PGM Programming Assignment: Decision Making

1 Overview

You have learned to how to model uncertainty in the world using graphical models and how to perform queries on those models to ask questions about the world. You have also learned the basics of decision making in this uncertain world using the principle of maximum expected utility. In this assignment, you will explore how to represent decision making in the framework of graphical models along with some of the algorithmic issues that arise. Specifically, you will learn how to represent influence diagrams as a sort of Bayes Net and perform inference over this representation. By the end of this assignment, you will have applied these new techniques to solve a real-world problem in medical care and treatment.

This assignment has a companion quiz. As you go through this assignment, you will be asked to answer the appropriate questions in the companion quiz.

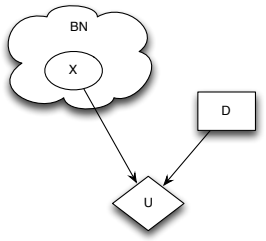
2 Background - Arrhythmogenic Right Ventricular Dysplasia (ARVD)

ARVD is a heart disease that is characterized by progressive replacement of cardiac muscle by adipose (fat) cells and/or fibrosis (scar tissue), starting in the right ventricle. ARVD patients can have potentially deadly arrhythmias (irregularities in the heart rate). There is a large heritable component to the disease - up to half of cases are linked to a family history of the disease. It is a progressive disease, and people with ARVD can remain asymptomatic for a long time without suspecting that they have this condition. Currently, most patients with ARVD have an implantable cardioverter defibrillator (ICD) surgically implanted. These devices are quite effective in reducing arrhythmias.

However, ARVD can be difficult to diagnose with certainty. While having known genetic risk factors confer susceptibility to developing ARVD, it is far from certain. Furthermore, diagnostic tests that directly measure some characteristic of heart function are not perfect, with some false positives and false negatives occurring. Different diagnostic tests also have different risks - getting some electrodes put on your chest during a stress test has different risks than surgery. How can you make sense of all this? In this assignment, we will bring the machinery of graphical models and decision theory to bear on these sorts of problems.

3 Notation and Definitions

3.1 Influence diagrams - random variables, decision nodes, and utility nodes.



We use the influence diagram representation that was introduced in lecture. A simple influence diagram is shown above, where X is a random variable node in some Bayes Net. D represents a decision we get to make. U is our utility, and in this assignment we focus on how to make decision D to maximize the expected utility. More formally, an influence diagram $\mathcal{I}$ is a directed, acyclic graph with three kinds of nodes:

- The first kind of node represents a random variable, denoted $X \in \mathcal{X}$. Each X has an associated factor that is its CPD in a Bayes Net, $P(X|Pa_X)$. We draw random variables as ovals. We will represent these factors using the same factor structure you used in previous assignments. In our ARVD example, we might have a random variable that represents the presence or absence of a particular genetic risk factor, and that random variable could depend on the presence or absence of the risk factor in the patient's parents.

- The second kind of node represents a decision, $D \in \mathcal{D}$, which we draw with a rectangle. A decision node also has a factor associated with it that is a deterministic CPD. In our ARVD example, our decision could be whether or not to surgically implant an ICD. The "CPD" for decision node D is called the decision rule, δ_D, because it tells us what action to take based on the inputs, i.e. it maps joint assignments of Pa_D to an action. Specifically, **a decision rule for D is represented by a CPD that assigns probability 1 to single action for each joint assignment to the parents of D, with zeros for all other actions.** Because we model a decision node as a random variable with a deterministic CPD, we can consider $\mathcal{I}$ to be a Bayes Net as far as the random variables and decision nodes are concerned.

  An optimal decision rule for decision node D is one which results in the maximum expected utility. Different decision rules can result in the same expected utility, so an optimal decision rule is not necessarily unique. In this assignment, we restrict our attention to decision problems that have exactly one decision node and represent decision nodes using the same factor structure we use for random variables. By convention, the decision variable itself is the first variable listed in the structure, i.e. `D.var(1)`, so the parents are `D.var(2:end)`.

- The third kind of node represents a utility node, $U \in \mathcal{U}$. These nodes are drawn as diamonds. Utility nodes never have children (whereas random variable nodes and decision nodes may both appear anywhere in $\mathcal{I}$). They are also associated with factors, but of a different sort than usual. Utility factors map each possible joint assignment of their parents, Pa_U, to a utility value. For instance, in our ARVD example, we could have utility node which maps the presence or absence of bad outcomes due to ARVD to a numerical

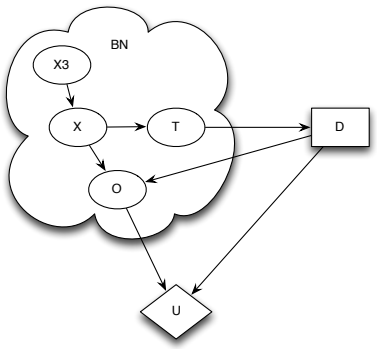
utility value. Utility factors can have negative values (one reason we were careful to not restrict factors to be non-negative early on in the course) since it is possible to have negative utility. In this assignment, we represent $U \in \mathcal{U}$ as factors over the parents of U using the same factor structure we use for random variables and decision nodes. Note that we do not introduce new random variables for the utility nodes themselves. Thus, `U.var` is a vector of the parents of U with no entry for U itself.

Because we are using regular CPDs for random variables $X \in \mathcal{X}$, deterministic CPDs for decision nodes $D \in \mathcal{D}$, and utility factors for utility nodes $U \in \mathcal{U}$, our data structure I is simply a struct array with three fields:

- I.RandomFactors - a list of normalized factors, one for each random variable $X \in \mathcal{X}$. These are simply the CPDs for each random variable.
- I.DecisionFactors - a list of deterministic factors, i.e. decision rules δ_D for each decision variable $D \in \mathcal{D}$. Again, note that we restrict ourselves to single decision problems in this assignment.
- I.UtilityFactors - a list of utility factors, one for each utility factor $U \in \mathcal{U}$.

We have also provided some utility functions for you to use - `PrintFactor.m` and `VariableElimination.m` - that you may find useful. PrintFactor prints out factors in an easily readable format, while VariableElimination runs Sum-Product on a list of factors, marginalizing out the specified variables. These functions are documented further in their respective implementation files. **We have also provided some test data and results in `TestCases.m` to help you debug your code.** `TestCases.m` contains definitions of some simple influence diagrams and the expected results of various inference tasks you are asked to implement.

4 Calculating Expected Utility Given A Decision Rule



Consider the above model for our decision process. The node X represents ARVD, and can have values x^1 or x^2 (for having ARVD or not, respectively). The node D is a decision node, and the decision we must make is whether or not to surgically implant an ICD (D can take values d^1 and d^2 for not having surgery or having surgery, respectively). We make this decision based on the result of an observation T of a test for ARVD; thus T is the sole parent of D. Because not every person with ARVD suffers from his/her disease, we introduce another random variable that is a function of ARVD. We will call this variable O for 'outcome', and it represents whether

an undesirable outcome occurred. O has parents X and D because the probability of a bad outcome is a function of having ARVD and is modified by having an ICD implanted. The node U is the utility node, and it has 2 parents: O and D. In other words, the utility depends on the decision we make regarding treatment and on whether or not we have ARVD. Obviously, there is little (or even negative) utility if we do not have ARVD and we implant an ICD. But if we do have ARVD then having an ICD implanted might save our life, which obviously has high utility. How do we evaluate the expected utility of a given decision rule?

First, let's say that our decision rule is to have surgery if $T = t^2$ and to not have surgery if $T = t^1$. In our representation for decision rules as deterministic CPDs, this is:

$$\begin{bmatrix} d^1t^1 & \rightarrow & 1 \\ d^2t^1 & \rightarrow & 0 \\ d^1t^2 & \rightarrow & 0 \\ d^2t^2 & \rightarrow & 1 \end{bmatrix}.$$

Now, if we delete the utility node U from the above influence diagram, we have a regular Bayes Net. We can thus run an inference algorithm such as variable elimination on it to get the marginal distribution over the parents of U, $P(O, D)$. We represent this distribution as a factor mapping joint assigments of O and D to their respective probabilities, for example:

$$\begin{bmatrix} d^1o^1 & \rightarrow & 0.15 \\ d^2o^1 & \rightarrow & 0.7 \\ d^1o^2 & \rightarrow & 0.1 \\ d^2o^2 & \rightarrow & 0.05 \end{bmatrix}$$

Now consider the following utility factor for U:

$$\begin{bmatrix} d^1o^1 & \rightarrow & 300 \\ d^2o^1 & \rightarrow & 50 \\ d^1o^2 & \rightarrow & -400 \\ d^2o^2 & \rightarrow & -500 \end{bmatrix}.$$

When we multiply these factors, we are multiplying the utility for each joint assignment to Pa_U by its probability. The result of summing out over the joint assignments is just the expected utility. In other words, marginalizing the product of probability factors and a utility factor yields an expected utility:

$$EU\left[\mathcal{I}\left[\sigma\right]\right] = \sum_{\mathcal{X} \cup \mathcal{D}} \prod_{X \in \mathcal{X}} P(X|Pa_X)\delta_D U$$

If you do the factor multiplication for the two factors shown above and then sum up the results you should get 15. You will now implement a function for computing an expected utility and use it to compute expected utilities for some influence diagrams.

- `SimpleCalcExpectedUtility.m (5 pts)` This function takes an influence diagram with a single decision node (which has a fully specified decision rule) and a single utility node, and returns the expected utility. Hint - you can use the provided function, VariableElimination, to perform Sum-Product to get the marginal on the parents of the utility node.

- `Quiz Question 1.` (Answer the quiz questions online.) We have provided an instantiated influence diagram `FullI` (complete with a decision rule for D) in the file `FullI.mat`. What is the expected utility for this influence diagram?

- `Quiz Question 2.` Run `ObserveEvidence.m` on `FullI` to account for the following: We have been informed that variable 3 in the model, which models an overall genetic risk for ARVD, has value 2 (indicating the presence of genetic risk factors). Then run `SimpleCalcExpectedUtility` on the modified influence diagram. What happened to the expected utility? (Hint – ObserveEvidence does not re-normalize the factors so that they are again valid CPDs unless the normalize flag is set to 1. – If you do not use the normalize flag, you can use `NormalizeCPDFactors.m` to do the normalization.)

5 Maximum Expected Utility - Naive Approach

In the previous scenario, we had a simple decision process and a fully specified decision rule. The decision rule corresponded to choosing to have surgery to implant an ICD if the test T came back positive for ARVD, and to not have surgery otherwise. But was this the right decision? Furthermore, what if we have many more inputs into our decision, such as other test results? A naive approach to figuring out the best decision rule is to explicitly enumerate all possible decision rules and to run our function SimpleCalcExpectedUtility on each, keeping the one that results in the highest expected utility.

- `Quiz Question 3.` Why can we explicitly enumerate all the possible decision rules for whether or not to implant an ICD even though in general we cannot enumerate over all possible CPDs?
- `Quiz Question 4.` Let a decision node D take on d values. Let it have m parents that can each take on n values. How many possible decision rules δ_D are there?

6 Maximum Expected Utility With Expected Utility Factors

We can avoid enumerating all possible decision rules by noting the following:

$$\begin{aligned} EU\left[\mathcal{I}\left[\delta_{\mathcal{D}}\right]\right] &= \sum_{\mathcal{X}\cup\mathcal{D}}\prod_{X\in\mathcal{X}} P(X|Pa_X)\delta_D U \\ &= \sum_{D,Pa_D}\delta_D \sum_{\mathcal{X}-Pa_D}\prod_{X\in\mathcal{X}} P(X|Pa_X)U \\ &= \sum_{D,Pa_D}\delta_D \mu_{-D}\left(D,Pa_D\right) \end{aligned}$$

$\mu_{-D}\left(D,Pa_D\right)$ is a factor that is almost what we want to calculate, except we haven't multiplied in the deterministic factor δ_D. This factor doesn't depend on our choice of decision rule. Furthermore, note that δ_D is deterministic and simply picks out particular entries from $\mu_{-D}\left(D,Pa_D\right)$. Thus, given $\mu_{-D}\left(D,Pa_D\right)$ we can obtain an optimal decision rule, δ_C^*, by simply scanning through $\mu_{-D}\left(D,Pa_D\right)$:

$$\delta_D^*\left(pa_D\right) = \underset{d\in\mathcal{D}}{\operatorname{argmax}}\mu_{-D}\left(d,pa_D\right)$$

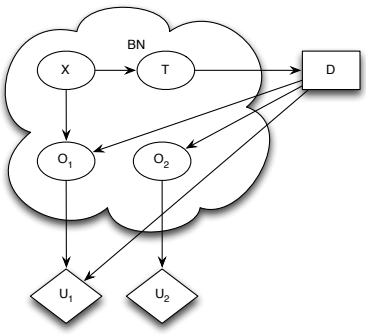
where pa_D denotes a joint assignment to the parents of D. Let's say that we have a decision node D with actions $\{d^0, d^1\}$. D has a single parent X that takes on values $\{x^0, x^1\}$. We have calculated the expected utility factor:

$$\mu_{-D}(D, X) = \begin{bmatrix} x^0 d^0 & \rightarrow & 10 \\ x^0 d^1 & \rightarrow & 1 \\ x^1 d^0 & \rightarrow & 2 \\ x^1 d^1 & \rightarrow & 5 \end{bmatrix}$$

We can read off the optimal decision rule by noting that if $X = x^0$the expected utility for $d^0$is 10, while the expected utility for action $d^1$is 1. Thus, the optimal decision should choose $d^0$when $X = x^0$. We can similarly deduce that the best action to take when $X = x^1$is d^1.

- `CalculateExpectedUtilityFactor.m (15 Pts)` This function takes an influence diagram $\mathcal{I}$ that has a single decision node D and returns the expected utility factor of $\mathcal{I}$ with respect to D. You may again find the provided function, VariableElimination, useful.
- `OptimizeMEU.m (15 Pts)` This function takes an influence diagram $\mathcal{I}$ that has a single decision node D and returns the maximum expected utility and a corresponding optimal decision rule for D. You should use your implementation of CalculateExpectedUtilityFactor in this function.
- `Quiz Question 5.` Consider an influence diagram with 1 decision node D that can take on d values. Let D have m parents that can each take on n values. Assume that running sum-product inference takes $O(S)$ time. What is the run-time complexity of running `OptimizeMEU` on this influence diagram?

7 Multiple Utility Factors



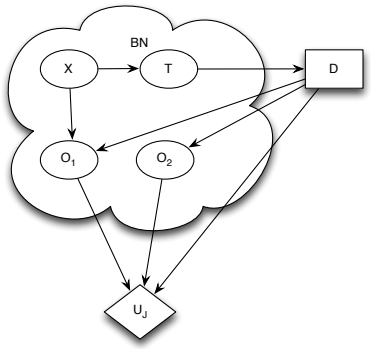
Up until now, we have assumed that there is only one utility node. But we often want to model utility as a function of many variables. This could impose a significant computational and knowledge engineering cost as the joint utility factor becomes exponentially large. As discussed in lecture, we can often decompose the utility into separate terms, each of which has relatively small scope. The total utility is then the sum of these components. This could be handy if the intervention we are considering, surgery to implant an ICD, has it own risks.

Concretely, we will model the risks of surgery as another random outcome variable, O_2, which is a child of D. Note that we have renamed our original outcome O_1. O_2 models an unlikely but

very bad outcome of surgery, such as serious infection. It is natural to model the utility of this as a distinct utility, U_2, which assigns a large negative utility to the event of having a catastrophic accident, and a small positive utility to avoiding such a fate. We can also assign utility values to outcomes such as the patient giving up competitive athletics (ARVD patients are recommended to not engage in strenuous exercise) - some patients would consider this a negative impact on their quality of life.

Mapping these disparate events to the same utility scale, allowing us to consider how to make decisions in a single coherent framework, is an important aspect of decision theory. So how do we deal with having multiple utility nodes in our model, shown in the figure above? You will now implement two simple approaches to dealing with multiple utility nodes. First, we will learn how to combine factors so we create a single joint factor. Second, we will take advantage of the linearity of expectations.

7.1 Joint utility factors



One way to deal with multiple utility factors is to combine the utilities for the different nodes into a single joint utility factor, reducing the model to the one shown above. We could then run OptimizeMEU on our new influence diagram. How do we combine utility factors? Basically, we do the same thing as FactorProduct, except that we add entries instead of multiplying them. Some of you may have already written a helper function that does this when adding factors in log-space for max sum inference in PA4. For instance, consider the following factors from our model:

$$U_1 = \begin{bmatrix} o_1^0 d^0 & \rightarrow 1 \\ o_1^1 d^0 & \rightarrow 2 \\ o_1^0 d^1 & \rightarrow 3 \\ o_1^1 d^1 & \rightarrow 4 \end{bmatrix}, U_2 = \begin{bmatrix} o_2^0 & \rightarrow 10 \\ o_2^1 & \rightarrow 20 \end{bmatrix}$$

$$U_1 + U_2 = \begin{bmatrix} o_1^0 d^0 o_2^0 & \rightarrow 11 \\ o_1^1 d^0 o_2^0 & \rightarrow 12 \\ o_1^0 d^1 o_2^0 & \rightarrow 13 \\ o_1^1 d^1 o_2^0 & \rightarrow 14 \\ o_1^0 d^0 o_2^1 & \rightarrow 21 \\ o_1^1 d^0 o_2^1 & \rightarrow 22 \\ o_1^0 d^1 o_2^1 & \rightarrow 23 \\ o_1^1 d^1 o_2^1 & \rightarrow 24 \end{bmatrix}$$

The situation would be somewhat more complicated if we had overlapping scopes for the two utility factors, but the basic idea is the same.

- `OptimizeWithJointUtility.m (15 Pts)` This function takes an influence diagram with a single decision node D and possibly many utility nodes and returns the MEU and corresponding optimal decision rule for D. Hint - you will have to write a function that implements the Factor Sum operation. If you wrote this function for Programming Assignment 4, feel free to use it here!

7.2 Linearity of Expectations

An alternative method is to make use of the linearity of expectations. When we have multiple utility nodes, we have to account for them all in our expression for the expected utility given δ_D:

$$\begin{aligned} EU\left[\mathcal{I}\left[\sigma\right]\right] &= \sum_{\mathcal{X}\cup\mathcal{D}} \prod_{X\in\mathcal{X}} P(X|Pa_X)\delta_D \sum_{U_i\in\mathcal{U}} U_i \\ &= \sum_{D,Pa_D} \delta_D \left[\sum_{U_i\in\mathcal{U}} \left(\sum_{\mathcal{X}-Pa_D} \prod_{X\in X} P(X|Pa_X)U_i\right)\right] \\ &= \sum_{D,Pa_D} \delta_D \left[\sum_{U_i\in\mathcal{U}} \mu_{-D,U_i}\right] \end{aligned}$$

In other words, we can compute the expected utility factor with respect to D and each utility node separately and combine them later. We denote the expected utility factor with respect to D and U_i as μ_{-D,U_i}.

- `OptimizeLinearExpectations.m (15 Pts)` This function takes an influence diagram $\mathcal{I}$, with a single decision node and possibly with multiple utility nodes and returns an optimal decision rule along with the corresponding expected utility. Verify that the MEU and optimal decision rule is the same as that returned by OptimizeWithJointUtility.
- `Quiz Question 6.` When does it make sense to use `OptimizeWithJointUtility` instead of `OptimizeLinearExpectations` for optimizing the utility?

8 The Value of Perfect Information And Utility Functions

Test	Sensitivity	Specificity
T1	99.9%	75%
T2	75%	99.9%
T3	99.9%	99.9%

Finally, let's explore how we can use the concept of the Value of Perfect Information to decide among some different options for diagnostic tests. Let's say that we have several options for our diagnostic test summarized in the table above. Two of the tests are quite cheap but have either poor sensitivity or specificity[1], while the last test has excellent sensitivity and specificity but is quite expensive. We would like to know which, if any, of the tests we should pay for.

[1] Sensitivity is defined as the probability of a test result indicating ARVD given that ARVD is in fact present, i.e. $P(t^2|x^2)$. Specificity is defined as the probability of a test result indicating ARVD is not present given that ARVD is in fact not present, i.e. $P(t^1|x^1)$.

We can approach this question via the concept of the Value of Perfect Information (VPI), which was introduced in lecture. The basic idea of VPI is that the value of information is the change in the MEU we can achieve given that information. This change is often measured relative to an influence diagram, $\mathcal{I}$ in which there is no edge between T and D - i.e. the decision is made without the information. Concretely, given an influence diagram $\mathcal{I}$, we can run one of the algorithms we implemented above to get the MEU. Then we can modify the influence diagram by adding an edge between T and D, yielding a modified influence diagram $\mathcal{I}_{T \to D}$ which we can similarly optimize to find the MEU. The value of perfect information (in utility units) is then $VPI = MEU\left[\mathcal{I}_{T \to D}\right] - MEU\left[\mathcal{I}\right]$.

- `Quiz Questions 7-9.` We want to know which, if any, the above test options are worth performing. First, you should take the influence diagram in `TestI0.mat` and use any method you wish to find the maximum expected utility (note that it has multiple utility nodes). This provides the baseline for comparison. Then, for each of the test options, start with the influence diagram in TestI0.mat and modify the model so that the test random variable, 11, encodes the above test characteristics (the sensitivity and specificity) in its CPD. Note that the test depends on a single variable, 1, which represents whether or not ARVD is present and has value 1 if not present, 2 if present. Make the decision D dependent on the test variable, and optimize to get the MEU. Assume that the utility of money is given by the function $100 \cdot \ln(d+1) \, \forall d \geq 0$ where d is a dollar amount. How much money (at most) would you actually pay for each of these tests? Which, if any, of these tests are actually worth performing (individually)?

9 Conclusion

Congratulations! You have completed Programming Assignment 6 and now hopefully have some appreciation of how useful the framework of graphical models can be for addressing questions about what we should do in an uncertain world. The techniques you implemented in this assignment are used in many real-world scenarios, from choosing which diagnostic tests to order for patients, to deciding where to next point a robot's camera.

If you found this material interesting, there is a lot more in the textbook about two advanced techniques one can employ in performing inference over influence diagrams. First, when dealing with multiple utility nodes, we resorted to either computing a joint utility factor or using the linearity of expectations. Both of these techniques can have significant computational costs - the first because it computes a large joint factor and the second because we have to compute an expected utility factor for each utility node (thus requiring that we run variable elimination or some other inference algorithm multiple times). It turns out that there is a generalization of variable elimination that allows us to run variable elimination directly on the factors, avoiding these costs. This is explained in detail in Section 23.4.3 of the textbook.

Second, we have limited ourselves to decision making scenarios in which we only had one decision node. In reality, we often must make a series of decisions. For instance, in a more realistic analysis, we might have another decision node that represents the choice to perform a diagnostic test or not. Having multiple decision nodes complicates matters because our optimization must find the joint assignment to the decision rules that achieves the globally optimal expected utility. In some circumstances, it is possible to perform this joint optimization efficiently using a sort of coordinate descent in the space of decision rules for the different decisions; this is covered in Sections 23.5.2 and 23.5.3.