Safety of Flight Prediction for Small Unmanned Aerial Vehicles Using Dynamic Bayesian Networks

Meghan C. Burns

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Aerospace Engineering

Craig A. Woolsey, Chair
Mayuresh J. Patil
Richard E. Adams

May 09, 2018
Blacksburg, Virginia

Keywords: Dynamic Bayesian Network, Unmanned Aerial Systems, Risk

Safety of Flight Prediction for Small Unmanned Aerial Vehicles Using Dynamic Bayesian Networks

Meghan C. Burns

Abstract

This thesis compares three variations of the Bayesian network as an aid for decision-making using uncertain information. After reviewing the basic theory underlying probabilistic graphical models and Bayesian estimation, the thesis presents a user-defined static Bayesian network, a static Bayesian network in which the parameter values are learned from data, and a dynamic Bayesian network with learning. As a basis for the comparison, these models are used to provide a prior assessment of the safety of flight of a small unmanned aircraft, taking into consideration the state of the aircraft and weather. The results of the analysis indicate that the dynamic Bayesian network is more effective than the static networks at predicting safety of flight.

Safety of Flight Prediction for Small Unmanned Aerial Vehicles Using Dynamic Bayesian Networks

Meghan C. Burns

Abstract (General Audiences)

This thesis used probabilities to aid decision-making using uncertain information. This thesis presents three models in the form of networks that use probabilities to aid the assessment of flight safety for a small unmanned aircraft. All three methods are forms of Bayesian networks, graphs that map causal relationships between random variables. Each network models the flight conditions and state of the aircraft; two of the networks are static and one varies with time. The results of the analysis indicate that the dynamic Bayesian network is more effective than the static networks at predicting safety of flight.

Acknowledgments

First, I would like to thank my advisor, Dr. Craig Woolsey for guiding me through the project and helping me to grow as both a student and a professional. I would also like to thank Dr. Mayuresh Patil and Mr. Richard Adams for serving on my committee and inspiring an interest in risk analysis. I have special gratitude to all of my friends and family who helped me make edits and supported me without question.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview of Problem

Virginia Tech student pilots frequently fly small unmanned aerial systems (sUAS) for research at Kentland Experimental Aerial Systems (KEAS) Laboratory on Kentland Farm. When planning missions, it is critical that the pilot decides whether or not a mission is safe to fly. At the small scale of a research laboratory, this particular challenge can serve as an analog for much weightier decisions faced daily by the private, commercial, and military aviation communities.

A pilot must consider many factors that affect flight safety before he or she will decide to proceed. For example, pilots will deflect all control surfaces before a flight to know that the actuators are fully functional. Without testing the actuators, there is a higher chance of taking off before realizing there is an issue. A pilot must also consider how many people will be around and where they will be standing during the flight to reduce the possibility of harm to the crew or bystanders. A pilot must be sure that there is communication between the transmitter and the aircraft over the full operational range, and it is important to consider how the aircraft will react if that link is lost. It is a good idea to check the propulsion system,

including any batteries and motors, of the vehicle before flying because a loss of thrust could be disastrous. Knowing the weather forecast is crucial when flying small aircraft; sUAS flight activities at the KEAS Lab are limited to visual flight rules (VFR) conditions with relatively calm winds. These factors are common for many sUAS, but each individual vehicle will have its own additional factors that affect the flight safety.

Often, the decision to fly is made when a pilot is experienced enough to recognize when certain conditions are conducive to a safe and successful flight. However, humans have subjective judgment so given identical circumstances, two pilots may arrive at different conclusions about mission safety. Although expertise is a valuable resource when a pilot is deciding whether or not to fly, humans have limited capabilities to process and account for historical data. Another issue with relying solely on pilot experience, especially with a transient population of students, is that when the student pilot graduates and leaves the university, his expertise is lost to others in the research group. This thesis describes a method for automated decision support that can help a pilot to make more informed and repeatable decisions about whether it is safe to fly a sUAS at the KEAS Lab.

1.2 Approach

The problem of deciding whether or not it is safe to fly is made challenging because the information necessary to make this decision is uncertain. A Bayesian network can help a person make rational decisions given uncertain information. Briefly, a Bayesian network is a directed, acyclic graph whose nodes represent conditions and whose edges represent causal relationships. In a simple illustration involving two nodes, the first node might represent "*Soil Quality*" while the second represents "*Crop Yield*". The quality of the soil has an effect on how many plants grow. Knowing information about *Soil Quality* can give a farmer information on what to expect the *Crop Yield* to be.

Bayesian networks have been successfully introduced to the medical diagnostics field. Much

like the case of an expert pilot deciding whether or not it is safe to fly, diagnosing patients typically relies on the subjective judgment of doctors. Expertise accumulated over a career is lost to the profession upon the doctor's retirement. Bayesian networks are implemented to combine the data from past patients and knowledge from adept doctors to successfully diagnose patients at a higher rate than doctors can achieve alone. Doctors are able to use Bayesian networks to explore the likelihood of the patient having a certain disease given the patient's symptoms. The doctor can then make an informed decision based on the probabilities output by the network. Knowing that Bayesian networks have been helpful to doctors deciding a medical diagnosis, it may be possible to apply a Bayesian network to the pilot's "safe to proceed" problem by finding similarities between the two situations. In this analogy, the sUAS is equivalent to the patient and the various conditions affecting flight safety are synonymous to patient symptoms.

However, there is a significant difference in the sUAS problem from medical diagnostics. Once a diagnosis is made there is typically no further analysis and the doctor moves on to treatment. When deciding if it is safe enough to fly, a pilot must consider the flight conditions in the recent past, what is currently happening, and how conditions may change through the future over the duration of the possible flight. This issue suggests the need for a dynamic Bayesian network, where the likelihoods associated with causal relationships vary in time, as this allows the user to input information that they know at various time steps to output probabilities for future time steps. Dynamic networks, like static networks, allow for historical data and expertise to be useful. However, dynamic networks are able to relate events throughout time whereas static networks do not have the means to do so. The purpose of this thesis is to explain how to implement a dynamic Bayesian network into the sUAS problem and to assess whether dynamic Bayesian networks will actually be a useful tool for assessing the safety of a planned flight operation.

1.3 Literature Review

The main source of inspiration for this thesis is Daphne Koller's book *Probabilistic Graphical Models: Principles and Techniques* [9] and her corresponding course. Dr. Koller is a professor at Stanford University who successfully uses Bayesian networks to improve the field of medical diagnostics. The problems of interest in the medical field are so large and complex that it can be difficult or impossible for a human analyst to construct a network model that expresses the correct causal relationships and to determine the conditional probability parameters that quantify these relationships. Thus, in medical applications, automated network learning and parameter learning are essential for constructing effective Bayesian network models. As briefly discussed in Section 1.2, the safe-to-proceed example is treated in analogy to a diagnosis. In the same way a doctor can look at the probabilities produced by a Bayesian network to decide how to treat a patient, a pilot can look at the probabilities and decide to ground the sUAS or not.

It is very common to apply dynamic Bayesian networks (DBNs) to unmanned vehicles. However, the typical application is local path planning. The method by which DBNs are used for local path planning involves "unrolling" a discrete-time sequence of Bayesian networks whose nodes are not only interconnected at each instant in time but also from time-step to time-step. The DBN can use sensors to make observations, then determine what the best path will be based on threats and obstacles. Gao, Ren, and Chen [6] were able to successfully use a type of DBN to create an adaptive path for a UAS when there are unexpected obstacles. In cases used for local path planning, it is suitable to unroll the DBN to only a few time steps. Taking more than a few time steps can cause lengthy in-flight computation and, in any case, the predictions would not be very accurate too far in the future if the environment is dynamic. The DBNs used in path planning are run in real-time as the vehicle moves through its environment, which is not the case in the safe-to-proceed example. As a result, the network in the safe-to-proceed example is less constrained by computation time. The time steps can also be further apart in the safe-to-proceed example than in path planning

applications because time scale at which conditions, such as weather, vary in the flight planning problem is much larger than the time scale at which a robot encounters unknown obstacles.

Three students, de Kock [4], Haingura [7], and Kent [8] from the University of Cape Town performed a very similar analysis as this thesis on using dynamic Bayesian networks for weather forecasting through GeNIe and SMILE. The students gathered data from weather stations around South Africa to use for training. The forecasting capabilities of their system are not very accurate beyond one day, which indicates that a 24 hours of prediction in the safe-to-proceed example is just within the plausible range of accurately predicting weather. Predicting weather is no easy task, but accurately predicting outcomes in the near future is not impossible.

Another instance of dynamic Bayesian networks used for predicting weather is the work of Doshi-Velez who introduces a variant of DBNs called "infinite dynamic Bayesian networks" and then uses weather prediction to validate their effectiveness [5]. Doshi-Velez describes an infinite dynamic Bayesian network (iDBN) as "a nonparametric, factored state-space model that generalizes dynamic Bayesian networks" which is able to explore anything that may be hidden in a DBN like nodes and structures. The iDBN was able to successfully rebuild known weather patterns in the continental United States of America given training data. The iDBN method is complex compared to what is used for the safe-to-proceed example, suggesting it may be impractical for the given application.

Using a tool to assess risk before flying is recommended by the Federal Aviation Administration (FAA). The FAA has a tool called the Flight Risk Assessment Tool (FRAT) [1]. FRAT asks about things like how the pilot is feeling, how fast the winds are, how many hours a pilot has had flying the vehicle, and many other relevant questions. In answering these questions, the pilot checks boxes with corresponding numerical values. The pilot then matches the sum of the checked values to a chart that shows levels of risk. A higher score is associated with a higher risk. The risk evaluation criteria used by the FAA for general

aviation pilots are shown in Table 1.1.

Table 1.1: The risk criteria formed by the FAA for FRAT.

Pilot	Experience (Hours)	Low Risk	Moderate Risk	High Risk
VFR	< 100 in type	5 to 15	15 to 20	> 20
	> 100 in type	15 to 20	20 to 25	> 25
IFR	< 100 in type	20 to 25	25 to 30	> 30
	> 100 in type	25 to 30	30 to 35	> 35

Table 1.1 is copied directly from the FAA's document describing FRAT. The "Pilot" column is the distinction of whether the pilot is trained to fly under Visual Flight Rules (VFR) or Instrument Flight Rules (IFR). When a pilot is more experienced in the given aircraft type and has more training, the pilot can accept more risk. This is very similar to what is done in the safe-to-proceed example explored in this thesis, but FRAT does not incorporate probabilistic information, does not have the predictive capability of a DBN, and is not tailored towards sUAS pilots. FRAT is most comparable to the non-learning, static Bayesian network of Section 3.3.1 which is the most limited of the three approaches considered here. A major goal of this thesis is to show that a better tool can be created for the same purpose as FRAT by using DBNs.

Chapter 2

Dynamic Bayesian Networks: Background

2.1 Modeling a System Using Probabilistic Graphs

A system is a device or process, or a collection of them, whose state is affected by some set of inputs and may be observed through some set of outputs. Engineers develop and use mathematical models of systems for a variety of reasons. For example, models of dynamical systems can be used for control design, state estimation, and prediction. In the example considered in this thesis, a system model is needed for prediction. In general, a system model is a collection of subsystem models whose various inputs and outputs may be interconnected within the larger system. A subsystem's state evolves according to some behavioral rules, which define the subsystem's natural evolution as well as the effect of any subsystem inputs. A system model may be deterministic or uncertain. For a deterministic model, the behavior of various subsystems in response to their initial state and inputs is precisely known. For an uncertain model, the behavior is unknown but might be characterized using probability theory. In the context of deciding whether or not it is safe to perform a particular flight mission, the model represents the environment and conditions an sUAS would be operating

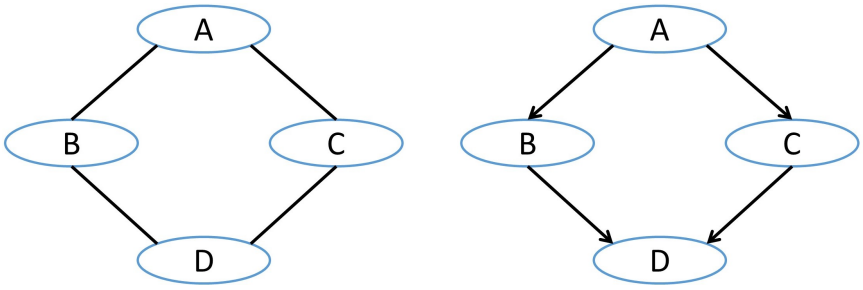
in and the prediction is the likelihood that the mission will fail.

In a Bayesian network model of an uncertain system, each node represents a subsystem and the state of that subsystem is a collection of absolute or conditional probabilities. Absolute probabilities define the likelihood that a subsystem is in a particular state and produces a particular output without reference to any other process. Conditional probabilities represent the likelihood that a subsystem is in a particular state given knowledge about the state of some causally related subsystem. An "observation" is information about the system from an external source, such as a sensor or human. An observation of a particular state will change its probability. For example, there is a variable for the probability of having a cloudy day and it is true that there is a cloud in the sky. This is noted in the model as an observed variable and the variable's output is 1 because there is a 1/1 probability the current day is cloudy given the observation.

Declarative representation is when a model of a system is built that is able to separate knowledge and reasoning. This type of model representation allows algorithms that are completely separate from the model to be applied to the system and is advantageous because the model can be improved without needing to alter the algorithms or vice-versa. As with any model, there is uncertainty. Observations cannot always be made of the entire system so there is uncertainty in the actual state of the system being observed. Observations themselves can have uncertainty because of the inability to observe completely or properly. Uncertainty leads to the necessity of making conclusions using the knowledge of both what is probable and what is possible. Probability makes it possible to still consider outcomes in the model that are not impossible but are very unlikely to occur. Improbable outcomes can be grouped together and still considered. Grouping improbable outcomes is more realistic than ignoring them, yet simpler than considering each unlikely event individually. For example, it is very unlikely to cut into a cake to discover that the flavor is roast beef, tomato soup, or hot salsa. Instead of listing these outcomes of possible flavors individually, they can be grouped together as an outcome called "Savory". The set of possible outcomes could be {"Chocolate","Vanilla", "Fruity", or "Savory"} which is much simpler than listing each of

the unlikely savory possibilities. Probabilistic models built using this approach are extremely useful in modeling real world events.

Structured probabilistic models represent systems whose state can be characterized by random variables. The values of these state variables can be helpful in estimating values of other random variables given a joint probability distribution. A type of structured probabilistic is the probabilistic graphical model (PGM). PGMs provide a graph based approach for representing systems with a large number of random variables in a less onerous way than a complex joint probability distribution. Recall that a graph is a collection of nodes connected by directed edges. For a PGM, the nodes represent subsystems and the edges may represent causal relationships. These models are typically easy for a human to understand and interact with and are very useful for inferring a current state given a set of observations. PGMs can be built using a combination of human judgment and expertise, physical principles, and quantitative measurements. Each random variable is associated with a node in the model. In the pictorial representations of graphs, or networks, used in this paper, a node will be a circle or oval. Edges, also known as arcs, between nodes represent the relationship between the two variables and will be denoted with arrows, indicating the direction in which information flows within the model. Two common forms of PGM are Bayesian networks (BNs) and Markov networks (MNs). These PGMs use directed and undirected edges respectively. Dependencies and independencies between variables are determined through the edges. A proper factorization of the probability distributions emerges from the independencies. Examples of both types of graphical models are shown in Figure 2.1.



(a) A Markov network example. (b) A Bayesian network example.

Figure 2.1: Examples of PGMs.

Information can flow in any direction in a Markov network. For example, if each node represents a sick or healthy child, Figure 2.1(a) can represent the spread of a cold. Child A and child D do not know each other and child B and child C do not know each other. If the children do not meet, the cold cannot be transmitted directly between two children. Child B has a cold and can spread germs to child A or D. Child A or D is then able to give the cold to child C. If child B gives the cold to child D and then recovers, child B can catch the cold back from child D. In a similar scenario, Figure 2.1(b) represents a network which allows the possibility that a child gains immunity to the cold. Child A has already had the cold and can no longer catch or transmit it. A cold from child B will never be able to infect child C. If child B gives the cold to child D, child B will not be susceptible to the illness again, as in Figure 2.1(a).

2.2 Probability Theory

To understand Bayesian networks, some basics of probability theory must be clear. Starting off, a probability is the unitless measurement of likelihood of an event or outcome occurring. For example, an opaque jar contains 10 cookies; 3 are oatmeal raisin cookies and 7 are chocolate chip cookies. A cookie is taken from the jar at random. The probability of a cookie being chocolate chip is $\frac{7}{10}$ or 0.7. In this example, *Cookie Flavor* is what is known

as a random variable, which is a variable that represents a random event. All variables in Bayesian networks are random variables so the occurrence of outcomes is random. However, like the cookie example, each outcome has an associated probability that informs an educated guess of what the outcome may be. In this thesis, all random variables have a discrete distribution, meaning that there is a finite number of possible outcomes. The alternative to a discrete distribution is a continuous distribution where possible outcomes are defined by some subset of all real numbers.

Consider the set of random variables $\{A, B\}$. The fact that the values of random variables can be related motivates the notion of joint probability. A joint probability of random variables A and B, $P[A, B]$, is the probability that both A and B are true. The joint probability of A and B can also be expressed as the product of two individual probabilities through the chain rule:

$$P[A, B] = P[A]P[B] \tag{2.1}$$

The chain rule can be extended through an infinite set of random variables. Returning to the cookie example, imagine the first cookie drawn from the jar is oatmeal raisin so it is put back in the jar. Luckily, the next cookie drawn is chocolate chip. The probability that an oatmeal raisin cookie is drawn and then a chocolate chip cookie (or vice versa) is $0.3 \times 0.7 = 0.21$

If B is observed, its effect on the probability of A is called the conditional probability, $P[A|B]$. If $P[A|B] = P[A]$ then A and B are independent of each other. If $P[A|B]$ equals any other value, the variables are not independent. The general expression for the conditional probability is:

$$P[A|B] = \frac{P[A, B]}{P[B]} \tag{2.2}$$

Again, this can be explained by the cookie example. The probability of the second cookie being chocolate chip given that the first cookie is oatmeal raisin is $\frac{0.21}{0.3} = 0.7$. Because P[ChocolateChip|OatmealRaisin] = P[ChocolateChip], the events are independent. Now say that first cookie drawn was not put back in the jar and is instead thrown in the trash.

The probability of drawing a chocolate chip cookie on the second try is now $\frac{7}{9}$ which is dependent on the outcome of the first draw because the total number of cookies in the jar changes.

A relationship between $P[A|B]$ and $P[B|A]$ is given by Bayes' rule[1]:

$$P[A|B] = \frac{P[B|A]P[A]}{P[B]} \tag{2.3}$$

Bayes' rule is the base of the graphical models previously discussed and is very important in probability theory. An example of how to apply Bayes' rule to solve a problem is expressed in Example 2.2.1.

Example 2.2.1. *Bayes' Rule*

A litter of 10 white puppies is born. The vet determines that 4 of the puppies are male and 6 are female, which is representative of the entire population of this particular breed. All puppies of this breed are born white and 55% will develop spots as they mature. 70% of adult spotted dogs are female. What is the probability that if the family selects a female puppy, it will develop spots?

To begin, define the probability that is sought: P[Spots|Female]
Next, organize known facts:

- P[Female|Spots] = 0.70 & P[Male|Spots] = 0.30
- P[Female] = $\frac{6}{10}$ & P[Male] = $\frac{4}{10}$
- P[Spots] = 0.55 & P[Plain] = 0.45

Applying Bayes' rule:
$$P[\text{Spots}|\text{Female}] = \frac{\text{P[Female|Spots]P[Spots]}}{\text{P[Female]}} = \frac{(0.70)(0.55)}{(\frac{6}{10})} \approx 0.64$$

A conditional probability distribution (CPD) is the distribution of one variable given the possible values of another. In this thesis, CPD will refer to a table of probabilities for possible values. Table 2.1 shows an example of a CPD for $P[B|A]$ in tabular form.

[1]Reverend Thomas Bayes first expressed Bayes' rule in 1763 [2]

Table 2.1: Example of a CPD for $P[B|A]$

	$B = b_1$	$B = b_2$	$B = b_3$
$A = a_1$	0.25	0.50	0.25
$A = a_2$	0.10	0.30	0.60
$A = a_3$	0.75	0.25	0.00

From the CPD for $P[B|A]$, the probabilities of B are determined from knowing information about A. For example, if $A = a_1$ then the probability that $B = b_2$ is 0.50. Note that each row sums to 1 signifying that if A is assigned a value, B B will take one of the three given values with probability one. By assigning a definite value to A, any uncertainty about the variable is removed: $P[A] = 1$. Thus, by the definition of conditional probability, $P[B] = P[B|A]P[A] = P[B|A] \times 1 = P[B|A]$.

2.3 Bayesian Networks

As discussed briefly before in the description of structured probabilistic models, a Bayesian network uses directed edges to define causal relationships between variables. Nodes are associated with each variable and represent conditions. In a Bayesian network, the visualization of a series of interconnected nodes is called a graph, G. Graphs compactly represent both conditional dependencies between variables and a factorized joint distribution. It is assumed in this thesis that G is a good representation of the actual system. Markov networks are also represented with graphs, but for Bayesian networks the graphs are acyclic meaning that information flows in only one direction. Relationships between nodes are shown by the edges. In the language of graphical modeling, a "parent" node has a causal effect on a "child" node, which is indicated by an arrow (edge) emanating from the parent and terminating at the child; see Figure 2.2. There is no limit to how many parents or children a node can have.

When a node has no children, it is called a "leaf" node.

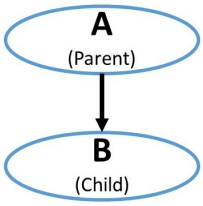
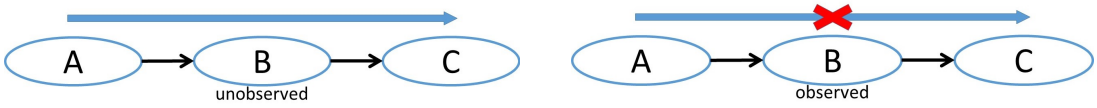


Figure 2.2: Visual representation of the relationship between a parent and child node

Knowing which nodes are independent of each other is necessary to properly understand the probabilities produced by a BN. Active flow of influence between two variables means that the state or outcome of one variable is able to affect the outcome of another variable, and these variables are not independent. Say there is a chain of three nodes as shown in Figure 2.3: A, B, and C. Node A is a parent to one child, B. Node B has one parent, A, and one child, C. Node B is the only parent to C. If B is unobserved, then the flow of influence from A to C is in tact, as in Figure 2.3(a). If B is observed then C is conditionally independent from A because the flow of influence is inactive, as in Figure 2.3(b). When the flow of influence between two nodes is inactive, they are considered d-separated [9, pg. 69-72].



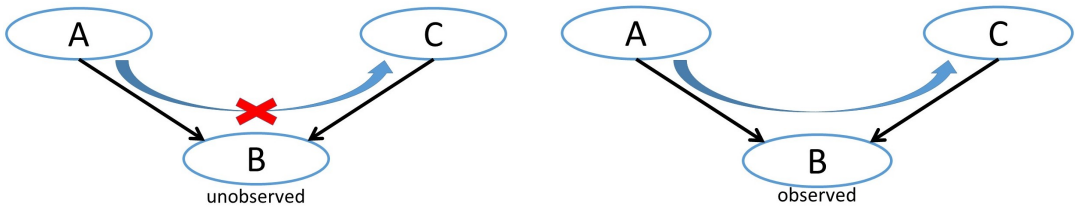
(a) $C \not\perp A$ (b) $C \perp A|B$

Figure 2.3: The flow of influence through a chain of nodes

To further explain the flow of influence through a chain, imagine A denotes *lightning has hit a tree*, B denotes *The tree falls*, and C denotes *a sound is made*. An Appalachian Trail hiker stays in a shelter during a storm. While sleeping, he is woken by a loud noise. He thinks it is possible that lightening has struck a tree, causing it to fall and make a loud sound. The

flow of influence connects A to B and then C, linking the sound made back to the initial lightening strike as in Figure 2.3(a). Another hiker slept through the noise and wakes up in the morning to see a tree has fallen near the shelter. Unaware of the storm, he sees no evidence of what has caused the tree to fall. B now has a probability of 1. He assumes that by falling, this tree has made a sound although he has no observation; the flow of influence between B and C is still active. Because the probability of B cannot change after being observed, it does not matter what the probability of lightening striking the tree is when the second hiker makes his assumption that the tree falling made a sound; his assumption is made exclusively from knowing that the tree has fallen. This reveals that A is unable to influence C, as in Figure 2.3(b)

Now, say that the edge from node B to node C is reversed so C becomes another parent of B, as in Figure 2.4. Information flowing to the child from a parent cannot flow back up into another parent of that child, so A and C are independent and D-separated, illustrated in Figure 2.4(a). Observing B in this case reconnects the flow of influence and activates what is called a v-structure [9, pg. 71], shown in Figure 2.4(a).



(a) $C \perp A$ (b) $C \not\perp A|B$

Figure 2.4: The flow of influence through a v-structure

Continuing on with the tree example will help explain the flow of influence in Figure 2.4. A and B do not change, but now suppose C denotes *the trunk is rotten.* Assume that lightening strikes and wood rot are the only reasons a tree can fall. Without observing if the tree has fallen, the event of lightening striking the tree is independent of the trunk of that tree being rotten. If the first hiker knows that lightening struck a tree, the probability of the trunk

being rotten is unchanged if he has not seen whether or not the tree has fallen; there is no flow of influence between A and C. The second hiker knows that the tree has fallen. The cause of the tree falling could be either a lightening strike or a rotten trunk. The second hiker goes to the fallen tree and sees that the wood inside is not rotten, giving C a probability of zero. The only remaining explanation for the tree falling is a lightening strike. Because knowing the probability of C affected the probability of A, they are not independent and the flow of influence is active between A and C.

Each random variable node has a CPD that assigns probabilities to each possible outcome based on the values of the parent nodes. The probabilities in the CPDs are then used in the chain rule for Bayesian networks to calculate a probability for all variables in the network.

$$P[X_1, \ldots, X_n] = \prod_{i=1}^{n} P(X_i | Pa_{X_i}^G) \tag{2.4}$$

where $Pa_{X_i}^G$ is a parent of the i^{th} node in G. Each factor $P(X_i|Pa_{X_i}^G)$ is a CPD. A distribution P with the same domain of random variables $(X_1, \ldots, X_n)$ as G factorizes according to G if P can be conveyed in the form above.

Example 2.3.1. *Bayesian Network Probabilities*

Consider the graph in Figure 2.5. The CPDs for each node are listed in Table 2.2, Table 2.3, Table 2.4, and Table 2.5

a) Use the chain rule for Bayesian networks to calculate the probabilities of each variable.

b) How are the probabilities affected if A is observed to be a_2?

c) How are the probabilities affected if D is observed to be d_2?

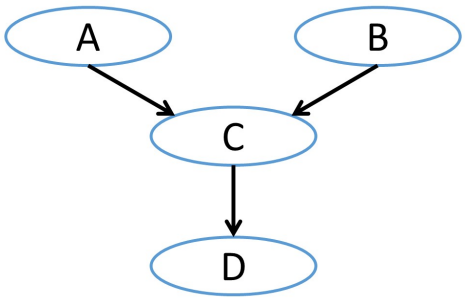


Figure 2.5: Simple graph structure for Example 2.3.1

Table 2.2: Probability distributions for $P[A]$

$A = a_1$	$A = a_2$
0.2	0.8

Table 2.3: Probability distribution for $P[B]$

$B = b_1$	$B = b_2$	$B = b_3$
0.6	0.3	0.1

Table 2.4: CPD for $P[C|A,B]$

		$C=c_1$	$C=c_2$
$A=a_1$	$B=b_1$	0.3	0.7
	$B=b_2$	0.4	0.6
	$B=b_3$	0.5	0.5
$A=a_2$	$B=b_1$	0.7	0.3
	$B=b_2$	0.6	0.4
	$B=b_3$	0.2	0.8

Table 2.5: CPD for $P[D|C]$

	$D=d_1$	$D=d_2$
$C=c_1$	0.75	0.25
$C=c_2$	0.45	0.55

a)

Starting with A and B is simple because their probabilities are simply the values in Table 2.2 and Table 2.3 respectively. At C, the probabilities are joint probabilities of A, B, and C so the chain rule of Bayesian networks is necessary. Once the joint probabilities are found, sum over the other variables to get the probabilities sought. This summation is called marginalization. For example, to marginalize B for $P[A,B]$, sum over B. Finding the joint probabilities and marginalization is all done in one step below by applying the summation directly to the chain rule of Bayesian networks:

$$P[C] = \sum_A \sum_B P[A]P[B]P[C|A,B]$$

Expanding this summation gives the results of conditioning on c_i:

$$P[C = c_1] = \sum_{i=1}^{2} \sum_{j=1}^{3} P[A_i]P[B_j]P[C = c_1|A_i, B_j] = 0.5660$$

$$P[C = c_2] = \sum_{i=1}^{2} \sum_{j=1}^{3} P[A_i]P[B_j]P[C = c_2|A_i, B_j] = 0.4340$$

Applying the chain rule and marginalization to find $P[D]$ is a simple extension of the previous calculations because now $P[C]$ is known.

$$P[D] = \sum_{C} P[C]P[D|C]$$

Similar to $P[C]$:

$$P[D = d_1] = \sum_{i=1}^{2} P[C_i, A, B]P[D = d_1|C_i] = 0.6198$$

$$P[D = d_2] = \sum_{i=1}^{2} P[C_i, A, B]P[D = d_2|C_i] = 0.3802$$

It can be helpful to replace the nodes in the graph by their respective probability tables because it is easier to visualize how observations can change the probabilities of other variables. This is done in Figure 2.6.

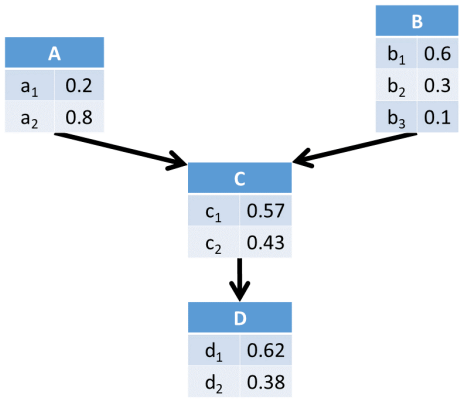


Figure 2.6: Probabilities for Example 2.3.1

b)

If A is observed to be a_2, then $P[A = a_1] = 0$ and $P[A = a_2] = 1$. Because B is independent of A, $P[B]$ is unchanged. The rest of the probabilities are not independent of A like in part (a). To calculate the probabilities of C, the same process is used as in the original probability calculation, but with new values of $P[A]$.

$P[C = c_1] = 0.620$

$P[C = c_2] = 0.380$

$P[D = d_1] = 0.636$

$P[D = d_2] = 0.364$

The probability changes are shown in Figure 2.7.

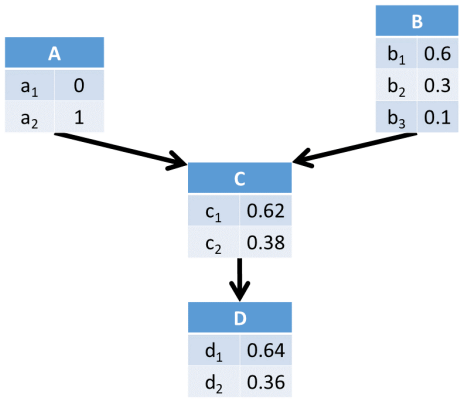


Figure 2.7: Probabilities for Example 2.3.1 with A observed to take the value $A = a_2$

c)

The process of recalculating the probabilities uses Bayes' rule when D is observed. The probabilities for A, B, and C are now all joint with D. A and B are also joint with C so the values for C must be found first.

Start by finding $P[C|D]$, $P[A|C]$, and $P[B|C]$ with the original conditions using Bayes' rule, then individual probabilities can be organized into CPDs:

$$P[C|D] = \frac{P[D|C]P[C]}{P[D]}$$

	$C = c_1$	$C = c_2$
$D = d_1$	0.6849	0.3151
$D = d_2$	0.3722	0.6278

$$P[A|C] = \sum_B \frac{P[A]P[B]P[C|A,B]}{P[C]}$$

	$A = a_1$	$A = a_2$
$C = c_1$	0.1237	0.8763
$C = c_2$	0.2995	0.7005

$$P[B|C] = \sum_A \frac{P[A]P[B]P[C|A,B]}{P[C]}$$

	$B = b_1$	$B = b_2$	$B = b_3$
$C = c_1$	0.6572	0.2968	0.0459
$C = c_2$	0.5253	0.3041	0.1705

From the problem statement for part (c), $P[D = d_1] = 0$ and $P[D = d_2] = 1$ because D is observed to be d_2. Use these new values to find the joint probabilities of C with D, then of A with C and D, and B with C and D by using the chain rule for Bayesian networks. Marginalize to get the desired probabilities.

$$P[C] = \sum_D P[D]P[C|D]$$

$C = c_1$	$C = c_2$
0.3722	0.6278

Now having found $P[C]$:

$$P[A] = \sum_C P[C]P[A|C]$$

$A = a_1$	$A = a_2$
0.2341	0.7659

$$P[B] = \sum_C P[C]P[B|C]$$

$B = b_1$	$B = b_2$	$B = b_3$
0.5744	0.3014	0.1241

Filling out the graph like in Figure 2.6 and Figure 2.7 forms Figure 2.8.

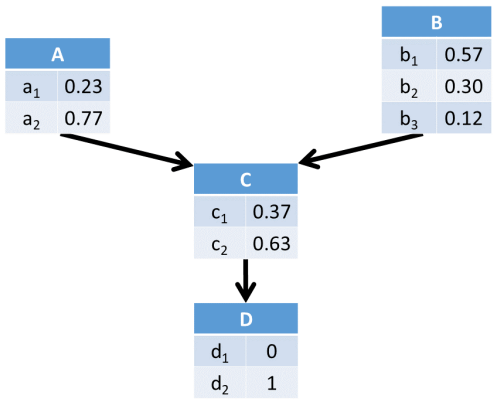


Figure 2.8: Probabilities for Example 2.3.1 with D observed to take the value $D = d_2$

A lot can be learned from Example 2.3.1 apart from understanding how probabilities are calculated. With any network, causal reasoning like in part (b) can be used to understand the sensitivities of overall likelihoods. Simply observe a variable, note how the probabilities change, and repeat the process. It is possible that the graph can be simplified by finding a nodes that have insignificant contributions and eliminating them without much consequence. A simpler graph leads to fewer calculations. As with all types of system modeling, the idea is

to model the system well enough with minimal computational cost. By performing evidential reasoning like in part (c), the user of a BN can explore the most likely causal sequence that led to a particular outcome. Intercausal reasoning was not used in Example 2.3.1, but it is a combination of causal and evidential reasoning. In Example 2.3.1, intercausal reasoning would allow the user to make inferences on B given observations on D, a descendant of B, and A, a non-descendant of B. Reasoning is why Bayesian networks are used so it is important to understand how to make conclusions from reasoning.

2.4 Dynamic Bayesian Networks

Dynamic Bayesian networks are an extension of static Bayesian networks and are useful when a system's state is changing over time. As stated in Section 1.2, static Bayesian networks can be very useful for a one-time inference such as a medical diagnosis. However, in many situations it is more useful to be able to predict outcomes in the upcoming period of time. For instance, a restaurant may want to predict the number of customers that will arrive each hour for the rest of the day so an efficient number of employees is always present. Each hour is a time step and at each time step the graph structure is the same. What makes the dynamic network different than the static network is that the graph is not only copied from time step to time step, but variables from one time step may influence variables in the next time step. The portions of the graph that are repeated, the shaded box in Figure 2.9, are called the temporal plate. The temporal plate is used to simplify the graph of a DBN. Nodes may be outside of the temporal plate, but those nodes are static and will not be repeated through the time steps.

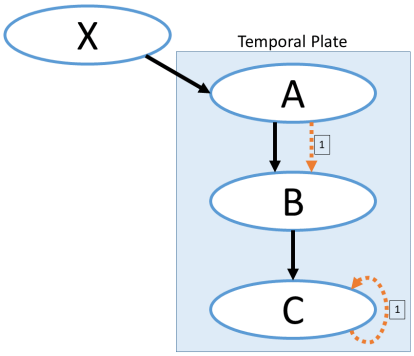


Figure 2.9: Dynamic Bayesian network with a static node and temporal plate.

There are two main types of edges in DBNs both of which can be seen in Figure 2.9. The solid edges are intra-time-slice edges that connect nodes within the time slice. The dashed edges are inter-time-slice edges that connect nodes from one time step to the next. The node at the beginning of a dashed edge is a parent while the node at the end of the edge is the corresponding child. Graphs must be acyclic at any one instant in time, but it is acceptable for a node's value in one time step to affect that same node's value in the future. The number in the box associated with a dashed edge gives the number of time steps from the time when the parent node attains a given value to the time when its child experiences the effect.

Another way to look at a DBN is by "unrolling" it. Unrolling a DBN means that the graph is expanded to a certain number of time steps, expanding the temporal plate. Figure 2.10 shows the temporal plate from Figure 2.9 unrolled over three time steps.

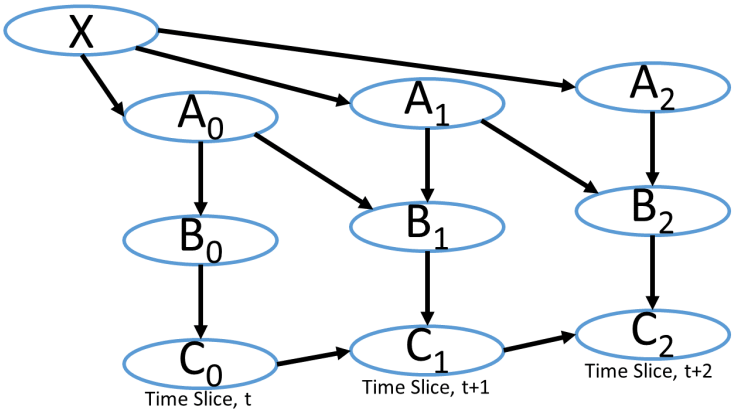


Figure 2.10: Dynamic Bayesian network unrolled to three steps.

In Figure 2.10, the edge from A_0 to B_0 is an intra-time-slice edge, the same solid edge from A to B in Figure 2.9. The edge from A_0 to B_1 is an inter-time-slice edge which is the same as the dashed edge from A to B in Figure 2.9. The edge from C_0 to C_1 is a special kind of inter-time-slice edge called a persistent edge. Its respective edge in Figure 2.9 is the curved dashed edge from C to itself. A persistent edge is formed when a variable directly influences its corresponding variable in the next time step. Unrolling a DBN is useful when forming predictions of future time steps because the user can visualize how probabilities evolve over time. Another advantage to unrolling a DBN is that observations can be made for a variable at a specific time step so the user can perform different types of reasoning.

Calculating the probabilities at each time step is exactly the same as performing calculations on a static network, by working through the chain of influencing nodes. As the DBN is unrolled, CPDs for nodes unaffected by the previous time step are equivalent to the original CPDs. In Figure 2.10, A_0, A_1, and A_2 all have the same CPD. B_0 and B_1 will have different CPDs because B_1 is conditional on both A from the current time step and A from the previous time step. The CPD will be the same for B_1 and B_2. C behaves similarly to B except after the initial time step, it becomes conditional on its own previous value along with its intra-time-slice parent. When the DBN is unrolled to reveal more time steps, the calculations get tedious and difficult to keep track of by hand. For the most part, these

calculations are done by computers with specialized software.

2.5 Learning Parameters

Usually, the CPDs are not known for a Bayesian network. If the graph structure is already created, then values for the CPDs can be estimated given a set of training data. To understand Bayesian network parameter estimation consider an unknown parameter, θ, a single value that represents the likelihood of some outcome. The goal is to estimate θ so it can then be used to predict the next event's outcome. For example out of 100 tosses of a loaded die, 16 of those trials result in a 2. An estimation of θ is 0.16 so 0.16 can be used as the probability to predict the outcome (i.e., a die roll of 2) of the next toss. Since there are not infinite tosses of the die, the estimation of θ from 100 tosses is not equal to θ. The true value of θ might be 0.5, though this means the odds of only seeing 16 outcomes of 2 is not likely. Increasing the number of tosses should bring the estimation closer to θ. Therefore when parameters are learned using a finite data set, the probabilities assigned to outcomes are estimated values of θ based on the data given.

It is possible to use the maximum likelihood estimation (MLE) to find θ, but this approach is not able to account for any prior knowledge that exists about θ. It is more common to use a joint probabilistic model for Bayesian network parameter learning, which will be explained using Koller's description [9, ch. 17]. When some information is known about θ already, a joint distribution seen in Equation (2.5) relates a vector $\boldsymbol{\theta}$ of scalar probabilities θ_i to a set of previously made observations, D.

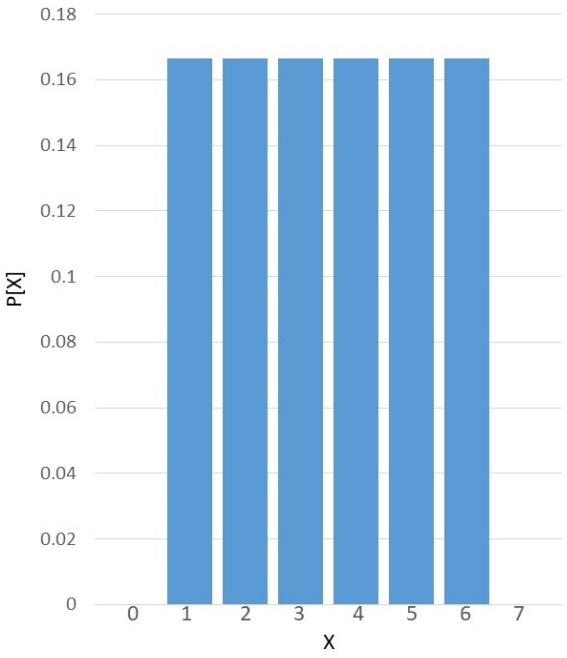
$$P[D, \boldsymbol{\theta}] = P[D|\boldsymbol{\theta}]P[\boldsymbol{\theta}] \tag{2.5}$$

$P[D|\boldsymbol{\theta}]$ is the likelihood function, the probability that the data set is D given the values of the parameters. $P[\boldsymbol{\theta}]$ is called the "prior distribution" for values possible in $\boldsymbol{\theta}$, or "prior" for short. Priors are assumed to be independent of each other. Applying Bayes' rule makes it possible to get $P[\boldsymbol{\theta}|D]$, the "posterior distribution", shown in Equation (2.6) which is the

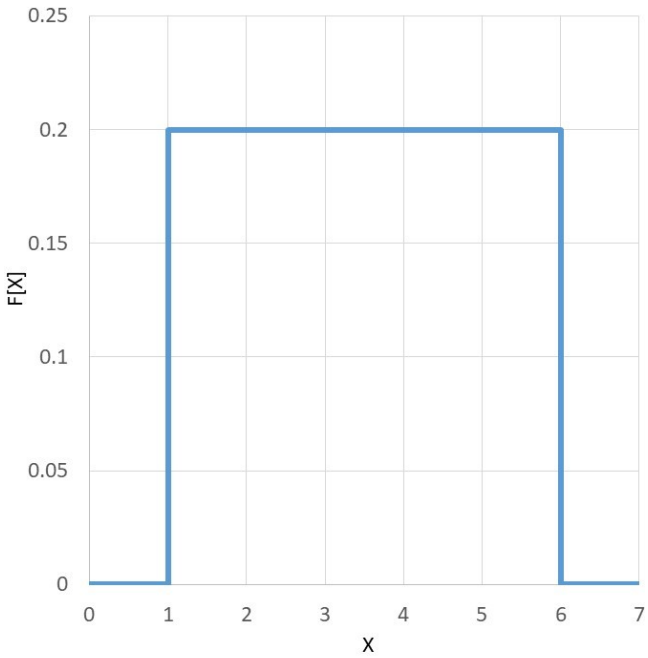
final goal for the parameter estimation.

$$P[\boldsymbol{\theta}|D] = \frac{P[D|\boldsymbol{\theta}]P[\boldsymbol{\theta}]}{P[D]} \tag{2.6}$$

Each entry in $\boldsymbol{\theta}$ is treated as a random variable. The distribution can be uniform or non-uniform. A uniform distribution is either a continuous or discrete distribution defined by a set of maximum and minimum values, a and b. If the value of the variable lies within the bounds, its probability is constant and equal to $\frac{1}{b-a}$ if the distribution is continuous. The value is equal to and $\frac{1}{n}$ if the distribution is discrete. If the value lies outside of the bounds, the probability of that value occurring is 0. Figure 2.11 shows examples of a discrete distribution and a continuous distribution.



(a) Probability mass function for a discrete uniform distribution.



(b) Probability density function for a continuous distribution function.

Figure 2.11: Uniform distributions between 1 and 6.

If the distribution is binary and non-uniform, the most common distribution to use for θ is the Beta distribution. A probability resulting from a Beta distribution is shown in Equation

(2.7).

$$P[\theta|x(1),...x(M)] = \theta^{\alpha_1+M_1-1}(1-\theta)^{\alpha_0+M_0-1} \tag{2.7}$$

M is the number of trials with outcome x, M_1 is the number of successes from M trials, and M_0 is the number of failures from M trials. Shape parameters affect the shape of the distribution. The shape parameters of the Beta distribution are α_1 and α_2. The expected value of θ_i is α_i divided by the sum of the shape parameters, α. Figure 2.12 shows what the probability density function of the Beta distribution looks like and how the shape parameters affect the shape.

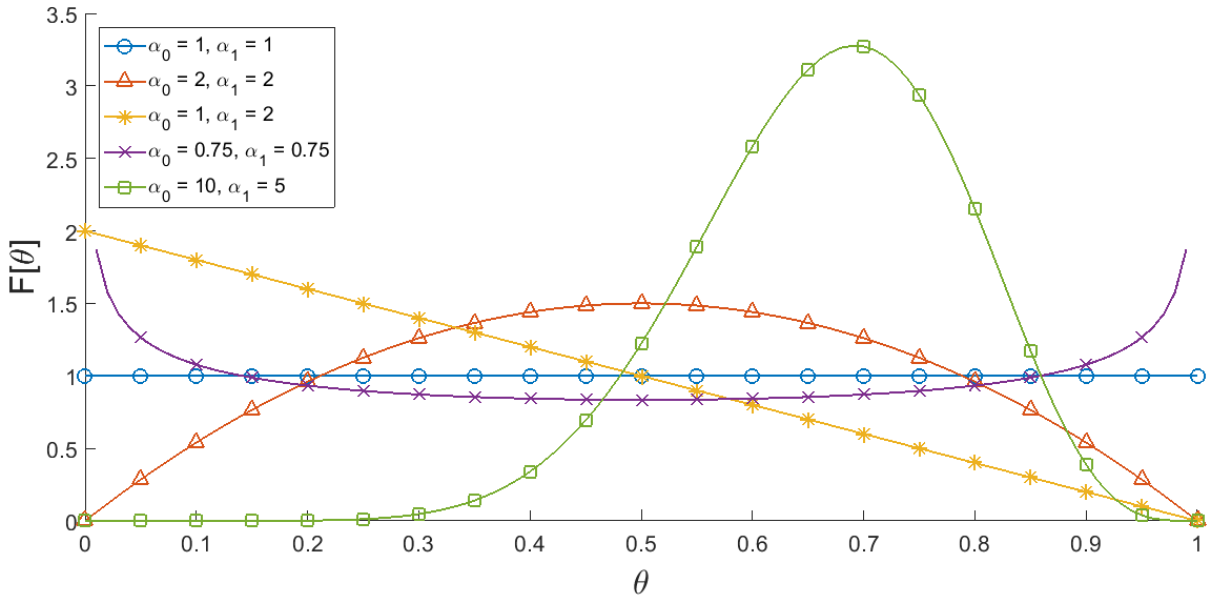


Figure 2.12: Probability density function of the Beta distribution given various shape parameters.

The Dirichlet distribution, an adaptation of the Beta distribution, replaces the Beta distribution when the outcomes are not binary. The Dirichlet distribution is given by Equation (2.8).

$$P[\theta] \propto \prod_k \theta_k^{\alpha_k-1} \tag{2.8}$$

The above distributions are examples of the most common types of distributions for priors, which are used to solve for the CPDs of each variable. The CPDs are solved for by calculating the joint probability of the conditional probability of X_i given the parents Pa_{X_i} and associated parameters $\boldsymbol{\theta}_{X_i|Pa_{X_i}}$ with the results from the posterior. In a graph with set X of n random variables, Equation 2.9 predicts the CPDs for the $M+1^{th}$ trial.

$$P[X_1(M+1),...,X_n(M+1)|D] =$$

$$\prod_i \int P[X_i(M+1)|Pa_{X_i}(M+1),\boldsymbol{\theta}_{X_i|Pa_{X_i}}]P[\boldsymbol{\theta}_{X_i|Pa_{X_i}}|D]d\boldsymbol{\theta}_{X_i|Pa_{X_i}} \tag{2.9}$$

Again, the procedure represented by Equation (2.9) is typically done using is done by specialized software that will accept a training data set, predictions for $\boldsymbol{\theta}$, and a graph.

One of the convenient aspects of parameter learning is that a complete data set for all variables is not necessary, meaning there can be instances where values are left blank. To arrive at CPDs when there is data missing, the parameters need to calculated in an iterative process called "expectation maximization" (EM). This thesis uses software that implements EM to learn parameters because the training data set is incomplete. EM optimizes likelihood functions of the known data set given the missing data values and the parameters in an iterative process which will again be explained using Koller's methods [9, Ch. 19]. The process begins with the "expectation step" (E-step) where the current estimation, $\boldsymbol{\theta_t}$, is used to calculate the "expected sufficient statistics" (ESS) in Equation (2.10).

$$\bar{M}_{\boldsymbol{\theta_t}}[x,\boldsymbol{u}] = \sum_m P[x,\boldsymbol{u}|o(m),\boldsymbol{\theta_t}] \tag{2.10}$$

In the notation used by Equation (2.10), x is a variable and $\boldsymbol{u}$ is the set of parent variables of x. A data case is a single point in the data set containing information on all of the variables. A partially observed data case is called $o(m)$ where m indicates the specific data case out of the M cases. In this thesis, the values in $\boldsymbol{\theta_0}$ are selected according either to hand crafted distributions or uniform distributions.

The next step is called the "maximization step" (M-step) where the likelihood is maximized relative to the ESS. The M-step is shown in Equation (2.11).

$$\boldsymbol{\theta_{t+1}}(x|\boldsymbol{u}) = \frac{\bar{M}_{\boldsymbol{\theta_t}}[x, \boldsymbol{u}]}{\bar{M}_{\boldsymbol{\theta_t}}[\boldsymbol{u}]} \tag{2.11}$$

The M-step is what provides the parameters for the CPDs. The E-step and M-step are repeated until both $\bar{M}_{\boldsymbol{\theta_t}}$ and $\boldsymbol{\theta_{t+1}}$ converge to fixed values. It is not practical to calculate the parameters by hand, and with a large data set may not even be possible. In some networks, the graph structure may be unknown. There is a separate learning process given a data set that allows for network structure learning, where the data is used to not only estimate parameters but also to infer how the variables are causally related.

Chapter 3

Building a Dynamic Bayesian Network for the Safe-to-Proceed Example

3.1 Review of Problem and Introduction to GeNIe Software

As mentioned in the introduction, the objective of this thesis is to present a method is to aide pilots who are deciding whether or not to fly an sUAS on a particular mission, given conditions at a particular time. Flight safety is affected by many factors and a Bayesian network can make a pilot's decision more informed by providing a data-driven decision aide. Bayesian networks combine past pilot experience with measured or estimated data to reduce subjectivity and to aid the continuation of operations after pilots move on from their role and are replaced.

This analysis explores the application of a simple network in three forms: a static Bayesian network without learning, a static Bayesian network with learning, and a dynamic Bayesian

network with learning. Pilot expertise is introduced through the selection of important variables and how they interact with each other along with the initial conditional probability distributions. The base graph structure is the same for all three cases. Like with all applications of BNs and system modeling, the proposed network is a model for reality and all models are inherently incomplete. The graph structure for the safe-to-proceed example being explored is not realistic and is only intended to demonstrate the use of Bayesian networks in a notional application.

The software selected for this analysis is called GeNIe Modeler with the SMILE Engine [3] and will be referred to as GeNIe and SMILE, respectively. GeNIe and SMILE were created by the Decision Systems Laboratory at the University of Pittsburgh and currently licensed by BayesFusion, LLC. GeNIe is free to use for academic purposes. GeNIE is the user interface for SMILE, the reasoning engine, but SMILE can also be used independently through a Java wrapper for users who create their own interface tools. GeNIe is designed for teaching and research so the tool works well for exploring the safe-to-proceed problem. Within GeNIe, all of the processes described in Chapter 2 are performed so no manual calculations are necessary. Users have the ability to build a graph from scratch with nodes and probability tables or to create a network learned from data that can be input. Once a graph is built, users can see how observing nodes affects probabilities instantaneously as in Example 2.3.1. This tool makes investigating the application of different types of BNs to the safe-to-proceed problem very efficient. GeNIe is very intuitive to use, but it is crucial to have a thorough understanding of Bayesian networks and probabilistic graphical models to ensure that the results being produced are interpreted correctly.

3.2 Relevant Variables

To begin the process of building a model that can be used for reasoning, variables were decided that would be important in determining the safety of flight for a small unmanned

vehicle at Kentland Farm. To have the example be easy to follow and understand, the model was kept very simple with only eight variables. The variable representing the output of the Bayesian network, and used in predicting the safety of flight is *Flight Safety*. This variable is a leaf node because it has no children. All other variables affect *Flight Safety* either directly or indirectly. An important factor in safety of flight for any vehicle is weather which is highly dependent on the time of year, *Day*, and time of day, *Time*. Incorporating the day and time as variables allows safety of flight to be studied as a function of these variables. For example, having *Day* and *Time* variables may be helpful to a pilot who has experience flying in daylight hours in summer but now has a need or interest to fly at night in the winter. The three variables that were chosen to represent weather at Kentland Farm are *Precipitation*, *Temperature*, and *Wind Speed*. The data for training the weather variables comes from the Virginia Agricultural Experiment Station (VAES) at Kentland Farm [11] and covers most months between 1999 and 2017. It is intuitive that *Precipitation* and *Wind Speed* may affect Flight Safety, but the effect of *Temperature* may not be as obvious. Lithium batteries serve as the vehicle's power supply. The operating temperature changes how well lithium batteries function [10], thus another variable is introduced, *Battery Safety*. There is another parent to *Battery Safety*, *Battery Charged* which is a binary representation of whether or not the battery was fully charged before use. Table 3.1 contains the description of variable outcomes and variable relationships for the eight variables in the model.

Table 3.1: Description of variables in the safe-to-proceed network

Variable	Discrete Outcomes	Parents	Children
Day	$[1, \ldots, 365]$	None	*Precipitation* *Temperature* *Wind Speed*
Time	$[0, \ldots, 23]$	None	*Precipitation* *Temperature* *Wind Speed*
Precipitation	True False	*Day* *Time*	*Flight Safety*
Temperature	Below 0^0C 0^0C to 20^0C Above 20^0C	*Day* *Time*	*Batt. Safety*
Wind Speed	0 to 5 kph 5 to 10 kph 10 to 15 kph 15 to 20 kph Above 20 kph	*Day* *Time*	*Flight Safety*
Batt. Charged	True False	None	*Batt. Safety*
Batt. Safety	Low Risk Medium Risk High Risk	*Temperature* *Batt. Charged*	*Flight Safety*
Flight Safety	Low Risk Medium Risk High Risk	*Precipitation* *Batt. Safety* *Wind Speed*	None

All of the relationships that are listed in Table 3.1 are shown in graph form in Figure 3.1.

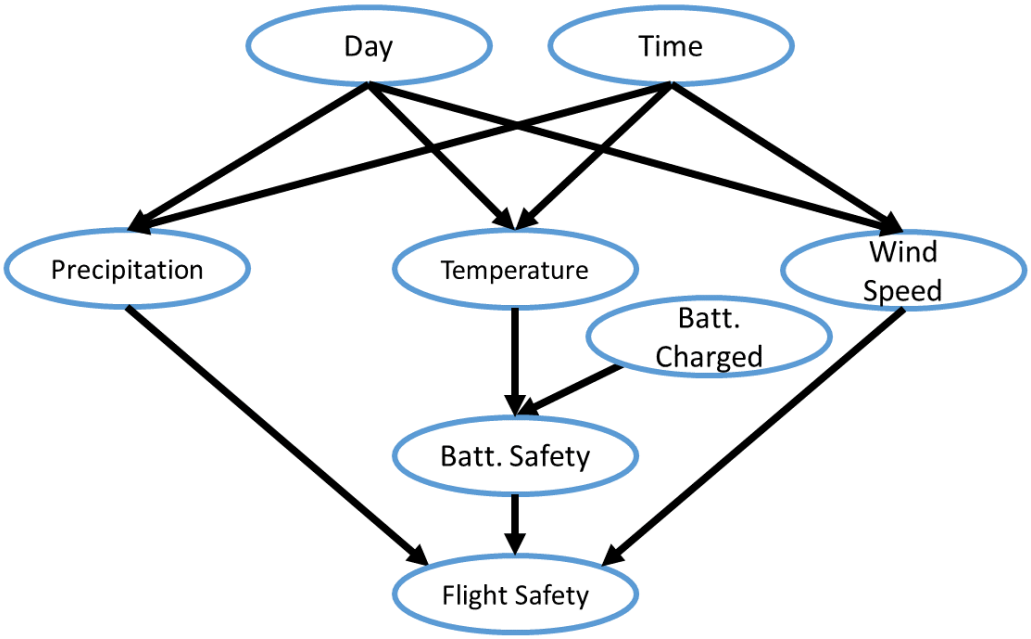


Figure 3.1: The base graph structure of the Bayesian network

In the graph, it is easy to see which nodes are directly connected to other nodes. *Day* and *Time* are independent of each other unless any other variable is observed except *Battery Charged* because v-structures will be formed allowing the flow of influence. If the weather variables are all observed, then *Flight Safety* is independent from both *Day* and *Time* meaning that it no longer matters what day it is or what time it is. More active and inactive flows of influence can be explored following the basic rules from Figure 2.3 and Figure 2.4.

Each node has an initial CPD with values based on educated guesses derived from experience. Later in this thesis, learning the parameters will be applied to try and get more accurate probabilities. When the the terms "learning network" or "network with learning" are used in this thesis, they mean learning the parameters only. (While methods exist to learn the network structure from data, these methods are not applied here.) The initial CPDs can be used to boost the learning process. In later sections, the effectiveness of defining an initial

CPD before learning the CPD is explored.

Day and *Time* have a uniform distribution. A number of assumptions are made to simplify the example. It is assumed that it is equally likely for each day of the year and each hour of the day to occur. Leap days do not exist, sunrise and sunset occur at the same time every day, and there is no climate change as years pass. Because there are so many combinations of day and time possible ($365 \times 24 = 8760$), the CPDs for *Precipitation*, *Temperature*, and *Wind Speed* are quite long. The initial weather CPDs match what is common throughout the year.

Table 3.2: The CPD for *Precipitation*

Day	*Time*	*Precipitation*	
		True	False
[1,90]	[0,23]	0.30	0.70
[91,212]	[0,23]	0.50	0.50
[213,365]	[0,23]	0.30	0.70

Table 3.2 shows a compressed CPD for the probabilities of precipitation. *Precipitation* is assumed to have no correlation to time of day so the probabilities are constant relative to time. *Precipitation* is assumed to be more likely in late spring and early summer months than any other time of the year.

Table 3.3: The CPD for *Temperature*

Day	*Time*	*Temperature*		
		Below 0^0C	0^0C to 20^0C	Above 20^0C
[1,151]	[0,7]	0.40	0.50	0.10
	[8,16]	0.30	0.60	0.10
	[17,23]	0.40	0.50	0.10
[152,273]	[0,7]	0.20	0.50	0.30
	[8,16]	0.10	0.60	0.30
	[17,23]	0.20	0.50	0.30
[274,365]	[0,7]	0.40	0.50	0.10
	[8,16]	0.30	0.60	0.10
	[17,23]	0.40	0.50	0.10

Table 3.3 shows a compressed CPD for the probabilities of temperature ranges. *Temperature* will have higher values in hours of the day with sunlight and months in late spring, summer, and early fall.

Table 3.4: The CPD for *Wind Speed*

Day	*Time*	*Wind Speed*				
		0-5 kph	5-10 kph	10-15 kph	15-20 kph	Above 20 kph
[1,264]	[0,11]	0.50	0.20	0.15	0.10	0.05
	[12,15]	0.05	0.40	0.35	0.10	0.10
	[16,23]	0.50	0.20	0.15	0.10	0.05
[265,355]	[0,11]	0.20	0.30	0.30	0.10	0.10
	[12,15]	0.05	0.40	0.35	0.10	0.10
	[16,23]	0.20	0.30	0.30	0.10	0.10
[356,365]	[0,11]	0.50	0.20	0.15	0.10	0.05
	[12,15]	0.20	0.30	0.30	0.15	0.05
	[16,23]	0.50	0.20	0.15	0.10	0.05

Table 3.4 shows a compressed CPD for the probabilities of each range of wind speeds.

It is assumed *Wind Speed* increases in both fall months and afternoon hours. *Battery Charged* is another variable with a uniform distribution that will not be shown in tabular form. It is assumed that the pilot has been trained to follow a pre-flight checklist so the probability of the battery having a full charge is 0.9 and the probability that the battery is not fully charged is 0.1. The CPD for *Battery Safety* is shown in Table 3.5.

Table 3.5: The CPD for *Battery Safety*

Batt. Charged	*Temperature*	*Batt. Safety*		
		Low Risk	Medium Risk	High Risk
True	Below 0^0C	0.20	0.50	0.30
	0^0C to 20^0C	0.80	0.15	0.05
	Above 20^0C	0.50	0.30	0.20
False	Below 0^0C	0	0	1
	0^0C to 20^0C	0	0	1
	Above 20^0C	0	0	1

The *Battery Safety* CPD shows that whenever the battery is known to be not fully charged, the outcome is always high risk. When the battery is fully charged, the battery functions optimally in the middle temperature range. Battery capacity decreases slightly at high temperatures and significantly at below freezing temperatures so this knowledge affects the expected risk. These values would depend on the actual battery specifications in a realistic scenario.

Table 3.6: The CPD for *Flight Safety* when *Precipitation* is true

Precipitation	*Battery Safety*	*Wind Speed*	*Flight Safety*		
			Low Risk	Medium Risk	High Risk
True	Low Risk	0 to 5 kph	0.30	0.40	0.30
		5 to 10 kph	0.25	0.35	0.40
		10 to 15 kph	0.20	0.30	0.50
		15 to 20 kph	0.15	0.25	0.60
		Above 20 kph	0.10	0.20	0.70
	Medium Risk	0 to 5 kph	0.30	0.30	0.40
		5 to 10 kph	0.20	0.30	0.50
		10 to 15 kph	0.10	0.30	0.60
		15 to 20 kph	0.10	0.20	0.70
		Above 20 kph	0.10	0.10	0.80
	High Risk	0 to 5 kph	0.20	0.30	0.50
		5 to 10 kph	0.10	0.30	0.60
		10 to 15 kph	0.10	0.20	0.70
		15 to 20 kph	0.10	0.10	0.80
		Above 20 kph	0.00	0.10	0.90

Table 3.7: The CPD for *Flight Safety* when *Precipitation* is false

Precipitation	*Battery Safety*	*Wind Speed*	*Flight Safety*		
			Low Risk	Medium Risk	High Risk
False	Low Risk	0 to 5 kph	0.60	0.30	0.10
		5 to 10 kph	0.45	0.35	0.20
		10 to 15 kph	0.30	0.40	0.30
		15 to 20 kph	0.25	0.30	0.45
		Above 20 kph	0.20	0.20	0.60
	Medium Risk	0 to 5 kph	0.30	0.30	0.40
		5 to 10 kph	0.25	0.35	0.40
		10 to 15 kph	0.20	0.40	0.40
		15 to 20 kph	0.175	0.30	0.525
		Above 20 kph	0.15	0.20	0.65
	High Risk	0 to 5 kph	0.20	0.30	0.50
		5 to 10 kph	0.15	0.30	0.55
		10 to 15 kph	0.10	0.30	0.60
		15 to 20 kph	0.10	0.20	0.70
		Above 20 kph	0.10	0.10	0.80

Table 3.6 and Table 3.7 together show the CPD for *Flight Safety* which has more parents than *Battery Safety* so the table becomes more complicated. As the values of *Precipitation*, risk of *Battery Safety*, and *wind speed* increase, the risk of *Flight Safety* tends to increase. Of course for a real system, the values in the *Flight Safety* CPD would depend on a specific vehicle's sensitivity to the parent variables. All of the CPDs above are used in the chain rule discussed in Example 2.3.1 but the computations are performed using GeNIe.

3.3 Safe-to-Proceed Analysis Using Bayesian Networks

3.3.1 Static Bayesian Network without Learning

The predetermined CPDs from the previous section are applied to the nodes in Figure 3.1 in GeNIe. Only the weather variables will have CPDs that change when the learning of parameters is introduced to the model because the learning data set only has weather information. With no variables observed, the probabilities of each variable are shown in Figure 3.2.

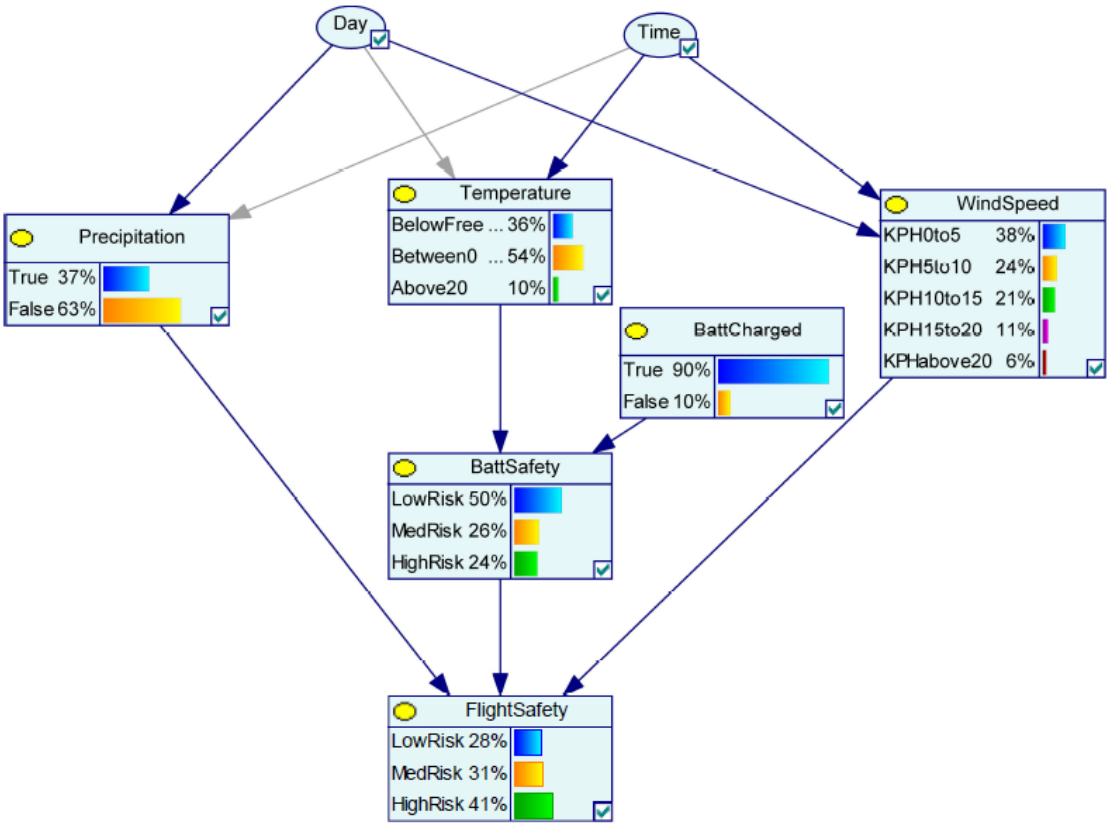


Figure 3.2: The GeNIe results from a static network without learning parameters

The probabilities in Figure 3.2 are not for any specific day or time. What the distributions are showing is that if a day and time are selected randomly, the probabilities of the optimal conditions are 0.63 for no precipitation, 0.54 for moderate temperatures, and 0.38 for low

winds. These probabilities produce a distribution for *Flight Safety* with a 0.28 probability of a low risk flight, a 0.31 probability of a medium risk flight, and a 0.41 probability of a high risk flight. The distribution is skewed, meaning the probabilities increase as risk increases. The values decided for the CPD may be incorrect, of course. To determine how good the predictions are, the weather of the first day of fall (day 265) at 2:00PM (hour 14) in 2017 is compared to actual data from the Montgomery WeatherSTEM [12]. This date is outside of the range of data that will be used for learning parameters so it can be compared through all of the networks being analyzed to determine how well the networks predict the actual weather. More days and times will be explored in detail following the discussion of the dynamic Bayesian network with learning parameters. When the day and time are set to observed at day 265 and hour 14, the model shows the probability distributions in Table 3.8.

	Precipitation
True	0.30
False	0.70

	Wind Speed
0-5 kph	0.20
5-10 kph	0.30
10-15 kph	0.30
15-20 kph	0.15
Above 20 kph	0.05

	Batt. Charged
True	0.90
False	0.10

	Temperature
Below 0^0C	0.30
0^0C to 20^0C	0.60
Above 20^0C	0.10

	Batt. Safety
Low Risk	0.53
Med. Risk	0.24
High Risk	0.23

	Flight Safety
Low Risk	0.26
Med. Risk	0.31
High Risk	0.42

Table 3.8: The series of probability distributions for the 265th day at 2:00PM without learning parameters

According to WeatherSTEM there was no precipitation, the temperature was above 20^0C, and the wind speed was between 5 and 10 kph. The only weather variable that is not accu-

rately predicted by the model is *Temperature*. When the weather variables are set observed to the actual values, the probability distribution for *Battery Safety* becomes 0.45, 0.27, and 0.28 for low risk, medium risk, and high risk, respectively. The probability distribution for *Flight Safety* becomes 0.31, 0.34, and 0.35 for low risk, medium risk, and high risk, respectively. Neither of these variables have significant probability changes. At this point, it is unclear how good the initial CPDs for weather are so the next step is to use parameter learning and compare. It is assumed that parameter learning will only improve the model.

3.3.2 Static Bayesian Network with Learning

The weather data collected from Kentland Farm amounts to over 160,000 data points, all of which are used for training. VAES provides data for each month from each year in separate Excel sheets. All 211 months need to be in the same data sheet for GeNIE to be able to import the training data, which is a time consuming task. While this may seem like a lot of data, it actually amounts to only 19 points for each day and time. GeNIe estimates the parameters for *Precipitation*, *Temperature*, and *Wind Speed* either using random seeds, the original CPDs, or normal distributions for all variables. The latter two options are what will be compared. To begin, the nodes for *Day*, *Time*, *Battery Charged*, *Battery Safety*, and *Flight Safety* are fixed because there is no data for these variables. GeNIe runs first with the original CPDs and then again with uniform CPDs. Both runs come to identical conclusions about the probability distributions. However, when using the original CPDs the learning process takes approximately 30 minutes while using uniform CPDs consistently takes under 10 seconds to complete. Given this major computational difference for the same results, it is clear that a comprehensive training data set can replace the need for determining preliminary CPDs for the parameters that will be learned. The results produced from learning the parameter probability distributions for *Precipitation*, *Temperature*, and *Wind Speed* and the children of these variables are shown in Figure 3.3.

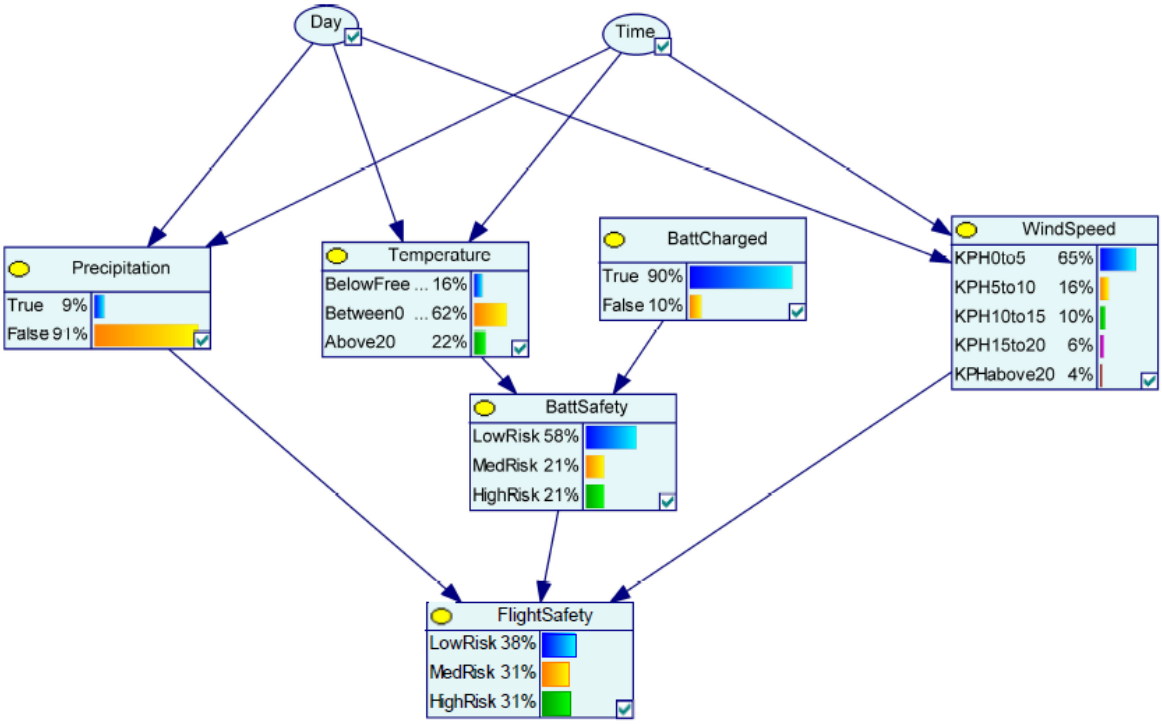


Figure 3.3: The GeNIe results from a static network with learning weather parameters

A major difference between the distributions with learning and without is that when no variables are observed, the probabilities for all of the desirable conditions increase when learning is incorporated. This indicates that the values selected for the earlier network were overly conservative. Now, the outcome with the highest probability for *Flight Safety* is "Low Risk". This is because the safest weather conditions all have the highest probability. There is no day or time selected so the probabilities in Figure 3.3 are general for any day and time; the figure shows the likely conditions for the average hour out of all 8760 hours in a year. Using the static BN after learning the parameters, a pilot can conclude that more often than not, it is safe to fly a mission. To pull out a specific hour, a day and time must be assigned as observations. Then, the network will output probabilities pertaining only to that specific hour. As an example of a specific hour, the last day of fall is input in the model like in Section 3.3.1. The new probabilities are given in Table 3.9.

	Precipitation
True	0.03
False	0.97

	Wind Speed
0-5 kph	0.48
5-10 kph	0.27
10-15 kph	0.12
15-20 kph	0.06
Above 20 kph	0.06

	Batt. Charged
True	0.90
False	0.10

	Temperature
Below 0^0C	0.02
0^0C to 20^0C	0.28
Above 20^0C	0.70

	Batt. Safety
Low Risk	0.52
Med. Risk	0.24
High Risk	0.24

	Flight Safety
Low Risk	0.35
Med. Risk	0.31
High Risk	0.35

Table 3.9: The series of probability distributions for the 265th day at 2:00PM for the model with learned weather variables

After learning the parameters, the model now fails to accurately pick the wind speed range but does predict the temperature range correctly. However, the *Flight Safety* probabilities for this imaginary aircraft are not exceptionally sensitive to the changes. Even though the weather was partially predicted incorrectly, the probability of a high risk flight does not drop lower than the probability of a low risk flight. Although one time of one day is not enough to prove that this model is not a good fit, the results from the static learning network are suspicious enough to try and apply a dynamic network.

3.3.3 Dynamic Bayesian Network with Learning

Recall that an important characteristic of DBNs is that the introduction of time slices and the temporal plate imply that outcomes from hour to hour are no longer independent. This is beneficial because now observations of the recent past are able to help predict the weather for the rest of the day, although observations are not necessary. The same training data from

Section 3.3.2 is used to train the DBN. However, the data must be adapted to a form that suits the temporal plate. All data for each individual day is transposed and combined into one row. The way training data was arranged for this example produces important rules for how a user needs to organize observations. Each row of data contains either incomplete or complete data for the entirety of a single day. Because both *Day* and *Time* are on the temporal plate, *Day* must be observed for the same value on all 24 time slices or not observed at all. *Time* must be observed chronologically from hour 0 to hour 23 for the ascending time steps. A 24 hour time period cannot be explored outside of these bounds only because of the training data. If the data is organized to be formed of randomly selected, sequential 24 hour hour periods, the single day constraint would disappear at the cost of possibly repeated data or loss of data. As the network currently exists, it is not possible to use the previous day to influence the day being analyzed. This limitation could be eliminated, but at the cost of increasing calculations necessary by adding time steps to the temporal plate or significantly increasing the amount of training data needed. A student or pilot training his own network could explore the reorganization of the training data and even changing the number of time slices that the temporal plate expands to. It is important for a user to know what data is used for training and how it is formed to properly interpret the results.

Once the user understands the constraints, the parameters may undergo the learning process. Again, the CPDs are set to uniform values. *Day*, *Time*, *Battery Charged*, *Battery Safety*, and *Flight Safety* are held fixed because these parameters do not have training data. The results from training *Precipitation*, *Temperature*, and *Wind Speed* with no variables observed are shown in Figure 3.4

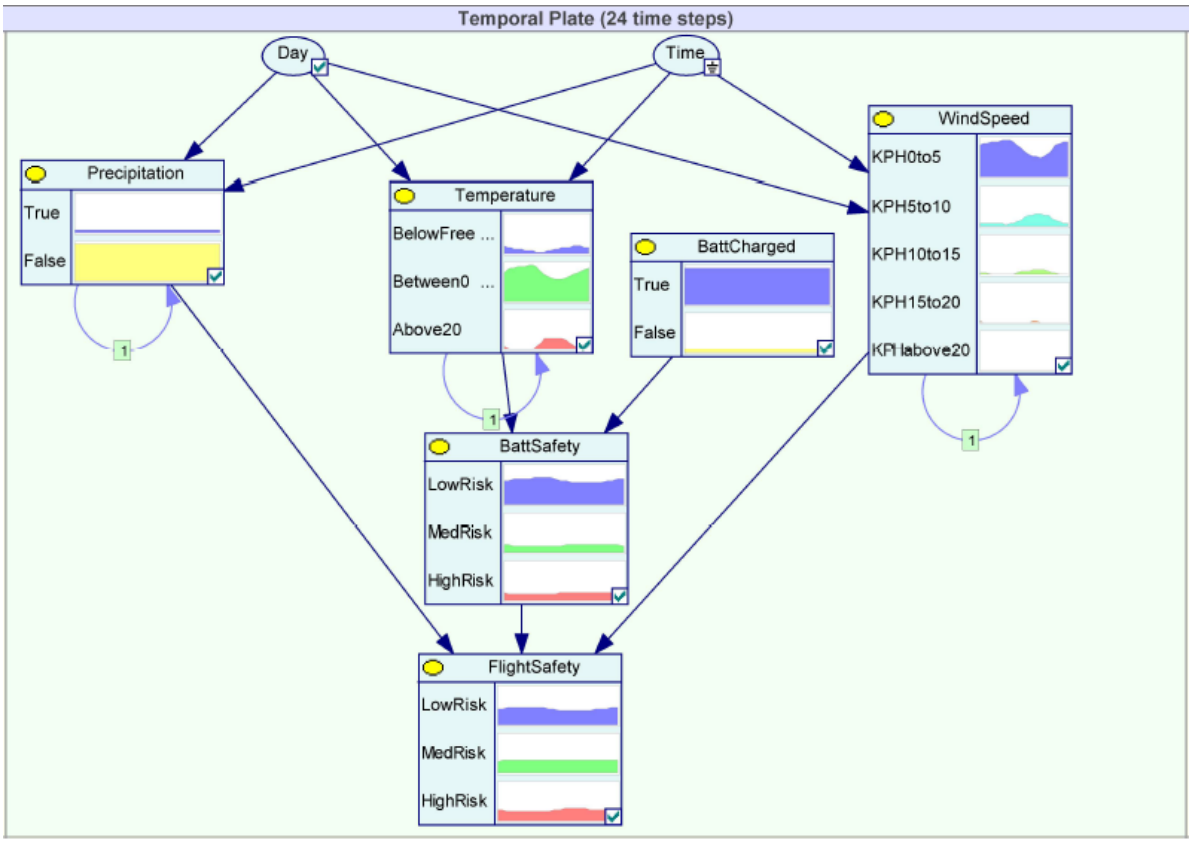


Figure 3.4: The GeNIe results from a dynamic network with learning weather parameters

Notice that now probabilities are shown as an area chart rather than a single value. The x-axis of the charts goes from the first time step to the last time step corresponding to hour 0 and hour 23. The y-axis is the probability. Like all probabilities, at each time step for each variable the sum of probabilities of outcomes will equal one, the equivalent of shading the entire area in the area chart.

Day 265 is revisited but instead of only getting information on 2:00PM, the whole day is predicted. To get the probabilities in the same tabular format as the other two methods, each node needs to be opened so the probabilities for 2:00PM can be extracted which is done for Table 3.10.

	Precipitation
True	0.01
False	0.99

	Wind Speed
0-5 kph	0.51
5-10 kph	0.28
10-15 kph	0.11
15-20 kph	0.05
Above 20 kph	0.05

	Batt. Charged
True	0.90
False	0.10

	Temperature
Below 0^0C	0.00
0^0C to 20^0C	0.33
Above 20^0C	0.67

	Batt. Safety
Low Risk	0.53
Med. Risk	0.23
High Risk	0.24

	Flight Safety
Low Risk	0.36
Med. Risk	0.31
High Risk	0.33

Table 3.10: The series of probability distributions for the 265th day at 2:00PM for the model with learned weather variables for a dynamic model

All three weather variables are correctly predicted for 2:00PM, hour 14. Even though the temperature is higher than 20^0C, *Battery Safety* has its highest probability associated with "Low Risk". Looking at the day as a whole, the probability for "High Risk" increases toward the afternoon when the temperature and wind speed are expected to increase for *Battery Safety* and subsequently *Flight Safety*. Setting observations for weather is more effective when predicting future time steps in the dynamic Bayesian network because there is flow of influence between time slices. The benefits of observing variables is discussed in the next section where a more in depth analysis of predictions is explained.

3.4 Case Study

In addition to day 265, ten random days are sampled between 1 August 2017 and 31 March 2018 because these dates lie outside of the VAES available data set that was used for training.

The complete list of Julian days explored are 223, 249, 260, 265, 301, 332, and 356 from 2017 and 6, 49, 77, and 86 from 2018 totaling 164 hours of data points collected from WeatherSTEM. The three methods for comparison to actual data are the methods discussed in Section 3.3 with no observations made other than *Day* and *Time*. This information is then compared to the dynamic Bayesian network when there is a variable observed. For brevity, the three methods will be referred to as G_1, G_2, and G_3 respectively.

- G_1 - static BN with no learning
- G_2 - static BN with learning
- G_3 - DBN with learning

The prediction from the three models is chosen as the outcome with the highest probability for each variable. If the prediction matches the actual weather recorded from that day and time, the prediction is considered correct. Some time steps have equal probabilities for multiple outcomes. If one of the outcomes with equal probabilities matches the recorded data, the prediction is still considered correct.

The discussion in this section focuses on correct versus incorrect predictions, but one of the major attributes of Bayesian networks is the use of probabilities of each event to predict subsequent probabilities. This is useful for the safe-to-proceed example because in general, a pilot does not care about the weather specifically, only how it affects the safety of flight. This analysis takes a conservative stance on flight safety where of course, accurate predictions for weather are preferred, but it is seen as better to over-conservatively predict worse than actual conditions than to incorrectly predict better than actual conditions. When different outcomes have similar probabilities each has a similar impact rather than the outcome with a slightly higher probability being completely dominant.

3.4.1 Precipitation Prediction

Regarding *Precipitation*, all three models chose "False" 100% of the time. The outcome of "False" is correct for 95% of the actual precipitation from those times. Table 3.11 holds the summary of all predictions made for the sample days.

Table 3.11: The predictions for *Precipitation* for the 11 days

WeatherSTEM		G_1		G_2		G_3	
		Correct	Incorrect	Correct	Incorrect	Correct	Incorrect
True	13	0	0	0	0	0	0
False	251	251	13	251	13	251	13

As Table 3.11 shows, neither G_1, G_2, nor G_3 ever predict "True" either correctly or incorrectly. All three predicted "False" 251 times correctly and 13 times incorrectly. It is clear that the models will never correctly predict a value of "True" because in the defined CPD for precipitation, the probability for "False" is always higher than the probability for "True". This can negatively impact the *Flight Safety* outcome by falsely increasing the probability for low risk. Many multirotors have exposed electronics and unexpected rain can fatally damage the vehicle in flight. With the two static models, more human intuition is needed to make an informed decision.

A way that G_3 has an advantage over the static models is the ability to take an observation and make better predictions for future time steps. Say a large-scale experiment is being planned for a week in September, three months from now, and the experiment requires 6 hours of uninterrupted flight. The group planning the experiment would like to know the likelihood of flying on any given day during that week supposing that it is raining at 8AM. On day 249, September 6th, the collected data indicates that there was precipitation from 8:00AM through 11:00AM. Testing the observation of rain at 8:00AM in G_3, *Precipitation* is

now predicted to be "True" until 2:00PM on that day. Although this overshoots the actual time period of precipitation seen in the 2017 data, the probabilities indicate that if the day starts with rain then the mission will not be able to be completed because "High Risk" is predicted for *Flight Safety*. Of course, the final decision on the day of flight is made by the pilot who can be influenced by the *Flight Safety* prediction, but the group may decide to pick a different day in that week that should have dryer weather. When the probabilities for outcomes are close, the pilot may choose to accept the risks even if "Medium Risk" or "High Risk" have higher probabilities.

3.4.2 Temperature Prediction

The predictions for *Temperature* are more diverse than those for *Precipitation* between the models. The general results are in Table 3.12.

Table 3.12: The predictions for *Temperature* for the 11 days

WeatherSTEM		G_1		G_2		G_3	
		Correct	Incorrect	Correct	Incorrect	Correct	Incorrect
Below 0^0C	41	0	0	15	17	15	17
0^0C to 20^0C	187	187	77	160	34	160	36
Above 20^0C	36	0	0	28	10	26	10

G_1 never predicts any outcome but "0^0C to 20^0C". The temperature predictions for G_1 are completely expected given Table 3.3 of the CPD. When there are no variables observed, G_2 and G_3 perform similarly with around 76% correct predictions, an improvement over the 71% from G_1. For the two learning models when an incorrect prediction is made, the prediction is most likely "0^0C to 20^0C". These models have a very difficult time predicting when the temperature will be below 0^0C getting only $\frac{15}{41}$ predictions correct when the temperature is

known to be below 0^0C. When the temperature is above 0^0C the models perform much better with correct predictions 85% of the time for "0^0C to 20^0C" and 72% of the time for "Above 20^0C".

Again, when an observation is made, the G_3 results significantly improve. Taking a closer look at day 6, the entire 24 hour period was below freezing according to WeatherSTEM. Without any observations, the only time G_3 predicted the temperature correctly was from 5:00AM through 9:00AM then again from 7:00PM through 11:00PM. With an observation of below freezing added at midnight, the only incorrect predictions are now from 1:00PM and 6:00PM, drastically improving the total time predicted incorrectly. From 1:00PM and 6:00PM the probabilities of "Below 0^0C" and "0^0C to 20^0C" are almost even, with "0^0C to 20^0C" just barely beating "Below 0^0C". Adding another observation further along in the day will yet again improve the results as the observation gets closer in time to the previously incorrect period of predictions. For this particular day, the *Flight Safety* outcomes are about 30% for all three possibilities which will put a lot of weight on the pilot's intuition for a flight decision.

3.4.3 Wind Speed Prediction

Wind Speed was the most difficult variable for the models to predict. Below, in Table 3.13, are the results from G_1, G_2, and G_3.

Table 3.13: The predictions for *Wind Speed* for the 11 days

WeatherSTEM		G_1		G_2		G_3	
		Correct	Incorrect	Correct	Incorrect	Correct	Incorrect
0 to 5 kph	176	99	61	172	73	172	80
5 to 10 kph	50	26	74	11	5	5	5
10 to 15 kph	26	0	4	2	1	1	1
15 to 20 kph	12	0	0	0	0	0	0
Above 20 kph	0	0	0	0	0	0	0

G_1 only predicted the correct wind speed 47% of the time. While G_2 and G_3 have slightly more impressive success rates, at 70% and 67% respectively, these models still have a poor prediction capability for any category other than "0 to 5 kph". As the wind speed increases, the probability of making a correct prediction decreases. Surprisingly, G_1 is able to predict "5 to 10 kph" the best of the three models, but at the cost of incorrectly predicting "5 to 10 kph" 74 times. Because high winds are so uncommon in the training data compared to "5 to 10 kph", graphs with unobserved variables almost always predicted "5 to 10 kph". Although the models predict "5 to 10 kph" well when that is actually the wind speed, about 30% of all predictions for "5 to 10 kph" are incorrect for G_1 and almost 40% incorrect for G_2 and G_3.

Exploring day 77, a quite windy day, gives significant insight into the improvements of predictions for G_3 when there is an observation. With "5 to 10 kph" observed at 8:00AM, the remainder of the day during daylight hours has predicted speeds above 5 kph. The true wind speed is "5 to 10 kph" for most of the day until evening when it dies down. Predictions are now for wind speeds higher than the actual value, unlike the unobserved case where winds were predicted to be lower. Although still incorrect, the predictions push the *Flight Safety* probabilities to more conservative values, meaning a pilot is less likely to accept a

risk that is lower than it should be. Adding wind evidence is unlike adding observations to the other weather variables because it does not seem to have much effect on the time steps prior to the observation, but observations in *Precipitation* and *Temperature* do have a large effect on prior time steps.

3.5 Computational Costs

The computer performing the computations for this analysis is a Dell Latitude XT3 with Intel® Core™ i7-26400M processor and 8.00 GB of installed RAM. It has a 64-bit operating system running Windows 10.

Beginning with the static BN with no learned parameters, computational costs are minimal as no learning process is involved. When a parameter is observed, probabilities update immediately. However, it is quite labor intensive to manually build CPDs. For this reason, the CPDs were extremely simplified to the point where the model loses usefulness as discovered in the analysis of randomly sampled days in Section 3.4.

As seen in Section 3.3.2, applying the learning process to the parameters introduces wait time while the computer performs its calculations. A significant decrease in time, an impressive 29:50 minutes, was discussed in Section 3.3.2 when choosing uniform CPDs for learning over using the CPDs formed for the non-learning model (Table 3.2, Table 3.3, Table 3.4, Table 3.5, Table 3.6, and Table 3.7). Both initial conditions produced the same results. However, recall that when using the initial CPDs, the total time for learning was 30 minutes (versus the 10 seconds when using uniform CPDs). Referring back to Section 2.5, the results for the two initial conditions are the same because as the size of training data, M, increases, the initial conditions become less significant. A reasonable conclusion to make is that by getting the same learning results from both types of initial CPDs, a uniform distribution is sufficient with the current set of training data. Learning the parameters introduces a slight increase in computing time, but ten seconds is faster than a human is able to input CPDs in all of

the nodes' properties. This network is static so it comes with similar positive and negative aspects as the static version without learning. There is still no wait time when observed variables are updated.

The DBN uses between 25 and 30 minutes to complete the learning process given uniform CPDs. A user may be able to decrease this training time by making assumptions about flight times. If a pilot knows he will only be flying during daylight hours, the list of possible outcomes of time decreases which simplifies the children CPDs. It also will shorten the training data. While this time is much greater than the learning time for the static network, analyzing the outcomes is much less work for a user. The values still update immediately, but 24 hours of probabilities are output at a time in useful charts through GeNIe like Figure 3.4.

Chapter 4

Conclusions

4.1 Benefits of DBNs over Static BNs

As seen in Section 3.4 the methods using learning parameters, the static BN and the DBN, have very similar capabilities of predicting future weather when no variables are observed. These predictions are correct the majority of the time, but consistently fail to predict outcomes with a lower probability. Outcomes with a lower probability are often associated with higher risk. If a pilot uses this information to decide whether or not to fly, more risk may be involved than is anticipated. However, given as little as one observation in the DBN for a 24-hour time period drastically improves the possibility of the model making correct predictions for time steps after the observation. An observation in the static network has no effect on the probabilities in future time steps because all variables are independent across time steps. When a variable is observed, the DBN becomes much better at predictions but tends to be on the conservative side meaning conditions are predicted to be worse than they turn out to be. For a pilot trying to determine the safe-to-proceed it is better to be more cautious in most situations so conservative predictions may be acceptable. To make observations, a pilot simply needs to visit WeatherSTEM in the morning before flying. The pilot will be able to quickly set a handful of observations to the time slices. Checking the weather is already a

standard part of any good pilot's pre-flight routine, so applying observations will not cause a large burden.

Another benefit of the DBN is how information for the full 24 hours of the day selected is visible on one screen. The capability of viewing how weather will progress through the day is more convenient for a pilot in the form output by the DBN than the static BN. A pilot can not only view predictions throughout the day, but a pilot can also input weather data predicted from the WeatherSTEM report as observations in the time steps to view only the *Battery Safety* and *Flight Safety* predictions. By doing this, probabilities are removed from the weather variables so there is no more possibility of other outcomes to occur in the model. While the WeatherSTEM forecast is almost definitely better at predicting the weather, weather predictions using traditional meteorology are not accurate all of the time so by setting these predictions as observations the assumption becomes that the WeatherSTEM predictions are always correct. Probabilities for the weather variables incorporates some uncertainty, but observing the weather variables using the weather forecast does not account for uncertainty in the observations. Whether or not the pilot decides to input a complete weather forecast, the risk predictions will be displayed in an advantageous manner by the DBN.

The tools and methods described in this thesis can help an sUAS operator to build a network that suits his aircraft's specific requirements. The operator can choose variables that are relevant to the flight environment and variables that have an impact on the safety of flight. If a pilot builds such a network for an sUAS, it can be constantly updated as it is passed down to future students and pilots as more data and observations are collected. The static network with no learning will not be reliable in most flight situations. Adding in the learning parameters makes some improvements, but is overall cumbersome and will not include effects of known conditions when making future predictions. The observation from this thesis is for a pilot to use a dynamic Bayesian network with learning parameters to aid the decision of whether or not a vehicle is safe to fly.

4.2 Drawbacks of DBNs

A drawback for the use of DBNs is the preparation and construction of the network. Careful consideration must be put into which variables should be represented in the network and how each variable relates to every other variable. Increasing time steps, number of nodes, and training data will all cause a significant impact on the number of calculations that need to be performed. It is important to include only the variables that are necessary and only expand the temporal plate to the minimum number of useful time steps. If the variables have been selected and there is training data available, then it is possible to shift the responsibility of connecting variables to a computer through structure learning. This requires extensive historical data, but reduces the need for human expertise. Structure learning will also increase the computation time significantly, so someone constructing a DBN needs to weigh the benefits of structure learning and manually constructing a network.

Another drawback of DBNs is the time necessary for learning the parameters which depends on how much training data is available. As the amount of training data increases, the time it takes to learn parameters increases as well. For the main example used in this thesis, the parameters take 30 minutes to learn with the full set of available data beginning in January of 1999 and ending in July of 2017. When the data set is halved, the learning process reduces to about 14 minutes. This difference in training time shows that time can be saved by determining the minimum set of data points that will provide useful results. Dividing the test data to only include half of the 17 years did change the probabilities by a few hundredths, but general probability trends remained similar. Because of this, it is likely that the complete data set used for the analysis is not the optimal size and can be reduced. Performing the optimization on data size will take even more computation so there needs to be a proper trade-off between time it takes to optimize and time saved in the learning process.

4.3 Topics for Future Consideration

There are some research areas in this thesis that can be expanded on in future work. The network that was used in the main example of this thesis was created by hand. However, network structure learning could be applied to the example as well. There are many different learning algorithms that can learn the network structure from data and it may be useful to compare the networks built with different methods and to the network used in this thesis. Another research area is exploring how changing the form of the training data changes the utility of the DBN. For example, the user may only be interested in risk over the next 6 hours so data would need to be grouped to accommodate this change. It would also be useful to organize the training data in a way that allows the previous day to affect the day in question. An analysis on how much training data is needed for a properly trained set of parameters is an important future research area. Decreasing the number of data points could drastically reduce the computational cost, but it must not be done in a way that significantly decreases the quality of the parameters.

There is also future work possible on the implementation of a DBN in larger scale risk analysis on more complex systems. An organization could explore what it will cost in time and money to create and use a DBN for risk prediction of an existing aircraft. This entails figuring out how much man power is needed to design such a network and how much computation must be done. It is also important to examine the future upkeep, re-training, and improvements compared to the initial effort. Then, the DBN method for estimating risk must be compared for cost, time, and effectiveness to any currently used methods and other possible ways to predict risk like Monte Carlo simulations and fault tree analysis.

Bibliography

[1] Federal Aviation Administration. Flight Risk Assessment Tools.

[2] Thomas Bayes. An essay towards solving a problem in the doctrine of chances. By the late Rev. Mr. Bayes, F.R.S. communicated by Mr. Price, in a letter to John Canton, A.M.F.R.S. *Philosophical Transactions (1683-1775)*, pages 370–418, 1763.

[3] BayesFusion, LLC. GeNIe Modeler with SMILE Engine 2.2.2601.0 Academic. https://www.bayesfusion.com/, Feb 2018.

[4] Matthew de Kock. Weather forecasting using dynamic Bayesian networks: report on dynamic Bayesian network learning. Department of Computer Science, University of Cape Town, 2008.

[5] Finale Doshi-Velez, David Wingate, Joshua Tenenbaum, and Nicholas Roy. Infinite dynamic Bayesian networks. International Machine Learning Society, Massachusetts Institute of Technology, 2011.

[6] X Gao, J Ren, and D Chen. Developing an effective algorithm for dynamic UAV path planning with incomplete threat information. In *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, volume 226, pages 413–421. SAGE Publications Sage UK: London, England, 2012.

[7] Pascal Haingura. Weather forecasting using dynamic Bayesian networks: weather visualization system. Department of Computer Science, University of Cape Town, 2008.

[8] Michael Kent. Weather forecasting with Bayesian networks: causal modeling. Department of Computer Science, University of Cape Town, 2008.

[9] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.

[10] Jim Turner. Lithium battery performance in cold temperatures. https://optibike.com/lithium-battery-performance-in-cold-temperatures/, Dec 2009.

[11] Virginia Agricultural Experiment Station at the Virginia Tech College of Agriculture and Life Sciences. College Farm: weather data. https://vaes.vt.edu/college-farm/weather/2017weather.html.

[12] WeatherSTEM. VT Kentland Farm. https://montgomery.weatherstem.com/kentlandfarm.