CT111: Introduction to Communication Systems

End-of-Semester Project

In the end-of-semester project, you will develop Message Passing Hard Decision and Soft Decision Channel Decoder for the LDPC Codes for both the BEC and the BSC.

Project Groups and Evaluations

▷ For the final project, you are required to work in the group that you have been assigned with. Consult your mentor TAs for the group assignments. There will be ~ 10 students per group.

▷ Each TA will supervise and evaluate a total of three groups.

▷ At the end of the semester, e.g., *either* before or *after* your final exams are over, you will make the final presentation of your project work.

▷ At the time of evaluation, each group should come prepared with a presentation (e.g., in Powerpoint) of your work, and should have a working code. You will be asked to demonstrate the operation of the program. The presentation format will be conveyed to you.

▷ You are strongly encouraged to use C++ or Python for this project. Matlab is okay also but please take permission from your Supervisor TA if you develop your decoder in Matlab.

▷ Although you will be working as a part of a group, the project grades will be individually awarded. It is in your best interests, both from the learning point of view as well as getting a good evaluation, that each one of you does not sit on the sidelines or be satisfied with some secondary tasks (preparing the presentation, understanding someone else's code/logic, etc.). Instead you should insist on engaging in the actual program development and working as if this is your own individual project. Do not hope or expect that someone else in your group will do a thorough study of the algorithm performance, instead be proactive and take up any such task yourself.

Honor Code

You will be required to submit the following pledge at the time of the interim and final evaluations of the project work. This should be the opening page of your project presentation slides.

- We declare that
  - → The work that we are presenting is our own work.

→ We have not copied the work (the code, the results, etc.) that someone else has done.

→ Concepts, understanding and insights we will be describing are our own.

→ We make this pledge truthfully. We know that violation of this solemn pledge can carry grave[1] consequences.

- Signed by: all the members of the project group.

Technical Deliverable

Following are the items that you (i.e., each project group) is expected to *deliver* at the conclusion of the project:

1. Models of (i) the binary erasure channel (BEC(p)) with the erasure probability of p, and (ii) the binary symmetric channel (BSC(p)) with bit flipping probability of p.

2. Performance (expressed[2] as (i) convergence of the algorithm and (ii) the probability of decoding success) of the LDPC coding scheme for different values of p for the three LDPC $\mathbf{H}$ matrices given to you for

   (a) Hard Decision Decoding for BEC(p),

   (b) Soft Decision Decoding for BEC(p),

   (c) Hard Decision Decoding for BSC(p), and

   (d) Soft Decision Decoding for BSC(p).

Your work will be evaluated on the basis of how well you are able to deliver the above tasks. You will be judged on the basis of whether (i) you fall short of meeting the above deliverable, (ii) you have met these deliverable, or (iii) you have exceeded the expectations and done an outstanding job. To ensure that your evaluations are in category (iii), you should perform detailed studies required for these deliverable, and do a deep analysis and thinking on *why* does your code behave in the manner that you have observed.

Technical Instructions

A total of 3 LDPC codes are given to you.

▹ Two parity check $\mathbf{H}$ matrices of two different LDPC codes are placed on the Google folder. The third LDPC code is stated on the last page of `Channel Coding Part 5` in the Google Classroom.

▹ *Do not* implement an encoder (i.e., $\mathbf{G}$ matrix) for any of these three LDPC codes. Instead assume that an all-zero codeword has been transmitted.

Following are several technical notes about the project.

1. Build a model for the binary erasure channel (BEC(p)) and the binary symmetric channel (BSC(p)) with the erasure probability of p.

---

[1] E.g., the entire project group will be awarded 0 marks even if there is a slightest hint of plagiarism. It will be an extremely *foolish* mistake to engage in the plagiarism, or to accept someone in your group who is attempting to plagiarize.

[2] This is explained later.

2. Build message passing hard decision and soft decision decoders for the BEC and for the BSC.

   ▷ Your code should automatically convert the pattern of ones and zeros in this parity check matrix **H** into a Tanner Graph model. The connections between the bit nodes and the check nodes of this bipartite graph are based on the locations of ones in **H**. Make your code generic enough so that it works with any matrix **H** given as the input.

   ▷ Your model of Tanner Graph should perform the message passing from bit nodes to variable nodes and vice versa. This is described in `Channel Coding Part 5` in the Google Classroom.

      → Recall that in the bipartite Tanner Graph, each variable node is connected to a total of d_v check nodes, and each check node is connected to a total of d_c variable nodes.

      → Each check node is an SPC code. It enforces the even parity among all the bit nodes that are connected to it. Thus the message passed from the CN to the VN is based on the SPC decoding.

      → Similarly, each bit node is essentially a repetition code. It requires that all the messages are the same (either 1 or 0). Thus, the message passed from the VN to the CN is based on the majority vote decoding of the repetition codes.

      → Refer to `Channel Coding Part 5` in the Google Classroom to understand the message computations at the VNs and the CNs for the BSC and the BEC for the soft decision and hard decision decoding.

3. Demonstration that your software is working correctly:

   (a) Spot checks: here you will (may be manually) erase (for the BEC) or flip (for the BSC) some bits out of the transmitted codeword and show that the decoder is able to recover the original message bits.

   (b) Monte Carlo Simulations: For each value of p for the BEC and the BSC, and for each of the three LDPC codes), perform N_{sim} Monte Carlo experiements. Keep N_{sim} to be some large number such as 10000. Let the simulation experiemental index be denoted as n, where $n = 1, 2, \ldots, N_{sim}$.

      i. *Probability of Successful Decoding (one of the deliverables).* You are asked to evaluate the performance of your algorithms in terms of the probability of decoding success. To evaluate this probability, you will need to implement the following logic:

         ▷ At n^{th} simulation experiment, set a flag F_n to 1 if the decoding is successful. Your code should set $F_n = 1$ at n^{th} trial if the decoder is able to unerase all the erasures introduced by the BEC, or it is able to unflip all the bits that have been flipped by the BSC. If the erasures (for BEC) or the bit flips (for the BSC) remain (after a maximum number of iterations, say, 100), F_n is to be set to 0.

         ▷ Probability of Successful Decoding is calculated after completing N_{sim} simulation trials as $\dfrac{1}{N_{sim}} \displaystyle\sum_{n=1}^{N_{sim}} F_n$.

      ii. *Algorithm Convergence (one of the deliverables).* To show that your message passing decoder is working correctly, store the number of erasures for the BEC (and the number of bit errors for the BSC) as a function of the iteration index (one iteration of the message passing is said to have been completed when the messages complete one cycle, i.e., go from from bit nodes to check nodes, and then check nodes to bit nodes). Plot this number, after averaging it over all N_{sim} experiments, as a function of iteration index and see that this plot is converging to zero as the number of iterations increase.