EE 583 Probabilistic Graphical Models

Homework 3

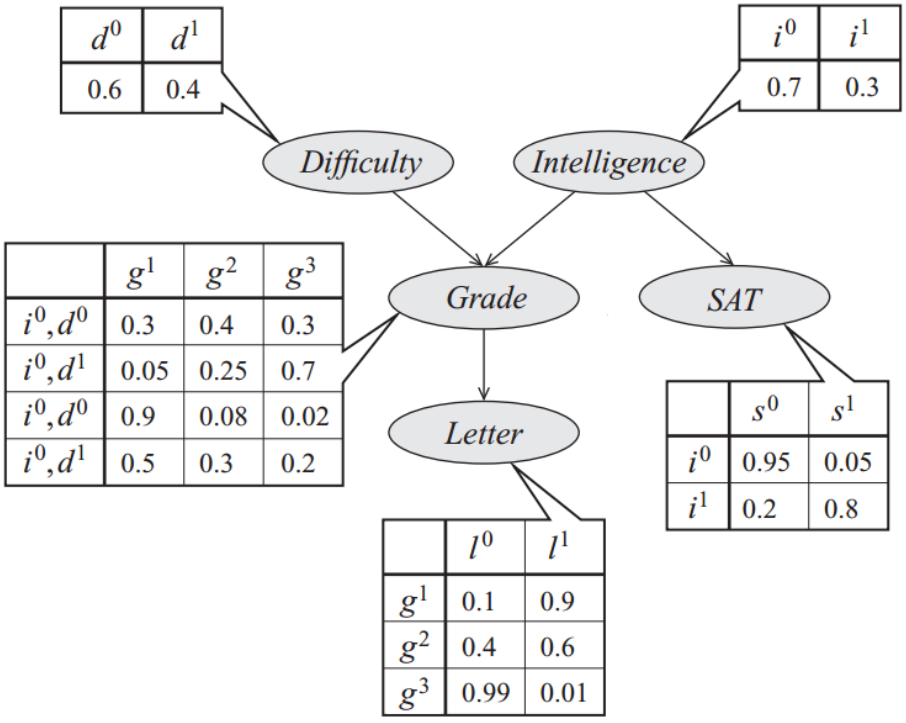
Due on Monday, April 25, 2016

1. **[40 pts]** Implement a MATLAB program `loopyBP.m` that performs the sum-product algorithm of loopy belief propagation on factor graphs. The program should be able to take a Bayesian network on discrete variables as input, with an appropriate data structure of your choice (edges + CPT tables or edges + factors) that encodes the underlying directed acyclic graph, the random variables and their conditional probability distributions given their parents, and it should return the converged result for marginal distributions of the involved variables. Specifically, given a Bayesian network of N discrete variables $\boldsymbol{X} = (X_n, n = 1, \dots, N)$ indexed by numbers, a suggested data structure for the input may consist of the following arrays
   - `Edges`: An $M \times 2$ array listing the directed edges of the Bayesian network, where M is the number of edges. In each row of `Edges`, the first entry carries the index of the source variable of the corresponding edge, and the second entry carries the index of the target variable. Note that using `Edges` array, the parent indices of each variable can also be readily extracted. Example: For $X_1 \to X_2 \leftarrow X_3$, we can specify `Edges = [1 2;3 2]`
   - `Factors`: An $N \times 1$ cell array for conditional probabilities of the Bayesian network. For each variable X_n, `Factors{n}` carries the conditional probability table (CPT) for X_n given its parents in the graph. In particular, `Factors{n}` will be a $|\mathrm{Pa}(n)| + 1$ dimensional array, where array dimensions are arranged according to some predefined order of node indices of the involved variables, such that how to read the CPT will be well defined. One such convention would be listing the different values of the argument variable X_n along the first dimension of `Factors{n}`, while reserving the other dimensions for $\boldsymbol{X}_{\mathrm{Pa}(n)}$ according to some fixed topological order (e.g. increasing node indices) in $\mathrm{Pa}(n)$. Following the same example $X_1 \to X_2 \leftarrow X_3$ above, assuming that $\{X_1, X_2, X_3\}$ are all binary, `Factors{2}` will be a $2 \times 2 \times 2$ array, with `Factors{2}(1,2,1)` carrying the conditional probability $P(X_2 = 0 | X_1 = 1, X_3 = 1)$. Note that, the correspondence between the CPT dimensions of `Factors{n}` and parent indices $\mathrm{Pa}(n) \subset \{1, \dots, N\}$ can be readily established by using the parent indices for each variable.

   `loopyBP.m` should run the sum-product algorithm of loopy belief propagation until convergence, and return the marginal distributions $P(x_n)$ of individual variables as an $N \times K$ array, where $K = |Val(X_n)|$, $n = 1, \dots, N$, is the number of possible values per variable. Use a synchronous message update schedule, in which all factor-to-variable messages are updated given the current variable-to-factor messages, and then all variable-to-factor messages are updated given the current factor-to-variable messages. Initialize the algorithm by setting the variable-to-factor messages equal to 1 for all states. Be careful to normalize messages at each step to avoid numerical underflow (i.e. for the message from factor f_m to variable X_n (respectively, from variable X_n to factor f_m), $\sum_{x_n} \mu_{m \to n}(x_n) = 1$ (respectively, $\sum_{x_n} \mu_{n \to m}(x_n) = 1$) should be satisfied.) Note that factors are nothing but the CPT tables listed in the cell array `Factors`. Convergence should be met whenever the total absolute change in messages falls below some threshold.

2. **[20 pts]** Implement a second MATLAB program `margBruteForce.m` that takes the same input structure as `loopyBP.m`, and again evaluates for each $n = 1, \ldots, N$, the marginal $P(x_n)$, but now in a brute force way, by element-wise multiplying the CPTs in `Factors` and summing out variables along dimensions that do not correspond to X_n.

3. **[20 pts]** Run your `loopyBP.m` and `margBruteForce.m` on the following Bayesian network with corresponding CPTs attached to each node. Note that numbers in superscripts indicate the values the variable can take on. For `loopyBP.m`, monitor the convergence of messages by plotting the sum of absolute message updates against the iteration count. Report the converged marginal distributions and show that they are the same for both `loopyBP.m` and `margBruteForce.m`.



d^0	d^1
0.6	0.4

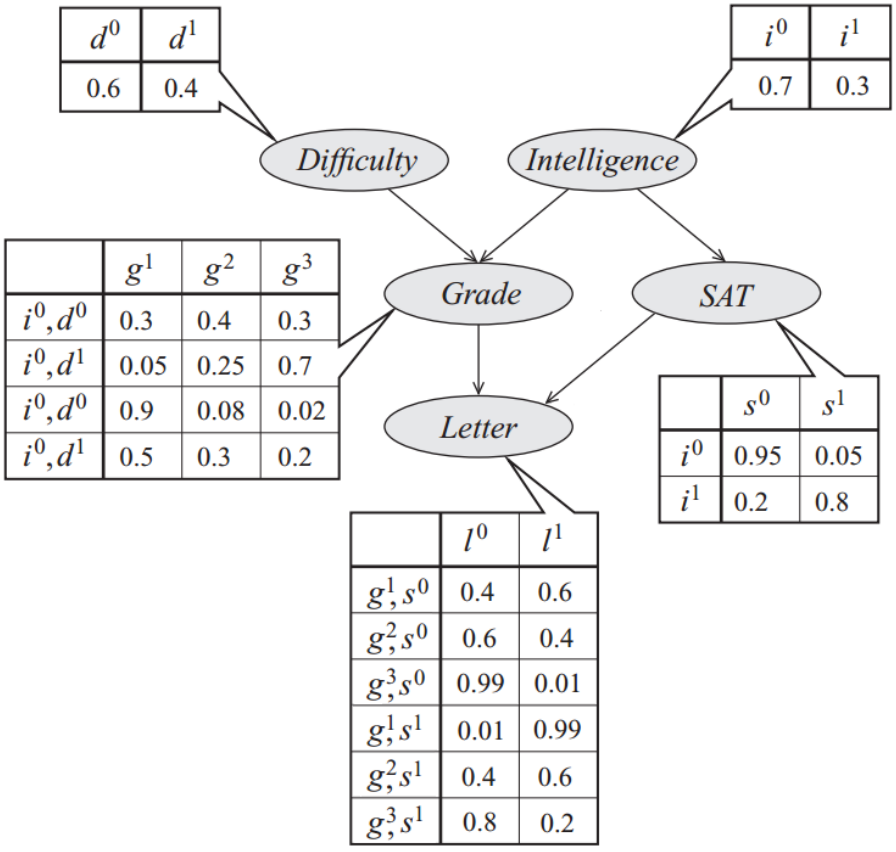
i^0	i^1
0.7	0.3

	g^1	g^2	g^3
i^0, d^0	0.3	0.4	0.3
i^0, d^1	0.05	0.25	0.7
i^0, d^0	0.9	0.08	0.02
i^0, d^1	0.5	0.3	0.2

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6
g^3	0.99	0.01

4. **[20 pts]** Repeat Problem 3 with the following Bayesian network, where now there is a loop due to the additional edge from "SAT" to "Letter". Again check if the found marginal distributions are the same for both `loopyBP.m` and `margBruteForce.m`.



d^0	d^1
0.6	0.4

i^0	i^1
0.7	0.3

	g^1	g^2	g^3
i^0, d^0	0.3	0.4	0.3
i^0, d^1	0.05	0.25	0.7
i^0, d^0	0.9	0.08	0.02
i^0, d^1	0.5	0.3	0.2

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

	l^0	l^1
g^1, s^0	0.4	0.6
g^2, s^0	0.6	0.4
g^3, s^0	0.99	0.01
g^1, s^1	0.01	0.99
g^2, s^1	0.4	0.6
g^3, s^1	0.8	0.2

In your homework submission, please provide a brief report including your pasted program code, output results and corresponding discussion.