4. Satish Bagavetyev

a) Belief propagation was implemented using C++ and VMFL library. I used VMFL primarily to access CPTs. I used additional 2d-arrays for storing messages. One (called Values) for storing messages generated to factors, and another (called fromfacts) to other direction.
I use two blocks of code for two stages of algorithm, and loop between them until convergence.
I also created a structure corresponding to possible possible probability assignments for simplicity.
Normalization of variable probabilities happens before assesment of convergence.

b) Cloudy: (0.5, 0.5)
Sprinkler: (0.3, 0.7)
Rain: (0.5, 0.5)
WetGrass: (0.602, 0.398)

True Marg. prob:

C	S	R	W
0	0	0	0.5 · 0.5 · 0.8 · 0.01
0	0	1	0.5 · 0.5 · 0.2 · 0.90
0	1	0	0.5 · 0.5 · 0.8 · 0.90
0	1	1	0.5 · 0.5 · 0.2 · 0.99
1	0	0	0.5 · 0.9 · 0.2 · 0.01
1	0	1	0.5 · 0.9 · 0.8 · 0.90
1	1	0	0.5 · 0.1 · 0.2 · 0.90
1	1	1	0.5 · 0.1 · 0.8 · 0.99

$0.002 + 0.045 + 0.18 + 0.0495 + 0.324 + 0.009 + 0.0396 + 0.0009 = 0.65.$

$P(W) = (0.65, 0.35)$

The numbers are different due to Belief propagation being approximate algorithm on loopy networks.
Marginal probabilities are calculated according an incomplete knowledge about the global state, but belief, composed of neighbors "beliefs" bout their state.

c) Multiplying all incoming messages takes $O(kf)$ time. Then each outgoing message needs to be divided depending on where it goes, by the incoming message from that node. (Cache lookup takes const time if messages stored in indexable array).