Lab 2

CS 3793/5233 – Fall 2013
Tom Bylander, Instructor

assigned September 13, 2013
due midnight, October 7, 2013

In Lab 2, you will complete a program for a Bayesian network for determining the probabilities of ABO blood types. Based on knowing the family relationships and some of the blood types of the family, the program will be able to determine the probabilities of the blood types for the other members of the family.

The initial program is lab2.zip, which you can download from the course web site. This code includes methods for making inferences from a Bayesian network and some of the probabilistic relationships and population assumptions for the genetics of the ABO blood group system. It is based on the variable elimination algorithm in Section 6.4.1. The code implements the algorithm. You need to put Bayesian networks together for different family relationships and call upon the inference mechanism.

Background Knowledge

[Heavily borrowed from http://en.wikipedia.org/wiki/ABO_blood_group_system#Inheritance]

Blood groups are inherited from both parents. The ABO blood type is controlled by a single gene (the ABO gene) with three alleles: O, A, and B. An allele is an alternative form of the same gene.

With one notable exception, a person has two copies of each gene, one from the mother and one from the father. That is, the person has an ABO gene from the mother (O allele or A allele or B allele), and an ABO gene from the father (O allele or A allele or B allele). With two copies, this results in nine possible combinations of alleles (listing the mother first): OO, OA, OB, AO, AA, AB, BO, BA, AB. Which parent an allele comes from does not affect the blood type, so the OA combination and AO combination have the same result, OB and BO have the same result, and AB and BA have the same result.

The O allele results in blood type O, The A allele results in blood type A, and the B allele results in blood type B. Both the A allele and B allele are dominant over the O allele, so only OO people have type O blood. Individuals with AA or OA or AO have type A blood, and individuals with BB or OB or BO have type B blood. AB and BA people have type AB blood, because the A allele and the B allele express a special dominance relationship: codominance.

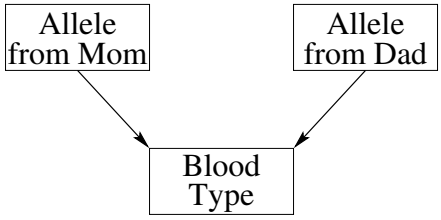
Note that it is possible for a type A mother and a type B father to have a type O child because one parent might have the AO combination, and other parent BO. That is, the child

might receive an O allele from the mother and an O allele from the father.

Probabilistic Relationships

Blood Type

A person's blood type is a function of the person's alleles, so the Bayesian network would have the following subgraph:

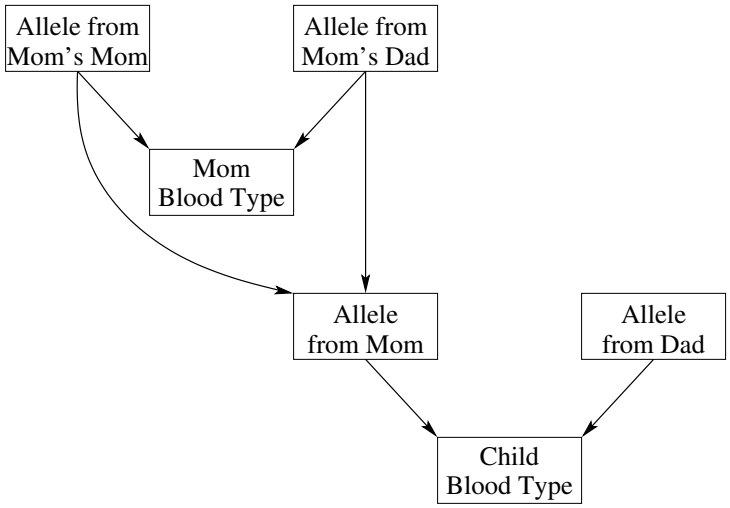


with the following probability table for the blood type:

		P(Blood Type | Alleles)			
Allele from Mom	Allele from Dad	O	A	B	AB
O	O	1.0	0.0	0.0	0.0
O	A	0.0	1.0	0.0	0.0
O	B	0.0	0.0	1.0	0.0
A	O	0.0	1.0	0.0	0.0
A	A	0.0	1.0	0.0	0.0
A	B	0.0	0.0	0.0	1.0
B	O	0.0	0.0	1.0	0.0
B	A	0.0	0.0	0.0	1.0
B	B	0.0	0.0	1.0	0.0

Allele

A Bayesian network that includes the mother would look like:



with the following probability table for the allele from the mother:

Allele from Mom's Mom	Allele from Mom's Dad	P(Allele from Mom | Mom's Alleles) O	A	B
O	O	1.0	0.0	0.0
O	A	0.5	0.5	0.0
O	B	0.5	0.0	0.5
A	O	0.5	0.5	0.0
A	A	0.0	1.0	0.0
A	B	0.0	0.5	0.5
B	O	0.5	0.0	0.5
B	A	0.0	0.5	0.5
B	B	0.0	0.0	1.0

The allele from the father would have the same probability table with, of course, different alleles being referenced.

Allele Priors

For a finite Bayesian network, the modeling of family relationships needs to stop at some point. This means that the alleles of some number of people will have no parent alleles specified. In that case, the Bayesian network needs a probability table that specifies the prior probabilities of the alleles:

P(Allele)		
O	A	B
0.6	0.3	0.1

In reality, these values vary over different population subgroups (look further down on the Wikipedia web page). The numbers in the table are crude approximations for the United States. The one-digit accuracy allows hand computation for simple Bayesian networks. **(Shared extra credit 100 pts.) Work backward from the numbers given for the United States to come up with prior probabilities to three decimal places. Show your work.**

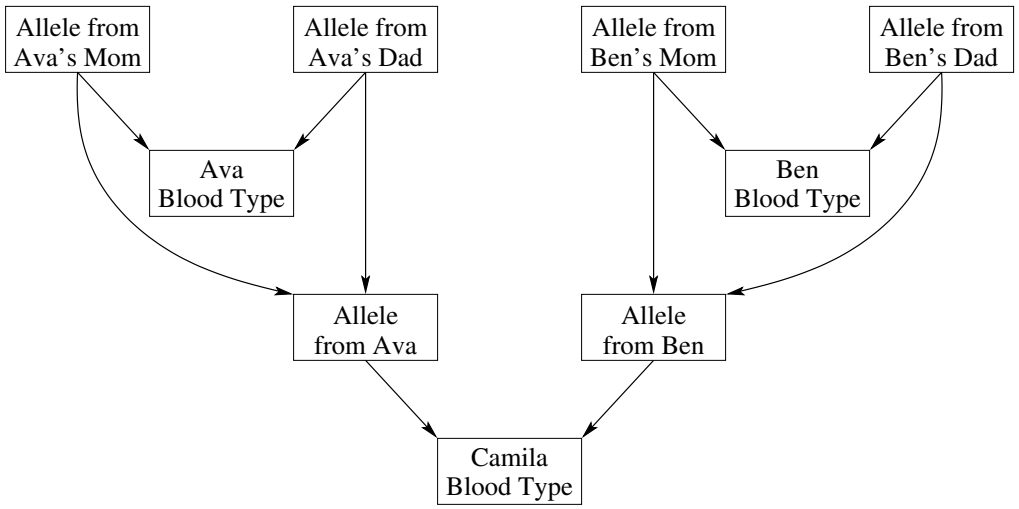
Example Network

`data.txt` in `lab2.zip` contains problems to solve using Bayesian networks. The second problem is:

```
start
person Ava
person Ben
person Camila parents Ava Ben
observe Camila bloodtype AB
infer Ava bloodtype
# should be O: 0.00, A: 0.45, B: 0.35, AB: 0.20
end
```

This problem involves three people: Ava, Ben and Camila, where Ava and Ben are Camila's genetic parents. We know that Camila has type AB blood. What is the probability of Ava having a certain ABO blood type? The comment shows the answer in this case, e.g., 0.20 is the probability that Ava has type AB blood given that Camila has type AB blood.

This problem corresponds to the following Bayesian network:



Nine probability tables are needed to fully define this network. The allele prior probability table will be used four times: Ava's two alleles and Ben's two alleles. The blood type table will be used three times, once for each person. The alleles with a parent table will be used twice, once for Camila's allele from Ava and once for Camila's allele from Ben.

The variable elimination algorithm represents these tables as *factors* (see Section 6.4.1). Observing Camila as type AB will zero all the values associated with types O, A and B for Camila. Then the variable elimination algorithm multiplies factors and sums out variables until only Ava's blood type is left.

The Programming in Lab 2

Although you students would undoubtedly learn more if you had to implement Bayesian networks and variable elimination from scratch, your instructor has decided to try to simplify the creation and operation of Bayesian networks for you. The disadvantage is trying to reverse-engineer the instructor's design while trying to implement the missing pieces. Hopefully, this explanation and the comments in the code will help.

Your Tasks

This section summarizes the parts you must finish to complete this assignment. The remaining sections are intended to give you enough context to understand what else is going on.

1. Fill in the right probabilities for the factors that are created in `ABOFactors.java`. All the numbers are provided above. It should relatively straightforward to fill them in.
2. Complete the methods in `ABOMethods.java`.
   (a) `ABOMethods.createFactors` is called for any line that starts with "person". This needs to create the three variables and the three factors for the person named on that line (two alleles and one bloodtype). You will also need to find and use the variables for the parents' alleles. These should have been created for previous "person" lines.
   (b) `ABOMethods.observeEvidence` is called for any line that starts with "observe". This needs one line of code to assign the right value to v.
   (c) `ABOMethods.runNetwork` is called for any line that starts with "infer". This needs one line of code to assign the right value to v.

The Components of lab2.zip

- `Interact.java` runs `Tester.java` (the "environment") at the same time as `ABO.java` (the "agent").
- `Tester.java` reads and prints `data.txt`, and tests whether the output of `ABO.java` matches the solutions in the comments of `data.txt`.
- `ABO.java` reads networks in the format of `data.txt`. A "start" line creates a `BayesianNetwork` object. A "person" line results in a call to `ABOMethods.createFactors`. An "observe" line results in a call to `ABOMethods.observeEvidence`. An "infer" line results in a call to `ABOMethods.runNetwork`. An "end" line is ignored, and a "quit" line quits. Most of the rest of the program is trying to ensure that the lines are correctly specified. The main method allows you to run `ABO.java` by itself.
- `BayesianNetwork.java` defines a Bayesian network as a set of variables and a set of factors. Some error checking is done. You will need to use the `addVariable`, `findVariable` and `addFactor` methods. Calling the other methods as needed is already coded. `BayesianNetworkTest.java` shows how examples in the textbook are implemented.
- `Variable.java` defines variables for Bayesian networks.. All a variable does is keep track of a name (String) and a set of possible values (an array of Strings). The possible values for alleles and blood types are in `ABO.java`. You will need to create `Variable` objects in the `ABOMethods.createFactors` method. There are two things to keep in mind when using `Variable` objects in this program.
  - For each problem, you must not create duplicate names for variables. For example, the name `"Allele from Ava"` in the above figure is a bad name because Ava might have more than one child. A better name would be `"Camila's Allele from Ava"`.
  - Generally, a value is not referred to by name elsewhere in the program. Instead, an index into the array of values is used to refer to a particular value. For example in `ABO.ALLELES`, `"O"` is value 0, `"A"` is value 1, and `"B"` is value 2.
- `Factor.java` defines factors as a set of variables and an assignment of a `double` to each combination of values of the variables. See `FactorTest.java` for examples from the textbook.

  Each probability table maps to a factor. For example, using X_1, X_2 and Y to denote the names of the variables for the blood type probability table, then one value in the probability table is:

$$P(Y = \text{AB} \mid X_1 = \text{B}, X_2 = \text{A}) = 1.0$$

  and could be formally stated in a factor f as:

$$f(X_1 = \text{B}, X_2 = \text{A}, Y = \text{AB}) = 1.0$$

In `Factor.java`, these assignments are represented as indexes.

$$f(2, 1, 3) = 1.0$$

This is because `"B"` is at index 2 of `ABO.ALLELES`, `"A"` is at index 1 of `ABO.ALLELES`, and `"AB"` is at index 3 of `ABO.BLOODTYPES`.

In the program, one parameter for the `get` and `set` method is an int array of indexes. This is because a factor can have any number of variables. The code that should correspond to the above assignment is in `ABOFactors.java`.

```
factor.set(new int[] { 2, 1, 3 }, 0.0);
```

Changing the 0.0 to 1.0 (and setting the correct values for other assignments) is part of this lab.

- `ABOFactors.java` defines the factors that you need for this lab. However, you need to change the probability values so they correspond to the above probability tables. You don't need to call any Factor methods, but you do need to add `Factor` objects to a `BayesianNetwork` object.

Turning in Your Lab

Submit the folder containing all your Java files to Blackboard. Your program should run under the following conditions: after all the .class files are deleted, then:

```
javac Interact.java
java Interact
```

should solve the problems.

The score of your lab will primarily depend on the performance of the program. In addition to the nine problems in `data.txt`, your program will be tested on an additional 11 problems. Your score will be the percentage of problems correctly solved.

Testing Your Program

A correct implementation will produce the following output (give or take some roundoff) when Interact is run. Also, the names of your variables might differ from mine.

```
class Tester start
class Tester person Ava
class Tester infer Ava bloodtype
class ABO Variable Ava blood type: [O, A, B, AB]
class ABO 0 0.36
class ABO 1 0.44999999999999996
class ABO 2 0.13
class ABO 3 0.06
CORRECT
class Tester end
class Tester
class Tester start
class Tester person Ava
class Tester person Ben
class Tester person Camila parents Ava Ben
class Tester observe Camila bloodtype AB
class Tester infer Ava bloodtype
class ABO Variable Ava blood type: [O, A, B, AB]
class ABO 0 0.0
class ABO 1 0.44999999999999996
class ABO 2 0.35000000000000003
class ABO 3 0.19999999999999996
CORRECT
class Tester end
class Tester
class Tester start
class Tester person Ava
class Tester person Ben
class Tester person Camila parents Ava Ben
class Tester observe Camila bloodtype AB
class Tester observe Ben bloodtype A
class Tester infer Ava bloodtype
class ABO Variable Ava blood type: [O, A, B, AB]
class ABO 0 0.0
class ABO 1 0.0
class ABO 2 0.7000000000000001
class ABO 3 0.29999999999999993
CORRECT
class Tester end
```

```
class Tester
class Tester start
class Tester person Ava
class Tester person Ben
class Tester person Camila parents Ava Ben
class Tester observe Camila bloodtype O
class Tester infer Ava bloodtype
class ABO Variable Ava blood type: [O, A, B, AB]
class ABO 0 0.5999999999999999
class ABO 1 0.29999999999999993
class ABO 2 0.09999999999999998
class ABO 3 0.0
CORRECT
class Tester end
class Tester
class Tester start
class Tester person Ava
class Tester person Ben
class Tester person Camila parents Ava Ben
class Tester observe Camila bloodtype O
class Tester observe Ben bloodtype A
class Tester infer Ava bloodtype
class ABO Variable Ava blood type: [O, A, B, AB]
class ABO 0 0.6
class ABO 1 0.3
class ABO 2 0.09999999999999999
class ABO 3 0.0
CORRECT
class Tester end
class Tester
class Tester start
class Tester person Ava
class Tester person Ben
class Tester person Camila parents Ava Ben
class Tester observe Camila bloodtype A
class Tester infer Ava bloodtype
class ABO Variable Ava blood type: [O, A, B, AB]
class ABO 0 0.24
class ABO 1 0.66
class ABO 2 0.04
class ABO 3 0.06
CORRECT
class Tester end
class Tester
class Tester start
```

```
class Tester person Ava
class Tester person Ben
class Tester person Camila parents Ava Ben
class Tester observe Camila bloodtype A
class Tester observe Ben bloodtype B
class Tester infer Ava bloodtype
class ABO Variable Ava blood type: [O, A, B, AB]
class ABO 0 0.0
class ABO 1 0.9
class ABO 2 0.0
class ABO 3 0.1
CORRECT
class Tester end
class Tester
class Tester start
class Tester person Ava
class Tester person Ben
class Tester person Camila parents Ava Ben
class Tester person David parents Ava Ben
class Tester observe Camila bloodtype AB
class Tester infer David bloodtype
class ABO Variable David blood type: [O, A, B, AB]
class ABO 0 0.09
class ABO 1 0.33749999999999997
class ABO 2 0.2075
class ABO 3 0.365
CORRECT
class Tester end
class Tester
class Tester
class Tester start
class Tester person Ava
class Tester person Ben
class Tester person Camila parents Ava Ben
class Tester person David parents Ava Ben
class Tester observe Camila bloodtype O
class Tester infer David bloodtype
class ABO Variable David blood type: [O, A, B, AB]
class ABO 0 0.64
class ABO 1 0.2625
class ABO 2 0.08249999999999999
class ABO 3 0.015
CORRECT
class Tester end
class Tester quit
```