Assignment 5

CSU520, Spring 2008
Due: Wednesday, April 2

Part I. Pencil-and-paper exercise

1. For the Bayes net in Figure 14.11a (p. 510), use either the method of enumeration or the method of variable elimination to compute exactly the conditional probability $P(Rain = true|Sprinkler = true, WetGrass = true)$. You must show all your steps to receive credit. (You will get a chance to check the correctness of your final result when you do Problem 3 below.)

Part II. Computer exercises

Look in the `programs/bayes-net` subdirectory of the course web site. Examine the files `test-net.lisp` and `burglar-alarm-net.lisp` there to see how to represent a Bayes net with discrete-valued variables for use with the Lisp program provided for you.

2. Follow these examples to create a corresponding Lisp representation of the Bayes net shown in Figure 14.11a (the same one you used in Problem 1 above). Turn in the Lisp file for this network.

3. Now load into Lisp the files `enumeration.lisp` and `bayes-net-utils.lisp`, together with the network you've created in Problem 3. Then use the function `query-by-enumeration` to compute the following three conditional probabilities:

- $P(Rain = true|Sprinkler = true, WetGrass = true)$
- $P(Rain = true|Sprinkler = false, WetGrass = true)$
- $P(Rain = true|WetGrass = true)$

Turn in dribble output showing the corresponding function calls and their results, and then indicate exactly what these 3 probabilities are. (The results will be 3 *distributions*, but you are to explicitly provide 3 *numbers*.) Do the relative values of these 3 probabilities make sense intuitively? Explain briefly.

4. Write a program to compute $P(Rain = true|Sprinkler = true, WetGrass = true)$ in this same Bayes net using stochastic simulation with likelihood weighting. Your program should accept as an input parameter the number of samples to generate. Perform some informal experiments with different numbers of samples and collect output from these experiments. Comment on any variability you observe in these results and the extent to which these results differ from that obtained from exact calculation (in Problem 3 and Problem 1). Turn in the output file along with your source code for this program and your comments on the results.

Extra Credit Version of Problem 4. Note that in the above program you are free to hardcode in the topology of the network and all the conditional probability tables as given in Figure 14.11a. For extra credit, you may instead write a much more general Lisp program that accepts as input any Bayes net encoded in the generic manner used as input to `query-by-enumeration.lisp`, like those defined in `test-net.lisp` and `burglar-alarm-net.lisp`. It should also accept as input a specification of the query variable X and the vector of evidence variables and their values $(\mathbf{E}, \mathbf{e})$, together with the number of random samples to

generate. Given all this, it should then perform stochastic simulation with likelihood weighting to compute the conditional distribution $\mathbf{P}(X|\mathbf{E} = \mathbf{e})$ approximately. For simplicity, your program may assume that all variables take on only the values `true` and `false`. (The main way this simplifies the program is that it is a little more involved to generate discrete random values from a set of $n > 2$ possible values.) Then give your program as input the Bayes net representation you created for Problem 2, and have it compute approximate values of $P(Rain = true|Sprinkler = true, WetGrass = true)$, performing the same experimentation described and recording your observations, just as described above.

Suggestion for writing this more generic program: Have your code call functions from `bayes-net-utils.lisp` wherever possible and simply load this file with your file to run your program. Particular functions from that file that should be useful as helpers are `get-values`, `topological-sort-vars`, `normalize`, `prob-given-parents`, and `get-val`. You may also find it helpful to examine `enumerate.lisp` closely and borrow ideas from that code.