Adarsh Srivastava

adarsh.srivastava@u.nus.edu

A0254358X

- factor_product: Here we have just multiplied the individual probabilities at the indexes calculated.
- factor_marginalize: We are marginalizing one variable at a time. For each variable, we construct a new resultant factor object `out`. For each row of out, we figure out which rows of the preceding factor should've been summed together to give the probability of this. The rows that should be summed together are the ones going over all values of the variable being marginalized keeping everything else constant.
- observe_evidence: For each factor in the list, we figure if there are any observed variables in its list. If yes, we figure their index in `var` and check for all values at that index not equal to the observed value in the assignments. We set all such assignments' value to `0`.
- factor_sum: Here we have just summed the individual probabilities at the indexes calculated.
- compute_joint_distribution: Joint distribution is simply the factors multiplied together. We repeatedly multiply the factors together using `factor_product`.
- compute_marginals_naive: We find the joint distribution, marginalize the variables other than `V` away and normalize the result.
- compute_marginals_bp: I have used the exact notation and function names as used in slides.
  - `collect`, `distribute` and `compute_marginal` are self explanatory.
  - For `send_message` we save the list of factors to multiply, and then multiply them together using `compute_joint_distribution`, finally storing the messages for later use in `messages`