FUZZY BAYESIAN NETWORKS FOR PROGNOSTICS AND HEALTH MANAGEMENT

by

Nicholas Frank Ryhajlo

A professional project submitted in partial fulfillment
of the requirements for the degree

of

Master of Science

in

Computer Science

MONTANA STATE UNIVERSITY
Bozeman, Montana

July, 2013

APPROVAL

of a professional project submitted by

Nicholas Frank Ryhajlo

This professional project has been read by each member of the professional project committee and has been found to be satisfactory regarding content, English usage, format, citations, bibliographic style, and consistency, and is ready for submission to The Graduate School.

Dr. John W. Sheppard

Approved for the Department of Computer Science

Dr. John Paxton

Approved for The Graduate School

Dr. Ronald W. Larsen

STATEMENT OF PERMISSION TO USE

In presenting this professional project in partial fulfillment of the requirements for a master's degree at Montana State University, I agree that the Library shall make it available to borrowers under rules of the Library.

If I have indicated my intention to copyright this professional project by including a copyright notice page, copying is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law. Requests for permission for extended quotation from or reproduction of this professional project in whole or in parts may be granted only by the copyright holder.

Nicholas Frank Ryhajlo

July, 2013

TABLE OF CONTENTS

1. INTRODUCTION 1

Problem 1

2. BACKGROUND 5

Bayesian Networks 5
Bayesian Network Example 6
Bayesian Inference - Example 1 7
Bayesian Inference - Example 2 9
Bayesian Inference 10
Continuous Values 11
Virtual Evidence 12
Fuzzy Sets 13
Fuzzy Membership Functions 14
Fuzzy Set Operations 16
Fuzzy Random Variables 16
α-Cuts 18

3. RELATED WORK 21

Fuzzy Fault Trees 21
Fuzzy Bayesian Networks 23
Coefficient Method 26
Virtual Evidence 30

4. FUZZY BAYESIAN NETWORKS 33

Notation 33
Approach 34
Simple Example of a Fuzzy Bayesian Network 35
Complexity Reduction Techniques 43
Effects of Fuzzy Membership Functions 44
Sequential Calculations 45
Combining Fuzzy Evidence 49
Detailed Example 50

5. EXPERIMENTS 59

Experimental Design 59

TABLE OF CONTENTS – CONTINUED

Doorbell Circuit 59
ATML Circuit 61
Li-Ion Battery Network 66
Experimental Results 70
Doorbell Network 71
Manipulate Membership Values 71
Real Values as Evidence 73
ATML Network 75
Manipulate Membership Values 76
Real Values as Evidence 77
Battery Network 79
Effects of Each Test Variable 80
Battery Degradation 81

6. CONCLUSION 84

Summary 84
Future Work 86

REFERENCES CITED 88

APPENDICES 92

APPENDIX A: Membership Functions 93
APPENDIX B: Evaluation Networks 96

LIST OF TABLES

Table Page

1 Conditional Probability Table for Cloudy node 7

2 Conditional Probability Table for Sprinkler node 7

3 Conditional Probability Table for Rain node 7

4 Conditional Probability Table for Wet Grass node 7

5 Short list of Fuzzy Operators 17

6 Values of tests from ATML fuzzy fault tree example 23

7 Conditional Probability Table for Resistor Short node 25

8 Conditional Probability Table for Current Test Node 25

9 Conditional Probability Table for Virtual Evidence Node 31

10 Conditional Probability Table for Resistor Short node 36

11 Conditional Probability Table for Current Test Node 36

12 Conditional Probability Table for V_CAC node 41

13 Conditional Probability Table for V_0DC node 41

14 Conditional Probability Table for C2 Open node 41

15 Input ranges for each Battery variable 80

16 Seven battery degradation test points 81

LIST OF FIGURES

Figure Page

1 Example Bayesian Network 6
2 Example Bayesian Network with a Virtual Evidence Node 13
3 Different Types of Membership Functions 15
4 Visual representation of Fuzzy Random Variables 18
5 Different Types of Membership Functions 19
6 Sample Fault Tree of ATML Circut 22
7 Simple Example Bayesian Network 24
8 Membership Functions for Example Network 25
9 Simple Example Bayesian Network with a Virtual Evidence Node 30
10 Simple Example Bayesian Network 35
11 Membership Functions for Example Network 36
12 Membership Functions for Small ATML Network 40
13 Subset of the ATML Networks 40
14 Examples of possible network strucures 46
15 Doorbell network with a hidden node 51
16 Circuit diagram for the doorbell test circuit 59
17 Doorbell diagnostic Bayesian network structure 60
18 Fuzzy Membership Function for the *Voltage* test 61
19 ATML test circuit 62
20 ATML diagnostic network 64
21 Battery network structure 67
22 Membership value of Battery Diagnosis varying *volt* membership 71
23 Membership value of *SW-O* with varying *Voltage* membership 73
24 Battery Diagnosis with varying Battery Voltage 74

LIST OF FIGURES – CONTINUED

Figure Page

25 Membership values for *SW-O* with varying Battery Voltage 74
26 Membership values with varying *Voltage* all other tests *Fail* 75
27 Membership value of *Q1 C Open* while varying $V_B DC\ 1$ 76
28 Membership values with varying Battery Voltage other tests *fail* 78
29 Membership values for individually varying V_{BE} and $V_E DC\ 1$ 79
30 Sweeps of battery capacity membership values 82
31 FBN predicted vs actual battery capacity 83
32 Membership functions for ATML network 94
33 Membership functions for ATML network (continued) 95

ABSTRACT

In systems diagnostics it is often difficult to define test requirements and acceptance thresholds for these tests. A technique that can be used to alleviate this problem is to use fuzzy membership values to represent the degree of membership of a particular test outcome. Bayesian networks are commonly used tools for diagnostics and prognostics; however, they do not accept inputs of fuzzy values. To remedy this we present a novel application of fuzzy Bayesian networks in the context of prognostics and health management. These fuzzy Bayesian networks can use fuzzy values as evidence and can produce fuzzy membership values for diagnoses that can be used to represent component level degradation within a system. We developed a novel execution ordering algorithm used in evaluating the fuzzy Bayesian networks, as well as a method for integrating fuzzy evidence with inferred fuzzy state information. We use three different diagnostic networks to illustrate the feasibility of fuzzy Bayesian networks in the context of prognostics. We are able to use this technique to determine battery capacity degradation as well as component degradation in two test circuits.

1

INTRODUCTION

In our lives we rely on the smooth operation of many electrical and mechanical systems. Some of these systems are more important than others, and if a failure occurs, the consequences can be dire. To help maintain proper operation of systems, engineers and scientists attempt to model these systems to monitor system health, diagnose problems, and predict failures.

Problem

Bayesian networks are a typical tool used to measure system states and diagnose problems. Such Bayesian networks rely on tests that are performed within a system. These tests can be voltage measurements, current measurements across resistors, component or ambient temperature, or functional tests like a light coming on or a buzzer buzzing. The outcome of these tests are typically used as *evidence* within a Bayesian network that relates this evidence to possible faults or failures within a system.

Within a Bayesian network, diagnoses are represented as random variables with a probability distribution over the particular part or component being good, or being a candidate for failure. This relationship between being good, or a candidate for failure can provide valuable information not only to system designers, but to the system operators and people performing maintenance. When these tests are part of regular operation of a system, they can provide real-time information to all three parties mentioned above.

When a component starts to go out of specification, the tests that are performed will also begin to deviate from the normal operational levels. If, for example, an airplane were able to monitor the health of its own systems and be able to diagnose problems in real-time, this information could be presented to a pilot who would be able to take preventive actions before a full failure occurs, endangering assets and lives. This self diagnosis would also be able to improve the maintenance process by reducing the amount of testing and diagnosing maintenance crews would be required to perform.

A diagnostic Bayesian network, in conjunction with evidence, can be used to calculate the probability a component is good or is a candidate to fail. However, it might be more valuable if a system would be able to represent levels of component and system degradation instead of probability of failure. The ability to represent levels of degradation would be very useful, for fault prognostics. Prognostics is a discipline that focuses on predicting future failure. More specifically it focuses on predicting the time at which a system will no longer function correctly.

Understanding the level of degradation would aid in prognostics by possibly making it easier to predict failures and thus schedule maintenance prior to failure occurring. Being able to schedule maintenance efficiently would help prevent expensive and possibly life threatening catastrophic failures of systems, while at the same time not wasting resources replacing components more then necessary. This is sometimes called "condition based maintenance" or "just in time maintenance."

An approach to creating a method to determine a level of system degradation has been previously performed in [1]. The process presented by that work provided a method of representing *gray-scale health*, which is a value in the range $[0, 1]$ representing system health. This gray-scale health is estimated by using fuzzy fault trees. The gray-scale health measure that is created from the fuzzy fault tree is the fuzzy

membership value for a particular component failure. The fuzzy membership value is a number in the range $[0, 1]$ that represents the *degree of membership* of a particular set. A traditional "crisp" set has membership values of either 0 or 1. These crisp sets are what are typically used in fault trees and Bayesian networks.

This application of fuzzy sets and the fuzzy fault trees to estimate gray-scale health was the original inspiration for the work reported here. This work focuses on being able to create a similar gray-scale health estimate with a Bayesian network instead of a fault tree. Similar to the previous work, the fuzzy membership function will help to determine a level of degradation.

In contrast with the method developed here, fault trees produce one, crisp answer, or diagnosis from a set of tests. The addition of fuzziness into the fuzzy fault tree softens this crisp result. In a Bayesian network, the diagnoses are probabilities, not crisp outcomes. The fact that the Bayesian networks output probabilities for each individual diagnosis allows the diagnostic reasoner to be much more flexible than a basic fault tree in that it can represent multiple, concurrent failures as well as varying probabilities of failures.

To solve the problems with Bayesian networks mentioned above, we propose to enhance Bayesian Networks with fuzzy sets to create a Fuzzy Bayesian Network. This solution uses fuzzy membership values in conjunction with a Bayesian network to determine the level of degradation within a system. This system is able to use fuzzy membership values similar to evidence in the network, and output a fuzzy membership value as a level of degradation. This solution is explained in more detail in Chapter 4.

Integrating the probabilities from the Bayesian network with the fuzzy membership functions becomes difficult because there are two independent measures of uncertainty being used to produce one value to represent system degradation. Probabilities

are not fuzzy membership values, and fuzzy membership values are not probabilities. Each represent slightly different concepts, even though they are both represented by a real number in the interval $[0, 1]$. This makes the integration of these two concepts difficult because we want to be able to preserve both the probabilities calculated from the Bayesian network and the fuzzy membership values throughout the calculations.

A benefit of using fuzzy evidence within a Bayesian network is that the model can become much more expressive than a traditional Bayesian network. This is because Bayesian networks, similar to fault trees, typically rely on crisp evidence. The outcome of a particular test will be either a *Pass* or a *Fail*, and the network can use that as evidence in its calculations. However, when taking measurements like voltage or current, a lot of information and expressive power is lost by mapping those continuous measurements into either a *Pass* or a *Fail*.

When dealing with continuous measurements it can be very difficult to specify *exactly* a level at which a test goes from passing to failing. When using a Fuzzy Bayesian Network, this does not need to be done. The measurements that are involved in the test can be mapped directly into the network by a fuzzy membership function. This fuzzified value is then used by the network in coming up with a level of degradation.

The overall problem this work is focusing on is the definition of the framework of a Fuzzy Bayesian Network within the context of prognostics and health management that will give levels of degradation for each measured component.

BACKGROUND

Bayesian Networks

Bayesian networks are probabilistic models corresponding to joint probability distributions that utilize conditional dependencies among random variables. Bayesian networks are used in a wide variety of domains, such as image processing, search, information retrieval, diagnostics, and many others. A Bayesian network uses observations, or evidence, and previously determined conditional probabilities to give the probability of a certain state.

More formally, a Bayesian network $\mathcal{B}$ is a directed, acyclic graph whose vertices correspond to random variables of a distribution, and the edges correspond to conditional dependencies between random variables. Each vertex has an associated conditional probability distribution: $P(X_i|\text{Pa}(X_i))$, where $\text{Pa}(X_i)$ are the parents of vertex X_i. The lack of an edge between two vertices indicates there is no direct interaction between the two nodes. However, these nodes can still interact in certain circumstances. An example of this is a V-structure where a common child of the two nodes is known.

Bayesian networks are a way of representing joint probability distributions in a more compact way by using conditional dependencies among the random variables. Instead of needing to enumerate the entire joint probability distribution we can just use the product rule from probability to get the following:

$$P(X_1, ..., X_n) = P(X_1) \prod_{i=2}^{n} P(X_i|X_1, ..., X_{i-1})$$

Bayesian networks are able to exploit conditional independence, which is represented in the directed acyclic graph $\mathcal{G}$, to reduce the model's complexity and yield the fol-

lowing:

$$P(X_1, ..., X_n) = \prod_{i=2}^{n} P(X_i|\text{Pa}(X_i))$$

Bayesian networks are frequently used because the models they use are often easier to understand than other graphical models, like Artificial Neural Networks. Additionally, even without the use of evidence, it can be much easier to tell what a particular network is representing and how it will behave in the presence of evidence. Bayesian networks are generally easy for domain experts to construct because of their reliance on conditional probabilities and not arbitrary weights like other graphical models.

Bayesian Network Example

To better illustrate Bayesian networks, we present an example from [2]. Assume we have the network in Figure 1, and the conditional probability tables in Tables 1, 2, 3, and 4. In the network representations we use in this project, we represent query (diagnosis) nodes as ovals, evidence nodes a diamonds, and hidden nodes as squares. Hidden nodes are random variables that do not have evidence applied to them, but are also not queried. Hidden nodes do however have conditional probability tables, which do effect the calculations performed in the inference process.

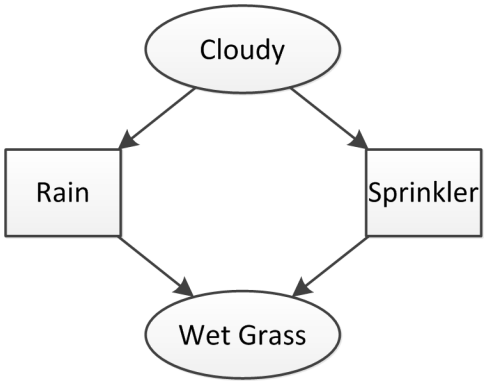


Figure 1: Example Bayesian Network

Table 1: Conditional Probability Table for Cloudy node

P(¬Cloudy)	P(Cloudy)
0.5	0.5

Table 2: Conditional Probability Table for Sprinkler node

Cloudy → Sprinkler	P(¬Sprinkler)	P(Sprinkler)
¬Cloudy	0.5	0.5
Cloudy	0.9	0.1

Table 3: Conditional Probability Table for Rain node

Cloudy → Rain	P(¬Rain)	P(Rain)
¬Cloudy	0.8	0.2
Cloudy	0.2	0.8

Table 4: Conditional Probability Table for Wet Grass node

Rain ∧ Sprinkler → Wet Grass		P(¬Wet Grass)	P(Wet Grass)
¬Rain	¬Sprinkler	1.00	0.00
¬Rain	Sprinkler	0.1	0.9
Rain	¬Sprinkler	0.1	0.9
Rain	Sprinkler	0.01	0.99

Bayesian Inference - Example 1. There is a lot of information stored within this network. With this network we can ask questions like, "What is the probability the grass is wet, given the sky is cloudy?" This question then takes the form

P(Wet Grass|Cloudy). Since we know, or have *evidence* that the sky is cloudy, we can use Table 2 to give us the probability of the sprinkler being on when the sky is cloudy, P(Sprinkler|Cloudy) = 0.1 because, by the directed edge in the graphical structure, we know that *Sprinkler* is conditionally dependent on *Cloudy*. Similarly, we can use Table 3 to give us the probability of *Rain* when the sky is cloudy, P(Rain|Cloudy) = 0.8, also because by the graphical structure we know that *Rain* is conditionally dependent on *Cloudy*.

Now that we have the updated beliefs for the random variables *Rain* and *Sprinkler*, we can update the belief for *Wet Grass*. To do this we need to get the conditional distribution from Table 4 and multiply it by the updated beliefs we have for *Rain* and *Sprinkler*. In the following example, for brevity the random variables *Rain*, *Sprinkler*, *Cloudy* and *Wet Grass* will be represented as R, S, C and W respectively.

$$\begin{aligned}
P(W|C) &= P(S|C)P(R|C)P(W|R,S) \\
&\quad + P(\neg S|C)P(R|C)P(W|R,\neg S) \\
&\quad + P(S|C)P(\neg R|C)P(W|\neg R,S) \\
&\quad + P(\neg S|C)P(\neg R|C)P(W|\neg R,\neg S) \\
&= 0.1 \times 0.8 \times 0.99 + (1-0.1) \times 0.8 \times 0.9 \\
&\quad + 0.1 \times (1-0.8) \times 0.9 + (1-0.1) \times (1-0.8) \times 0.0 \\
&= 0.7452
\end{aligned}$$

Thus, we find that P(Wet Grass|Cloudy) = 0.7452. This was a fairly simple process since we are propagating the beliefs in the direction of the conditional dependencies. Due to this, we only really need to look up the probabilities in the conditional probability tables and multiply or add where appropriate.

This example relied heavily on *marginalization*. Marginalization is an important technique in evaluating Bayesian networks and performing Bayesian inference.

Marginalization is the process of summing over all states of a variable to eliminate it, or marginalize it. More formally, given two random variables X and Y:

$$P(X) = \sum_{y \in \text{Val}(Y)} P(X, y)$$

Another valuable technique, similar to marginalization, is the process of *conditioning*. We use conditioning to calculate the probability of a state assignment. Formally, given two random variables X and Y:

$$P(X) = \sum_{y \in \text{VAL}(Y)} P(X|y)P(y)$$

Both of these processes are key to the task of probabilistic inference, and are used very often.

Bayesian Inference - Example 2. We can go the other direction in the network and ask the question, "What is the probability it is cloudy, given the grass is wet?", this is written as $P(\text{Cloudy}|\text{Wet Grass})$. To calculate this we need to apply Bayes' rule, which is defined for events A and B as Equation 1.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{1}$$

We can apply Bayes' rule to our current problem to get:

$$P(\text{Cloudy}|\text{Wet Grass}) = \frac{P(\text{Wet Grass}|\text{Cloudy})P(\text{Cloudy})}{P(\text{Wet Grass})}$$

We know from the previous example that $P(\text{Wet Grass}|\text{Cloudy}) = 0.7452$. We also know from Table 1 that $P(\text{Cloudy}) = 0.5$, so all we still need to calculate is $P(\text{Wet Grass})$. This is done by summing over the variable *Cloudy*. This is done by calculating $P(\text{Wet Grass}|\neg\text{Cloudy})$ just like above, then adding it to

P(Wet Grass|Cloudy) which was calculated earlier.

$$\begin{aligned} P(\text{Wet Grass}) &= P(\text{Wet Grass}|\text{Cloudy}) + P(\text{Wet Grass}|\neg\text{Cloudy}) \\ &= 0.7452 + 0.549 \\ &= 0.6417 \end{aligned} \tag{2}$$

Now we can use these probabilities to fill in Bayes' rule from above.

$$\begin{aligned} P(\text{Cloudy}|\text{Wet Grass}) &= \frac{P(\text{Wet Grass}|\text{Cloudy})P(\text{Cloudy})}{P(\text{Wet Grass})} \\ &= \frac{0.7452 \times 0.5}{0.6417} \\ &= 0.5758 \end{aligned} \tag{3}$$

Thus using the Bayesian network and Bayes' rule, the probability of it being cloudy given that the grass is wet is 0.5758.

Bayesian Inference

The process used above is called Bayesian inference. Inference is the task of computing the posterior probability distribution for a set of query variables, given some observed event. This event is manifested as an assignment of values to a set of evidence variables. Typically X is used to denote the query variable, $\mathbf{E}$ denotes the set of evidence variables $E_1, ..., E_m$, and $\mathbf{e}$ is a particular observed event. Additionally, $\mathbf{Y}$ is used to denote the set of nonevidence, nonquery variables $Y_1, ..., Y_l$, which are called hidden variables. Typically queries to a Bayesian network are of the form $P(X|\mathbf{e})$ [3]. In the example above where we were evaluating P(Cloudy|Wet Grass), the evidence variable was *Wet Grass*, and the query variable was *Cloudy*. *Rain* and *Sprinkler* were hidden variables.

As can be seen, even in this small example, there is the definite possibility for an exponential blowup when performing inference. The method for performing inference

presented here is called exact inference. The problem of inference in graphical models is NP-hard. Unfortunately, approximate inference, and the use of approximate methods to perform inference is also NP-hard [4]; however, approximate inference is easier to manage as a trade off between accuracy and complexity. All inference that is used in this work is exact inference; addressing approximate inference is beyond the scope of this project.

Continuous Values

In all of the examples and descriptions seen so far, the assumption is made that the random variables in Bayesian networks have discrete states. For example, there is no distinction made between slightly damp grass, and grass that is soaking wet. This can make using Bayesian networks difficult when evidence comes in the form of sensor inputs. Sensors, such as thermometers, rain gauges, volt meters and others do not usually return a discrete value, but rather a continuous value. There are a few techniques that can be used with Bayesian networks to handle continuous valued data.

The first method is to use a binning discretization method. This is where the range of values is split into bins. This can work well; however, determining bin width and number is problem dependent. It can be difficult to get a proper balance between expressive power and accuracy. If the data is split into too many bins, then it can be difficult to learn the parameters of a network because there is not enough sample data spread across the bins. Similarly, if the data is not split into enough bins, expressive power of the network, and of the evidence can be lost. Similar to this problem of choosing the proper number of bins, the borders of the bins must also be chosen carefully in order to prevent the problems mentioned above.

The process of binning adapts continuous values into discrete states which can be used directly in a Bayesian network. An alternative to this method are Gaussian Bayesian Networks. Gaussian Bayesian networks are defined in [4] to be a Bayesian network all of whose variables are continuous, and where all of the continuous probability distributions are linear Gaussians. An example of this for a variable Y which is a linear Gaussian of its parents $X_1, ..., X_k$ is defined as:

$$p(Y|x_1, ..., x_k) = \mathcal{N}\left(\beta_0 + \beta_1 x_1 + ... + \beta_k x_k; \sigma^2\right) \tag{4}$$

As can be seen, the variable is defined to be drawn from a Gaussian distribution which is defined by its parents and a variance. This is a very powerful method for modeling continuous values directly in a Bayesian network.

Virtual Evidence

Virtual evidence is not a method for mapping a continuous values into a Bayesian network. Virtual evidence is a probability of evidence. Thus virtual evidence is a method for incorporating the uncertainty of evidence into a Bayesian network [5].

Virtual evidence is used by adding a *virtual evidence node* as a child of a regular evidence node in a network. Using the network from the previous example, we can add virtual evidence to the node *Cloudy* in Figure 2. Evidence is then set as virtual evidence on the *VE Cloudy* node, not the *Cloudy* node directly. This virtual evidence is set by manipulating the conditional probability table for *VE Cloudy*. Then since *VE Cloudy* is a descendant of *Cloudy*, we use Bayesian inference to update $P(\text{Cloudy})$.

If we want to set the virtual evidence as Cloudy $= 0.75$ and $\neg$Cloudy $= 0.25$ then we can calculate $P(\text{Cloudy}|\text{VE Cloudy})$ in Equation 5.

$$P(\text{Cloudy}|\text{VE Cloudy}) = \frac{P(\text{VE Cloudy}|\text{Cloudy})P(\text{Cloudy})}{P(\text{VE Cloudy})} \tag{5}$$

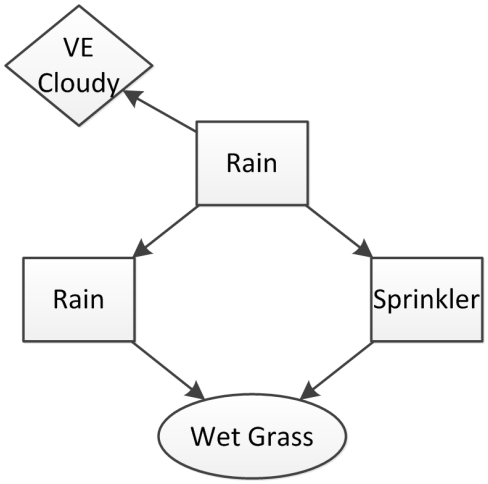


Figure 2: Example Bayesian Network with a Virtual Evidence Node

Typically virtual evidence is applied to discrete states. For example, in the context of system health testing, a test can either Pass or it can Fail. However, it can be difficult if not impossible to define specific thresholds that determine a pass condition or a failure condition. In addition to this limitation, these networks do not represent degradation, but probability of particular state.

Fuzzy Sets

In traditional set theory an object is either a member of a set or it is not. These type of sets are called *crisp sets*. Crisp sets, like those used in the Bayesian network above, are very common for representing evidence, and outcomes. Often, objects in the real world do not fit cleanly into crisp sets. Typically we define sets in terms of imprecise, linguistic variables. For example, the set of "tall" people has a very imprecise, or fuzzy, meaning.

In the context of system health monitoring the Montana State University Space Science and Engineering Laboratory came across the need for fuzzy sets to represent

unsafe conditions on their FIREBIRD satellite. It was difficult to determine good thresholds for alarm values for things because not wanting to have alarms being triggered all the time, and when there is no problem, but at the same time wanting to have alarms trigger when first entering an unsafe situation. To account for these imprecise boundaries fuzzy sets can be used.

Let X be a space of objects with an element denoted x, such that $x \in X$. A *fuzzy subset* A of X is characterized by a *membership function*, $\mu_A(x)$, which associates each point in X to a real number on the interval [0,1]. The *membership value*, which is the value of the membership function for a point x in X, represents the "degree of membership" of x in set A [6]. Consequently, the closer $\mu_A(x)$ is to 1, the higher the degree of membership of x in A.

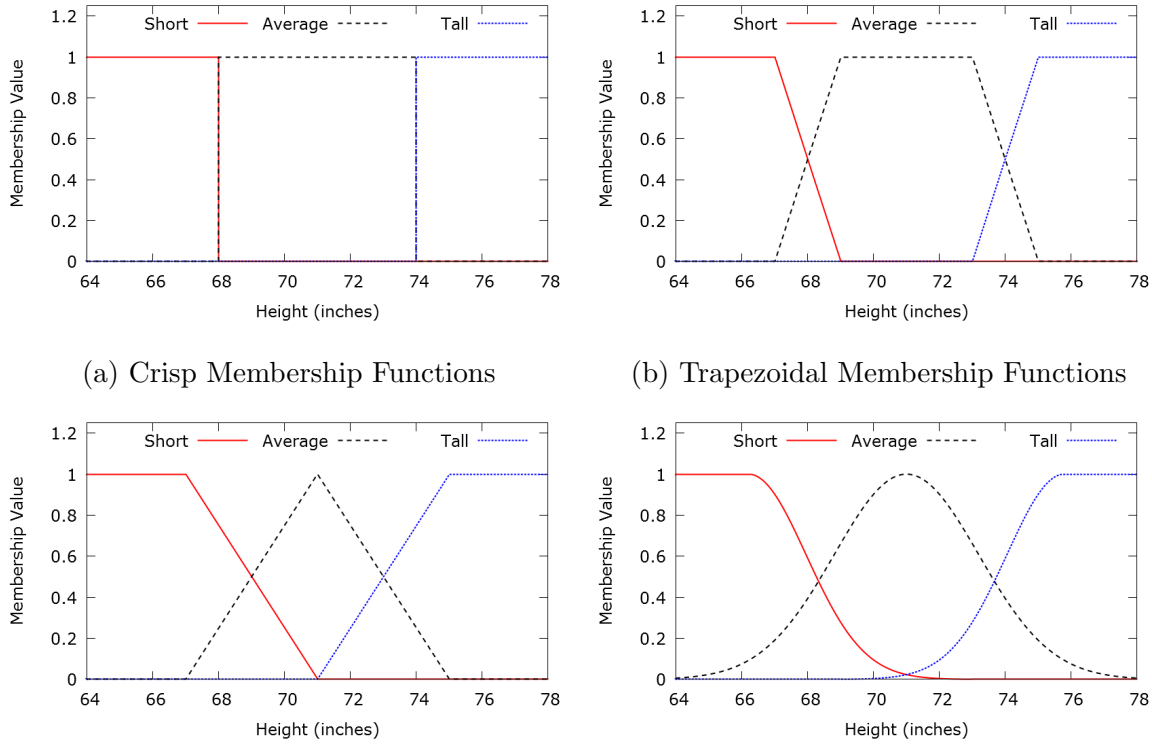
Using this definition of fuzzy sets, we can also say that crisp sets are a special case of fuzzy sets. When the membership function return either 0 or 1, it is a crisp set. A conceptual way to think about fuzzy sets is that every object is a member of every set, just to different degrees.

Fuzzy Membership Functions

The practical necessity fuzzy sets can easily be shown by using an example. As stated before, when considering the height of an individual, intuitively there is not a specific, hard boundary between someone who is short, average height, and tall. In the realm of crisp set theory, if someone is below, say 68 inches in height, that person is short, and if between 68.0001 inches and 74 inches in height, then that person is of average height. A graphical example of this can be seen in Figure 3a.

We can represent this example using the fuzzy sets *Short*, *Average*, and *Tall* in Figure 3. This figure shows some of the most common types of membership functions

applied to the human height example mentioned before: trapezoidal (Figure 3b), triangular (Figure 3c), and Gaussian (Figure 3d). When using the fuzzy membership functions shown in Figure 3b, if someone is 68 inches tall, they are a member of *Average* with degree of 0.5, and a member of *Tall* with 0.5.



(a) Crisp Membership Functions

(b) Trapezoidal Membership Functions

(c) Triangular Membership Functions

(d) Gaussian Membership Functions

Figure 3: Different Types of Membership Functions

Fuzzy membership functions are commonly misrepresented or misinterpreted as probability functions, however, they measure very different things. With probabilities, if someone has a 50% chance of being tall or 50% chance of being short, they have equal probability of being tall or short, but that does not mean they are equally tall and short like would be the case with fuzzy sets. Additionally, unlike a probability distribution that must sum to 1 over all possible values, membership values do not

have this requirement. The only requirement placed on them is they must be a real value on the interval [0,1].

Fuzzy Set Operations

Within classical set theory there are operators that can be used on sets, such as *Union*, *Intersection*, *Set Difference*, *Cartesian Product*.

The classical set operations have also been defined in the context of fuzzy sets. However, there can be multiple definitions for various fuzzy set operations. While multiple definitions of the same operator can be correct, multiple definitions are useful because different scenarios may require different definitions of the same operator. For example, t-norm is a binary operation that generalizes the intersection operator. Two examples of fuzzy t-norms or intersections are: $\mu_{A\cap B}(x) = \min\left[\mu_A(x), \mu_A(x)\right] \forall x$ and $\mu_{A\cap B}(x) = \mu_A(x) \cdot \mu_B(x) \forall x$. These t-norms are referred to as the Gödel t-norm and the product t-norm respectively. Both of these are valid t-norms even though they have different definitions. Fuzzy operators routinely have multiple definitions because in different contexts, different definitions of the same operator might be needed.

Table 5 is a short list of fuzzy operators. The list is primarily compiled from [7] and [8]. This table provides the definitions for all of the operators we will be using in the rest of this work.

Fuzzy Random Variables

Fuzzy random variables were introduced by Kwakernaak [9] [10] and enhanced by Puri and Ralescu [11] to model imprecisely valued functions represented by fuzzy sets that are associated with random experiments [12]. Kwakernaak introduced Fuzzy Random Variables in 1978 as "random variables whose values are not real, but fuzzy

Table 5: Short list of Fuzzy Operators

Containment	$A \subset B$	$\mu_A(x) \leq \mu_B(x)\ \forall x$
Equality	$A = B$	$\mu_A(x) = \mu_B(x)\ \forall x$
Complement	A'	$\mu'_A(x) = 1 - \mu_A(x)\ \forall x$
Union (s-norm)	$A \cup B$	$\mu_{A\cup B}(x) = \max[\mu_A(x), \mu_A(x)]\ \forall x$
	$A \cup B$	$\mu_{A\cup B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x) \cdot \mu_B(x)\ \forall x$
Intersection (t-norm)	$A \cap B$	$\mu_{A\cap B}(x) = \min[\mu_A(x), \mu_A(x)]\ \forall x$
	$A \cap B$	$\mu_{A\cap B}(x) = \mu_A(x) \cdot \mu_B(x)\ \forall x$
Product	AB	$\mu_{AB}(x) = \mu_A(x) \cdot \mu_B(x)\ \forall x$
Sum	$A \oplus B$	$\mu_{A\oplus B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x) \cdot \mu_B(x)\ \forall x$

numbers" [13]. Central to the concept of the FRV is a concept of "windows of observation." Windows of observation correspond to linguistic interpretations of traditional random variables. An example of this is the task of classifying people by age. The actual age is represented by an ordinary random variable X. However, when we perceive people, we typically assign a linguistic variable to their age. This perceived random variable, ξ, which can be conceptualized through the use of linguistic variables, or fuzzy sets.

Thus a fuzzy random variable is a mapping from the sample space, Ω, of the random variable to the class of normal convex fuzzy subsets. Thus, every instance in the sample space is mapped to its own fuzzy membership function. In Figure 4, we can see for each ω_i there is a corresponding membership function. In the conext of the age example, ω_i would be an observation of a person, $\xi(\omega_i)$ is the mapping that defines the perception of that persons age, and finally $x(\omega_i)$ is the actual person's age.

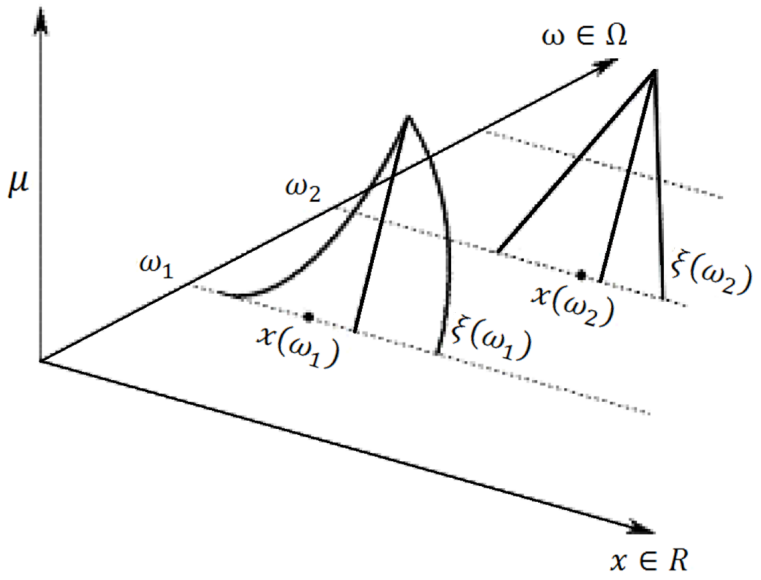


Figure 4: Visual representation of Fuzzy Random Variables

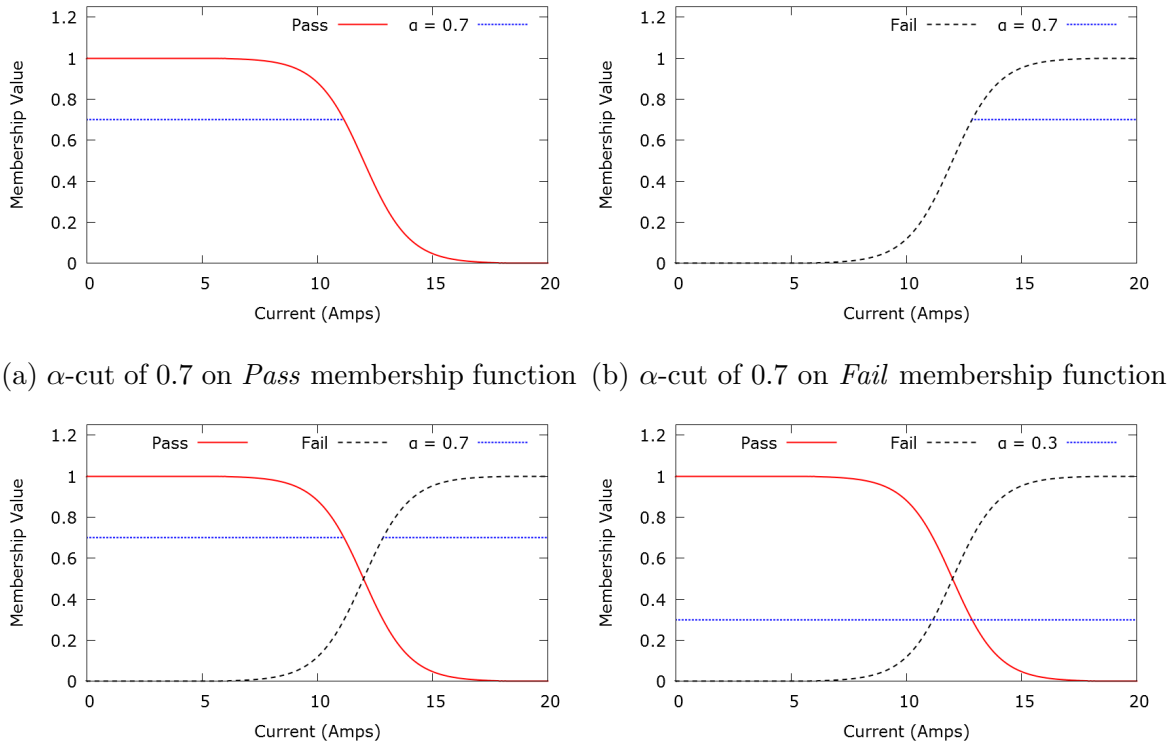
Often, these mappings are also defined with particular α-cuts. So, essentially a FRV is a mapping from an event $\omega \in \Omega$ to a fuzzy membership function, which can have a α-cut applied to it. A graphical example of these windows with fuzzy random variables can be seen in Figure 4. In this figure each ω represents an event. Then with each event, there is a window, which is represented by $\xi(\omega)$. Each of these membership functions are specific to each observation of each instance of the random variable X.

α-Cuts

Fuzzy sets and fuzzy membership functions provide a very descriptive framework for describing situations in more detail than crisp sets. This added detail, however, makes computation with fuzzy sets and fuzzy variables much more complicated. One way to attempt to rectify this situation is to use what are called α-cuts. α-cuts are a

technique to decompose fuzzy sets into a collection of crisp sets [14]. An α-cut is a real value on the range $[0, 1]$ that defines a "cut" membership for a membership function. Typically many of these α values are defined to different levels of membership value. These cuts, in the case of Fuzzy-Random Variables, are used to represent levels of uncertainty in the membership.

Since this is a fairly abstract concept added on top of the abstract concept of a fuzzy membership function, it is best to illustrate this with an example. Assume we are measuring current across a resistor to monitor if the part has failed. As a resistor fails (by shorting), the current across the resistor will go up dramatically. The membership functions for modeling this scenario are modeled in Figure 33.



(a) α-cut of 0.7 on *Pass* membership function (b) α-cut of 0.7 on *Fail* membership function

(c) α-cut of 0.7 (d) α-cut of 0.3

Figure 5: Different Types of Membership Functions

Figures 5a and 5b represent the membership functions for a resistor passing or failing a resistor short test respectively. The α-cut shown has a value of $\mu = 0.7$. Anywhere the membership value falls below the indicated line, the membership function for that test becomes 0. Figure 5c is a combination of Figures 5a and 5b. This α-cut forms a crisp set that contains elements of the domain associated with membership values that are greater than or equal to the α value. In this example, a current of 10 Amps have membership values of: $\mu_{\text{Pass}}(10 \text{ Amps}) = 1$ and $\mu_{\text{Fail}}(10 \text{ Amps}) = 0$, and would thus result in a *Pass*.

The use of α-cuts can be useful for an operator to discretize fuzzy values. However, it can lead to unexpected results if not careful. In Figure 5c at 11.5 Amps and $\alpha = 0.7$, the membership values are: $\mu_{\text{Pass}}(11.5 \text{ Amps}) = 0$ and $\mu_{\text{Fail}}(11.5 \text{ Amps}) = 0$. This means it is neither a pass or a fail. This is because, as can be seen in the Figure, there is a break in the α cut line because both states have membership values less then 0.7. This may be a desirable outcome, but it is something to be aware of. Similarly, in Figure 5d, at 11.5 Amps and $\alpha = 0.3$, the membership values are: $\mu_{\text{Pass}}(11.5 \text{ Amps}) = 1$ and $\mu_{\text{Fail}}(11.5 \text{ Amps}) = 1$. This means that the test both passed and failed at the same time.

Since the α-cut influences how selective the membership function is, it is often used as a method to define and limit uncertainty in fuzzy values. This is primarily done in Fuzzy Random Variables. This technique is also used as a pre-processing step to enable the use of techniques that require crisp sets, like Bayesian networks.

RELATED WORK

Fuzzy Fault Trees

Fuzzy fault have been used previously to calculate levels of degradation within a system. In [1] fuzzy fault trees were used to create a level of *gray-scale health*. Fuzzy fault trees are based on the combination of fuzzy sets and fault trees. Traditional fault trees are a model used in failure analysis which utilizes Boolean states and relations between states to find either a diagnosis to a problem, or to recommend an action to resolve the problem.

Fault trees are graphical models, much like flow charts, that represent a process that shows relationships of test outcomes graphically. An example fault tree is given in Figure 6[1]. This fault tree is used to diagnose the ATML test circuit in Appendix A. The ATML test circuit was created to demonstrate the Automatic Test Markup Language. The fuzzy fault tree behaves just like a normal fault tree when membership values for tests are either 0 or 1. If an outcome is 0 or 1, the corresponding path of the fault tree is taken. However, if a membership value is between 0 and 1, all paths with non-zero membership values must be taken. A way to think about this at that point is to create multiple instances of the fault tree taking each path separately but maintaining the fuzzy membership value the whole way through the tree.

For example, given the fault tree in Figure 6, we use the results from specific tests in Table 6. The actual fuzzy membership functions are not given, but the corresponding fuzzy membership values are given for each test value measured.

First we start with the V_{CC} resistance test, which passes with a membership of 1. We then move to the V_0 AC voltage test, which fails with a membership value of 1.

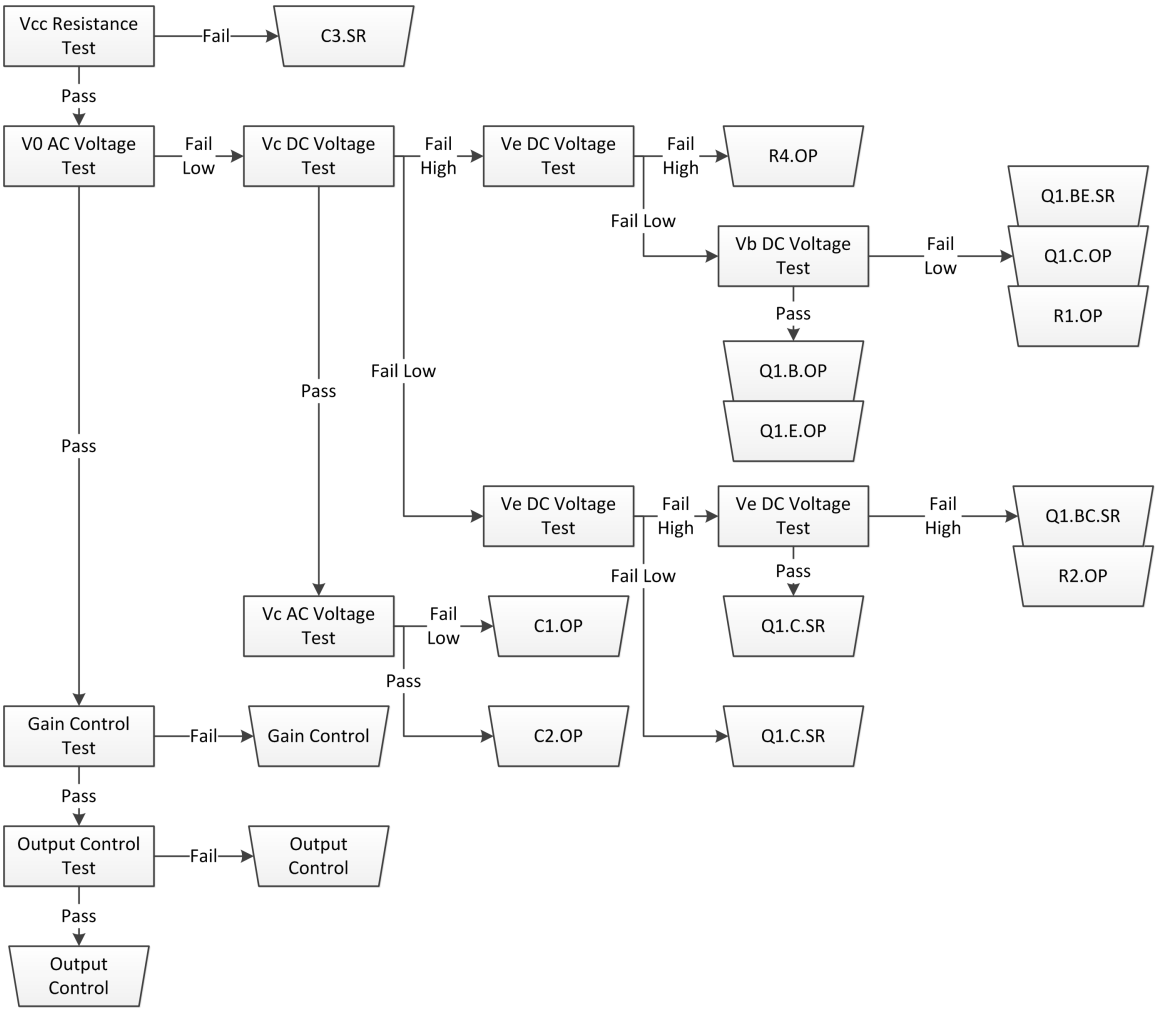


Figure 6: Sample Fault Tree of ATML Circut

We move to the V_C DC voltage test, which at 4.5 volts, fails low with a membership value of 1. Next we move to the V_E DC voltage test which at 0.6 volts fails high with a membership value of 1. Up to this point, the fuzzy fault tree has behaved like a regular fault tree because there has been a crisp outcome at each test.

The final test in this instance of the fuzzy fault tree is the V_B DC voltage test, which has a value of 1.21 volts. This does not yield a crisp outcome and is a pass with a membership value of 0.25, and a fail high with a membership value of 0.75. Since we have two possibilities with non-zero membership values, we have to enumerate both

Table 6: Values of tests from ATML fuzzy fault tree example

Test	Value	Outcome (Membership Value)
V_{CC} Resistance Test	12.5 KΩ	Pass (1)
V_0 AC Voltage Test	0.85 V	Fail (1)
V_C DC Voltage Test	4.5 V	Fail Low (1)
V_E DC Voltage Test	0.6 V	Fail High (1)
V_B DC Voltage Test	1.21 V	Pass (0.25), Fail High (0.75)

of them. The first possibility is what arises when V_B DC voltage passes, which is a diagnosis of Q1.C.SR. This diagnosis has a fuzzy membership value of 0.25, which defuzzifies based on a predefined membership function to a candidate value of 0.66. The other possible outcome arises when V_B DC fails high. In this case the diagnosis is the ambiguity group R2.OP and Q1.BC.SR, and since there is a fuzzy membership value of 0.75 for this route, this defuzzifies to candidate values of 0.72 for these two diagnoses. Thus the outcome of the fuzzy fault tree is a degredation level of 0.66 for Q1.C.SR and 0.72 for R2.OP and 0.72 for Q1.BC.SR.

If multiple tests had non-zero outcomes, we would have to combine the fuzzy membership values with a t-norm which is propagated along the path in the fuzzy fault tree. This t-norm was implicitly propagated in the example above as a 1.0 at each step until the V_B DC voltage test.

Fuzzy Bayesian Networks

Bayesian networks are very powerful tools and are used many different situations and domains. They are a useful and compact method for representing joint proba-

bility distributions. Similarly fuzzy sets are able to represent data in linguistic terms that help to improve understandability. Additionally, the fuzzy membership function provides a nice framework for representing degrees of membership in a set.

Combining these two ideas can be conceptually difficult because the meaning of a fuzzy membership value and a probability are very different, yet are represented similarly (a real number on the range [0,1]). Nevertheless, Fuzzy Bayesian Networks are not uncommon in the literature. There are many different methods for integrating these two tools presented by various authors. Many of these techniques differ from each other because they are often being used to represent different things. In addition to different techniques, nearly every work uses different notation. This can make it difficult to understand the similarities and differences between the various techniques.

To better facilitate the comparison of the techniques, a common Bayesian network will be used to illustrate the mechanisms in each method presented. Assume we have a simple network that is used to diagnose a resistor short with a test measuring the current across the resistor. This network is represented in Figure 7. This network has a test node, *Current Test* and a diagnosis node, *Resistor Short*. The test node is treated like an evidence node, and the diagnosis node is a query variable. The conditional probability tables for this Bayesian network are presented in Tables 7 and 9, as well as a plot of the fuzzy membership functions in Figure 8. The membership functions for each state are in Equations 6 and 7

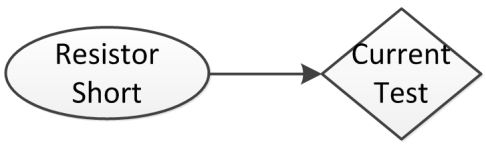


Figure 7: Simple Example Bayesian Network

Table 7: Conditional Probability Table for Resistor Short node

P(Resistor Short = True)	P(Resistor Short = False)
0.15	0.85

Table 8: Conditional Probability Table for Current Test Node

Resistor Short → CurrentTest	P(CurrentTest = High)	P(CurrentTest = Normal)
Resistor Short = True	0.99	0.01
Resistor Short = False	0.01	0.99

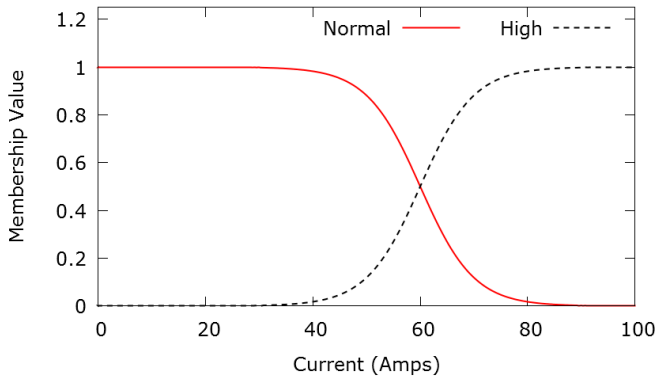


Figure 8: Membership Functions for Example Network

$$\mu_{high}(x) = \frac{1}{1 + e^{0.2x-60}} \tag{6}$$

$$\mu_{normal}(x) = 1 - \frac{1}{1 + e^{0.2x-60}} \tag{7}$$

Coefficient Method

The first technique for Fuzzy Bayesian Networks is presented in [15] and [16]. This technique applies a common approach used in many of the techniques, to weight the probability with the fuzzy membership value. The notation used in this work denotes a fuzzy set by putting a tilde over the set name. As an example, $\widetilde{A}$ is the fuzzy set corresponding to set A.

This technique uses what the authors call the "Fuzzy Bayesian equation." It supports fuzzy values on the evidence node, fuzzy values on the query node, or fuzzy values on both the evidence node and the query node. This technique combines the probabilities and fuzzy membership values into one value by multiplying the probabilities by the related fuzzy membership value.

First we consider when the evidence node is represented as a classic, crisp value and the query node is represented as a fuzzy variable.

$$P(\widetilde{A}|B) = \frac{\sum_{i \in I} \mu_{\widetilde{A}}(A_i)\, P(B|A_i)P(A_i)}{P(B)} \tag{8}$$

Where I represents the set of states in A, so i is an individual state in i. As we can see, this is very similar to the traditional Bayes' rule. The difference is that we enumerate each possibility for the variable A and weight it with the membership value of $\widetilde{A}$ for each state. This scenario would mean that we want to know the probability of each fuzzy value given crisp evidence.

Next we consider conditioning a crisp value on a fuzzy variable.

$$P(A|\widetilde{B}) = \frac{\sum_{i \in I} \mu_{\widetilde{B}}(B_i)\, P(B_i|A)P(A)}{P(\widetilde{B})} \tag{9}$$

This fuzzy Bayesian equation is similar to that from Equation 8. The primary difference is Equation 9 uses a *marginal fuzzy probability*. This marginal fuzzy probability

is given as follows:

$$P(\widetilde{X}) = \sum_{i \in I} \mu_{\widetilde{X}}(X_i) P(X_i) \tag{10}$$

Finally, we consider the case where both fuzzy values as evidence and fuzzy values on the query variable as well.

$$P(\widetilde{A}|\widetilde{B}) = \frac{\sum_{i \in I} \sum_{j \in J} \mu_{\widetilde{A}}(A_i) \mu_{\widetilde{B}}(B_j) P(B_j|A_i) P(A_i)}{P(\widetilde{B})} \tag{11}$$

This allows the use of linguistic variables on both ends of the inference process. The best way to illustrate this is to use the example network from Figure 7. If we assume the current measured across the resistor is 50 Amps, we know, based on the fuzzy membership functions from Figure 11 that $\mu_{\widetilde{T}}(\text{Normal}) = 0.881$ and $\mu_{\widetilde{T}}(\text{High}) = 0.119$. We set this as fuzzy evidence, and use Equation 9 from above to calculate $P(\text{Resistor Short} = \text{True}|\text{Current Test})$. For ease of notation we will refer to Resistor Short as R and Current Test as T. This means we will be calculating $P(R|\widetilde{T})$.

$$\begin{aligned}
P(R|\widetilde{T}) &= \frac{\sum_{i \in I} \mu_{\widetilde{T}}(T_i) P(T_i|R) P(R)}{P(\widetilde{T})} \\
&= \frac{\sum_{i \in I} \mu_{\widetilde{T}}(T_i) P(T_i|R) P(R)}{\sum_{i \in I} \mu_{\widetilde{T}}(T_i) P(T_i)} \\
&= \frac{\mu_{\widetilde{T}}(T_{\text{High}}) P(T_{\text{High}}|R) P(R) + \mu_{\widetilde{T}}(T_{\text{Normal}}) P(T_{\text{Normal}}|R) P(R)}{\mu_{\widetilde{T}}(T_{\text{High}}) P(T_{\text{High}}) + \mu_{\widetilde{T}}(T_{\text{Normal}}) P(T_{\text{Normal}})} \\
&= \frac{0.119 \cdot 0.946 \cdot 0.15 + 0.881 \cdot 0.002 \cdot 0.15}{0.119 \cdot 0.157 + 0.881 \cdot 0.843} \\
&= 0.023
\end{aligned} \tag{12}$$

Thus, according to this method, there is a probability of 0.023 of the resistor having been shorted given the results of the current test. Additionally, we can use form in Equation 8 to perform the query $P(\widetilde{R}_{\text{true}}|T_{\text{High}})$. For this example we will

assume $\mu_{\widetilde{R}}(\text{true}) = 0.31$ and $\mu_{\widetilde{R}}(\text{false}) = 0.69$. We assume the same conditional probability tables as before.

$$\begin{aligned} P(\widetilde{R}_{\text{true}}|T_{\text{High}}) &= \frac{\sum_{i\in I} \mu_{\widetilde{R}}(R_i)\, P(T_{\text{High}}|R_i)P(R_i)}{P(T_{\text{High}})} \\ &= \frac{\mu_{\widetilde{R}}(R_{\text{true}})P(T_{\text{High}}|R_{\text{true}})P(R_{\text{true}}) + \mu_{\widetilde{R}}(R_{\text{false}})P(T_{\text{High}}|R_{\text{false}})P(R_{\text{false}})}{P(T_{\text{High}})} \\ &= \frac{0.31 \cdot 0.99 \cdot 0.15 + 0.69 \cdot 0.01 \cdot 0.85}{0.157} \\ &= 0.3306 \end{aligned} \tag{13}$$

So, $P(\widetilde{R}_{\text{true}}|T_{\text{High}}) = 0.3306$ which means, given the current test was *High*, the probability that $\mu_{\widetilde{R}}(\text{true}) = 0.31$ and $\mu_{\widetilde{R}}(\text{false}) = 0.69$ is 0.3306. The final example of this technique is to use fuzzy values on both random variables. We again use the same conditional probability tables and we assume $\mu_{\widetilde{R}}(\text{true}) = 0.31$ and $\mu_{\widetilde{R}}(\text{false}) = 0.69$, as well as $\mu_{\widetilde{T}}(\text{Normal}) = 0.881$ and $\mu_{\widetilde{T}}(\text{High}) = 0.119$. To calculate $P(\widetilde{R}|\widetilde{T})$ we use the form given in equation 11.

$$
\begin{aligned}
P(\widetilde{R}|\widetilde{T}) &= \frac{\sum_{i\in I}\sum_{j\in J}\mu_{\widetilde{R}}(R_i)\mu_{\widetilde{T}}(T_j)P(T_j|R_i)P(R_i)}{P(\widetilde{T})} \\
&= \frac{\sum_{i\in I}\sum_{j\in J}\mu_{\widetilde{R}}(R_i)\mu_{\widetilde{T}}(T_j)P(T_j|R_i)P(R_i)}{\sum_{j\in J}\mu_{\widetilde{T}}(T_j)P(T_j)} \\
&= \frac{\sum_{i\in I}\sum_{j\in J}\mu_{\widetilde{R}}(R_i)\mu_{\widetilde{T}}(T_j)P(T_j|R_i)P(R_i)}{\mu_{\widetilde{T}}(T_{\text{Normal}})P(T_{\text{Normal}}) + \mu_{\widetilde{T}}(T_{\text{High}})P(T_{\text{High}})} \\
&= \frac{\sum_{i\in I}\sum_{j\in J}\mu_{\widetilde{R}}(R_i)\mu_{\widetilde{T}}(T_j)P(T_j|R_i)P(R_i)}{0.881\cdot 0.157 + 0.119\cdot 0.843} \\
&= \Big(\mu_{\widetilde{R}}(R_{\text{true}})\mu_{\widetilde{T}}(T_{\text{Normal}})P(T_{\text{Normal}}|R_{\text{true}})P(R_{\text{true}}) \\
&+ \mu_{\widetilde{R}}(R_{\text{true}})\mu_{\widetilde{T}}(T_{\text{High}})P(T_{\text{High}}|R_{\text{true}})P(R_{\text{true}}) \\
&+ \mu_{\widetilde{R}}(R_{\text{false}})\mu_{\widetilde{T}}(T_{\text{Normal}})P(T_{\text{Normal}}|R_{\text{false}})P(R_{\text{false}}) \\
&+ \mu_{\widetilde{R}}(R_{\text{false}})\mu_{\widetilde{T}}(T_{\text{High}})P(T_{\text{High}}|R_{\text{false}})P(R_{\text{false}})\Big)/0.2386 \\
&= \Big(0.31\cdot 0.881\cdot 0.99\cdot 0.15 + 0.31\cdot 0.119\cdot 0.01\cdot 0.15 \\
&+ 0.69\cdot 0.881\cdot 0.01\cdot 0.85 + 0.69\cdot 0.119\cdot 0.99\cdot 0.85\Big)/0.2386 \\
&= \frac{0.1149}{0.2386} \\
&= 0.4815
\end{aligned}
\tag{14}
$$

Where I is the set of states of $\widetilde{R}$, and J is the set of states of $\widetilde{T}$.

The primary reason we are not using this method is we need outputs of fuzzy values to represent component degradation. This method does not support the ability to output fuzzy values. This method can use fuzzy states to evaluate probabilities but the outputs are still just probabilities. Problem with this, and all FBN methods is that if the membership values do not sum to 1, then the probabilities that are produced also do not sum to 1. This is a problem because one of the axioms of probability is the assumption of unit measure, i.e., that the probability of some event happening in the entire sample space is 1. If the outcome of this network does not meet all the axioms of probability, the value is not a probability. Due to this problem,

the authors restrict the membership function to sum to 1 to help maintain the validity of the probability measures.

Virtual Evidence

The above method seems to make intuitive sense in the way it combines the probabilities and the fuzzy membership values. A large drawback of that method is it requires changes to the inference algorithm used because one of the central tools for exact inference, Bayes' rule, needs to be changed. This means that a custom inference engine must be used.

An alternative method of incorporating fuzzy membership values into a Bayesian network is to use virtual evidence, which is the technique is used in [17]. As was discussed in Chapter 2, virtual evidence is a method for incorporating uncertainty of evidence into a Bayesian network.

The process of using virtual evidence to incorporate fuzzy values into a Bayesian network is very straight forward. Once the virtual evidence node is added, fuzzy evidence is incorporated directly as virtual evidence. Virtual evidence is represented in manipulating the conditional probability table of the virtual evidence node. We illustrate this process with the example network given in Figure 7. The example network will be modified slightly to include a virtual evidence node attached to the *Current Test* node. This change can be seen in Figure 9.

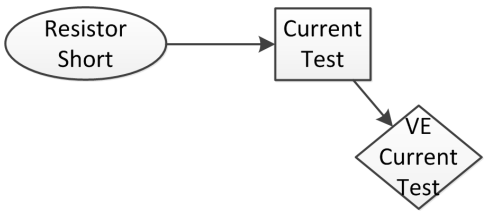


Figure 9: Simple Example Bayesian Network with a Virtual Evidence Node

Our example will assume a current measurement of 50 Amps just like in the previous example, which yields fuzzy membership values of μ(Normal) = 0.881 and μ(High) = 0.119. Since we are using the fuzzy membership values as the values in the virtual evidence node, we set 0.811 and 0.119 as the probability of evidence, and evaluate just like we did in Chapter 2. For these calculations, we use the same conditional probability tables that were used in the previous example (Tables 7 and 9). In addition to these, we also need to add the conditional probability table for the virtual evidence node (Table 9). This conditional probability table is set to match the fuzzy membership values we defined earlier. In the following calculations, we represent the virtual evidence node with *VE*, *Current Test* as *T*, and *Resistor Short* as *R*.

Table 9: Conditional Probability Table for Virtual Evidence Node

Current Test → VE Current Test	P(VE = High)	P(VE = Normal)
Current Test = High	0.119	0.881
Current Test = Normal	0.881	0.119

We can then use this information to calculate $P(R_{true}|\,T_{High})$ using the fuzzy values as virtual evidence. However, instead of using T as evidence, we are using VE as evidence, so what we are really solving for is $P(R_{true}|\,VE_{High})$.

$$\begin{aligned} P(R_{true}|\,VE_{High}) &= \frac{P(\,VE_{High}|R_{true})P(R_{true})}{P(\,VE_{High})} \\ &= \frac{P(\,VE_{High}|R_{true})P(R_{true})}{P(\,VE_{High}|\,T_{High}) + P(\,VE_{High}|\,T_{Normal})} \\ &= \frac{0.12662 \cdot 0.15}{0.119 + 0.881} \\ &= 0.0249 \end{aligned}$$

So as we can see, using fuzzy values as virtual evidence, we get $P(R_{true}|VE_{High}) = 0.0249$. This is a similar result to that achieved with the Coefficient Method presented above.

This method, similar to the Coefficient method, makes the assumption that the fuzzy membership value can be integrated directly with the probabilities in the network. However, unlike the Coefficient method that uses the membership value as a weight, this method assumes the membership value is a probability.

When we think about what virtual evidence actually is, it is a method for incorporating uncertainty of evidence into a Bayesian network. This is not exactly what the fuzzy membership value means. It is a grade of membership of that set, which is uncertainty of the state assignment, not uncertainty of the evidence.

FUZZY BAYESIAN NETWORKS

Notation

Representing both probabilities and fuzziness simultaneously requires some special notation. This notation is used in [2], [8], and [18]. A probability distribution can be represented by using curly braces and subscripted values. Assume we have a probability distribution T where there are two different states, hi and low. The individual probabilities for each state are as follows: $P(hi) = 0.6$ and $P(low) = 0.4$. This probability distribution T can be written as Equation 15.

$$T = \{hi_{0.6}, low_{0.4}\} \tag{15}$$

We can also assume the tuple ordering is fixed, which allows us to leave out the value names, so probability distribution T can be represented as: $T = \{0.6, 0.4\}$.

Similar to the notation for a probability distribution, we represent fuzzy states using square brackets and subscripted membership values. If we assume we have a fuzzy state S that has two possible fuzzy values hi and low, and have membership values of $\mu(hi) = 0.7$ and $\mu(low) = 0.3$. This fuzzy state S can be written in the form in Equation 31.

$$S = [hi_{0.7}, low_{0.3}] \tag{16}$$

Just like with the probability distribution, the notation can be reduced in size by assuming a consistent tuple ordering. We leave out the state names, so the fuzzy state S can be represented as $S = [0.7, 0.3]$.

Each of these two notions, probability distributions and fuzzy states are well understood and naturally can stand apart. The key to this method is to combine

the probability distribution and the fuzzy state without losing the information from either representation. This is done with a Fuzzy Probability Distribution, or FPD.

The two separate pieces of information, the fuzzy state and the probability distribution can then be combined using a notation that utilizes both of the notations above. This notation is for the Fuzzy Probability Distribution, which is a probability distribution that has a fuzzy state associated with it. A Fuzzy Probability Distribution on a variable X could look like the following:

$$X = [\{hi_{0.6}, low_{0.4}\}_{0.7}, \{hi_{0.4}, low_{0.6}\}_{0.3}]$$

This means that the probability distribution $\{hi_{0.6}, low_{0.4}\}$ has a fuzzy membership value of 0.7 and the probability distribution $\{hi_{0.4}, low_{0.6}\}$ has a fuzzy membership value of 0.3.

Approach

Our approach to Fuzzy Bayesian Networks is used in [18], [8], and is similar to the approach used in [2]. This approach utilizes the two distinct features, probability and fuzziness simultaneously with the Fuzzy Probability Distribution. Most other approaches (see Chapter 3) use some sort of method to combine fuzziness and probabilities. This technique is unique in that it is able to keep the two aspects separate, while still considering both of them.

One of the key aspects of this technique is the assumption that during belief propagation, the components within a variable, the fuzziness and the probabilities, should not directly interact. In a classic Bayesian network both a network structure and joint probability distribution must be defined. The joint probability distribution

must have one specific structure whereas the structure can have many joint probability distributions defined for it. Similarly, the fuzzy variables can use the structure of the network without directly influencing that structure or the probability distribution associated with it.

Our method is then able to side step the problem that plagues other techniques, how to combine fuzziness and probabilities. Our technique treats the propagation of probabilities and the propagation of fuzzy values as independent procedures that are combined with the Fuzzy Probability Distribution.

Since evidence in the FBN is represented as fuzzy values, not crisp values, more components to a variable must be propagated and kept track of. At the end of the process, the fuzzy membership values can be combined with the probabilities calculated from the Bayesian network if desired. This can be done by using a *product t-norm* or any other fuzzy conjunction.

Simple Example of a Fuzzy Bayesian Network

To illustrate, we use the same networks as in Chapter 3, shown again in Figure 10 for reference. This Bayesian network was constructed as a simple diagnostic test of a resistor. The test measures current across a resistor, and if the resistor shorts, the current will increase dramatically.

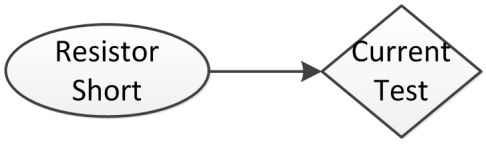


Figure 10: Simple Example Bayesian Network

Table 10: Conditional Probability Table for Resistor Short node

P(Resistor Short = True)	P(Resistor Short = False)
0.15	0.85

Table 11: Conditional Probability Table for Current Test Node

Resistor Short → CurrentTest	P(CurrentTest = High)	P(CurrentTest = Normal)
Resistor Short = True	0.99	0.01
Resistor Short = False	0.01	0.99

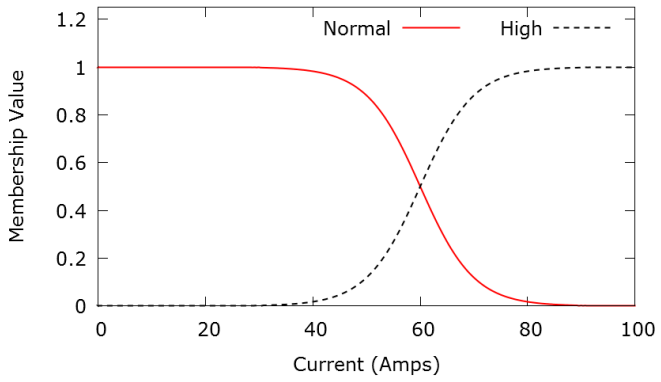


Figure 11: Membership Functions for Example Network

The *Current Test* node is the evidence node, and the *Resistor Short* node represents the query variable. We can now use this network to perform some simple, sample calculations to illustrate the usage of the FBN.

We first assume we start with a current reading of 50 Amps across the resistor. Using the membership functions μ_{High} and μ_{Normal}, we can calculate that the fuzzy membership value for *High* is $\mu_{High}(50\text{Amps}) = 0.119$ and the membership value for *Normal* is $\mu_{Normal}(50\text{Amps}) = 0.881$. We can use the notation from the previous section to write this as a fuzzy state as follows:

$$T = [high_{0.119}, normal_{0.881}] \tag{17}$$

We can now use the conditional probability tables given with the Bayesian network to calculate $P(T_{high})$ and $P(T_{normal})$.

$$\begin{aligned} P(T_{high}) &= P(R_{true})P(T_{high}|R_{true}) + P(R_{false})P(T_{high}|R_{false}) \\ &= 0.15 \cdot 0.99 + 0.85 \cdot 0.01 \\ &= 0.157 \end{aligned} \tag{18}$$

$$\begin{aligned} P(T_{normal}) &= P(R_{true})P(T_{normal}|R_{true}) + P(R_{false})P(T_{normal}|R_{false}) \\ &= 0.15 \cdot 0.01 + 0.85 \cdot 0.99 \\ &= 0.843 \end{aligned} \tag{19}$$

We can represent these values in the form for the probability distributions as follows:

$$T = \{high_{0.157}, normal_{0.843}\} \tag{20}$$

Now that we have the values from the fuzzy state in Equation 17 and the probability distribution from Equation 20, we can take the next logical step forward and calculate the fuzzy probability distribution for the node *Resistor Short*. We have all the information we need to make this calculation.

First we start by calculating the probabilities. Since we are using fuzzy data as evidence, and not crisp data we have to calculate $P(R|T = high)$ and $P(R|T = normal)$. We are only calculating the probability that there is a resistor short. We use Bayes' rule, the values we calculated from Equations 18, 19, and the conditional probabilities found in Tables 14 and 11.

$$P(R = true|T = high) = \frac{P(T = high|R = true)P(R = true)}{P(T = high)}$$
$$= \frac{0.99 \cdot 0.15}{0.157} \tag{21}$$
$$= 0.945$$

$$P(R = true|T = normal) = \frac{P(T = normal|R = true)P(R = true)}{P(T = normal)}$$
$$= \frac{0.01 \cdot 0.15}{0.843} \tag{22}$$
$$= 0.002$$

$$P(R = false|T = high) = \frac{P(T = high|R = false)P(R = false)}{P(T = high)}$$
$$= \frac{0.01 \cdot 0.85}{0.157} \tag{23}$$
$$= 0.054$$

$$P(R = false|T = normal) = \frac{P(T = normal|R = false)P(R = false)}{P(T = normal)}$$
$$= \frac{0.01 \cdot 0.85}{0.843} \tag{24}$$
$$= 0.998$$

So far, the probability calculations have been pretty standard except that we calculated the probabilities for every possible set of crisp evidence. Otherwise, nothing special has been done. Next we need to propagate the fuzzy state from the *Current Test* node (Equation 17) to the *Resistor Short* node, similar to what we did with the probabilities.

Since there is no new fuzzy information to incorporate at the *Resistor Short* node that is not already contained in the fuzzy state from the *Current Test* node, we can just apply the fuzzy state directly to the probability distribution calculated in Equations 21, 22, 23, and 24. This then results in a Fuzzy Probability Distribution at the *Resistor Short* node of

$$
\begin{aligned}
R = \Big[& \{P(R = true|T = high), P(R = false|T = high)\}_{\mu_{high}(50\text{Amps})}, \\
& \{P(R = true|T = normal), P(R = false|T = normal)\}_{\mu_{normal}(50\text{Amps})} \Big] \\
= \Big[& \{true_{0.945}, false_{0.054}\}_{0.119}, \{true_{0.002}, false_{0.998}\}_{0.881} \Big]
\end{aligned} \tag{25}
$$

Finally, once the Fuzzy Probability Distribution has been obtained at the query node, we can reduce this to a fuzzy expected value. This fuzzy expected value is calculated using a product t-norm of each component, yielding a fuzzy expected value for the *Resistor Short* node of:

$$
\begin{aligned}
R &= \Big[\{true_{0.945}, false_{0.054}\}_{0.119}, \{true_{0.002}, false_{0.998}\}_{0.881} \Big] \\
&= \Big[true_{0.945\times0.119+0.002\times0.881}, false_{0.054\times0.119+0.998\times0.881} \Big] \\
&= \Big[true_{0.114}, false_{0.886} \Big]
\end{aligned} \tag{26}
$$

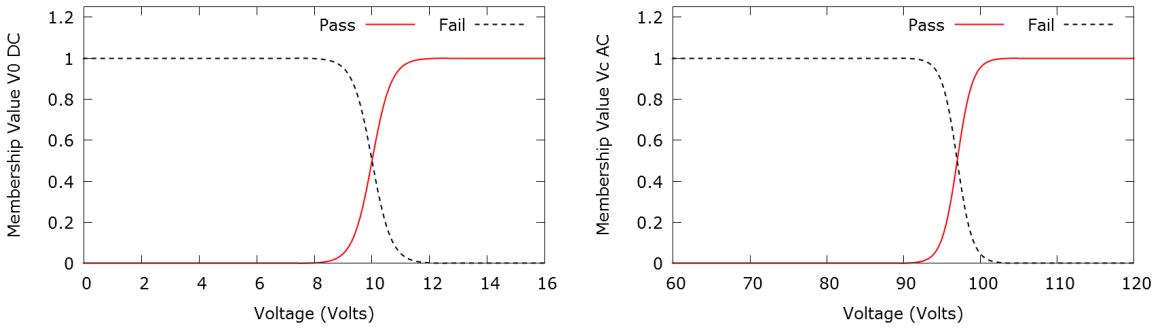
This example is fairly simple because the fuzzy values do not need to be combined with any others. The more interesting situation arises when there are multiple fuzzy events that contribute to an outcome. To illustrate this, we use a slightly more complex example.

This example uses a subset of the ATML network (Figure 13). The full network is presented in Appendix A. This network uses two voltage measurements. One of these measurements is a DC voltage (V_0 *DC*) and the other is of an AC measurement (V_C *AC*) at different parts of the ATML circuit. These two are related to the capacitor *C2* failing open.

The failure condition for V_C *AC* is a low voltage. Due to this, we use the fuzzy membership function in Equation 27, which is shown in Figure 12b. The failure condition for V_0 *DC* is a high voltage. Because of this, we use the fuzzy membership function in Equation 28. This membership function is mapped in Figure 12a.

$$\mu_{V_C\ AC\ =\ Pass}(x) = 1 - \frac{1}{1 + e^{x-97}}$$
$$\mu_{V_C\ AC\ =\ Fail}(x) = \frac{1}{1 + e^{x-97}} \tag{27}$$

$$\mu_{V_0\ DC\ =\ Pass}(x) = \frac{1}{1 + e^{3\cdot(x-10)}}$$
$$\mu_{V_0\ DC\ =\ Fail}(x) = 1 - \frac{1}{1 + e^{3\cdot(x-10)}} \tag{28}$$



(a) Membership functions for $V_0\ DC$ (b) Membership functions for $V_C\ AC$

Figure 12: Membership Functions for Small ATML Network

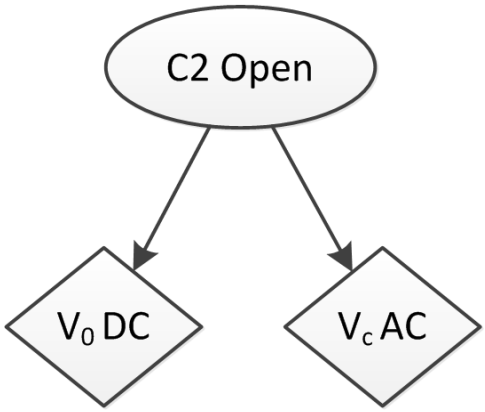


Figure 13: Subset of the ATML Networks

Table 12: Conditional Probability Table for $V_C AC$ node

C2 Open $\rightarrow V_C AC$	P($V_C AC$ = Pass)	P($V_C AC$ = Fail)
C2 Open = Good	1	0
C2 Open = Candidate	0.5	0.5

Table 13: Conditional Probability Table for $V_0 DC$ node

C2 Open $\rightarrow V_0 DC$	P($V_0 DC$ = Pass)	P($V_0 DC$ = Fail)
C2 Open = Good	1	0
C2 Open = Candidate	0.5	0.5

Table 14: Conditional Probability Table for C2 Open node

P(C2 Open = Good)	P(C2 Open = Candidate)
0.9896	0.0104

In this example we will assume the measurement taken for test $V_C\ AC$ is 99 Volts AC and the measurement taken for the test $V_0\ DC$ is 9.5 Volts DC. Using Equations 28 and 27 we get the membership values of:

$$\begin{aligned} \mu_{V_C\ AC\ =\ Pass}(99\text{Volts}) = 0.1824 \quad \mu_{V_C\ AC\ =\ Fail}(99\text{Volts}) = 0.8176 \\ \mu_{V_0\ DC\ =\ Pass}(9.5\text{Volts}) = 0.8808 \quad \mu_{V_0\ DC\ =\ Fail}(9.5\text{Volts}) = 0.1192 \end{aligned} \tag{29}$$

For now, we set these membership values aside and calculate the probabilities for $P(C2\ Open | V_0\ DC,\ V_C\ AC)$. We need to calculate the probabilities for each of the possible permutations of $C_2\ Open,\ V_0\ DC$ and $V_C\ AC$ using standard Bayesian inference.

To find the fuzzy state for the node *C2 Open*, we is to enumerate all possible state assignments for all of the variables involved. The fuzzy probability distribution for

C2 Open is calculated in Equation 30.

$$
\begin{aligned}
C2\ Open = \Big[& \{P(C2 = Good | V0DC = Pass,\ VCAC = Pass), \\
& P(C2 = Candidate | V0DC = Pass,\ VCAC = Pass)\}_{\mu(V0DC = Pass)\mu(VCAC = Pass)}, \\
& \{P(C2 = Good | V0DC = Fail,\ VCAC = Pass), \\
& P(C2 = Candidate | V0DC = Fail,\ VCAC = Pass)\}_{\mu(V0DC = Fail)\mu(VCAC = Pass)}, \\
& \{P(C2 = Good | V0DC = Pass,\ VCAC = Fail), \\
& P(C2 = Candidate | V0DC = Pass,\ VCAC = Fail)\}_{\mu(V0DC = Pass)\mu(VCAC = Fail)}, \\
& \{P(C2 = Good | V0DC = Fail,\ VCAC = Fail), \\
& P(C2 = Candidate | V0DC = Fail,\ VCAC = Fail)\}_{\mu(V0DC = Fail)\mu(VCAC = Fail)} \Big]
\end{aligned}
$$

$$
\begin{aligned}
C2\ Open &= \Big[\{0.9974, 0.0026\}_{0.8808 \cdot 0.1824}, \{0,1\}_{0.8808 \cdot 0.8176}, \{0,1\}_{0.1192 \cdot 0.1824}, \{0,1\}_{0.1192 \cdot 0.8176} \Big] \\
&= \Big[\{0.9974, 0.0026\}_{0.1607}, \{0,1\}_{0.7201}, \{0,1\}_{0.0217}, \{0,1\}_{0.0974} \Big]
\end{aligned} \tag{30}
$$

Finally, now that we have the fuzzy probability distribution for the query node, we need to collapse it into a single fuzzy state. This is done in equation 31.

$$
\begin{aligned}
C2\ Open = \Big[& 0.9974 \cdot 0.1607 + 0 \cdot 0.7201 + 0 \cdot 0.0217 + 0 \cdot 0.0974, \\
& 0.0026 \cdot 0.1607 + 1 \cdot 0.7201 + 1 \cdot 0.0217 + 1 \cdot 0.0974 \Big] \\
= & \big[0.1603, .8396 \big]
\end{aligned} \tag{31}
$$

As is pretty easy to see, even with this small example, the number of calculations needed to perform inference on a fuzzy Bayesian network can be excessive. Due to this problem, we discuss a few methods to reduce the complexity of this process.

Complexity Reduction Techniques

As can be seen in the last example, there is a definite potential for an exponential explosion in complexity when using this technique. In general, a random variable that has k parents and each of the k parents has a fuzzy state with m components, the updated fuzzy state will be of size m^k. Then assuming all the variables have k parents, the grandchildren will have an updated fuzzy state of size m^{k^k} [8]. This is referred to as a *fuzzy state size explosion* or FSSE [18].

To combat the FSSE, four different methods are presented in [8]. The first of which is a process of removing components that do not have a substantial impact on the overall computation. This can be done by removing fuzzy states that have small membership values from the calculation, then normalizing the remainder of the fuzzy values. While this technique would reduce the amount of computation required, it would not reduce it substantially. Additionally, we would now need to set the thresholds as to what is a *minor* impact. However, more significantly, this will remove some of the expressive power of the Fuzzy Bayesian Network. Finally, this technique does not work well when there are only two states per variable, because removing one results in a crisp state, which defeats the purpose of the FBN. For these reasons, we chose not to use this method to reduce complexity.

In the second technique presented, the full fuzzy probability distribution is calculated for each node; however, before using the full FPD to update the states of the next level of nodes, the components are clustered so that FPDs that specify similar distributions are combined. Similar to the first technique, important information would be discarded. The third technique is to use approximate inference such as

Markov Chain Monte Carlo methods. This was discounted because we wanted to use exact inference to get a better sense of the meaning of the results.

The final technique is called linear collapse, and the process collapses an FPD into a fuzzy state, which is then used as evidence in the next layer of computation. The new evidence is determined by weighting each component by its fuzzy membership value. The results are then summed to create a single fuzzy state, and the process repeats until the query node is reached. We can represent this process mathematically in Equations 32 and 33. The first step is to create a set, A, of all the combinations of the previous layers' nodes' states. This is represented in Equation 32 for nodes B and C with states t and f.

$$A = \{\{B_t, C_t\}, \{B_f, C_t\}, \{B_t, C_f\}, \{B_f, C_f\}\} \tag{32}$$

We then use Equation 33 to collapse the fuzzy probability distribution for variable Q with n states.

$$FS = \left[\sum_{i=0}^{|A|} P(Q_1 \mid A_i) \cdot \prod_{j=0}^{|A_i|} \mu\left(A_{i_j}\right), ..., \sum_{i=0}^{|A|} P(Q_n \mid A_i) \cdot \prod_{j=0}^{|A_i|} \mu\left(A_{i_j}\right)\right] \tag{33}$$

The primary problem with this approach is that it confuses fuzzy values with the state probabilities, which strictly speaking is not correct. While all four methods yield approximations, we felt the fourth was to be preferred because we wanted to use exact Bayesian inference and avoid arbitrary information loss.

Effects of Fuzzy Membership Functions

Fuzzy membership functions are what allow the mapping of real world measurements like voltage and current to fuzzy membership values for use in the FBN. The choice of these functions has a large impact on how the model will behave.

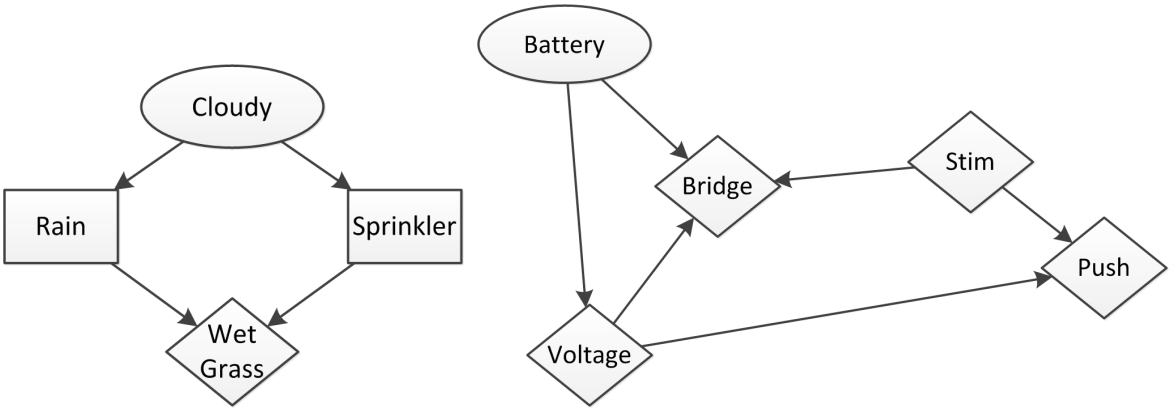
With traditional fuzzy membership functions, it is not necessary for all membership values to sum to 1 [6]. In this work we make the assumption that the fuzzy membership values *do* sum to 1. This is a fairly typical assumption to make when dealing with Fuzzy Bayesian Networks [16][8][2][18][19]. This assumption should not be too restrictive when a domain expert is designing a system because we typically think in terms of total membership anyway (membership values sum to 1).

If this assumption does not hold, the linear collapse could yield non-valid values for a fuzzy state by producing a number that is larger than 1, or make it impossible for numbers to be produced that sum to 1. Both of these situations violate the requirements of a fuzzy value. To prevent this from happening, in our implementation of this technique, all membership values are normalized to ensure they sum to 1 before they are used in any calculations.

Sequential Calculations

The linear collapse process requires sets of sequential calculations that start at a fuzzy evidence node and work towards the query node, propagating the fuzzy probability distribution, or in the case of linear collapse, the fuzzy state, to the next level of nodes, then repeating the process until the query node is reached. Propagation on the network presented in Figure 14a is pretty straightforward when *Wet Grass* is an evidence node, and *Cloudy* is the query node. First *Wet Grass* updates *Rain* and *Sprinkler*. Then the hidden nodes *Rain* and *Sprinkler* update the query node *Cloudy*.

The order of updating becomes less clear when paths to the query node are of unequal length. For example, in Figure 14b, what is the proper order of update assuming that *Bridge*, *Stim*, *Voltage* and *Push* are all fuzzy evidence nodes and



(a) Example of a balanced network (b) Example of an unbalanced network

Figure 14: Examples of possible network strucures

Battery is the query node? One possibility for updating is that *Push* should update *Voltage* and *Stim*. Then *Stim* and *Voltage* should update *Bridge*, then finally *Voltage* and *Bridge* would update *Battery*.

Another possible solution would be to have *Stim* update *Bridge* and *Push*. Next *Bridge* and *Push* would update *Voltage*, then finally *Bridge* and *Voltage* would update *Battery*. Each of these possible paths could yield different values for *Battery*. Due to this ambiguity, a method for ordering the execution must be defined.

We developed an approach for consistent update as follows. First we find the node that has the longest, shortest path to the query node. In this example it is a tie between *Stim* and *Push*, each of which have a distance of 2 from *Battery*. If these nodes were not connected to each other, they would both serve as the first step in execution, but since *Push* is the child of *Stim*, we give priority to the parent node, and have a step where the parent updates the child that is at the same depth. So the first execution step will be *Stim* updates *Push*.

Once all nodes at a particular level are updated, the next step has all the nodes at the same depth update the nodes that have a depth one less then their own. So

Stim and *Push* update *Bridge* and *Voltage* respectively. Now, we are finished with the nodes *Stim* and *Push* and have moved up a layer to the layer of all nodes that have a distance of 1 from the query node.

Once again, there is a dependency inside this layer, so this needs to be resolved before continuing on. Since we are giving priority to the parent node, we use *Voltage* to update *Bridge*. Once this has been done, we use all the nodes in this layer to update the nodes in the layer above it. In this case, there is only one node in the layer above. This node is *Battery*, which is the query node. Once the query node has been updated, the inference process is complete.

We also provide pseudo-code in Algorithms 1 and 2. Algorithm 1 is called by passing in a network structure G and a query node q. The process starts by finding the depth of the node that is furthest from the query node. The nodes which are distance i from the query node are passed into Algorithm 2 which will build each execution level using the given nodes as evidence nodes.

Within Algorithm 2, all the children of all of the evidence nodes are found. Then this list is iterated over. If a child of a node is also in the list of evidence nodes, then there is a dependency within the current level. To resolve this, the function calls itself with the conflicting node as the evidence node. This will return either a step, or a queue of steps which are added to the queue of steps that is currently being calculated. Finally, all the parents of the evidence nodes are added to list Q, and the step is created by using E as the evidence nodes, and the nodes Q as the query nodes at the particular level.

Finally, the queue is returned to Algorithm 1 which adds it to the queue. The process is then repeated, decrementing i until the depth reaches 1. Once this happens the execution queue has been fully assembled and is returned.

Algorithm 1: BuildExecutionQueue

Data: G network structure, q query node

Result: O queue of execution steps

begin

    for $i = MaxDepthFrom(q)$ **to** *1* **do**

        O.enqueue(BuildExecutionLevel(G, G.nodesAtDistance(i), O))

return O;

Algorithm 2: BuildExecutionLevel

Data: G network structure, E evidence nodes at level, O queue already built

Result: S queue of execution steps

begin

    for $e \in E$ **do**

        for $c \in e.children$ **do**

            if $c \notin O$ **then**

                $Q \longleftarrow c$;

    for $e \in E$ **do**

        if $e \in Q$ **then**

            S.enqueue(BuildExecutionLevel(G, e,O))

    for $e \in E$ **do**

        for $p \in e.parents$ **do**

            if $p \notin O$ **then**

                $Q \longleftarrow p$;

    S.evidenceNodes $\longleftarrow E$

    S.queryNodes $\longleftarrow Q$

return O

The notion of priority is handled in the if statement in Algorithm 2. Since there needs to be some method of assigning priority to nodes because we have to choose which one to evaluate first. Inference can flow in either direction, and priority could be set in a problem specific method.

This ordering is by no means the only valid ordering. In the future we hope to further investigate this execution ordering and try to better understand the effects of the choices made when defining the order.

Combining Fuzzy Evidence

In the previous section the idea of combining fuzzy states is prevalent. When propagating fuzzy probability distributions, this can be done by merging them together. Also, when the path from the fuzzy evidence nodes to the query node only has hidden nodes, like in Figure 14a, the fuzzy states can be applied directly, since there is no evidence present at the hidden nodes.

The situation changes when dealing with a network like the one presented in Figure 14b. In this network, fuzzy evidence nodes interact with other evidence nodes, each of which influence the state further up the execution line. There are three possible approaches to addressing these conflicts.

The first approach is to ignore evidence on nodes that need to be updated and over-write with the evidence from nodes that came before in the execution order. For example, with Figure 14b and the example ordering from the previous section, the fuzzy values for *Stim* would propagate to *Push*, which would eliminate the evidence assigned to the node *Push*. This is not a good solution because the only evidence that would affect effecting the query would be the fuzzy evidence applied at *Stim*.

The second approach, which is only slightly better, would be to ignore updated fuzzy states if a node has evidence. An example of this would be if both *Bridge* and *Stim* have fuzzy evidence. When *Stim* propagates its fuzzy state to *Bridge*, *Bridge* will ignore that propagated state because *Bridge* already has a fuzzy state set by evidence. This method is not much better then the previously presented one. In the example we have been using, *Voltage* and *Bridge* would be the only fuzzy evidence to have an impact on the query node.

Given the uncertainty in the evidence, the best approach is to combine the fuzzy states to make a new fuzzy state that incorporates both the evidence given for that node and the fuzzy state that is being propagated to it. To do this, we apply a fuzzy union operator.

Typically the fuzzy union operator is defined as $\mu_{A\cup B}(x) = \max\{\mu_A(x), \mu_B(x)\}$. However, we wanted to be able to incorporate both fuzzy states, so we used an alternate fuzzy union operator as follows:

$$\mu_{A\cup B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x)\mu_B(x)$$

This fuzzy union, after normalization, incorporates both fuzzy states into one unified fuzzy state which can then be used to continue the propagation process.

Detailed Example

In this section, we give a full example of the inference process using a diagnostic network for a simple doorbell. The structure of the network can be seen in Figure 15.

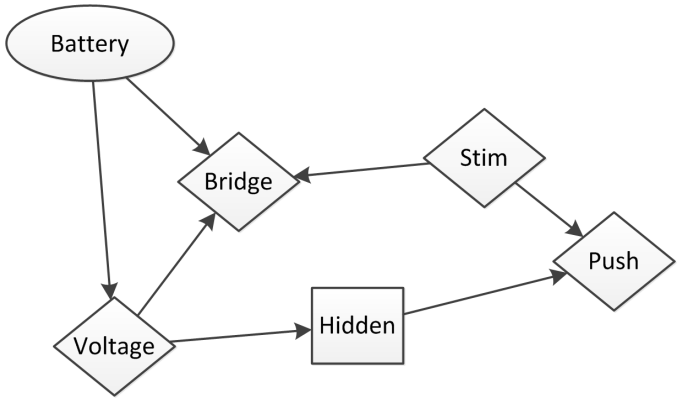


Figure 15: Doorbell network with a hidden node

For this example we will assume the fuzzy states for each evidence node are as follows:

$$\begin{aligned} \mathit{Bridge} &= [0.1, 0.9] \qquad \mathit{Voltage} = [0.2, 0.8] \\ \mathit{Stim} &= [0.7, 0.3] \qquad \mathit{Push} = [0.6, 0.4] \end{aligned} \tag{34}$$

Since we want the fuzzy state at the query node *Battery*, the first step is to determine the computation order.

We can see that the deepest node is *Push*. This will be the first node to use, and it will update the states of the evidence node *Stim* and the hidden node *Hidden* at the next layer up. Since there is no inter dependence at this layer there is nothing that needs to be resolved, and computation can move up to the next layer. However, since there is a dependence in this next layer, *Voltage* will be updated first, then *Bridge* will be updated with *Stim* and *Bridge*. Finally *Voltage* and *Bridge* will update *Battery* to get the fuzzy state for the desired query node.

The first calculations needed are to propagate the fuzzy state from *Push* to the nodes *Stim* and *Hidden*. The propagation to the node *Hidden* is performed in equation 35 and the propagation to the node *Stim* is done in equaton 36.

$$
\begin{aligned}
Hidden = & \\
& \Big[\{P(Hidden = Pass|Push = Pass), P(Hidden = Fail|Push = Pass)\}_{\mu(Push\ =\ Pass)}, \\
& \{P(Hidden = Pass|Push = Fail), P(Hidden = Fail|Push = Fail)\}_{\mu(Push\ =\ Fail)}\Big] \\
& = [\{0.1716, 0.8284\}_{0.6}, \{0.9999, 0.0001\}_{0.4}] \\
& = [0.1716 \cdot 0.6 + 0.9999 \cdot 0.4, 0.8284 \cdot 0.6 + 0.0001 \cdot 0.4] \\
& = [0.5029, 0.4971]
\end{aligned}
\tag{35}
$$

$$
\begin{aligned}
Stim = & \\
& \Big[\{P(Stim = Pass|Push = Pass), P(Stim = Fail|Push = Pass)\}_{\mu(Push\ =\ Pass)}, \\
& \{P(Stim = Pass|Push = Fail), P(Stim = Fail|Push = Fail)\}_{\mu(Push\ =\ Fail)}\Big] \\
& = [\{0.9999, 0.0001\}_{0.6}, \{0.9999, 0.0001\}_{0.4}] \\
& = [0.9999 \cdot 0.6 + 0.9999 \cdot 0.4, 0.0001 \cdot 0.6 + 0.0001 \cdot 0.4] \\
& = [0.9999, 0.0001]
\end{aligned}
\tag{36}
$$

The result from equation 35 is now the fuzzy state of *Hidden*, but the result of equation 36 is not the fuzzy states of *Stim*. Since *Stim* has fuzzy evidence of its own, the two fuzzy states need to be combined with the fuzzy union to get the actual fuzzy state for *Stim*. The fuzzy state for *Pass* is calculated in equation 37 and the FS for *Fail* is calculated in equation 38.

$$
\begin{aligned}
\mu(Stim\ =\ Pass) &= 0.7 + 0.9999 - 0.7 \cdot 0.9999 \\
&= 0.99997
\end{aligned}
\tag{37}
$$

$$\begin{aligned}\mu(Stim = Fail) &= 0.3 + 0.0001 - 0.3 \cdot 0.0001 \\ &= 0.30007\end{aligned} \tag{38}$$

These values need to be normalized, because they sum to 1.30004. The final, updated FS for *Stim* is then:

$$\begin{aligned}Stim &= \left[\frac{0.99997}{0.99997 + 0.30007}, \frac{0.30007}{0.99997 + 0.30007}\right] \\ &= [0.7692, 0.2308]\end{aligned} \tag{39}$$

Now that we have finished this level, we to the next level and since there is an inter-dependence at the next level, we update *Voltage* first with the fuzzy state from *Hidden*:

$$\begin{aligned}Voltage = & \\ &\Big[\{P(Voltage = Pass|Hidden = Pass), P(Voltage = Fail|Hidden = Pass)\}_{\mu(Hidden = Pass)}, \\ &\{P(Voltage = Pass|Hidden = Fail), P(Voltage = Fail|Hidden = Fail)\}_{\mu(Hidden = Fail)}\Big] \\ &= [\{0.9985, 0.0015\}_{0.5029}, \{0.9847, 0.0153\}_{0.4971}] \\ &= [0.9985 \cdot 0.5029 + 0.9847 \cdot 0.4971, 0.0015 \cdot 0.5029 + 0.0153 \cdot 0.4971] \\ &= [0.9916, 0.0084]\end{aligned} \tag{40}$$

To update the fuzzy state for *Voltage* we need to take the fuzzy union of this value and the fuzzy evidence, then normalize. This is done in equations 41, 42 and 43, as follows:

$$\begin{aligned}\mu(Voltage = Pass) &= 0.2 + 0.9916 - 0.2 \cdot 0.9916 \\ &= 0.9933\end{aligned} \tag{41}$$

$$\begin{aligned} \mu(\mathit{Voltage} = \mathit{Fail}) &= 0.8 + 0.0084 - 0.8 \cdot 0.0084 \\ &= 0.8017 \end{aligned} \tag{42}$$

$$\begin{aligned} \mathit{Voltage} &= \Big[\frac{0.9933}{0.9933 + 0.8017}, \frac{0.8017}{0.9933 + 0.8017}\Big] \\ &= [0.5534, 0.4466] \end{aligned} \tag{43}$$

Equation 43 now contains the fully updated fuzzy state for *Voltage*. Next we calculate the fuzzy state for *Bridge* using *Voltage* and *Stim*. The propagated fuzzy state is calculated in equation 44.

$$
\begin{aligned}
Bridge = & \\
& \Big[\{P(Bridge = Pass | Stim = Pass, Voltage = Pass), \\
& P(Bridge = Fail | Stim = Pass, Voltage = Pass)\}_{\mu(Stim\ =\ Pass)\cdot\mu(Voltage=Pass)}, \\
& \{P(Bridge = Pass | Stim = Pass, Voltage = Fail), \\
& P(Bridge = Fail | Stim = Pass, Voltage = Fail)\}_{\mu(Stim\ =\ Pass)\cdot\mu(Voltage=Fail)}, \\
& \{P(Bridge = Pass | Stim = Fail, Voltage = Pass), \\
& P(Bridge = Fail | Stim = Fail, Voltage = Pass)\}_{\mu(Stim\ =\ Fail)\cdot\mu(Voltage=Pass)}, \\
& \{P(Bridge = Pass | Stim = Fail, Voltage = Fail), \\
& P(Bridge = Fail | Stim = Fail, Voltage = Fail)\}_{\mu(Stim\ =\ Fail)\cdot\mu(Voltage=Fail)}\Big] \\
& = \Big[\{0.9999, 0.0001\}_{0.7692\cdot 0.5534}, \{0.0040, 0.9969\}_{0.7692\cdot 0.4466}, \\
& \{0.0090, 0.9910\}_{0.2308\cdot 0.5534}, \{0.0010, 0.9990\}_{0.2308\cdot 0.4466}\Big] \\
& = \Big[\{0.9999, 0.0001\}_{0.4257}, \{0.0040, 0.9969\}_{0.3435}, \\
& \{0.0090, 0.9910\}_{0.1277}, \{0.0010, 0.9990\}_{0.1031}\Big] \\
& = \Big[0.9999 \cdot 0.4257 + 0.0040 \cdot 0.3435 + 0.0090 \cdot 0.1277 + 0.0010 \cdot 0.1031, \\
& 0.0001 \cdot 0.4257 + 0.9969 \cdot 0.3435 + 0.9910 \cdot 0.1277 + 0.9990 \cdot 0.1031\Big] \\
& = [0.428, .572]
\end{aligned}
\tag{44}
$$

We need to take the union of the fuzzy state we just calculated with the fuzzy state presented to *Bridge* as evidence.

$$
\begin{aligned}
\mu(Bridge\ =\ Pass) &= 0.1 + 0.428 - 0.1 \cdot 0.428 \\
&= 0.4852
\end{aligned}
\tag{45}
$$

$$\begin{aligned} \mu(Bridge = Fail) &= 0.9 + 0.572 - 0.9 \cdot 0.572 \\ &= 0.9572 \end{aligned} \tag{46}$$

$$\begin{aligned} Bridge &= \left[\frac{0.4852}{0.4852 + 0.9572}, \frac{0.9572}{0.4852 + 0.9572}\right] \\ &= [0.3364, 0.6636] \end{aligned} \tag{47}$$

Finally, we update the fuzzy state for *Battery*, which is the query node, based on the fuzzy states we calculated for *Voltage* and *Bridge*.

$$
\begin{aligned}
Battery = & \\
& \Big[\{P(Battery = Pass | Bridge = Pass, Voltage = Pass), \\
& P(Battery = Fail | Bridge = Pass, Voltage = Pass)\}_{\mu(Bridge\ =\ Pass)\cdot\mu(Voltage=Pass)}, \\
& \{P(Battery = Pass | Bridge = Pass, Voltage = Fail), \\
& P(Battery = Fail | Bridge = Pass, Voltage = Fail)\}_{\mu(Bridge\ =\ Pass)\cdot\mu(Voltage=Fail)}, \\
& \{P(Battery = Pass | Bridge = Fail, Voltage = Pass), \\
& P(Battery = Fail | Bridge = Fail, Voltage = Pass)\}_{\mu(Bridge\ =\ Fail)\cdot\mu(Voltage=Pass)}, \\
& \{P(Battery = Pass | Bridge = Fail, Voltage = Fail), \\
& P(Battery = Fail | Bridge = Fail, Voltage = Fail)\}_{\mu(Bridge\ =\ Fail)\cdot\mu(Voltage=Fail)}\Big] \\
& = \Big[\{0.9999, 0.0001\}_{0.3364\cdot 0.5534}, \{0.0040, 0.9960\}_{0.3364\cdot 0.4466} \\
& \{0.8001, 0.1999\}_{0.6636\cdot 0.5534}, \{0.0020, 0.9980\}_{0.6636\cdot 0.4466}\Big] \\
& = \Big[\{0.9999, 0.0001\}_{0.1862}, \{0.0040, 0.9960\}_{0.1502} \\
& \{0.8001, 0.1999\}_{0.3672}, \{0.0020, 0.9980\}_{0.2964}\Big] \\
& = \Big[0.9999 \cdot 0.1862 + 0.0040 \cdot 0.1502 + 0.8001 \cdot 0.3672 + 0.0020 \cdot 0.2964, \\
& 0.0001 \cdot 0.1862 + 0.9960 \cdot 0.1502 + 0.1999 \cdot 0.3672 + 0.9980 \cdot 0.2964\Big] \\
& = [0.4812, 0.5188]
\end{aligned}
\tag{48}
$$

In the end, we find the final fuzzy state of *Battery* to be $[0.4812, 0.5188]$. An interesting caveat of this technique can be seen in the last step. The way we combine fuzzy states within a layer causes a non-linear relationship to be formed at the next level. For example, the combination of fuzzy states from from *Voltage* to *Bridge* causes a non-linear relationship to be formed between *Voltage* and *Battery*.

In the example just defined, the fuzzy state of *Bridge* becomes a function of the fuzzy states of both *Stim* and *Voltage*. Then, when the fuzzy state for *Battery* is calculated, the fuzzy values for *Voltage* and *Battery* are multiplied together which is analogous to multiplying a value x by a function of x, $f(x)$. The result is therefore non-linear with respect to x.

EXPERIMENTS

Experimental Design

In this chapter we show that this method works by using real diagnostic networks that have been used in other literature to demonstrate how this system actually behaves when being used in a model. For this work, we used three networks from the PHM literature [1], [20], and [21].

Doorbell Circuit

The doorbell circuit is possibly one of the simplest circuits that can be used. Due to the simplicity it is useful for simple demonstrations of diagnostic reasoners. In [21] this circuit was tested using both a fault tree and a traditional Bayesian network. We used the same Bayesian network except with the addition of fuzzy membership functions to produce levels of system degradation. The circuit is shown in Figure 16.

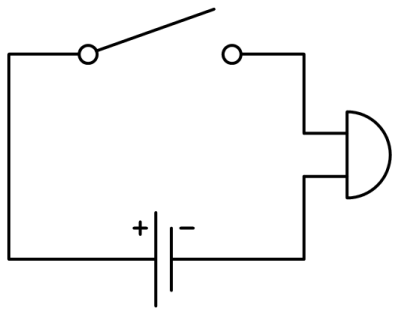


Figure 16: Circuit diagram for the doorbell test circuit

The simple doorbell circuit contains three main components: a clapper-type bell, a push button switch, and a battery. Each of the components listed can fail in specific ways. The battery could die, the switch may fail to close the circuit or get stuck closed

after the second push, the clapper on the bell can get stuck open causing the bell not to ring, and finally the solenoid on the bell can get stuck.

Each of these failure modes is represented in the Bayesian network as random variables: *Batt*, *SW-O*, *SW-C*, *Clap-O*, *Clap-C*, and *Sol-O* respectively. These are the possible diagnoses which will be represented by the Fuzzy Bayesian Network in Figure 17. The conditional probability tables are in Appendix A since no example calculations will be done here, they are not necessary in the text.

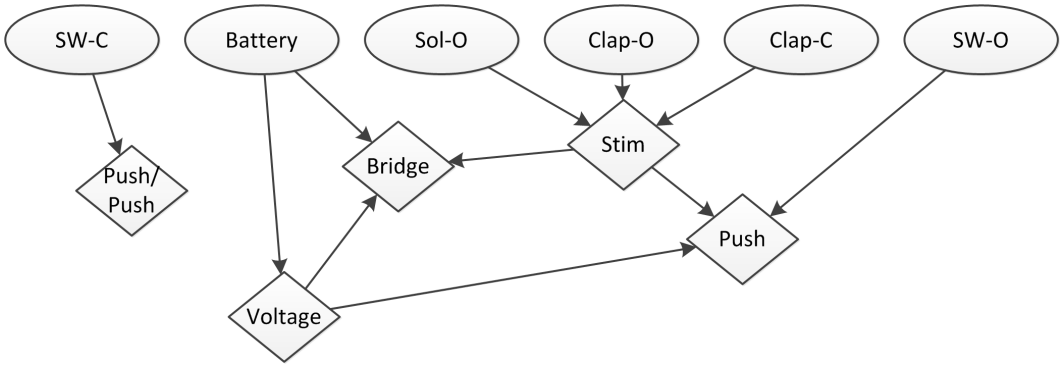


Figure 17: Doorbell diagnostic Bayesian network structure

There are five different tests that can be performed on this circuit to use as inputs to the diagnostic reasoner. The switch can be pushed, which is represented by the *Push* evidence node. The switch can be pushed twice in succession, which is represented by the *Push/Push* evidence node. The voltage of the battery can be measured, which is represented by *Voltage*. The bell can be stimulated directly, which is represented in the variable *Stim*. Finally, the switch can be bridged to forcibly close the circuit, which is then represented by the variable *Bridge*.

The tests *Push/Push*, *Bridge*, *Stim*, and *Push* typically only have natural *Pass* or *Fail* conditions. For example, when the switch is pushed, the bell usually rings or it does not. There is a situation where there could be a delay in the ring, or the ring could be weak, which could indicate a low battery voltage. However, we are limiting

the tests to what we consider to be the most natural application of fuzzy evidence, which is on the *Voltage* test. The *Voltage* test is a natural fit for a fuzzy membership function to map its voltage into the Bayesian network.

To create the fuzzy membership function for the *Voltage* test we assume the circuit uses a 9 volt battery. With this assumption, we use the following fuzzy membership function, which is plotted in Figure 18. Additionally, the conditional probability tables for this network are in Appendix B.

$$\begin{aligned} \mu_{Battery\,=\,Pass}(x) &= 1 - \frac{1}{1+e^{3(x-5.75)}} \\ \mu_{Battery\,=\,Fail}(x) &= \frac{1}{1+e^{3(x-5.75)}} \end{aligned} \tag{49}$$

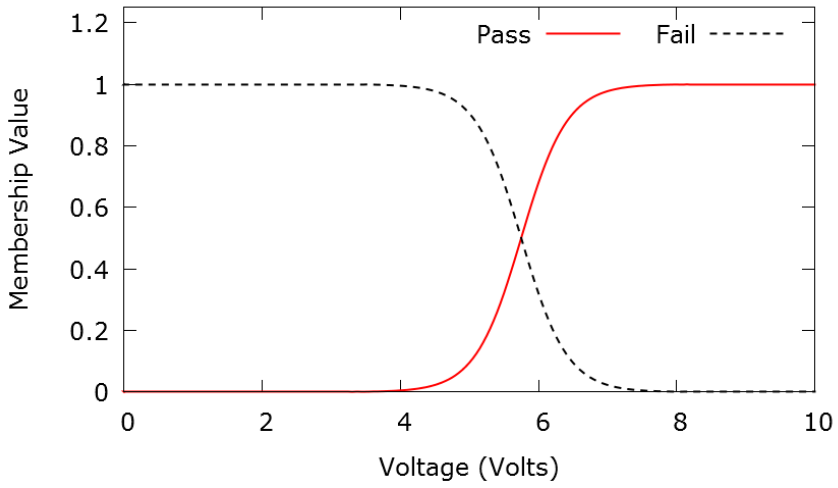


Figure 18: Fuzzy Membership Function for the *Voltage* test

ATML Circuit

The next network that we used to test the Fuzzy Bayesian Network framework is the ATML Circuit. This circuit was developed with funding from the US Navy

to demonstrate various IEEE standards for XML-based information exchange. The circuit is shown in Figure 19.

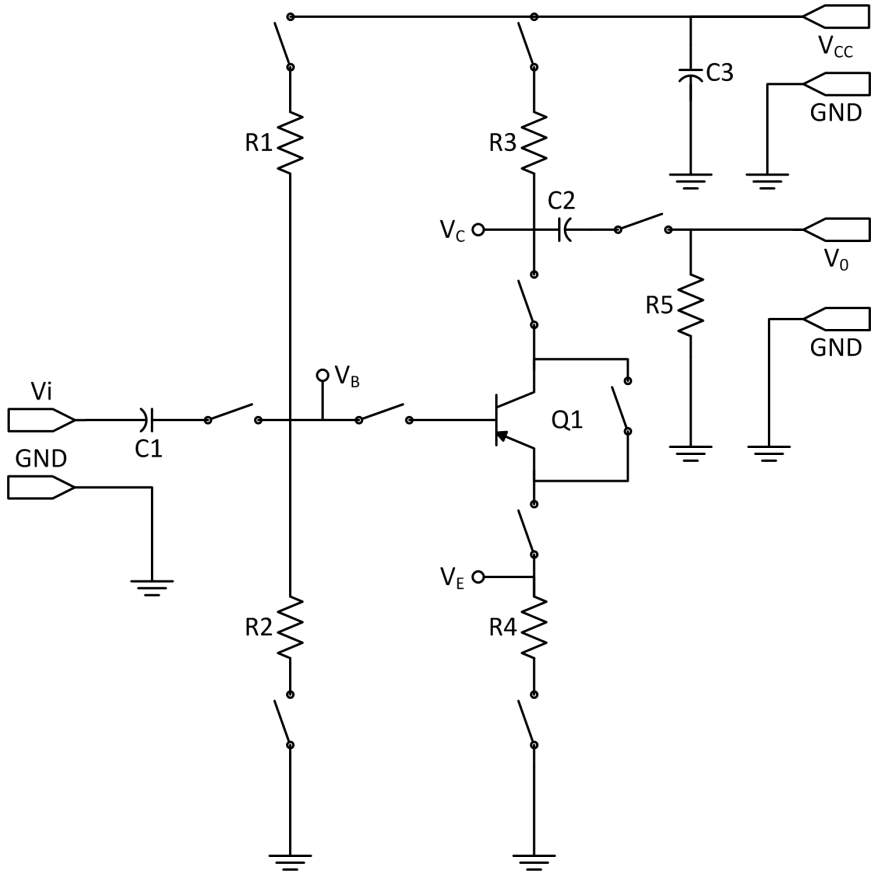


Figure 19: ATML test circuit

The ATML circuit can be tested by 11 possible tests. These tests correspond to evidence nodes in a diagnostic Bayesian network that was developed by Lockheed-Martin for a 2009 demonstration of the Automatic Test Markup Language (ATML). The eleven tests that are defined in the circuit are:

- AC voltage test at V_0
- DC voltage test at V_0
- AC voltage test at V_C

- DC voltage test at V_C
- Test for high DC voltage at V_B
- Test for low DC voltage at V_B
- Test for high DC voltage at V_E
- Test for low DC voltage at V_E
- Test for high DC voltage at V_C while bridging the base and emitter
- Test for low DC voltage at V_C while bridging the base and emitter
- V_{CC} continuity test

There are thirteen possible diagnoses that are represented in the Bayesian network:

- Capacitor short on C3
- Capacitor open on C2
- Capacitor open on C1
- Resistor open on R1
- Resistor open on R2
- Resistor open on R3
- Resistor open on R4
- Transistor Q1 collector open
- Transistor Q1 collector short
- Transistor Q1 base open

- Transistor Q1 emitter open
- Transistor Q1 base shorted to emitter
- Transistor Q1 base shorted to collector

In addition to the eleven evidence nodes and the thirteen query nodes, there are also four latent variables in the network. The latent variables are labeled $Q1_1$, $Q1_2$, $Q1_3$ and $Q1_4$. These hidden nodes were added to help make the size of the probability distributions for the failure modes of the transistor $Q1$ more manageable. The overall structure of the network can be seen in Figure 20. The conditional probability tables are included in Appendix B.

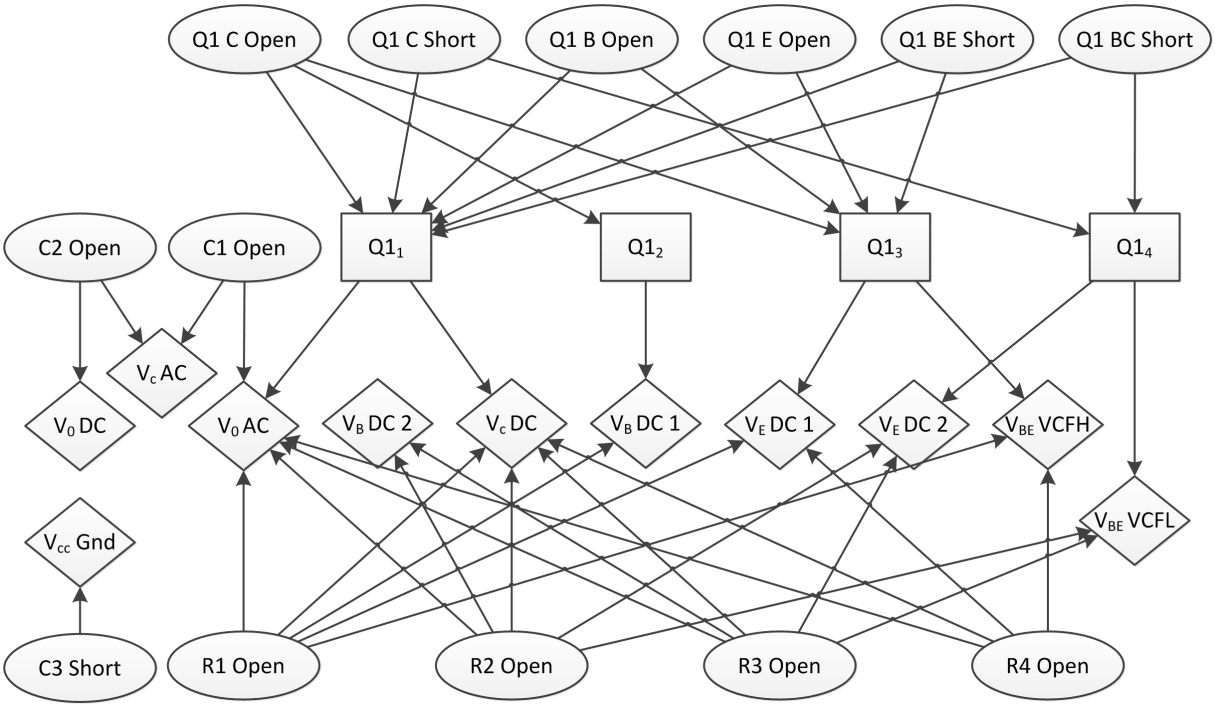


Figure 20: ATML diagnostic network

All of the tests are based on continuous values. Due to this, membership functions must be created for all of the tests to allow the actual measured values to be mapped

directly into the network. These membership functions were created with values that seemed to make sense but do not necessarily correspond to values that would be used in the real world. These membership functions are all plotted in Appendix A. The membership functions for the test nodes are as follows:

$$\begin{aligned} \mu_{V0AC=Pass}(x) &= e^{\frac{-1\cdot(x-10)^2}{2\cdot 0.6^2}} \\ \mu_{V0AC=Fail}(x) &= 1 - e^{\frac{-1\cdot(x-10)^2}{2\cdot 0.6^2}} \end{aligned} \tag{50}$$

$$\begin{aligned} \mu_{V0DC=Pass}(x) &= e^{\frac{-1\cdot(x-5)^2}{2\cdot 0.25^2}} \\ \mu_{V0DC=Fail}(x) &= 1 - e^{\frac{-1\cdot(x-5)^2}{2\cdot 0.25^2}} \end{aligned} \tag{51}$$

$$\begin{aligned} \mu_{VBDC1=Pass}(x) &= 1 - \frac{1}{1+e^{2\cdot(x-10)}} \\ \mu_{VBDC1=Fail}(x) &= \frac{1}{1+e^{2\cdot(x-10)}} \end{aligned} \tag{52}$$

$$\begin{aligned} \mu_{VBDC2=Pass}(x) &= \frac{1}{1+e^{2\cdot(x-20)}} \\ \mu_{VBDC2=Fail}(x) &= 1 - \frac{1}{1+e^{2\cdot(x-20)}} \end{aligned} \tag{53}$$

$$\begin{aligned} \mu_{VBEVCFH=Pass}(x) &= \frac{1}{1+e^{10\cdot(x-4.25)}} \\ \mu_{VBEVCFH=Fail}(x) &= 1 - \frac{1}{1+e^{10\cdot(x-4.25)}} \end{aligned} \tag{54}$$

$$\begin{aligned} \mu_{VBEVCFL=Pass}(x) &= 1 - \frac{1}{1+e^{10\cdot(x-2)}} \\ \mu_{VBEVCFL=Fail}(x) &= \frac{1}{1+e^{10\cdot(x-2)}} \end{aligned} \tag{55}$$

$$\begin{aligned} \mu_{VCAC=Pass}(x) &= e^{\frac{-1\cdot(x-25)^2}{2\cdot 1^2}} \\ \mu_{VCAC=Fail}(x) &= 1 - e^{\frac{-1\cdot(x-25)^2}{2\cdot 1^2}} \end{aligned} \tag{56}$$

$$\begin{aligned} \mu_{VCDC=Pass}(x) &= e^{\frac{-1\cdot(x-3.3)^2}{2\cdot 0.15^2}} \\ \mu_{VCDC=Fail}(x) &= 1 - e^{\frac{-1\cdot(x-3.3)^2}{2\cdot 0.15^2}} \end{aligned} \tag{57}$$

$$\begin{aligned} \mu_{VEDC1=Pass}(x) &= 1 - \frac{1}{1+e^{x-40}} \\ \mu_{VEDC1=Fail}(x) &= \frac{1}{1+e^{x-40}} \end{aligned} \tag{58}$$

$$\begin{aligned} \mu_{VEDC2=Pass}(x) &= \frac{1}{1+e^{x-60}} \\ \mu_{VEDC2=Fail}(x) &= 1 - \frac{1}{1+e^{x-60}} \end{aligned} \tag{59}$$

$$\begin{aligned} \mu_{VCC=Pass}(x) &= \frac{1}{1+e^{20\cdot(x-0.5)}} \\ \mu_{VCC=Fail}(x) &= 1 - \frac{1}{1+e^{20\cdot(x-0.5)}} \end{aligned} \tag{60}$$

Li-Ion Battery Network

The lithium-ion battery network was trained using a data set from the NASA Prognostics Center of Excellence data repository [22]. The data set contains data of charging and discharging cycles of lithium ion batteries at different temperatures. The purpose of this data set is to be able to develop models that can measure current battery health and overall charge degradation depending on various factors.

The network being used has a very simple structure. It consists of one diagnosis (query) node labeled *Capacity*, and five test (evidence) nodes labeled *Charge Current*, *Measured Current*, *Voltage Measured*, *Temperature* and *Charge Voltage*. The structure can be seen in Figure 21.

The most interesting feature of this network, unlike the previous networks, is that the states of each of the evidence nodes are not limited to *Pass* and *Fail*. They are five

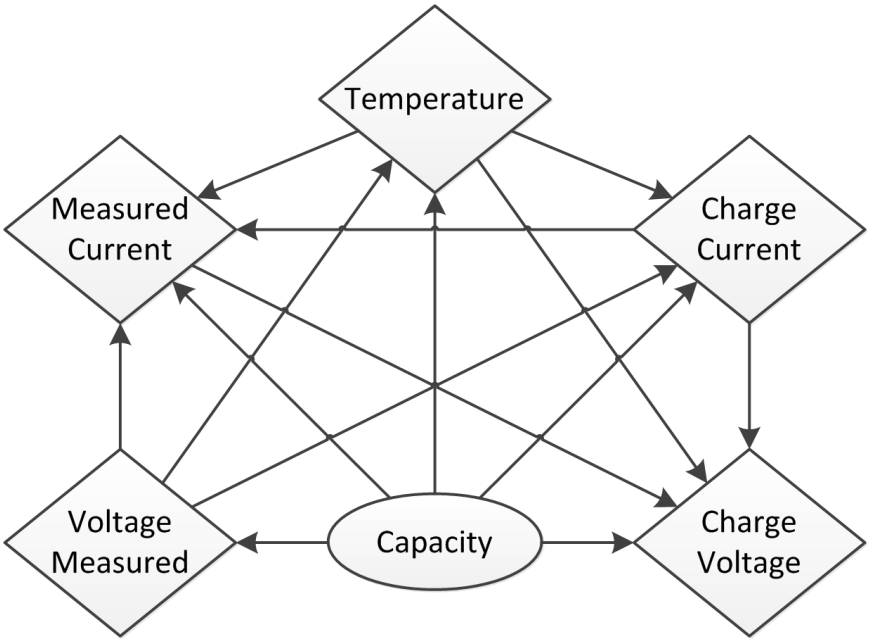


Figure 21: Battery network structure

different discrete states, which are labeled *TESTOUTCOME0*, *TESTOUTCOME1*, *TESTOUTCOME2*, *TESTOUTCOME3*, and *TESTOUTCOME4*.

These states do not map directly to simple measurements. Instead, they are mapped to intervals after PCA was performed on the input data. The process for using PCA in this context is outlined in [23]. This means the values that map to these states do not have an easy-to-understand interpretation and they are relative magnitudes within the transformed space. Nevertheless, we constructed fuzzy membership functions for each of the states, for each of the variables based on these transformed values.

All of the fuzzy membership functions were defined as logistic membership functions as follows:

Charge Current:

$$\mu_{TESTOUTCOME0} = \frac{1}{1 + e^{1.5 \cdot (-26.25 \cdot x)}} \tag{61}$$

$$\mu_{TESTOUTCOME1}(x) = \begin{cases} \frac{1}{1 + e^{-1.5 \cdot (-26.25 \cdot x)}} & \text{if } x \leq -18.625 \\ \frac{1}{1 + e^{10 \cdot (-11 \cdot x)}} & \text{if } x > -18.625 \end{cases}$$

$$\mu_{TESTOUTCOME2}(x) = \begin{cases} \frac{1}{1 + e^{-10 \cdot (-11 \cdot x)}} & \text{if } x \leq -10.7 \\ \frac{1}{1 + e^{10 \cdot (-10.4 \cdot x)}} & \text{if } x > -10.7 \end{cases}$$

$$\mu_{TESTOUTCOME3}(x) = \begin{cases} \frac{1}{1 + e^{-10 \cdot (-10.4 \cdot x)}} & \text{if } x \leq -8.7 \\ \frac{1}{1 + e^{5 \cdot (-7 \cdot x)}} & \text{if } x > -8.7 \end{cases}$$

$$\mu_{TESTOUTCOME4} = \frac{1}{1 + e^{-7 \cdot (-5 \cdot x)}}$$

Measured Current:

$$\mu_{TESTOUTCOME0} = \frac{1}{1 + e^{1.5 \cdot (-24 \cdot x)}} \tag{62}$$

$$\mu_{TESTOUTCOME1}(x) = \begin{cases} \frac{1}{1 + e^{-1.5 \cdot (-24 \cdot x)}} & \text{if } x \leq -20.4 \\ \frac{1}{1 + e^{1.5 \cdot (-16.8 \cdot x)}} & \text{if } x > -20.4 \end{cases}$$

$$\mu_{TESTOUTCOME2}(x) = \begin{cases} \frac{1}{1 + e^{-1.5 \cdot (-16.8 \cdot x)}} & \text{if } x \leq -14.6 \\ \frac{1}{1 + e^{1.5 \cdot (-12.4 \cdot x)}} & \text{if } x > -14.6 \end{cases}$$

$$\mu_{TESTOUTCOME3}(x) = \begin{cases} \frac{1}{1 + e^{-1.5 \cdot (-12.4 \cdot x)}} & \text{if } x \leq -4.5 \\ \frac{1}{1 + e^{1.5 \cdot (3.4 \cdot x)}} & \text{if } x > -4.5 \end{cases}$$

$$\mu_{TESTOUTCOME4} = \frac{1}{1 + e^{-1.5 \cdot (-3.4 \cdot x)}}$$

Voltage Measured:

$$\mu_{TESTOUTCOME0} = \frac{1}{1 + e^{1\cdot(-87\cdot x)}} \tag{63}$$

$$\mu_{TESTOUTCOME1}(x) = \begin{cases} \frac{1}{1 + e^{-1\cdot(-87\cdot x)}} & \text{if } x \leq -80.5 \\ \frac{1}{1 + e^{1\cdot(-74\cdot x)}} & \text{if } x > -80.5 \end{cases}$$

$$\mu_{TESTOUTCOME2}(x) = \begin{cases} \frac{1}{1 + e^{-1\cdot(-74\cdot x)}} & \text{if } x \leq -69.375 \\ \frac{1}{1 + e^{1\cdot(-64.75\cdot x)}} & \text{if } x > -69.375 \end{cases}$$

$$\mu_{TESTOUTCOME3}(x) = \begin{cases} \frac{1}{1 + e^{-1\cdot(-64.75\cdot x)}} & \text{if } x \leq -48.625 \\ \frac{1}{1 + e^{1\cdot(-32.5\cdot x)}} & \text{if } x > -48.625 \end{cases}$$

$$\mu_{TESTOUTCOME4} = \frac{1}{1 + e^{-1\cdot(-32.5\cdot x)}}$$

Temperature:

$$\mu_{TESTOUTCOME0} = \frac{1}{1 + e^{0.75\cdot(-475\cdot x)}} \tag{64}$$

$$\mu_{TESTOUTCOME1}(x) = \begin{cases} \frac{1}{1 + e^{-0.75\cdot(-475\cdot x)}} & \text{if } x \leq -413.4 \\ \frac{1}{1 + e^{0.45\cdot(-351.8\cdot x)}} & \text{if } x > -413.4 \end{cases}$$

$$\mu_{TESTOUTCOME2}(x) = \begin{cases} \frac{1}{1 + e^{-0.45\cdot(-351.8\cdot x)}} & \text{if } x \leq -332.65 \\ \frac{1}{1 + e^{0.5\cdot(-313.5\cdot x)}} & \text{if } x > -332.65 \end{cases}$$

$$\mu_{TESTOUTCOME3}(x) = \begin{cases} \frac{1}{1 + e^{-0.5\cdot(-313.5\cdot x)}} & \text{if } x \leq -259.625 \\ \frac{1}{1 + e^{0.5\cdot(-205.75\cdot x)}} & \text{if } x > -259.625 \end{cases}$$

$$\mu_{TESTOUTCOME4} = \frac{1}{1 + e^{-0.5\cdot(-205.75\cdot x)}}$$

Charge Voltage:

$$\mu_{TESTOUTCOME0} = \frac{1}{1 + e^{1.5\cdot(4.5\cdot x)}} \tag{65}$$

$$\mu_{TESTOUTCOME1}(x) = \begin{cases} \dfrac{1}{1 + e^{-1.5\cdot(4.5\cdot x)}} & \text{if } x \leq 9.25 \\ \dfrac{1}{1 + e^{1.5\cdot(14\cdot x)}} & \text{if } x > 9.25 \end{cases}$$

$$\mu_{TESTOUTCOME2}(x) = \begin{cases} \dfrac{1}{1 + e^{-1.5\cdot(14\cdot x)}} & \text{if } x \leq 18.75 \\ \dfrac{1}{1 + e^{1.5\cdot(23.5\cdot x)}} & \text{if } x > 18.75 \end{cases}$$

$$\mu_{TESTOUTCOME3}(x) = \begin{cases} \dfrac{1}{1 + e^{-1.5\cdot(23.5\cdot x)}} & \text{if } x \leq 25 \\ \dfrac{1}{1 + e^{1.5\cdot(26.5\cdot x)}} & \text{if } x > 25 \end{cases}$$

$$\mu_{TESTOUTCOME4} = \frac{1}{1 + e^{-1.5\cdot(26.5\cdot x)}}$$

To map the fuzzy membership value of *Capacity* to the actual capacity value given in the data we need to normalize the capacity data given. This is done assuming the maximum battery capacity is 2 Ahr. This allows us to get a percentage discharged that we can compare to the output of the *Capacity* node.

Experimental Results

In this section we use the networks presented in the previous section to show how this method of using Fuzzy Bayesian Networks works on the three test networks. This section contains selected instances of possible values in each of the networks. These instances were chosen either to illustrate a specific feature of the FBN or because the results were especially interesting.

Doorbell Network

The experiments performed on the doorbell network were broken into two groups. The first group is used to illustrate the influence of the fuzzy membership values as evidence on the behavior of the network. This is done by varying the membership values associated with the evidence in the network. The second group varies real-world values such as battery voltages, which are then translated into fuzzy membership values with the defined fuzzy membership functions.

Manipulate Membership Values. To get an initial sense of how the doorbell network behaves with fuzzy evidence instead of crisp evidence, we first varied the membership values directly. We varied the membership value for the *Voltage* test and used *Battery* as the query node. We expected to see a direct relationship between these two quantities. The results are shown in Figure 22.

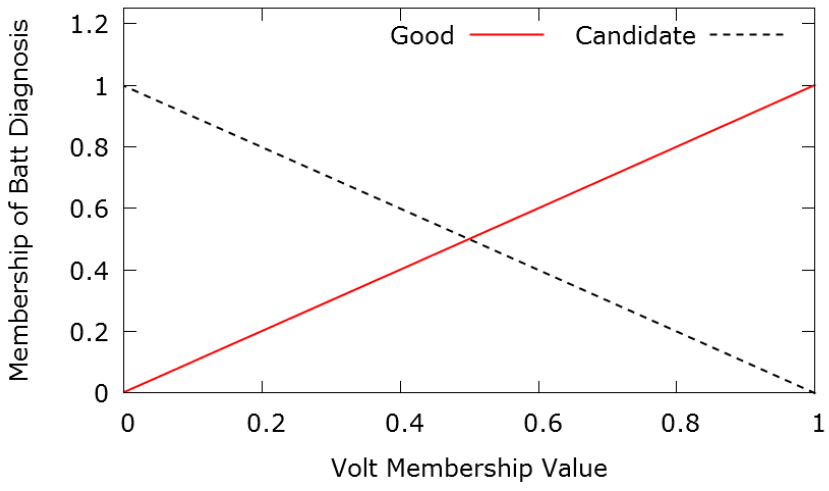


Figure 22: Membership value of Battery Diagnosis varying *volt* membership

In this first test, while varying the membership value for *Voltage*, the evidence on *Bridge*, *Stim* and *Push* were all held constant at a membership value of 1 for the

Pass condition for each test. For comparison, using crisp evidence on the *Voltage* node and traditional Bayesian inference we found the following. With evidence set as *Pass*, $P(\text{Good}) = 0.999999$, and with evidence set as *Fail* on *Voltage*, $P(\text{Good}) = 0.0039681748$.

Here the voltage membership value uses a linear change, so we expected, and observed, a linear relation to the membership value for *Battery*. A more interesting example of the relationships between the fuzzy evidence and the membership values for the diagnoses can be seen in Figure 23. In this second test, the evaluations were performed by only varying the membership value for the *Voltage* node whereas *Bridge*, *Stim*, and *Push* were all held constant at a membership value of 1 for *Pass* and 0 for *Fail*. Even though over the whole range of possible membership values $[0, 1]$ for the *Voltage* variable, the diagnosis of *SW-O* or Switch Open only changes from 0.96438008 to 0.999914647, the relation between the two is interesting. Unlike Figure 22 which shared a linear relationship between the membership value on *Voltage* and the membership value for the diagnosis node, this example does not have such a linear relationship.

The non-linearity in Figure 23 comes from the combination of Fuzzy evidence at other evidence nodes, as the fuzzy membership values are propagated through the network. This phenomenon is explained more thoroughly in Chapter 4. The second test used three steps of execution. First, it used the fuzzy evidence from the *Bridge* test node to update both the *Voltage* and *Stim* nodes. Then since there was already evidence applied to the *Voltage* and *Stim* nodes, the fuzzy union operator was used to combine the values propagated from *Bridge* and *Stim* or *Voltage*. The next step incorporates fuzzy values from the *Stim* and *Voltage* nodes into the *Push* node along with evidence that had already been set at that node. The final step was to propagate the fuzzy states to the diagnosis node *SW-O*. In this case, the effect

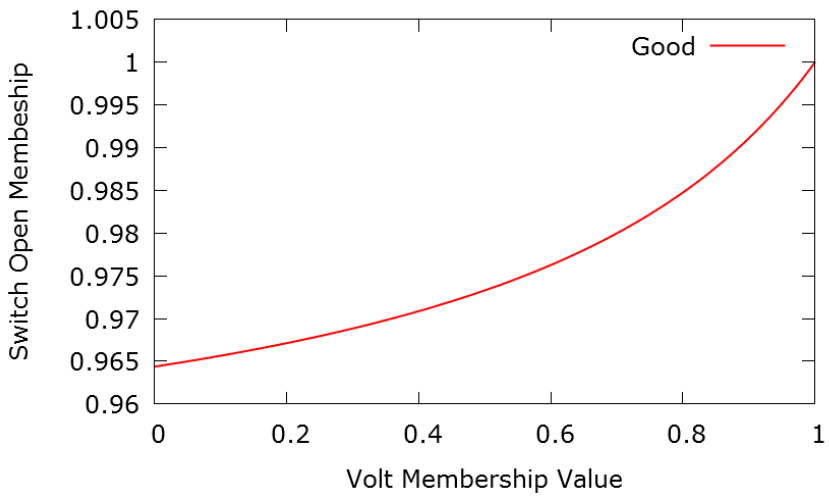


Figure 23: Membership value of *SW-O* with varying *Voltage* membership

is minimal; however, there are situations where the effect can become much more pronounced.

Real Values as Evidence. Now that we have illustrated how the doorbell network reacts to fuzzy evidence, we will use more realistic values as inputs to the network. Specifically, we will use the logistic fuzzy membership function defined in Equation 49. We look at only one test that can be represented in this network with a continuous valued input, which is the voltage test represented by the *Voltage* node. The rest of the tests have discrete results; either the doorbell rang or it did not. Due to this, we only vary the test results for the voltage measured from the battery.

The first test varies the battery voltage from 0 to 10 Volts. While this value is changing, the *Bridge*, *Stim*, and *Push* tests are all assumed to pass. This may not be the most realistic scenario, but it is a good baseline for investigating the impact of the fuzzy membership function of the evidence on the fuzzy values of the diagnoses.

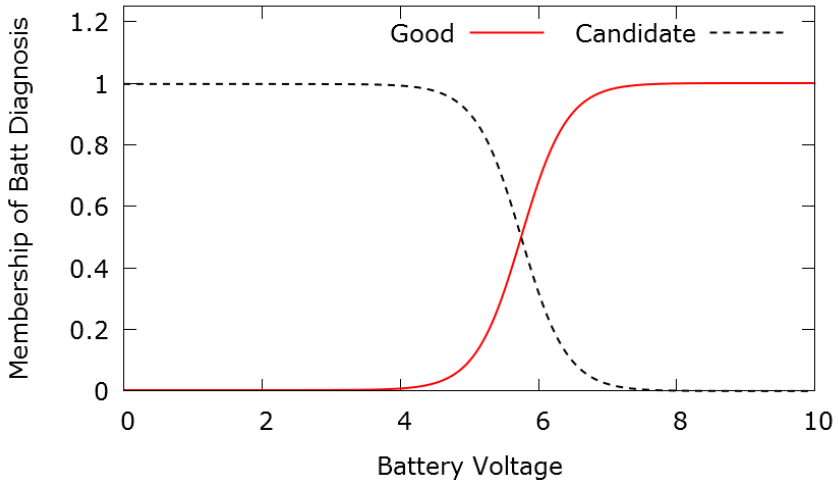
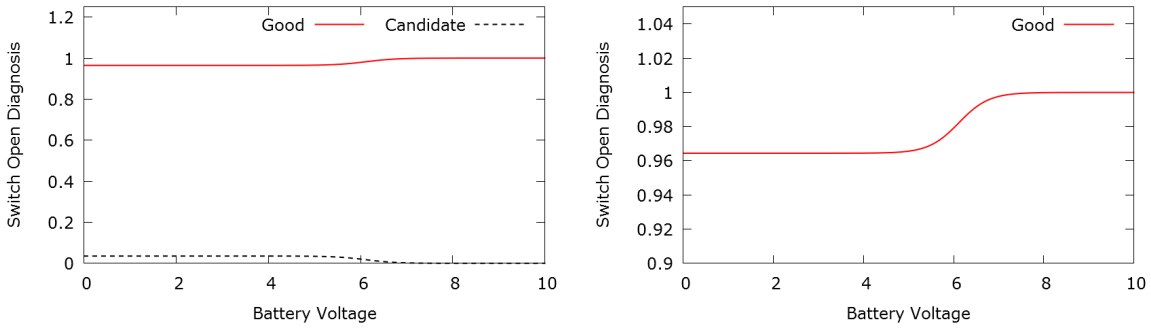


Figure 24: Membership value of Battery Diagnosis with varying Battery Voltage

The results of this experiment (Figure 24) look more as we might expect. The direct effect of the logistic fuzzy membership function is visible and seems to match well with the membership function shown in Figure 18. The more interesting plots come from looking at the Switch Open diagnosis node while varying the battery voltage.

Figure 25 shows the results of the same situation as above except the *SW-O* node is the query node instead of *Battery*.

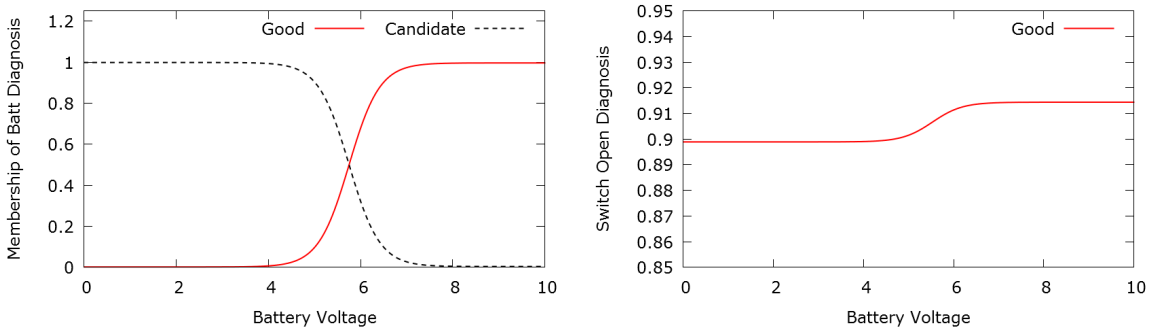


(a) Membership values for *SW-O*

(b) Zoom view of *Good* diagnosis

Figure 25: Membership values for *SW-O* with varying Battery Voltage

The non-linearity that we saw in Figure 23 is not easy to see in this example. This effect is obscured by the fuzzy membership function itself, whose shape dominates the plot. Figure 25b is a zoomed view of the *Good* diagnosis for *SW-O*, which is affected only slightly by the varying battery voltage.



(a) Membership values of *Battery* (b) Zoom view of *Good* diagnosis for *SW-O*

Figure 26: Membership values with varying *Voltage* all other tests *Fail*

In the next scenario, the battery voltage is being varied from 0 volts to 10 volts; however, unlike the previous test, the *Bridge*, *Stim* and *Push* tests all *Fail*. Figure 26 shows both the membership values for the *Battery* and *SW-O* diagnoses. Figure 26b is a zoomed in version of the *SW-O* diagnosis to make the trend easier to see. The same trends exist here that existed in the example where all the tests passed. The only difference is the actual membership values. The values for the *Good* diagnoses are lower in this case than the case where all the other test nodes were set to *Pass*.

ATML Network

The ATML network presents a different feature that can be present in Bayesian network structures that is not present in the Doorbell network structure. This is the concept of a hidden node. Hidden nodes are neither query nodes, nor are they evidence nodes. Thus, hidden nodes do not have evidence directly applied to them,

but they do have conditional probability tables associated with them and must be considered when performing inference.

Manipulate Membership Values. To illustrate how a layer of non-interconnected nodes behaves differently than a layer of interconnected nodes, we query the *Q1 C Open* diagnosis node. This node is connected to three hidden nodes, *Q1*, *Q2* and *Q3*. These hidden nodes are then connected to five different evidence nodes. They are V_0AC, V_CDC, $V_BDC\ 1$, $V_EDC\ 1$ and $V_{BE}VCFH$ respectively. For this scenario, all of these tests are assumed to have a state of *Pass* with a membership value of 1, except for $V_BDC\ 1$. The evidence node $V_BDC\ 1$ is varied from 0 to 1 to sweep its entire range of possible values. The effect of this on the membership values for *Q1 C Open* are plotted in Figure 27.

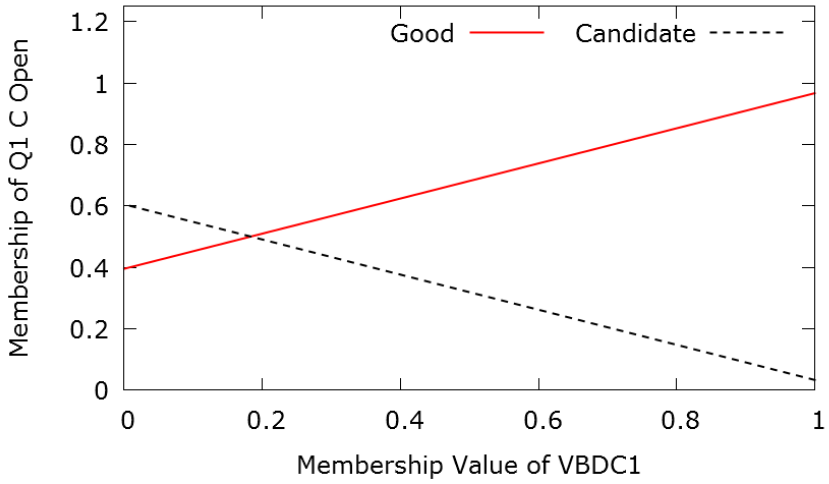


Figure 27: Membership value of *Q1 C Open* while varying $V_BDC\ 1$'s membership value

The inference process follows two steps. First, all of the evidence nodes: V_0AC, V_CDC, $V_BDC\ 1$, $V_EDC\ 1$ and $V_{BE}VCFH$, are evaluated and update the fuzzy states

of the nodes in the next layer up. In this case, all the nodes in the next layer are hidden nodes and do not have any pre-existing evidence associated with them; therefore, there is no need to use the fuzzy union operator to combine fuzzy evidence. The fuzzy states are propagated directly to these hidden nodes. Then the process is repeated at the layer of hidden nodes to propagate to the diagnosis node *Q1 C Open*.

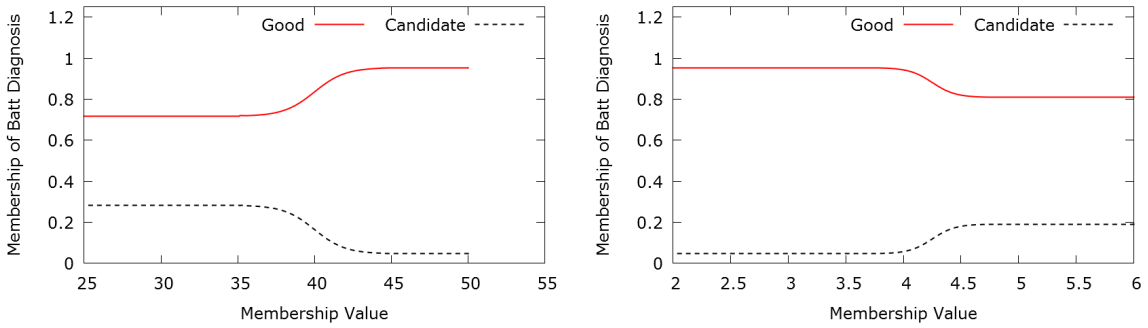
While the same mulit-step evaluation process is used to perform inference as in the example from the doorbell network, we do not see the same non-linearity we saw with the doorbell. This is because in the doorbell network, the successive steps were updating the fuzzy evidence of interconnected observed nodes, then using these updated fuzzy states with their own states to update the next level. This process is not performed on the hidden nodes because there is a layer between the evidence nodes and the query nodes which separates the network, preventing the non-linear relationship from forming.

Real Values as Evidence. Now that we have shown that a layer of non-interconnected nodes do not behave in the same way as a layer of interconnected nodes, we will use realistic values and the fuzzy membership functions from Equations 50 through 60.

This first scenario will consider the diagnosis node *Q1 BE Short*. This scenario corresponds to the diagnosis of the base and the emitter of transistor *Q1* being shorted. This is an interesting test because of the interaction between the voltage measurement for $V_E DC\ 1$ and $V_{BE} VCFH$. For this scenario, the only nodes that will have varying values will be $V_E DC\ 1$ and $V_{BE} VCFH$. All the other values will be held constant. We illustrate the effect in three steps: first only $V_E DC\ 1$ will be varied from 25 to 55 volts while the V_{BE} will be held at 3.3 volts, which is a pass condition

for $V_{BE}VCFH$; second the voltage at $V_EDC\ 1$ will be held constant at 50 volts, which is a *Pass* condition, and vary the V_{BE} voltage from 2 to 6 volts; third the V_{BE} and the $V_EDC\ 1$ voltages will be varied across their ranges from *Pass* conditions to *Fail* conditions.

The first scenario, varying $V_EDC\ 1$ with $V_{BE}VCFH$ constant is shown in Figure 28a. Figure 28b shows the scenario with $V_EDC\ 1$ held constant and $V_{BE}VCFH$ varied. The values plotted are the membership values of *Q1 BE Short*.



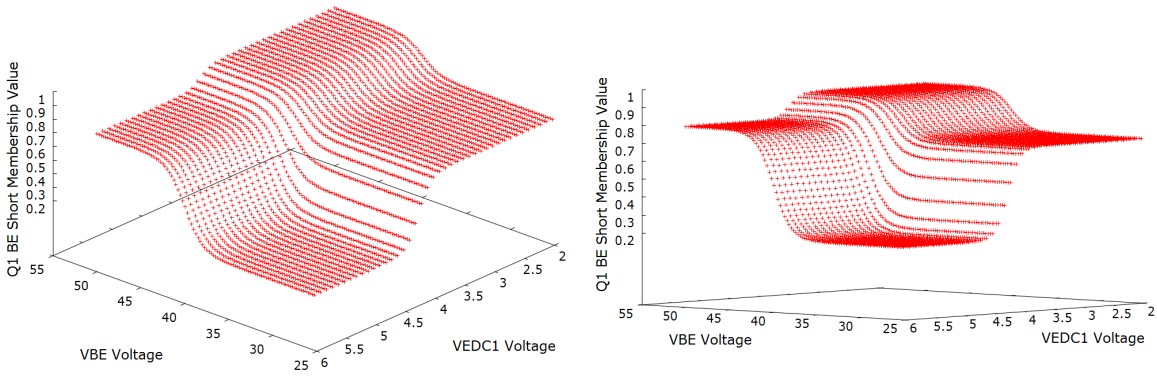
(a) Values for Battery diagnosis

(b) Zoom view of *Good* diagnosis

Figure 28: Membership values with varying Battery Voltage other tests *fail*

The scenario where the voltages for both $V_EDC\ 1$ and V_{BE} are varied is presented in Figure 29. The value plotted is the membership value of the *Q1* transistor's base being shorted to the emitter. Figures 29 and 29b are plots of the same data, just rotated differently to allow the data to be more easily visualized.

It is interesting to see the four flat plateaus in Figure 29, which correspond to when $V_{BE}VCFH$ and $V_EDC\ 1$ are both a *Fail*, when $V_{BE}VCFH$ is *Pass* and $V_EDC\ 1$ is *Failed*, when they both are *Pass*, and when $V_{BE}VCFH$ is *Fail* and $V_EDC\ 1$ is pass. The transitions between these plateaus are due to the shape of the membership functions. In this case, both membership functions are logistic functions. If this same



(a) Varying V_{BE} and $V_EDC\ 1$ (b) Varying V_{BE} and $V_EDC\ 1$ side view

Figure 29: Membership values for individually varying V_{BE} and $V_EDC\ 1$

scenario were run without using fuzzy evidences, but with crisp evidence similar plateaus would be seen, but there would not be the gradual transitions.

The gradual change between the states exemplifies one of the key advantages of using a fuzzy Bayesian network as opposed to a classic Bayesian network for diagnostics. The fuzzy membership functions are able to capture degrees of state assignments which can translate into a measure of degradation.

Battery Network

The third network we used for testing we learned from the NASA Li-Ion Battery data set, which evaluates battery capacity based on charge current, charge voltage, battery temperature, battery voltage, and battery current. This networks is designed to represent degradation of the battery in terms of capacity.

The Bayesian network was learned such that, for each of the five tests, there are five possible test outcomes. These are called, *TESTOUTCOME0*, *TESTOUTCOME1*, *TESTOUTCOME2*, *TESTOUTCOME3*, and *TESTOUTCOME4* as described previously.

Due to this transformation, we performed the experiments in two phases. First swept over the range of transformed values that are valid for each individual test. Second used real battery data from the repository to attempt to predict battery degradation.

Effects of Each Test Variable. To get a good understanding of the effects each variable has on battery capacity, we swept through the range of all valid assignments to each variable. The ranges for each variable are given in Table 15. In each test, the variables that are not being swept over are set to their maximum value, *TESTOUTCOME4*, with membership value of 1.0. This was chosen because, of the combinations tested, it yielded the greatest differences in the outputs, which made the plots more interesting. Figure 30 shows the results of the sweeps of each individual variable while holding the others constant. While these results do not give a full picture of how the network would behave, in practice they help to give an idea of the effects of the different *TESTOUTCOME*s. In order to get a full understanding of how the network would behave, we would need to sweep across every variable at the same time, which is not feasible, or easily presentable to the reader.

Table 15: Input ranges for each Battery variable

Network Variable	Minimum Value	Maximum Value
Charge Current	-30	10
Current Measured	-30	15
Temperature	-500	-150
Charge Voltage	-10	40
Voltage Measured	-100	-50

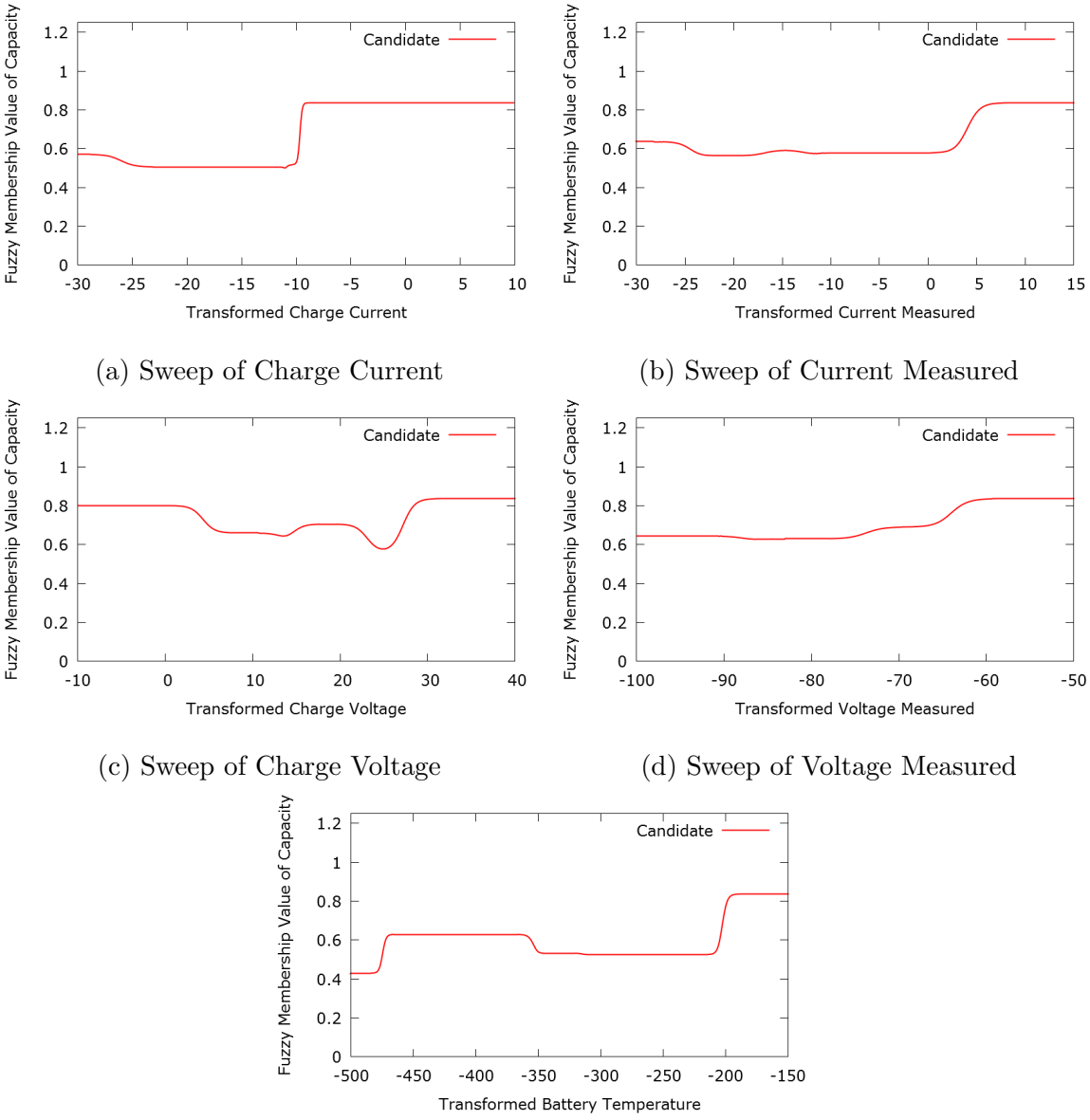
As we sweep across each variable, there are bumps and dips in the membership values in Figure . These changes happen when the fuzzy membership function values are changing between states. These figures are intended to help to illustrate the effect of the membership functions on the final result.

Battery Degradation. Next we apply scenarios that correspond to degradation of battery capacity. For each set of measurements, there is a corresponding battery capacity value derived from actual accelerated life testing that is used as a ground truth in our experiments. This capacity is a continuous value with 2.0 as maximum capacity, and as the capacity value decreases, the overall health of the battery decreases as well.

Table 16: Seven battery degradation test points

Voltage Measured	Measured Current	Temperature	Charge Current	Charge Voltage	Capacity
-16.0314	-2.5361	-42.4211	-47.9836	3.9497	1.8565
-89.1721	-7.0267	-452.0676	-7.1498	-94.2487	1.8041
-64.7714	-8.5038	-674.4137	-8.5751	-50.5853	1.7282
-59.0502	-9.1018	-676.0246	-9.1462	-43.3863	1.4156
-61.0147	-10.5288	-679.3794	-10.3951	-45.7369	1.351
-51.3241	-11.1308	-658.7028	-11.0458	-35.0896	1.2416
-90.6681	-26.5886	-867.1445	-26.5621	-109.7129	1.1857

Our experiments use seven sample data points to represent battery capacity degradation (see Table 16). In order to compare the results from the FBN to the battery capacity truth we need to match their scales. Since we are interested in quantifying



(a) Sweep of Charge Current

(b) Sweep of Current Measured

(c) Sweep of Charge Voltage

(d) Sweep of Voltage Measured

(e) Sweep of Battery Temperature

Figure 30: Sweeps of individual variables and the resulting battery capacity membership values

degradation, and we know the maximum capacity value is 2.0, we can normalize the given capacity value to be on the scale of $[0, 1]$. Figure 31 plots this normalized battery capacity as well as the predicted degradation from the Fuzzy Bayesian Network at

each test point. Notice that not only does the predicted value trend in the same direction as the actual battery capacity, the actual values are very similar to each other. This means that, at least in this circumstance, and model, a Fuzzy Bayesian Network can be used to estimate degradation of components within a system.

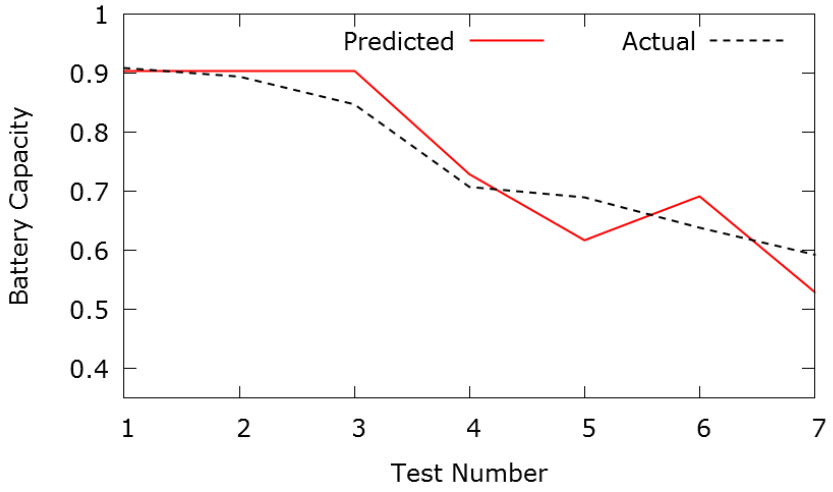


Figure 31: FBN predicted vs actual battery capacity

CONCLUSION

Summary

This work provided an indepth study of Fuzzy Bayesian Networks (FBN), and how Fuzzy Bayesian Networks can be applied in the context of diagnostics and prognostics. This work was motivated by the goal of incorporating fuzzy values into a probabilistic framework to model gray scale health with uncertainty. These fuzzy values were results of tests, which instead of returning either a crisp *Pass* or *Fail*, could return a fuzzy membership value for *Pass* and *Fail*. This information is able to give a diagnostic reasoner more information about the current state of a system to make better diagnoses. The key idea was to allow diagnoses to have soft outcomes, which could be used to represent degradation of a component.

The demonstration that inspired this work used fuzzy fault trees to demonstrate the viability of this idea. We expanded on the fuzzy fault trees by using a fuzzy Bayesian network for use in diagnostics. Fuzzy Bayesian networks are not uncommon in literature; however, what is lacking in FBN literature is a standard method for integrating fuzzy values into a Bayesian network. There are a few different methods for creating a FBN, all of which are used with different purposes and handle fuzzy values differently. Some of these have been presented in Chapter 3. We chose one of the methods around which to base our Fuzzy Bayesian Network.

The methodology we chose was based on work by Christopher Fogelberg [18]. This methodology keeps the actual process of inference within the Bayesian network separate from the fuzzy values, then represents them together in a *Fuzzy Probability*

Distribution, thus preserving both the probabilities and the fuzzy values separately, but in a combined representation.

When implementing this approach, we made two novel contributions. The first was related to the order of execution. The implemented method of evaluating a FBN requires the inference to be performed in steps, propagating the fuzzy evidence through the network toward the query node. The method in which the fuzzy states were to be propagated through a non-trivial network was left unspecified. Our approach is discussed more in Chapter 4.

The method we chose for evaluating FBNs suffers from an explosion in the size of fuzzy states as evidence is propagated through a network. To combat this combinatorial explosion, at each layer of evaluation we collapse the entire fuzzy probability distribution into a single fuzzy state using Fogelberg's method of linear collapse [8]. While this method does reduce the overall size of the fuzzy states, it conflates probability and fuzziness which is not necessarily correct.

This brings us to our second contribution, which is how to update the fuzzy evidence over multiple evidence nodes. Nodes within a network can only have one fuzzy state assignment at any given time. However, since evidence is represented as a fuzzy state, it is not clear how to propagate fuzzy evidence across other evidence nodes. A concrete example of this is shown in Chapter 4. To solve this problem, we take a fuzzy union between the fuzzy state being propagated and the fuzzy state set as evidence, then normalize the resulting state.

We implemented our approach to FBN-based diagnosis in C#, and we use the SMILE [24] library to perform the traditional Bayesian inference that is required for this process.

Our implementation was then tested with three different pre-defined, diagnostic networks. One models a doorbell circuit, one models a more complex demonstration

circuit, and the third network models Li-Ion battery capacity based on various parameters of the battery in operation. Each of the first two were run using various scenarios, and the effect of the fuzzy evidence values were demonstrated. The battery network was used to illustrate that the Fuzzy Bayesian Network can be used to model component degradation. With the battery network we calculated the degradation of the battery capacity at various levels of discharge and compared them with known battery capacity at the given points in time.

The results from the battery test show that it is possible to use the Fuzzy Bayesian Network to model component level degradation. The overall objective behind creating the Fuzzy Bayesian Network was to model component degradation similar to the fuzzy fault tree. The Bayesian network can be a more flexible tool for modeling than a fault tree, and this enhancement has been demonstrated. Additionally, the Fuzzy Bayesian Network can use the same fuzzy test outcomes as the fuzzy fault tree, and can output the same soft diagnosis values.

Future Work

Even though this is a more complete and thorough handling of Fuzzy Bayesian Networks than most, there are still places where more work is needed. First, the order in which the propagation occurs does matter. In the future, we would like to explore different methods for determining the execution order. The method used now was chosen because it yielded reasonable values. The current approach, however, may not be the best, or might be incorrect all together.

The second contribution of this work is the use of the fuzzy union to merge the fuzzy states at nodes with evidence. This technique, similar to determining the

execution order, has a large impact on the overall outcome at the query node. We plan to investigate other methods of merging the fuzzy state information and better understand what is happening because of the fuzzy union.

In addition to these areas of investigation, we also plan to perform more trials that represent component degradation. We plan to perform more experiments with the battery data set and network to better understand using a FBN to model degradation within this system. Additionally, we plan to perform similar degradation experiments on other networks and data sets to verify the results are not limited to this data set alone. Running experiments on other datasets may yield unexpected results that would give us greater insight into how the Fuzzy Bayesian Network operates.

REFERENCES CITED

[1] P.J. Donnelly, L.E. Sturlaugson, J.W. Sheppard. A standards-based approach to gray-scale health assessment using fuzzy fault trees. *IEEE AUTOTESTCON, 2012*, pages 174 –181, Sept. 2012.

[2] Christopher Fogelberg. Belief propagation in fuzzy Bayesian networks: A worked example. Shamal Faily, Standa Zivny, editors, *Proceedings of the 2008 Oxford University Computing Laboratory Student Conference*, October 2008.

[3] S. J. Russell, P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, edition 3rd, 2009.

[4] Daphne Koller, Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, 2009.

[5] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

[6] Lotfi A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

[7] Lotfi A. Zadeh. Probability measures of fuzzy events. *Journal of Mathematical Analysis and Applications*, 1968.

[8] Christopher Fogelberg, Vasile Palade, Phil Assheton. Belief propagation in fuzzy bayesian networks. Ioannis Hatzilygeroudis, editor, *1st International Workshop on Combinations of Intelligent Methods and Applications(CIMA) at ECAI'08*, pages 19–24, University of Patras, Greece, 21–22 July 2008.

[9] Huibert Kwakernaak. Fuzzy random variablesi. definitions and theorems. *Information Sciences*, 15(1):1 – 29, 1978.

[10] Huibert Kwakernaak. Fuzzy random variablesii. algorithms and examples for the discrete case. *Information Sciences*, 17(3):253 – 278, 1979.

[11] Madan L Puri, Dan A Ralescu. Fuzzy random variables. *Journal of Mathematical Analysis and Applications*, 114(2):409 – 422, 1986.

[12] Maria Angeles Gil, Miguel Lopez-Diaz, Dan A. Ralescu. Overview on the development of fuzzy random variables. *Fuzzy Sets and Systems*, 157(19):2546 – 2557, 2006.

[13] Arnold F Shapiro. A fuzzy random variable primer. *Actuarial Research Clearing House*, 2008.

[14] Hussam Hamrawi. *Type-2 Fuzzy Alpha-cuts*. Doctoral Dissertation, De Montfort University, 2011.

[15] Ali Ben Mrad, Veronique Delcroix, MohamedAmine Maalej, Sylvain Piechowiak, Mohamed Abid. Uncertain evidence in bayesian networks: Presentation and comparison on a simple example. Salvatore Greco, Bernadette Bouchon-Meunier, Giulianella Coletti, Mario Fedrizzi, Benedetto Matarazzo, RonaldR. Yager, editors, *Advances in Computational Intelligence*, Volume 299 series *Communications in Computer and Information Science*, pages 39–48. Springer Berlin Heidelberg, 2012.

[16] Hao Tang, Shi Liu. Basic theory of fuzzy bayesian networks and its application in machinery fault diagnosis. *Proceedings of the Fourth International Conference on Fuzzy Systems and Knowledge Discovery - Volume 04*, FSKD '07, pages 132–137, Washington, DC, USA, 2007. IEEE Computer Society.

[17] H. Pan, L. Liu. Fuzzy bayesian networks-a general formalism for representation, inference and learning with hybrid bayesian networks. *Neural Information Processing, 1999. 6th International Conference on ICONIP '99.*, Volume 1, pages 401–406 vol.1, 1999.

[18] Christopher Fogelberg. Fuzzy bayesian networks for network inference. Transfer Report, Computing Laboratory, Wolfson Building, Parks Road, Oxford, OX1-3QD, October 2008.

[19] Cesar A. Penz, Carlos A. Flesch, Silvia M. Nassar, Rodolfo C. C. Flesch, Marco A. de Oliveira. Fuzzy-bayesian network for refrigeration compressor performance prediction and test time reduction. *Expert Syst. Appl.*, 39(4):4268–4273, March 2012.

[20] J.W. Sheppard, S.G.W. Butcher, P.J. Donnelly. Demonstrating semantic interoperability of diagnostic reasoners via AI-ESTATE. *IEEE Aerospace Conference, 2010*, pages 1 –10, March 2010.

[21] J.W. Sheppard, S.G.W. Butcher, P.J. Donnelly, B.R. Mitchell. Demonstrating semantic interoperability of diagnostic models via AI-ESTATE. *IEEE Aerospace conference, 2009*, pages 1 –13, March 2009.

[22] B. Saha, K. Goebel. Battery data set. *NASA Ames Prognostics Data Repository*, 2007.

[23] Liessman Sturlaugson, John Sheppard. Principal component analysis preprocessing with bayesian networks for battery capacity estimation. *I2MTC*, pages 98–101. IEEE, 2013.

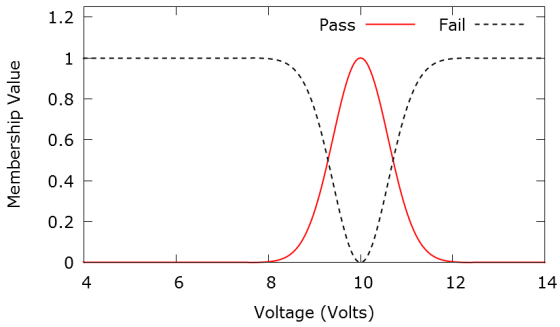
[24] Marek J. Druzdzel. SMILE: Structural Modeling, Inference, and Learning Engine and GeNIe: a development environment for graphical decision-theoretic models

(Intelligent Systems Demonstration). *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99)*, pages 902–903. AAAI Press/The MIT Press, 1999.

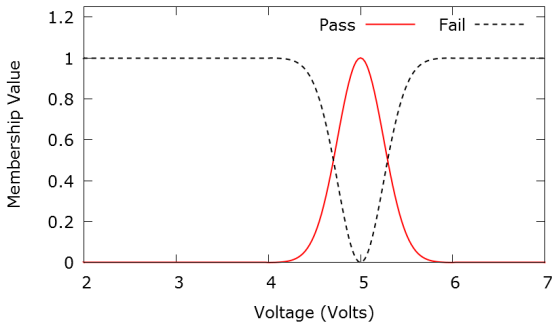
APPENDICES

APPENDIX A

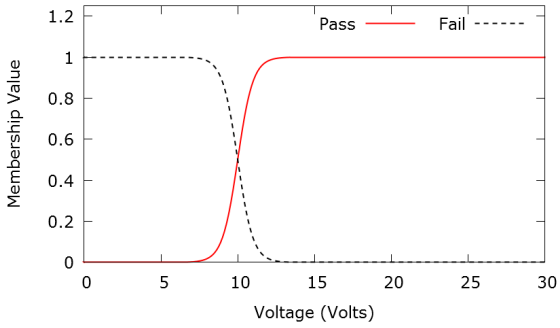
MEMBERSHIP FUNCTIONS



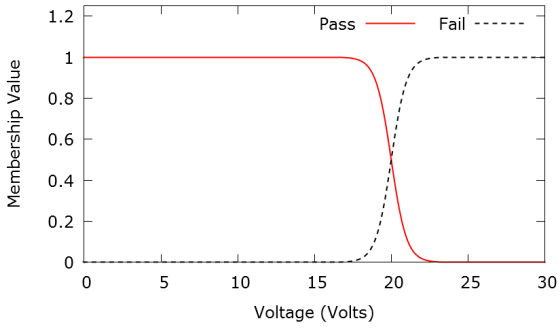
(a) V0 AC membership function



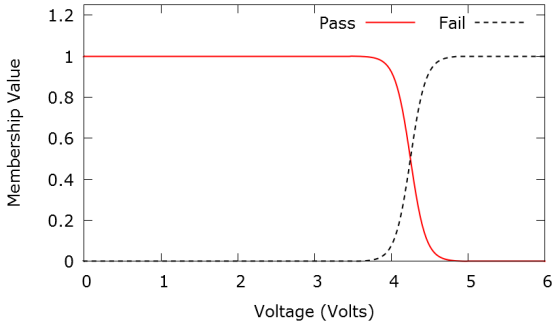
(b) V0 DC membership function



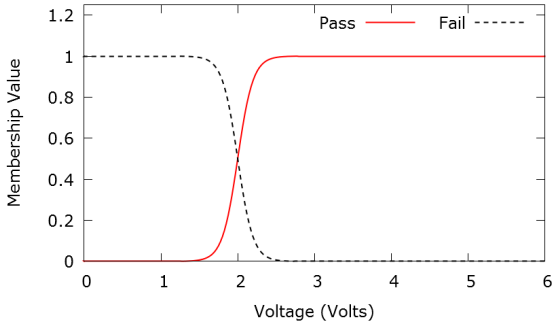
(c) VBDC1 membership function



(d) VBDC2 membership function

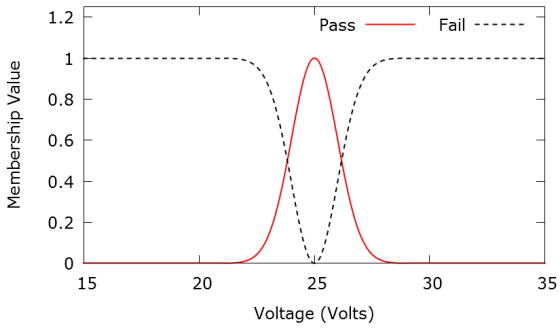


(e) VBEVCFH membership function

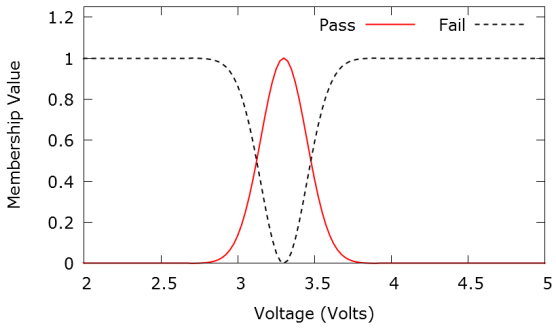


(f) VBEVCFL membership function

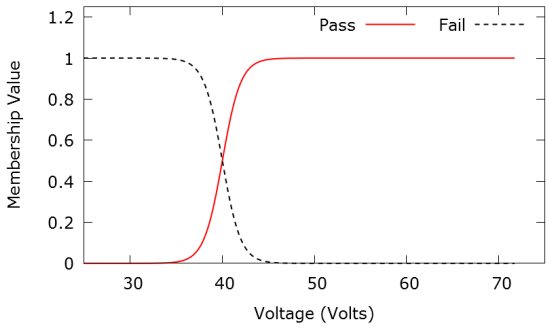
Figure 32: Membership functions for ATML network



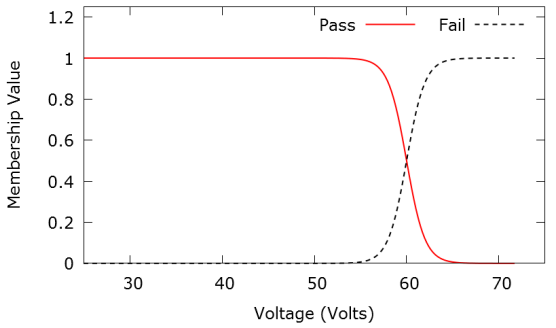
(a) VCAC membership function



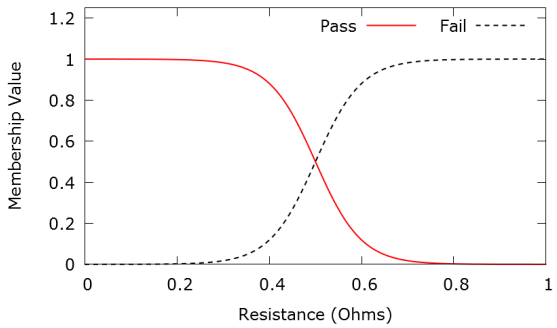
(b) VCDC membership function



(c) VEDC1 membership function



(d) VEDC2 membership function



(e) VCC membership function

Figure 33: Membership functions for ATML network (continued)

APPENDIX B

EVALUATION NETWORKS

Doorbell Conditional Probability Tables

```
<cpt id="SW-C">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.9995000000000001 0.0005</probabilities>
</cpt>
<cpt id="pushPush">
  <state id="Pass" />
  <state id="Fail" />
  <parents>SW-C</parents>
  <probabilities>0.9999900000000001 1e-005 0.03 0.97</
      probabilities>
</cpt>
<cpt id="Battery">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.995 0.005</probabilities>
</cpt>
<cpt id="Voltage">
  <state id="Pass" />
  <state id="Fail" />
  <parents>Battery</parents>
  <probabilities>0.9999900000000001 1e-005 0.001 0.999</
      probabilities>
</cpt>
<cpt id="Sol-O">
```

```
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.9999 0.0001</probabilities>
</cpt>
<cpt id="Clap-O">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.9997 0.0003</probabilities>
</cpt>
<cpt id="Clap-C">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.9996 0.0004</probabilities>
</cpt>
<cpt id="Stim">
  <state id="Pass" />
  <state id="Fail" />
  <parents>Sol-O Clap-O Clap-C</parents>
  <probabilities>0.9999900000000001 1e-005 0.01 0.99
      0.008999999999999999 0.991 0.005 0.995 0.011 0.989 0.004
      0.996 0.003 0.997 0.001 0.999</probabilities>
</cpt>
<cpt id="SW-O">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.9993 0.0007</probabilities>
</cpt>
```

```
<cpt id="Push">
  <state id="Pass" />
  <state id="Fail" />
  <parents>Voltage Stim SW-O</parents>
  <probabilities>0.9999900000000001 1e-005 0.01 0.99
      0.008999999999999999 0.991 0.005 0.995 0.008 0.992 0.004
      0.996 0.003 0.997 0.001 0.999</probabilities>
</cpt>
<cpt id="Bridge">
  <state id="Pass" />
  <state id="Fail" />
  <parents>Voltage Stim Battery</parents>
  <probabilities>0.9999900000000001 1e-005 0.01 0.99
      0.008999999999999999 0.991 0.005 0.995 0.008 0.992 0.004
      0.996 0.003 0.997 0.001 0.999</probabilities>
</cpt>
```

ATML Conditional Probability Tables

```
<cpt id="C3_Short">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.9895833333 0.01041666667</probabilities>
</cpt>
<cpt id="C2_Open">
  <state id="Good" />
  <state id="Candidate" />
```

```
  <probabilities>0.9895833333 0.01041666667</probabilities>
</cpt>
<cpt id="C1_Open">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.8229166667 0.1770833333</probabilities>
</cpt>
<cpt id="VCCGND">
  <state id="Good" />
  <state id="Candidate" />
  <parents>C3_Short</parents>
  <probabilities>1 0 0 1</probabilities>
</cpt>
<cpt id="V0DC">
  <state id="Pass" />
  <state id="Fail" />
  <parents>C2_Open</parents>
  <probabilities>1 0 0.5 0.5</probabilities>
</cpt>
<cpt id="VCAC">
  <state id="Pass" />
  <state id="Fail" />
  <parents>C2_Open C1_Open</parents>
  <probabilities>1 0 0.1666666666 0.8333333333999999 0.5 0.5
      0.08333333333335 0.91666666666665</probabilities>
</cpt>
<cpt id="R1_Open">
```

```
  <state id="Pass" />
  <state id="Fail" />
  <probabilities>0.8229166667 0.1770833333</probabilities>
</cpt>
<cpt id="R2_Open">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.90625 0.09375</probabilities>
</cpt>
<cpt id="R3_Open">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.90625 0.09375</probabilities>
</cpt>
<cpt id="R4_Open">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.90625 0.09375</probabilities>
</cpt>
<cpt id="Q1_C_Open">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.91 0.08999999999999997</probabilities>
</cpt>
<cpt id="Q1_C_Short">
  <state id="Good" />
  <state id="Candidate" />
```

```
  <probabilities>0.98 0.02000000000000002</probabilities>
</cpt>
<cpt id="Q1_B_Open">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.99 0.01000000000000001</probabilities>
</cpt>
<cpt id="Q1_E_Open">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.98 0.02000000000000002</probabilities>
</cpt>
<cpt id="Q1_BE_Short">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.69 0.31</probabilities>
</cpt>
<cpt id="Q1_BC_Short">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.991 0.009000000000000008</probabilities>
</cpt>
<cpt id="Q1_1">
  <state id="State0" />
  <state id="State1" />
  <parents>Q1_C_Open Q1_C_Short Q1_B_Open Q1_E_Open Q1_BE_Short
      Q1_BC_Short</parents>
```

```
  <probabilities>0.9 0.09999999999999998 0.09 0.91 0.92 0.08
     0.07000000000000001 0.9300000000000001 0.9 0.1 0.09 0.91
     0.92 0.08 0.29 0.71 0.94 0.06 0.02 0.98 0.28 0.72 0.94 0.06
     0.89 0.11 0.08 0.92 0.9300000000000001 0.07000000000000001
     0.38 0.62 0.6800000000000001 0.32 0.41 0.59 0.98 0.02 0.46
     0.54 0.95 0.05 0.05 0.95 0.89 0.11 0.04 0.96 0.94 0.06
     0.07000000000000001 0.9300000000000001 0.91 0.09 0.06 0.94
     0.91 0.09 0.06 0.94 0.91 0.09 0.15 0.85 0.95 0.05 0.09 0.91
     0.91 0.09 0.07000000000000001 0.9300000000000001
     0.9300000000000001 0.07000000000000001 0.08 0.92 0.88 0.12
     0.07000000000000001 0.9300000000000001 0.91 0.09 0.09 0.91
     0.85 0.15 0.45 0.55 0.89 0.11 0.02 0.98 0.9 0.1 0.08 0.92
     0.9300000000000001 0.07000000000000001 0.08 0.92 0.91 0.09
     0.07000000000000001 0.9300000000000001 0.94 0.06 0.05 0.95
     0.86 0.14 0.05 0.95 0.91 0.09 0.11 0.89 0.86 0.14 0.11 0.89
     0.86 0.14 0.05 0.95 0.9 0.1 0.07000000000000001
     0.9300000000000001</probabilities>
</cpt>
<cpt id="V0AC">
  <state id="Pass" />
  <state id="Fail" />
  <parents>C1_Open R1_Open Q1_1 R4_Open R3_Open R2_Open</parents>
  <probabilities>1 0 0.04 0.96 0.94 0.06 0.05 0.95 0.95 0.05 0.06
      0.94 0.94 0.06 0.06 0.94 0.1428571428 0.8571428571999999
     0.05714285712 0.94285714288 0.05714285712 0.94285714288
     0.02285714285 0.97714285715 0.05357142855 0.94642857145
     0.02142857142 0.97857142858 0.02142857142 0.97857142858
```

0.008571428568 0.991428571432 0.2666666667
0.7333333333000001 0.1066666667 0.8933333333 0.1066666667
0.8933333333 0.04266666668 0.95733333332 0.1 0.9 0.04 0.96
0.04 0.96 0.016 0.984 0.03809523808 0.96190476192
0.01523809523 0.98476190477 0.01523809523 0.98476190477
0.006095238092 0.993904761908 0.01428571428 0.98571428572
0.005714285712 0.9942857142879999 0.005714285712
0.9942857142879999 0.002285714285 0.997714285715
0.1666666667 0.8333333333 0.06666666668 0.93333333332
0.06666666668 0.93333333332 0.02666666667 0.9733333333300001
0.06250000001 0.93749999999 0.025 0.975 0.025 0.975 0.01
0.99 0.0238095238 0.9761904762 0.009523809520000001
0.99047619048 0.009523809520000001 0.99047619048
0.003809523808 0.996190476192 0.008928571425 0.991071428575
0.00357142857 0.99642857143 0.00357142857 0.99642857143
0.001428571428 0.998571428572 0.04444444446
0.9555555555400001 0.01777777778 0.98222222222 0.01777777778
0.98222222222 0.007111111112 0.992888888888 0.01666666667
0.98333333333 0.006666666668 0.993333333332 0.006666666668
0.993333333332 0.002666666667 0.997333333333 0.006349206349
0.993650793651 0.00253968254 0.99746031746 0.00253968254
0.99746031746 0.001015873016 0.998984126984 0.002380952381
0.997619047619 0.0009523809524 0.9990476190476
0.0009523809524 0.9990476190476 0.000380952381
0.999619047619</probabilities>
</cpt>
<cpt id="VBDC2">

```
  <state id="Pass" />
  <state id="Fail" />
  <parents>R2_Open R3_Open</parents>
  <probabilities>1 0 0.4 0.6 0.4 0.6 0.16 0.84</probabilities>
</cpt>
<cpt id="VCDC">
  <state id="Pass" />
  <state id="Fail" />
  <parents>R1_Open Q1_1 R4_Open R3_Open R2_Open</parents>
  <probabilities>1 0 0.4 0.6 0.79 0.21 0.16 0.84 0.8 0.2 0.15
      0.85 0.8100000000000001 0.19 0.06 0.94 0.82 0.18
      0.05714285712 0.94285714288 0.85 0.15 0.02285714285
      0.97714285715 0.6800000000000001 0.32 0.02142857142
      0.97857142858 0.92 0.08 0.008571428568 0.991428571432 0.91
      0.09 0.1066666667 0.89333333330.1066666667 0.8933333333
      0.04266666668 0.95733333332 0.1 0.9 0.4 0.6 0.4 0.6 0.016
      0.984 0.03809523808 0.96190476192 0.01523809523
      0.98476190477 0.01523809523 0.98476190477 0.006095238092
      0.993904761908 0.01428571428 0.98571428572 0.005714285712
      0.9942857142879999 0.005714285712 0.9942857142879999
      0.002285714285 0.997714285715</probabilities>
</cpt>
<cpt id="Q1_2">
  <state id="State0" />
  <state id="State1" />
  <parents>Q1_C_Open</parents>
  <probabilities>1 0 0.01 0.99</probabilities>
```

```
</cpt>
<cpt id="VBDC1">
  <state id="Pass" />
  <state id="Fail" />
  <parents>R1_Open Q1_2</parents>
  <probabilities>0.95 0.05000000000000004 0.3 0.7 0.96 0.04
      0.2 0.8</probabilities>
</cpt>
<cpt id="Q1_3">
  <state id="State0" />
  <state id="State1" />
  <parents>Q1_C_Open Q1_B_Open Q1_E_Open Q1_BE_Short</parents>
  <probabilities>0.9 0.1 0.1 0.9 0.99 0.01 0.11 0.89 0.96 0.04
      0.04 0.96 0.9300000000000001 0.07000000000000001 0.04 0.96
      0.95 0.05 0.01 0.99 0.9300000000000001 0.07000000000000001
      0.05 0.95 0.95 0.05 0.05 0.95 0.92 0.08 0.07000000000000001
      0.9300000000000001</probabilities>
</cpt>
<cpt id="VEDC1">
  <state id="Pass" />
  <state id="Fail" />
  <parents>R1_Open R4_Open Q1_3</parents>
  <probabilities>0.91 0.09 0.03 0.97 0.87 0.13 0.375 0.625 0.94
      0.06 0.2666666667 0.7333333333000001 0.84 0.16 0.1 0.9</
      probabilities>
</cpt>
<cpt id="Q1_4">
```

```
  <state id="State0" />
  <state id="State1" />
  <parents>Q1_BC_Short Q1_C_Short</parents>
  <probabilities>0.91 0.09 0.12 0.88 0.92 0.08 0.05 0.95</
      probabilities>
</cpt>
<cpt id="VEDC2">
  <state id="Pass" />
  <state id="Fail" />
  <parents>R2_Open R3_Open Q1_4</parents>
  <probabilities>0.88 0.12 0.07000000000000001 0.9300000000000001
       0.79 0.21 0.4 0.6 0.88 0.12 0.4 0.6 0.87 0.13 0.16 0.84</
      probabilities>
</cpt>
<cpt id="VBEVCFH">
  <state id="Pass" />
  <state id="Fail" />
  <parents>R4_Open R1_Open Q1_3</parents>
  <probabilities>0.85 0.15 0.07000000000000001 0.9300000000000001
       0.74 0.26 0.11 0.89 0.94 0.06 0.05 0.95 0.92 0.08 0.04
      0.96</probabilities>
</cpt>
<cpt id="VBEVCFL">
  <state id="Pass" />
  <state id="Fail" />
  <parents>R2_Open R3_Open Q1_4</parents>
```

```
  <probabilities>0.89 0.11 0.08 0.92 0.88 0.12 0.08 0.92 0.8 0.2
      0.06 0.94 0.6 0.4 0.07000000000000001 0.9300000000000001</
      probabilities>
</cpt>
```

Li-Ion Battery Conditional Probability Tables

```
<cpt id="Capacity">
  <state id="Good" />
  <state id="Candidate" />
  <probabilities>0.382822305899229 0.6171776941007711</
      probabilities>
</cpt>
<cpt id="Voltage_Measured">
  <state id="TESTOUTCOME0" />
  <state id="TESTOUTCOME1" />
  <state id="TESTOUTCOME2" />
  <state id="TESTOUTCOME3" />
  <state id="TESTOUTCOME4" />
  <parents>Capacity</parents>
  <probabilities>0.01262272089761571 0.1276297335203366
      0.0182328190743338 0.09350163627863488 0.748013090229079
      0.03221125943122461 0.04962275101567034 0.04004643064422519
      0.07254788160185723 0.8055716773070226</probabilities>
</cpt>
<cpt id="Temperature">
  <state id="TESTOUTCOME0" />
```

```
  <state id="TESTOUTCOME1" />
  <state id="TESTOUTCOME2" />
  <state id="TESTOUTCOME3" />
  <state id="TESTOUTCOME4" />
  <parents>Voltage_Measured Capacity</parents>
  <probabilities>0.8709677419354839 0.03225806451612903
     0.03225806451612903 0.03225806451612903 0.03225806451612903
     0.1130434782608696 0.8521739130434782 0.01739130434782609
     0.008695652173913044 0.008695652173913044 0.7256317689530686
      0.003610108303249098 0.08303249097472924 0.1588447653429603
      0.02888086642599278 0.12 0.005714285714285714 0.08
     0.5771428571428572 0.2171428571428571 0.6511627906976745
     0.02325581395348837 0.02325581395348837 0.02325581395348837
     0.2790697674418605 0.4295774647887324 0.007042253521126761
     0.007042253521126761 0.01408450704225352 0.5422535211267606
     0.9264705882352942 0.004901960784313725 0.004901960784313725
      0.004901960784313725 0.05882352941176471 0.2362204724409449
      0.003937007874015748 0.003937007874015748
     0.003937007874015748 0.7519685039370079 0.2593516209476309
     0.001870324189526185 0.08977556109725686 0.3229426433915212
     0.3260598503740648 0.003597122302158274 0.002877697841726619
      0.0007194244604316547 0.01546762589928058
     0.9773381294964029</probabilities>
</cpt>
<cpt id="Charge_Current">
  <state id="TESTOUTCOME0" />
  <state id="TESTOUTCOME1" />
```

```
<state id="TESTOUTCOME2" />
<state id="TESTOUTCOME3" />
<state id="TESTOUTCOME4" />
<parents>Voltage_Measured Capacity Temperature</parents>
<probabilities>0.03225806451612903 0.8387096774193548
    0.03225806451612903 0.03225806451612903 0.06451612903225806
    0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
    0.2 0.2 0.2 0.2 0.2 0.05882352941176471 0.5882352941176471
    0.05882352941176471 0.05882352941176471 0.2352941176470588
    0.009803921568627451 0.0196078431372549 0.009803921568627451
     0.009803921568627451 0.9509803921568627 0.1666666666666667
    0.1666666666666667 0.1666666666666667 0.1666666666666667
    0.3333333333333333 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
    0.01463414634146341 0.9560975609756097 0.004878048780487805
    0.004878048780487805 0.01951219512195122 0.2 0.2 0.2 0.2 0.2
     0.03703703703703704 0.8518518518518519 0.03703703703703704
    0.03703703703703704 0.03703703703703704 0.02083333333333333
    0.9166666666666666 0.02083333333333333 0.02083333333333333
    0.02083333333333333 0.08333333333333333 0.6666666666666666
    0.08333333333333333 0.08333333333333333 0.08333333333333333
    0.84 0.04 0.04 0.04 0.04 0.2 0.2 0.2 0.2 0.2
    0.05555555555555555 0.7777777777777777 0.05555555555555555
    0.05555555555555555 0.05555555555555555 0.009523809523809525
     0.8380952380952382 0.0761904761904762 0.038095238095238
    0.038095238095238 0.02380952380952381 0.5238095238095237
    0.09523809523809523 0.07142857142857143 0.2857142857142857
    0.03125 0.875 0.03125 0.03125 0.03125 0.2 0.2 0.2 0.2 0.2
```

0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.0625 0.4375 0.125
0.3125 0.0625 0.01538461538461539
0.4615384615384616 0.1538461538461539 0.1384615384615385
0.2307692307692308 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.1666666666666667 0.3333333333333333 0.1666666666666667
0.1666666666666667 0.1666666666666667 0.01234567901234568
0.1234567901234568 0.1728395061728395 0.1111111111111111
0.5802469135802468 0.005181347150259068 0.7046632124352332
0.1088082901554404 0.07253886010362695 0.1088082901554404
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.0625 0.125 0.0625 0.0625 0.6875 0.015625 0.015625 0.0625
0.109375 0.796875 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.005128205128205128
0.02564102564102564 0.005128205128205128
0.01025641025641026 0.9538461538461539 0.009523809523809525
0.3023809523809524 0.03571428571428572 0.03095238095238095
0.6214285714285714 0.1428571428571429 0.1428571428571429
0.1428571428571429 0.1428571428571429 0.4285714285714286
0.006756756756756757 0.972972972972973 0.006756756756756757
0.006756756756756757 0.006756756756756757
0.001915708812260536 0.0210727969348659
0.007662835249042145 0.2969348659003832 0.6724137931034483
0.00189753320683112 0.04174573055028463
0.005692599620493359 0.2106261859582543 0.7400379506641367
0.1428571428571429 0.07142857142857143 0.07142857142857143
0.07142857142857143 0.6428571428571428 0.08333333333333333
0.08333333333333333 0.08333333333333333 0.08333333333333333

0.6666666666666666 0.1666666666666667 0.3333333333333333
0.1666666666666667 0.1666666666666667 0.1666666666666667
0.02127659574468085 0.4468085106382979 0.2978723404255319
0.2127659574468085 0.02127659574468085 0.03454612274898934
0.0003675119441381845 0.07386990077177509
0.003307607497243661 0.8879088570378538</probabilities>
</cpt>
<cpt id="Measured_Current">
<state id="TESTOUTCOME0" />
<state id="TESTOUTCOME1" />
<state id="TESTOUTCOME2" />
<state id="TESTOUTCOME3" />
<state id="TESTOUTCOME4" />
<parents>Charge_Current Voltage_Measured Capacity Temperature</
parents>
<probabilities >0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.4285714285714286
0.1428571428571429 0.1428571428571429 0.1428571428571429
0.1428571428571429 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.84 0.04 0.04 0.04
0.04 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2

0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.5 0.125 0.125 0.125 0.125 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.3333333333333333 0.1666666666666667 0.1666666666666667
0.1666666666666667 0.1666666666666667 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.01020408163265306
0.01020408163265306 0.01020408163265306 0.9591836734693877
0.01020408163265306 0.8333333333333334 0.03333333333333333
0.06666666666666667 0.03333333333333333 0.03333333333333333
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.3571428571428571 0.2857142857142857
0.2142857142857143 0.07142857142857143 0.07142857142857143
0.1666666666666667 0.3333333333333333 0.1666666666666667
0.1666666666666667 0.1666666666666667 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.005 0.28 0.705
0.005 0.005 0.2 0.2 0.2 0.2 0.2 0.03703703703703704
0.03703703703703704
0.8518518518518519 0.03703703703703704 0.03703703703703704
0.1041666666666667 0.1041666666666667 0.6666666666666666
0.1041666666666667 0.02083333333333333 0.08333333333333333
0.08333333333333333 0.08333333333333333 0.6666666666666666
0.08333333333333333 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.05555555555555555 0.05555555555555555 0.7777777777777777
0.05555555555555555 0.05555555555555555 0.0108695652173913
0.0108695652173913 0.6086956521739131 0.358695652173913

0.0108695652173913 0.03846153846153846 0.03846153846153846
0.1538461538461539 0.7307692307692308 0.03846153846153846
0.03125 0.4375 0.46875 0.03125 0.03125 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.09090909090909091
0.09090909090909091 0.09090909090909091 0.6363636363636364
0.09090909090909091 0.02941176470588235
0.08823529411764705 0.1470588235294118 0.7058823529411764
0.02941176470588235 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.1666666666666667 0.1666666666666667 0.1666666666666667
0.3333333333333333 0.1666666666666667 0.07142857142857143
0.07142857142857143 0.07142857142857143 0.7142857142857142
0.07142857142857143 0.1714285714285714 0.02142857142857143
0.45 0.35 0.007142857142857143 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.1666666666666667
0.3333333333333333 0.1666666666666667 0.1666666666666667
0.1666666666666667 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2222222222222222
0.4444444444444444 0.1111111111111111 0.1111111111111111
0.1111111111111111 0.732824427480916 0.06106870229007633
0.06870229007633588 0.1297709923664122 0.007633587786259542
0.2 0.2 0.2 0.2 0.2 0.006756756756756757
0.006756756756756757 0.006756756756756757
0.006756756756756757 0.972972972972973 0.06666666666666667
0.06666666666666667 0.06666666666666667 0.06666666666666667
0.7333333333333333 0.03846153846153846 0.03846153846153846
0.3461538461538462 0.1538461538461539 0.4230769230769231
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.1666666666666667

0.1666666666666667 0.1666666666666667 0.1666666666666667
0.3333333333333333 0.04 0.04 0.04 0.04 0.84 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.08333333333333333 0.08333333333333333
0.08333333333333333 0.6666666666666666 0.08333333333333333
0.125 0.125 0.125 0.5 0.125 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.1666666666666667 0.1666666666666667 0.1666666666666667
0.3333333333333333 0.1666666666666667 0.07142857142857143
0.07142857142857143 0.07142857142857143 0.7142857142857142
0.07142857142857143 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.05555555555555555
0.05555555555555555 0.05555555555555555 0.7777777777777777
0.05555555555555555 0.04 0.04 0.04 0.84 0.04 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.125 0.125 0.125 0.5 0.125 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.05263157894736842 0.05263157894736842
0.05263157894736842 0.7894736842105263 0.05263157894736842
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.125 0.125 0.125
0.125 0.5 0.1428571428571429 0.1428571428571429
0.1428571428571429 0.2857142857142857 0.2857142857142857

0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.05555555555555555 0.05555555555555555
0.05555555555555555 0.05555555555555555 0.7777777777777777
0.004878048780487805
0.004878048780487805 0.004878048780487805 0.004878048780487805
0.9804878048780488 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.125 0.125 0.125 0.5 0.125
0.1428571428571429 0.1428571428571429 0.1428571428571429
0.4285714285714286 0.1428571428571429 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.1111111111111111 0.1111111111111111 0.1111111111111111
0.5555555555555556 0.1111111111111111 0.07692307692307693
0.07692307692307693 0.07692307692307693 0.6923076923076923
0.07692307692307693 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.07692307692307693
0.07692307692307693 0.07692307692307693 0.6923076923076923
0.07692307692307693 0.05555555555555555 0.05555555555555555
0.05555555555555555 0.7777777777777777 0.05555555555555555
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.09090909090909091
0.09090909090909091 0.09090909090909091 0.6363636363636364
0.09090909090909091 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2

0.2 0.2 0.2 0.2 0.2 0.1666666666666667 0.1666666666666667
0.1666666666666667 0.3333333333333333 0.1666666666666667
0.05882352941176471 0.05882352941176471 0.05882352941176471
0.7647058823529411 0.05882352941176471 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.006289308176100629
0.006289308176100629 0.006289308176100629
0.006289308176100629 0.9748427672955975
0.008695652173913044 0.008695652173913044
0.008695652173913044 0.008695652173913044
0.9652173913043478 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.07142857142857143 0.07142857142857143
0.07142857142857143 0.07142857142857143 0.7142857142857142
0.07692307692307693 0.07692307692307693
0.07692307692307693 0.1538461538461539 0.6153846153846154
0.1666666666666667 0.1666666666666667 0.1666666666666667
0.3333333333333333 0.1666666666666667 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.125 0.125 0.125 0.5 0.125
0.009900990099009901 0.009900990099009901
0.009900990099009901 0.9603960396039604
0.009900990099009901 0.1666666666666667 0.1666666666666667
0.1666666666666667 0.1666666666666667 0.3333333333333333
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.125 0.125 0.125
0.5 0.125 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.125 0.125 0.125 0.5
0.125 0.0625 0.0625 0.0625 0.75 0.0625 0.2 0.2 0.2 0.2 0.2

0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.05263157894736842
0.05263157894736842 0.05263157894736842 0.7894736842105263
0.05263157894736842 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.0196078431372549
0.0196078431372549 0.0196078431372549 0.9215686274509803
0.0196078431372549 0.04 0.04 0.04 0.84 0.04 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.06666666666666667 0.06666666666666667
0.06666666666666667 0.7333333333333333 0.06666666666666667
0.01818181818181818 0.01818181818181818
0.01818181818181818 0.9272727272727273 0.01818181818181818
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.005263157894736842 0.005263157894736842
0.005263157894736842 0.8736842105263157 0.1105263157894737
0.003773584905660377 0.003773584905660377
0.003773584905660377 0.9018867924528302
0.08679245283018867 0.1428571428571429 0.1428571428571429
0.1428571428571429 0.1428571428571429 0.4285714285714286
0.2 0.2 0.2 0.2 0.2 0.002816901408450704
0.008450704225352114 0.08450704225352113
0.9014084507042254 0.002816901408450704
0.002538071065989848 0.1091370558375635 0.1649746192893401
0.5177664974619289 0.2055837563451776 0.07692307692307693
0.07692307692307693 0.07692307692307693
0.07692307692307693 0.6923076923076923 0.08333333333333333
0.08333333333333333 0.08333333333333333

0.08333333333333333 0.6666666666666666 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.0004132231404958678
0.1669421487603306 0.02107438016528926 0.121900826446281
0.6896694214876032</probabilities>
</cpt>
<cpt id="Charge_Voltage">
<state id="TESTOUTCOME0" />
<state id="TESTOUTCOME1" />
<state id="TESTOUTCOME2" />
<state id="TESTOUTCOME3" />
<state id="TESTOUTCOME4" />
<parents>Temperature Charge_Current Capacity Measured_Current</
parents>
<probabilities>0.6000000000000001 0.1 0.1 0.1 0.1 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.8461538461538463 0.03846153846153846
0.03846153846153846 0.03846153846153846 0.03846153846153846
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.9727891156462585 0.006802721088435374
0.006802721088435374 0.006802721088435374
0.006802721088435374 0.951219512195122 0.01219512195121951
0.01219512195121951 0.01219512195121951 0.01219512195121951
0.9739130434782609 0.004347826086956522 0.004347826086956522
0.004347826086956522 0.01304347826086957 0.8840579710144928
0.02898550724637681 0.01449275362318841 0.01449275362318841
0.05797101449275362 0.2 0.2 0.2 0.2 0.2 0.5555555555555556
0.1111111111111111 0.1111111111111111 0.1111111111111111

0.1111111111111111 0.6000000000000001 0.1 0.1 0.1 0.1

0.6363636363636364 0.09090909090909091 0.09090909090909091

0.09090909090909091 0.09090909090909091 0.8571428571428571

0.03571428571428571 0.03571428571428571 0.03571428571428571

0.03571428571428571 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2

0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.717948717948718

0.1282051282051282 0.02564102564102564 0.02564102564102564

0.1025641025641026 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2

0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.7647058823529411

0.05882352941176471 0.05882352941176471 0.05882352941176471

0.05882352941176471 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2

0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.6666666666666666

0.1666666666666667 0.06666666666666667 0.03333333333333333

0.06666666666666667 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2

0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.7894736842105263

0.05263157894736842 0.05263157894736842 0.05263157894736842

0.05263157894736842 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2

0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2247191011235955

0.1385767790262172 0.1310861423220974 0.04868913857677903

0.4569288389513109 0.03703703703703704 0.03703703703703704

0.03703703703703704 0.03703703703703704 0.8518518518518519

0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2

0.9444444444444444 0.01388888888888889

0.01388888888888889 0.01388888888888889 0.01388888888888889

0.07692307692307693 0.07692307692307693 0.07692307692307693

0.07692307692307693 0.6923076923076923 0.2 0.2 0.2 0.2 0.2

0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2

0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.3333333333333333 0.1666666666666667
0.1666666666666667 0.1666666666666667 0.1666666666666667
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.1428571428571429 0.1428571428571429
0.1428571428571429 0.1428571428571429 0.4285714285714286
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.9603960396039604 0.009900990099009901
0.009900990099009901 0.009900990099009901
0.009900990099009901 0.08333333333333333
0.08333333333333333 0.08333333333333333 0.08333333333333333
0.6666666666666666 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.8518518518518519

0.03703703703703704 0.03703703703703704 0.03703703703703704
0.03703703703703704 0.2 0.2 0.2 0.2 0.2
0.006756756756756757 0.006756756756756757
0.006756756756756757
0.972972972972973 0.006756756756756757 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.7777777777777777 0.05555555555555555
0.05555555555555555 0.05555555555555555 0.05555555555555555
0.2 0.2 0.2 0.2 0.2 0.1666666666666667 0.1666666666666667
0.1666666666666667 0.1666666666666667 0.3333333333333333
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.3333333333333333 0.1666666666666667 0.1666666666666667
0.1666666666666667 0.1666666666666667 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.5555555555555556 0.1111111111111111
0.1111111111111111 0.1111111111111111 0.1111111111111111
0.5555555555555556 0.1111111111111111 0.1111111111111111

0.1111111111111111 0.1111111111111111 0.8888888888888888
0.02777777777777778 0.02777777777777778 0.02777777777777778
0.02777777777777778 0.5555555555555556 0.1111111111111111
0.1111111111111111 0.1111111111111111 0.1111111111111111
0.06666666666666667 0.06666666666666667 0.2
0.3333333333333333 0.3333333333333333 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2
0.9333333333333334 0.01666666666666667 0.01666666666666667
0.01666666666666667 0.01666666666666667 0.8947368421052631
0.02631578947368421 0.02631578947368421
0.02631578947368421 0.02631578947368421 0.04 0.04 0.04
0.04 0.84 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.25 0.375 0.125 0.125
0.125 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.6666666666666666 0.08333333333333333
0.08333333333333333 0.08333333333333333
0.08333333333333333 0.7777777777777777 0.05555555555555555
0.05555555555555555 0.05555555555555555
0.05555555555555555 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.9685534591194969 0.01257861635220126
0.006289308176100629 0.006289308176100629
0.006289308176100629 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.5 0.125 0.125 0.125 0.125
0.7142857142857142 0.07142857142857143 0.07142857142857143
0.07142857142857143 0.07142857142857143 0.2 0.2 0.2 0.2
0.2 0.1428571428571429 0.1428571428571429

0.1428571428571429 0.4285714285714286 0.1428571428571429
0.02941176470588235 0.02941176470588235
0.02941176470588235 0.8823529411764706 0.02941176470588235
0.1481481481481481 0.7962962962962963 0.02160493827160494
0.0308641975308642 0.00308641975308642 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.5 0.125 0.125 0.125 0.125 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.9387755102040816 0.03061224489795918 0.01020408163265306
0.01020408163265306 0.01020408163265306 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2
0.3333333333333333 0.1666666666666667 0.1666666666666667
0.1666666666666667 0.1666666666666667 0.6923076923076923
0.07692307692307693 0.07692307692307693
0.07692307692307693 0.07692307692307693 0.6666666666666666
0.1904761904761905 0.04761904761904762
0.04761904761904762 0.04761904761904762
0.06666666666666667 0.06666666666666667
0.06666666666666667 0.7333333333333333 0.06666666666666667
0.3333333333333333 0.1666666666666667 0.1666666666666667
0.1666666666666667 0.1666666666666667 0.5 0.125 0.125
0.125 0.125 0.5 0.125 0.125 0.125 0.125 0.875 0.03125
0.03125 0.03125 0.03125 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.4285714285714286 0.1428571428571429 0.1428571428571429

0.1428571428571429 0.1428571428571429 0.3333333333333333
0.1666666666666667 0.1666666666666667 0.1666666666666667
0.1666666666666667 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.8095238095238095
0.04761904761904762 0.04761904761904762
0.04761904761904762 0.04761904761904762 0.9804878048780488
0.004878048780487805 0.004878048780487805
0.004878048780487805 0.004878048780487805 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.5555555555555556 0.1111111111111111 0.1111111111111111
0.1111111111111111 0.1111111111111111 0.9652173913043478
0.008695652173913044 0.008695652173913044
0.008695652173913044
0.008695652173913044 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2 0.2
0.2 0.2 0.2 0.2 0.2 0.7647058823529411
0.05882352941176471 0.05882352941176471
0.05882352941176471 0.05882352941176471
0.6666666666666666 0.08333333333333333
0.08333333333333333 0.08333333333333333
0.08333333333333333 0.2 0.2 0.2 0.2 0.2
0.02127659574468085 0.02127659574468085
0.8936170212765957 0.0425531914893617 0.02127659574468085
0.01449275362318841 0.01449275362318841
0.4347826086956522 0.5217391304347826 0.01449275362318841
0.6238532110091744 0.1651376146788991
0.01834862385321101 0.1284403669724771
0.06422018348623854 0.07058823529411765

0.01176470588235294 0.01176470588235294
0.01176470588235294 0.8941176470588235 0.2 0.2 0.2 0.2
0.2 0.002450980392156863 0.002450980392156863
0.9877450980392156 0.004901960784313725
0.002450980392156863 0.01818181818181818
0.01818181818181818 0.8909090909090909
0.05454545454545454 0.01818181818181818
0.9481765834932822 0.03646833013435701
0.007677543186180422 0.003838771593090211
0.003838771593090211 0.5014766686355582
0.02953337271116362 0.03248670998227998
0.009450679267572357 0.4270525694034258</probabilities>
</cpt>