations with different contribution values. Therefore, there will be n_g different, non-overlapping W_{ki} vectors, $i = 1, ..., n_g$. The same breakdown of the $W_{u,k}$ vector must occur, making n_g different, non-overlapping $W_{u,ki}$ vectors whose elements are the probabilities of going

```
For i = 1 to n_g
  If c_v,i < c_k
    Append(s_1, {c_v,i})
For i = 2 to n_g+1
  For j = 1 to i-1
    For all p ∈ s_j
      If Sum( p ) + c_v,i-1 < c_k
        Append(s_i, Append(p,c_v,i-1))
l = n_g + 1
While ( Union(s_1,s_l-1,...,s_l-ng+1) ≠ ∅ )
  For i = 1 to n_g
    For j = l-n_g+1 to l
      For all p ∈ s_j
        If Sum( p ) + c_v,i < c_k
          Append(s_l, Append(p,c_v,i))
    l++
failpaths = Union( s_i, i∈[1,l] )
```

Fig. 3. Failure path algorithm

from a non-failing state associated with the ith subset of locations to all of the states in the unsafe set. The Q_k matrix breakdown is a little more complicated. This matrix of probabilities of transitions from all non-failing elements to all other non-failing elements must be broken into n_g^2 matrices. Each matrix $Q_{ki,j}$ has transition probabilities from each non-failing element associated with the ith subset of locations to each non-failing element in the jth subset of locations, where $i = 1, ..., n_g$ and $j = 1, ..., n_g$.

The procedure for finding the failure probability for group g_k is essentially summing the path probabilities of all the failure paths found with the algorithm described in the previous section plus the probability of initially reaching the unsafe set (a_k). The path probability for each of the paths with the initial transition into the non-failing set followed by a transition into the unsafe set is $W_{ki} \cdot W_{u,ki}$. All other paths include products of $Q_{ki,j}$ matrices for each non-failing transition after the initial one. The path probability for these paths has the form $W_{ki_1} \cdot Q_{ki_1,i_2} ... Q_{ki_{p-1},i_p} W_{u,ki_p}$, where p is the number of transitions in the non-failing set, including the initial one. If a transition into a subset of locations with a contribution value of zero is represented in a path, a modified $Q_{ki,j}$ matrix is used in the path probability calculation to account for the possibility of staying in that subset anywhere from one to an infinite number of time steps. This modified $Q_{ki,j}$ matrix is

$$Q'_{ki,0} = Q_{ki,0}(I - Q_{k0,0})^{-1} \tag{7}$$

because

$$(I - Q_k)^{-1} = \sum_{x=0}^{\infty} Q_k^x. \tag{8}$$

V. Completion Time and Location Uncertainty

Estimation uncertainty of the uncontrollable state variables drove the uniform and non-uniform completion time failure probability derivations. It is possible to add two other sources of uncertainty; first, in the completion time and second, with which location in a group an uncertain state variable element is associated. Methods for dealing with these sources of uncertainty in the failure probability calculation are addressed in this section.

A. Completion Time Uncertainty

It may not be possible to exactly know the completion time for a group. If there is a probability distribution over a finite number of possible completion times, the failure probability for that group can be calculated by finding the failure probability for each possible completion time. The total failure probability for the group is the sum of the failure probabilities for each possible completion time multiplied by the completion time probability. This procedure works for both the uniform and non-uniform completion cases.

B. Location Uncertainty

For the failure probability calculations presented thus far, it is assumed that each uncertain state variable element is uniquely associated with only one location in a group. There are cases, however, where the transitions between elements in a group are deterministic, but which location an uncertain state variable element is associated with is also dependent on the previous location(s) visited in the execution path. It may be a misnomer to call this location uncertainty, but the added dependence on execution path does change the problem in a significant way.

The method to calculate the group failure probability is very similar to the ones presented for the uniform and non-uniform completion cases. In fact, the only changes to the processes involve the matrices and vectors that are used in the path probability calculations. A composite transition matrix between all combinations of uncertain state variable elements is created from the original stationary Markov transition matrices and the estimation uncertainty for each uncertain state variable. However, if there are sets of elements that could be associated with more than one location, one representation of the values for each possible location must be present in the transition matrix. For example, if element j of $\prod_{i=1}^{n} m_i^2$ possible elements could be associated with either location a or b, then a row and column in the transition matrix would be added and labeled element j^b, and the original row and column associated with element j would be renamed j^a. The transition matrix would then be a $(m_i^2 + 1) \times (m_i^2 + 1)$ matrix. The transition probabilities from element j^a to j^b and vice versa are zero and if an element r had a non-zero transition probability to element j, that same transition probability would apply from r to one of the new j elements, while the transition probability from r to the other j element would be zero. This is true because the transitions between locations must still be deterministic.

The W_k and $W_{u,k}$ vectors and Q_k matrices also reflect the addition of the extra elements due to the location uncertainty. These vectors are calculated in a similar way to the uniform and non-uniform completion cases, however, the Q_k matrices

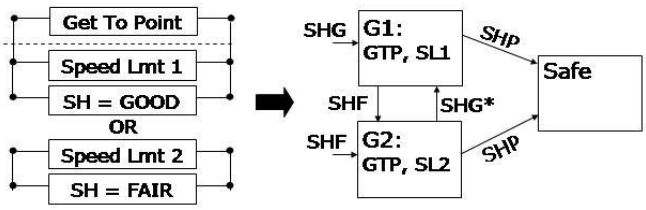

Fig. 4. Goal network and hybrid automaton for the non-uniform completion and location uncertainty examples. The transition marked with an asterisk does not exist in the location uncertainty example.

can be found by simply selecting the appropriate rows and columns of the composite transition matrix. The process for calculating the failure probability for the group continues as previously presented using these augmented matrices and vectors.

VI. Examples

A. Non-Uniform Completion Example

The task in this example is to drive a rover to a point, enforcing different speed limits for different sensor capabilities. The goal network and its hybrid automaton conversion are shown in Fig. 4. The unsafe set is as follows:

1) Location is G1 and sensor health is Fair or Poor
2) Location is G2 and sensor health is Poor
3) Location is Safe and sensor health is Good or Fair.

The goals in this group are completed when the rover reaches the point, however the two locations in the group drive the robot to the point at different rates. The speed limit in G1 is higher, and so the contribution value of location G1 is 1. If the speed limit in the second location, G2, is half the speed limit in G1, the contribution value of G2 is $1/2$.

The following vectors and matrices are found for this problem, using shorthand notation like $P(GF)$, which signifies the probability of starting in an actual state of Good and an estimated state of Fair. $W_{u,k1}$ and $W_{u,k2}$ are not shown, but are a scalar value and a 2×1 vector, respectively.

$$a_k = P(GP) + P(FG) + P(FP) + P(PG) + P(PF)$$

$$W_{k1} = P(GG) \qquad W_{k2} = \begin{bmatrix} P(GF) \\ P(FF) \end{bmatrix}$$

$$Q_{k1,1} = P(GG|GG) \qquad Q_{k2,1} = \begin{bmatrix} P(GG|GF) \\ P(GG|FF) \end{bmatrix}$$

$$Q_{k1,2} = \begin{bmatrix} P(GF|GG) & P(FF|GG) \end{bmatrix}$$

$$Q_{k2,2} = \begin{bmatrix} P(GF|GF) & P(FF|GF) \\ P(GF|FF) & P(FF|FF) \end{bmatrix}$$

For one given path for $c_k \geq 4$,

$$\sigma = \{G1, G2, G1, G2, G2, f\}, \tag{9}$$

the path probability is given as

$$P(\sigma) = W_{k1} \cdot Q_{k1,2} Q_{k2,1} Q_{k1,2} Q_{k2,2} W_{u,k2}. \tag{10}$$

The overall failure probability of this group is found given a completion time distribution of $[c_k, P(c_k)] = \{[4, 0.25], [5, 0.5], [6, 0.25]\}$, stationary Markov transition probabilities for the actual values of the sensor health, and several values for the estimation uncertainty associated with the sensor health. The results from these calculations are shown in Fig. 5.

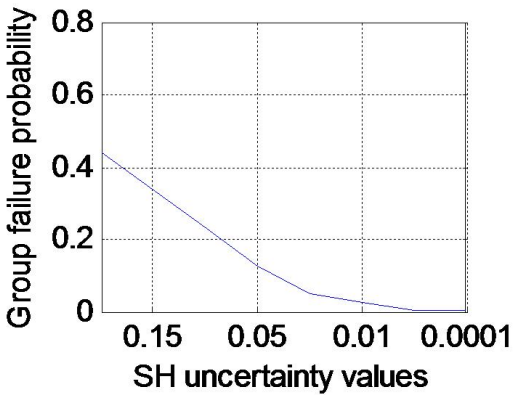

Fig. 5. Failure probabilities for various values of estimation uncertainty for non-uniform completion example

B. Location Uncertainty Example

The goal network is the same for this example, shown in Fig. 4, except that there is no transition back into G1 once G2 is reached. The location that any element with an estimated state of Good is associated with now has an execution path dependency. For this case, the set of possible uncertain elements would have to be supplemented with three new elements so that set S would be

$$S = \{GG^1, GG^2, GF, GP, FG^1, FG^2, FF, FP, PG^1, PG^2, PF, PP\}.$$

The matrices and vectors needed for this problem are updated from the ones derived for the previous example to include these new elements. Fig. 6 shows the sets that each of the elements are in, from which the W_k and $W_{u,k}$ vectors and Q_k matrices are derived. The probability of going from an element with a superscript of 1 to one with a superscript of 2 and vice versa is zero. Also, none of the elements with a superscript of 2 can be reached initially. Once the transition probabilities between the elements have been worked out and a completion time given, the problem is solvable using the same procedure as before.

VII. Conclusions and Future Work

This paper derives the failure probability of certain verifiable goal networks due to state variable estimation uncertainty. Methods to deal with completion time uncertainty and a type of location uncertainty were presented as extensions to the estimation uncertainty failure probability calculations. Two example goal networks were presented to illustrate the failure probability calculation procedures. The calculation of the failure probability for the different groups of a goal network can be used as a verification of the goal network in the presence of estimation uncertainty as well as a design

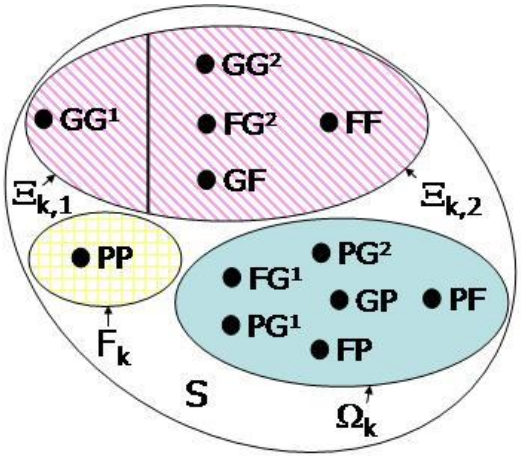

Fig. 6. Sets of elements for the location uncertainty example

tool to drive the design of goal networks or the choice of sensors and estimators to reduce the probability of failure.

Future work includes extending the calculation of a failure probability to include other types of uncertainty. Uncertainty in group transitions, controllable state variables, and the given probability distributions are all examples of possible types of uncertainty to include. The verification of goal networks in the presence of different forms of uncertainty, including estimation uncertainty, is an important problem, and this approach seems promising as a design tool for goal-based control programs.

VIII. Acknowledgements

The authors would like to gratefully acknowledge Michel Ingham, David Wagner, Robert Rasmussen, and the MDS team at JPL for feedback, suggestions, answered questions, and MDS and State Analysis instruction. This work was funded by NSF and AFOSR.

References

[1] G. Labinaz, M. M. Bayoumi, and K. Rudie, "A survey of modeling and control of hybrid systems," *Annual Reviews of Control*, 1997.

[2] R. Alur, T. Henzinger, and P.-H. Ho, "Automatic symbolic verification of embedded systems," *IEEE Transactions on Software Engineering*, vol. 22, no. 3, pp. 181–201, 1996.

[3] T. A. Henzinger, P.-H. Ho, and H. Wong-Toi, "HyTech: A model checker for hybrid systems," *International Journal on Software Tools for Technology Transfer*, 1997.

[4] K. Larsen, P. Pettersson, and W. Yi, "UPPAAL in a nutshell," *International Journal on Software Tools for Technology Transfer*, vol. 1, no. 1-2, pp. 134–152, 1997.

[5] G. Frehse, "PHAVer: Algorithmic verification of hybrid systems past HyTech," *International Conference on Hybrid Systems: Computation and Control*, 2005.

[6] S. Prajna, A. Jadbabaie, and G. J. Pappas, "Stochastic safety verification using barrier certificates," *IEEE Conference on Decision and Control*, 2004.

[7] M. Kwiatkowska, G. Norman, and D. Parker, "Probabilistic symbolic model checking with PRISM: a hybrid approach," *Int J Software Tools Technology Transfer*, vol. 6, pp. 128–142, 2004.

[8] S. Amin, A. Abate, M. Prandini, J. Lygeros, and S. Sastry, "Reachability analysis for controlled discrete time stochastic systems," *International Conference on Hybrid Systems: Computation and Control*, pp. 49–63, 2006.

[9] J. M. Braman and R. M. Murray, "Safety verification of fault tolerant goal-based control programs with estimation uncertainty." To appear, American Control Conference, 2008.

[10] M. Ingham, R. Rasmussen, M. Bennett, and A. Moncada, "Engineering complex embedded systems with State Analysis and the Mission Data System," *AIAA Journal of Aerospace Computing, Information and Communication*, vol. 2, pp. 507–536, December 2005.

[11] R. D. Rasmussen, "Goal-based fault tolerance for space systems using the Mission Data System," *IEEE Aerospace Conference Proceedings*, vol. 5, pp. 2401–2410, March 2001.

[12] J. M. Braman, R. M. Murray, and D. A. Wagner, "Safety verification of a fault tolerant reconfigurable autonomous goal-based robotic control system," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007.