Handout 5 Statistical Physics for Communication and Computer Science
Homework 5 March 24, 2011, INR 113 - 9:15-11:00

Problem 1 (Belief Propagation for $(3, 6)$ Ensemble and AWGN Channel)**.** In the first homework you have implemented a program which can generate random elements from a regular Gallager ensemble. We will now use this, together with the message-passing algorithm discussed in class, to simulate transmission over a BAWGN channel.

We will use elements from the $(3, 6)$-ensemble of length $n = 1024$. For every codeword we send we generate a new code. This way we get the so called *ensemble average*. As discussed in class last week, when transmitting with a binary linear code over a symmetric channel, we can in fact assume that the all-zero (in 0/1 notation) codeword was sent since the error probability is independent of the transmitted codeword. This simplifies our life since we do not need to implement an encoder. We assume that we send the codeword over a binary-input additive white Gaussian noise channel. More precisely, the input is ± 1 (with the usual mapping). The channel adds to each component of the codeword an independent Gaussian random variable with zero mean and variance σ^2. At the receiver implement the message-passing decoder discussed in class. It is typically easiest to do the computations with likelihoods. Since a random element from the $(3, 6)$ ensemble typically does not have a tree-like factor graph the scheduling of the messages is important. To be explicit, assume that we use a *parallel* schedule. This means, we start by sending all *initial* messages from variable nodes to check nodes. We then process these messages and send messages back from check nodes to all variable nodes. This is one *iteration*. For each codeword perform 100 iterations and then make the final decision for each bit.

Plot the negative logarithm (base 10) of the resulting bit error probability as a function of the capacity of the BAWGN channel with variance σ^2. This capacity does not have a closed form but can be computed by means of the numerical integral

$$C(\sigma^2) = \int_{-1}^{1} \frac{\sigma}{\sqrt{2\pi}(1-y^2)} e^{-\frac{(1-\sigma^2 \tanh^{-1}(y))^2}{2\sigma^2}} \log_2(1+y) dy.$$

If the code and the decoder where optimal and the length of the code were infinite, where should you see the phase transition (rapid decay of error probability)?

Problem 2 (Min-Sum And Block MAP Decoding)**.** In class we discussed how to compute the marginal of a multivariate function $f(x_1, \ldots, x_n)$ efficiently, assuming that the function can be factorized into factors involving only few variables and that the corresponding factor graph is a tree. We accomplished this by formulating a message-passing algorithm. The messages are functions over the underlying alphabet. Functions are passed on edges. The algorithm starts at the leaf nodes and we discussed how messages are computed at variable and at function nodes.

Recall from the derivation that the main property we used was the *distributive law*. Consider now the following generalization. Consider the so-called *commutative semiring* of extended real numbers (including ∞) with the two operations min and $+$ (instead of the usual operations $+$ and $*$).

(i) Show that both operations are commutative.

(ii) Show that the identity element under min is ∞ and that the identity element under $+$ is 0.

(iii) Show that the distributive law holds.

(iv) If we formally exchange in our original marginalization $+$ with min and $*$ with $+$, what corresponds to the marginalization of a function?

(v) What are the message passing rules and what is the initialization?

(vi) Show that in case the factor graph associated to a binary linear codes is a tree, we can solve the block-MAP decoding problem by using the above message-passing algorithm based on the min-sum algebra.

More generally, there is a message passing algorithm for any commutative semi-ring.