2 The Variable Elimination Algorithm (55 marks)

You will implement the variable elimination algorithm to perform inference in Bayesian networks with discrete random variables.

We have provided four Python files. Please read the detailed comments in the provided files carefully.

1. `factor.py` Defines a factor class for VEA. **Do not change this file.**
2. `vea.py` Contains empty functions for VEA operations. You will implement all of the functions in this file. **Do not change the function signatures.**
3. `example.py` Provides a demonstration of how to define the factors for a Bayesian network (for the Holmes scenario) and execute the variable elimination algorithm.

Here are some tips for implementing the variable elimination algorithm:

1. `restrict()`: Consider using splicing operations or `take`.
2. `multiply()`: Consider using the `numpy` broadcasting rules to multiply multidimensional arrays of different shapes.
3. `sum_out()`: Consider using the `sum` operation.
4. You may find it helpful to print the VEA operations and intermediate factors. See Appendix A for tips on printing the output of VEA. Note that this is optional.

Please complete the following tasks.

(a) Implement the empty functions in `vea.py`. Zip and submit this file only on Marmoset.

Marking Scheme: (49 marks)

Unit tests:

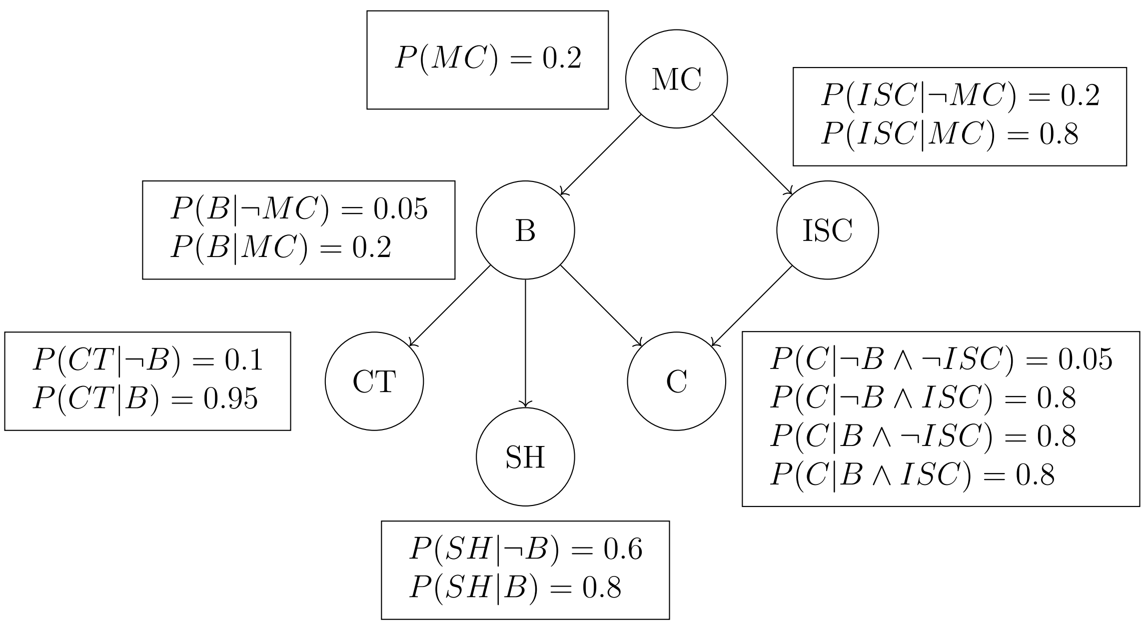
- `normalize`
  (1 public test + 2 secret tests) $\times$ 1 mark = 3 marks
- `restrict`
  (1 public test + 2 secret tests) $\times$ 2 marks = 6 marks
- `sum_out`
  (1 public test + 2 secret tests) $\times$ 2 marks = 6 marks

- `multiply`
  (1 public test + 7 secret tests) × 2 marks = 16 marks
- `vea`
  (1 public test + 5 secret tests) × 3 marks = 18 marks

Solutions: Code submiited.

(b) Below is a Bayesian network that appears in a recent review of uncertainties in Bayesian networks with discrete variables (Rohmer, (2020)). The network is a probabilistic model of brain cancer diagnosis. Note that the example is meant to be illustrative, as the network is simple and the conditional probability tables are fictional. The random variables are defined below:

Random Variable	Definition
MC	Metastatic cancer
B	Brain tumour
CT	CT scan is positive for brain tumour
SH	Severe headache
ISC	Increased serum calcium
C	Patient falls into coma



Using your implementation from the previous question, execute the Variable Elimination Algorithm to determine the probability that a patient has a brain tumour, given that their blood work shows an increased calcium concentration, and they do not report having severe headaches. Eliminate hidden variables in reverse alphabetical order. For example, you would eliminate MC before ISC and CT before C. For each random variable's domain, consider `False` as 0 and `True` as 1.

Marking Scheme: (2 marks)

- (2 marks) Correct answer

Solutions: $P(B|ISC \wedge \neg SH) = 0.0667$

(c) The complexity of VEA depends on the *treewidth* of the Bayesian network, which is the maximum number of random variables in a factor after summing out a variable. The order in which hidden variables are eliminated impacts the treewidth, thereby impacting the complexity.

Use your implementation of VEA to evaluate the probability of metastatic cancer given that a patient's CT scan is positive for a brain tumour (i.e., $P(MC = 1|CT = 1)$). Do not prune the network. Execute VEA for each possible order of eliminating the random variables and note the treewidth in each case. Produce a histogram of the treewidths for each execution of VEA.

Hint: You might find `itertools.permutation()` useful in determining the possible permutations of a list of strings.