An introduction to coding theory

Adrish Banerjee

Department of Electrical Engineering
Indian Institute of Technology Kanpur
Kanpur, Uttar Pradesh
India

Feb. 27, 2017

Lecture #14A: Decoding of low density parity check codes-I

Outline of the talk

- Decoding on BSC: Bit Flipping Algorithm

Outline of the talk

- Decoding on BSC: Bit Flipping Algorithm
    - Example 1: One transmission error case.

Outline of the talk

- Decoding on BSC: Bit Flipping Algorithm
  - Example 1: One transmission error case.
  - Example 2: Two transmission errors case.

Low-density parity check codes

1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0
0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1
1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0
0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1

- Example of a low density code matrix; n=20, j=3, k=4

Definitions

- The set of bits contained in a parity-check equation constitutes a **parity check set**.

Definitions

- The set of bits contained in a parity-check equation constitutes a **parity check set**.
- **Parity check set tree** is a representation of parity check set in a tree structure.

Definitions

- The set of bits contained in a parity-check equation constitutes a **parity check set**.
- **Parity check set tree** is a representation of parity check set in a tree structure.
  - An arbitrary bit d is represented by the node of the base of the tree.

Definitions

- The set of bits contained in a parity-check equation constitutes a **parity check set**.
- **Parity check set tree** is a representation of parity check set in a tree structure.
  - An arbitrary bit d is represented by the node of the base of the tree.
  - Each line rising from this node represents one of the parity-check sets containing d.

Definitions

- The set of bits contained in a parity-check equation constitutes a **parity check set**.
- **Parity check set tree** is a representation of parity check set in a tree structure.
  - An arbitrary bit d is represented by the node of the base of the tree.
  - Each line rising from this node represents one of the parity-check sets containing d.
  - The other nodes bits in these parity-check sets are represented by the nodes on the first tier of the tree.

Definitions

- The set of bits contained in a parity-check equation constitutes a **parity check set**.
- **Parity check set tree** is a representation of parity check set in a tree structure.
  - An arbitrary bit d is represented by the node of the base of the tree.
  - Each line rising from this node represents one of the parity-check sets containing d.
  - The other nodes bits in these parity-check sets are represented by the nodes on the first tier of the tree.
  - The lines rising from tier 1 to tier 2 of the tree represent the other parity-check sets containing the bits on tier 1.

Definitions

- The set of bits contained in a parity-check equation constitutes a **parity check set**.
- **Parity check set tree** is a representation of parity check set in a tree structure.
  - An arbitrary bit d is represented by the node of the base of the tree.
  - Each line rising from this node represents one of the parity-check sets containing d.
  - The other nodes bits in these parity-check sets are represented by the nodes on the first tier of the tree.
  - The lines rising from tier 1 to tier 2 of the tree represent the other parity-check sets containing the bits on tier 1.
  - The nodes on tier 2 represent the other bits in those parity-check sets.

Low-density parity check codes

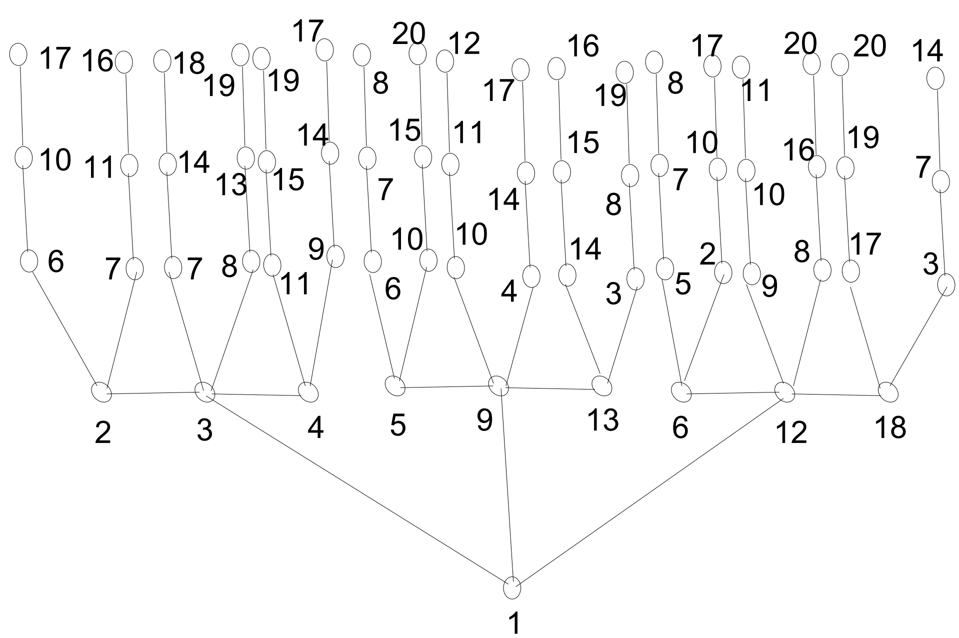
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0
0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1
1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0
0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0
0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1

- Example of a low density code matrix; n=20, j=3, k=4

Parity-check set

#	parity-check set
1	{1,2,3,4}
2	{5,6,7,8}
3	{9,10,11,12}
4	{13,14,15,16}
5	{17,18,19,20}
6	{1,5,9,13}
7	{2,6,10,17}
8	{3,7,14,18}
9	{4,11,15,19}
10	{8,12,16,20}
11	{1,6,12,18}
12	{2,7,11,16}
13	{3,8,13,19}
14	{4,9,14,17}
15	{5,10,15,20}

Parity-check set tree



Decoding on BSC: Bit-Flipping Algorithm

Example 1: Single transmission error case

Transmitted bits= {0,0,0,0,0,0,1,1,1,1,1,1,1,0,1,0,1,1,0,0}
Received bits={1,0,0,0,0,0,1,1,1,1,1,1,1,0,1,0,1,1,0,0}

- The first bit is received in error.

Decoding on BSC: Bit-Flipping Algorithm

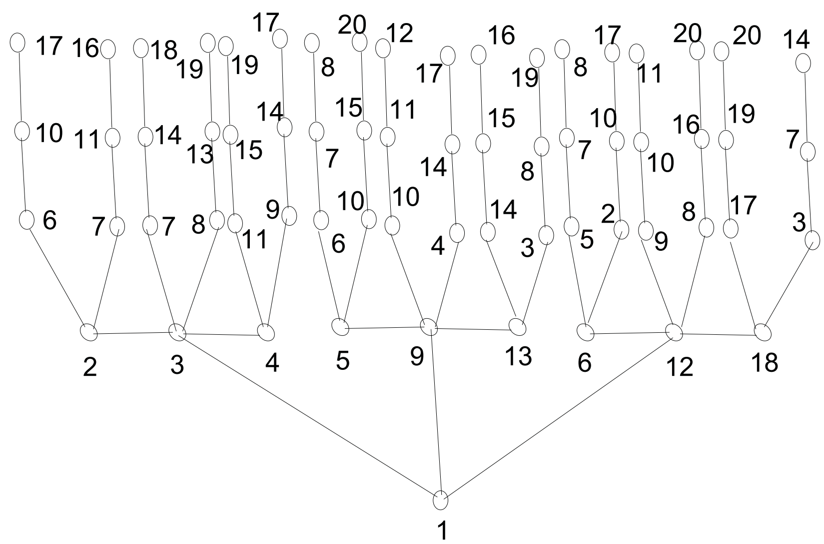
Example 1: Single transmission error case

Transmitted bits= {0,0,0,0,0,0,1,1,1,1,1,1,1,0,1,0,1,1,0,0}
Received bits={1,0,0,0,0,0,1,1,1,1,1,1,1,0,1,0,1,1,0,0}

- The first bit is received in error.
- Decoder will try to correct the error.

Decoding on BSC: Bit-Flipping Algorithm

- **Step 1:** Represent the code using parity check set tree.



Parity-check set

#	parity-check set
1	{1,2,3,4}
2	{5,6,7,8}
3	{9,10,11,12}
4	{13,14,15,16}
5	{17,18,19,20}
6	{1,5,9,13}
7	{2,6,10,17}
8	{3,7,14,18}
9	{4,11,15,19}
10	{8,12,16,20}
11	{1,6,12,18}
12	{2,7,11,16}
13	{3,8,13,19}
14	{4,9,14,17}
15	{5,10,15,20}

Decoding on BSC: Bit-Flipping Algorithm

- **Step 2:** First Iteration: Check the parity-check sets containing bit # 1 to see if they are satisfied.

Decoding on BSC: Bit-Flipping Algorithm

- **Step 2:** First Iteration: Check the parity-check sets containing bit # 1 to see if they are satisfied.
- All the three parity check set #1, 6, 11 are violated.

Decoding on BSC: Bit-Flipping Algorithm

- **Step 2:** First Iteration: Check the parity-check sets containing bit # 1 to see if they are satisfied.
- All the three parity check set #1, 6, 11 are violated.
- Since all three of the parity check-set containing bit # 1 are violated, there is a strong possibility that bit #1 is in error.

Decoding on BSC: Bit-Flipping Algorithm

- **Step 2:** First Iteration: Check the parity-check sets containing bit # 1 to see if they are satisfied.
- All the three parity check set #1, 6, 11 are violated.
- Since all three of the parity check-set containing bit # 1 are violated, there is a strong possibility that bit #1 is in error.
- Flip the first received bit # 1 from 1 to 0 and recompute the syndrome(check whether the parity constraints are satisfied).

Decoding on BSC: Bit-Flipping Algorithm

- **Step 2:** First Iteration: Check the parity-check sets containing bit # 1 to see if they are satisfied.
- All the three parity check set #1, 6, 11 are violated.
- Since all three of the parity check-set containing bit # 1 are violated, there is a strong possibility that bit #1 is in error.
- Flip the first received bit # 1 from 1 to 0 and recompute the syndrome(check whether the parity constraints are satisfied).
- All the parity equations containing bit 1 are satisfied, hence the first bit is decoded as 0.

Decoding on BSC: Bit-Flipping Algorithm

Example 2: Two transmission errors case

Transmitted bits= {0,0,0,0,0,0,1,1,1,1,1,1,1,0,1,0,1,1,0,0}
Received bits={1,1,0,0,0,0,1,1,1,1,1,1,1,0,1,0,1,1,0,0}

- First two bits are received in error.

Decoding on BSC: Bit-Flipping Algorithm

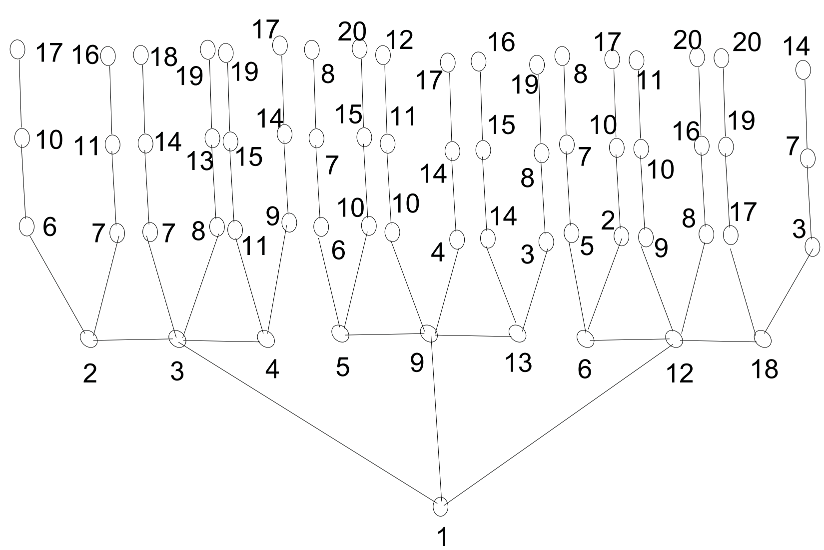
Example 2: Two transmission errors case

Transmitted bits= {0,0,0,0,0,0,1,1,1,1,1,1,1,0,1,0,1,1,0,0}
Received bits={1,1,0,0,0,0,1,1,1,1,1,1,1,0,1,0,1,1,0,0}

- First two bits are received in error.
- Decoder will try to correct bit #1 and 2.

Decoding on BSC: Bit-Flipping Algorithm

- **Step 1:** Construct parity check-set tree.



Parity-check set

#	parity-check set
1	{1,2,3,4}
2	{5,6,7,8}
3	{9,10,11,12}
4	{13,14,15,16}
5	{17,18,19,20}
6	{1,5,9,13}
7	{2,6,10,17}
8	{3,7,14,18}
9	{4,11,15,19}
10	{8,12,16,20}
11	{1,6,12,18}
12	{2,7,11,16}
13	{3,8,13,19}
14	{4,9,14,17}
15	{5,10,15,20}

Decoding on BSC: Bit-Flipping Algorithm

- **Step 2:** First Iteration: Check the parity-check set containing bit #1 to see if they are satisfied.

Decoding on BSC: Bit-Flipping Algorithm

- **Step 2:** First Iteration: Check the parity-check set containing bit #1 to see if they are satisfied.
- Two of the three parity check set #6 and 11 are violated. Parity check set #1 is satisfied.

Decoding on BSC: Bit-Flipping Algorithm

- **Step 2:** First Iteration: Check the parity-check set containing bit #1 to see if they are satisfied.
- Two of the three parity check set #6 and 11 are violated. Parity check set #1 is satisfied.
- Since majority of the parity-check set containing bit #1 are violated, there is a strong possibility that first bit is in error.

Adrish Banerjee Department of Electrical Engineering Indian Institute of Technology Kanpur Kanpur, Uttar Pradesh India
An introduction to coding theory

Decoding on BSC: Bit-Flipping Algorithm

- **Step 2:** First Iteration: Check the parity-check set containing bit #1 to see if they are satisfied.
- Two of the three parity check set #6 and 11 are violated. Parity check set #1 is satisfied.
- Since majority of the parity-check set containing bit #1 are violated, there is a strong possibility that first bit is in error.
- Flip the first received bit from 1 to 0 and recompute the syndrome(check whether the parity constraints are satisfied.

Decoding on BSC: Bit-Flipping Algorithm

- **Step 2:** First Iteration: Check the parity-check set containing bit #1 to see if they are satisfied.
- Two of the three parity check set #6 and 11 are violated. Parity check set #1 is satisfied.
- Since majority of the parity-check set containing bit #1 are violated, there is a strong possibility that first bit is in error.
- Flip the first received bit from 1 to 0 and recompute the syndrome(check whether the parity constraints are satisfied.
- Parity check-set #6 and 11 are satisfied, but #1 failed.

Decoding on BSC: Bit-Flipping Algorithm

- **Step 2:** First Iteration: Check the parity-check set containing bit #1 to see if they are satisfied.
- Two of the three parity check set #6 and 11 are violated. Parity check set #1 is satisfied.
- Since majority of the parity-check set containing bit #1 are violated, there is a strong possibility that first bit is in error.
- Flip the first received bit from 1 to 0 and recompute the syndrome(check whether the parity constraints are satisfied.
- Parity check-set #6 and 11 are satisfied, but #1 failed.
- Hence the first iteration is not sufficient to correct the errors.

Decoding on BSC: Bit-Flipping Algorithm

- **Step 3:** Second Iteration: Check the parity-check set containing bits in the first tier of the parity check-set tree to see if they are satisfied.

Decoding on BSC: Bit-Flipping Algorithm

- **Step 3:** Second Iteration: Check the parity-check set containing bits in the first tier of the parity check-set tree to see if they are satisfied.
- Parity check set containing bits 3(#8 and 13), 4(#9 and 14), 5(# 2 and 15), 9(# 3 and 14), 13(# 4 and 13), 12(# 3 and 10), 18(# 5 and 8) are satisfied. One of the parity check set containing bit 6(# 2) is also satisfied.

Decoding on BSC: Bit-Flipping Algorithm

- **Step 3:** Second Iteration: Check the parity-check set containing bits in the first tier of the parity check-set tree to see if they are satisfied.
- Parity check set containing bits 3(#8 and 13), 4(#9 and 14), 5(# 2 and 15), 9(# 3 and 14), 13(# 4 and 13), 12(# 3 and 10), 18(# 5 and 8) are satisfied. One of the parity check set containing bit 6(# 2) is also satisfied.
- Both the parity check set (# 7 and 12) containing bit 2 and one of the parity check set containing (# 7) containing bit 6 are violated.

Decoding on BSC: Bit-Flipping Algorithm

- **Step 3:** Second Iteration: Check the parity-check set containing bits in the first tier of the parity check-set tree to see if they are satisfied.
- Parity check set containing bits 3(#8 and 13), 4(#9 and 14), 5(# 2 and 15), 9(# 3 and 14), 13(# 4 and 13), 12(# 3 and 10), 18(# 5 and 8) are satisfied. One of the parity check set containing bit 6(# 2) is also satisfied.
- Both the parity check set (# 7 and 12) containing bit 2 and one of the parity check set containing (# 7) containing bit 6 are violated.
- Bit 2 is common in all of these three parity check set as well as the parity check #1 which was violated after first iteration.

Decoding on BSC: Bit-Flipping Algorithm

- **Step 3:** Second Iteration: Check the parity-check set containing bits in the first tier of the parity check-set tree to see if they are satisfied.
- Parity check set containing bits 3(#8 and 13), 4(#9 and 14), 5(# 2 and 15), 9(# 3 and 14), 13(# 4 and 13), 12(# 3 and 10), 18(# 5 and 8) are satisfied. One of the parity check set containing bit 6(# 2) is also satisfied.
- Both the parity check set (# 7 and 12) containing bit 2 and one of the parity check set containing (# 7) containing bit 6 are violated.
- Bit 2 is common in all of these three parity check set as well as the parity check #1 which was violated after first iteration.
- Hence, there is a strong possibility that second bit is in error.

Decoding on BSC: Bit-Flipping Algorithm

- Flip the second received bit #2 from 1 to 0 and recompute the syndrome(check whether the parity constraints are satisfied.

Decoding on BSC: Bit-Flipping Algorithm

- Flip the second received bit #2 from 1 to 0 and recompute the syndrome(check whether the parity constraints are satisfied.
- All the parity check sets at first tier are satisfied.

Decoding on BSC: Bit-Flipping Algorithm

- Flip the second received bit #2 from 1 to 0 and recompute the syndrome(check whether the parity constraints are satisfied.
- All the parity check sets at first tier are satisfied.
- Now we check the parity-check sets at zero tier(containing first bit) and they are also satisfied.

Decoding on BSC: Bit-Flipping Algorithm

- Flip the second received bit #2 from 1 to 0 and recompute the syndrome(check whether the parity constraints are satisfied.
- All the parity check sets at first tier are satisfied.
- Now we check the parity-check sets at zero tier(containing first bit) and they are also satisfied.
- Hence the first and second bits are decoded as 0's.

Decoding on BSC: Bit Flipping Algorithm

- The decoder computes all the parity checks and then changes any bit that is contained in more than some fixed number of unsatisfied parity-check equations.

Decoding on BSC: Bit Flipping Algorithm

- The decoder computes all the parity checks and then changes any bit that is contained in more than some fixed number of unsatisfied parity-check equations.
- Using these new values, the parity checks are recomputed, and the process is repeated until the parity checks are all satisfied.

Decoding on BSC: Bit Flipping Algorithm

- The decoder computes all the parity checks and then changes any bit that is contained in more than some fixed number of unsatisfied parity-check equations.
- Using these new values, the parity checks are recomputed, and the process is repeated until the parity checks are all satisfied.
- If the parity check sets are small, this decoding procedure is reasonable, since most of the parity-check sets will contain either one transmission error or no transmission errors.

Decoding on BSC: Bit Flipping Algorithm

- The decoder computes all the parity checks and then changes any bit that is contained in more than some fixed number of unsatisfied parity-check equations.
- Using these new values, the parity checks are recomputed, and the process is repeated until the parity checks are all satisfied.
- If the parity check sets are small, this decoding procedure is reasonable, since most of the parity-check sets will contain either one transmission error or no transmission errors.
- Thus when most of the parity-check equation checking on a digit are unsatisfied, there is a strong indication that that digit is in error.