Corrections:

- Slide IDC8-14:

Signals	Code signals	Distance square to "all-zero" signals
01 00 00	010 001 010 000 000	$d_1^2 + d_0^2$
$\vdots$	$\vdots$	$\vdots$

  should be corrected as

Signals	Code signals	Distance square to "all-zero" signals
01 00 00	010 001 010 000 000	$2d_1^2 + d_0^2$
$\vdots$	$\vdots$	$\vdots$

- Slide IDC8-38 should be immediately after Slide IDC 8-32.

- Slide IDC 8-33 (or Slide IDC 8-34 after moving Slide IDC 8-38 ahead):

$$P(S_4|S_3) = \begin{cases} \vdots & \vdots & \vdots \\ P(b|c) & = & P(m_4 - 1) \\ \vdots & \vdots & \vdots \end{cases}$$

  should be

$$P(S_4|S_3) = \begin{cases} \vdots & \vdots & \vdots \\ P(b|c) & = & P(m_4 = 1) \\ \vdots & \vdots & \vdots \end{cases}$$

- Slide IDC 8-36 (or Slide IDC 8-37 after moving Slide IDC 8-38 ahead): Change 10-123 to 8-33. Change the first 10-126 to 8-32 & 8-33. Change the second 10-126 to 8-36.

- Slide IDC 8-39 (or Slide IDC 8-38 after moving the original Slide IDC 8-38 ahead): l_1 and l_2 should be $\tilde{l}_1$ and $\tilde{l}_2$.

- Slide IDC 8-52:

$$P_j^1 = \frac{p_j^1 \prod_{k \in \mathrm{Bit}(j)} Q_{k,j}^1}{p_j^1 \prod_{k \in \mathrm{Bit}(j)} Q_{k,j}^1 + p_j^1 \prod_{k \in \mathrm{Bit}(j)} Q_{k,j}^1}$$

  should be

$$P_j^1 = \frac{p_j^1 \prod_{k \in \mathrm{Bit}(j)} Q_{k,j}^1}{p_j^1 \prod_{k \in \mathrm{Bit}(j)} Q_{k,j}^1 + (1 - p_j^1) \prod_{k \in \mathrm{Bit}(j)} (1 - Q_{k,j}^1)}$$

Additional sample problems for the final exam

1. (Continue from Additional Sample Problem 2 for the Final Exam (Part I))

   (a) From the list of all codewords, show that the convolutional code so constructed is a linear code.

   (b) Is this convolutional code a cyclic code? Justify your answer.

   (c) Is this convolutional code a polynomial code? Justify your answer.

   (d) What is the minimum pairwise Hamming distance $d_{\text{H,min}}$ of this linear code?

   (e) What is the minimum pairwise Euclidean distance square $d^2_{\text{E,min}}$ of the equivalent antipodally coded signals, provided $\sqrt{E}(-1)^{\boldsymbol{c}}$ is transmitted?

Solution.

(a) We can choose three linearly independent codewords among the eight codewords below:

message $m_0m_1m_2$	codeword $c_0^{(1)}c_0^{(2)}c_1^{(1)}c_1^{(2)}c_2^{(1)}c_2^{(2)}c_3^{(1)}c_3^{(2)}c_4^{(1)}c_4^{(2)}$
000	00 00 00 00 00
001	00 00 11 10 11
010	00 11 10 11 00
011	00 11 01 01 11
100	11 10 11 00 00
101	11 10 00 10 11
110	11 01 01 11 00
111	11 01 10 01 11

and confirm that

$$\begin{bmatrix} 000 \\ 001 \\ 010 \\ 011 \\ 100 \\ 101 \\ 110 \\ 111 \end{bmatrix} \underbrace{\begin{bmatrix} 1110110000 \\ 0011101100 \\ 0000111011 \end{bmatrix}_{3\times 10}}_{\text{generator matrix}} = \begin{bmatrix} 0000000000 \\ 0000111011 \\ 0011101100 \\ 0011010111 \\ 1110110000 \\ 1110001011 \\ 1101011100 \\ 1101100111 \end{bmatrix}$$

holds and hence this convolutional code is a linear code.

(b) The answer is negative because, for example, not all circular shifts of codeword 0000111011 are codewords.

(c) If the answer were positive, then the code polynomial must satisfy

$$c(X) = (a_0 + a_1X + a_2X^2)\underbrace{(1 + g_1X + \cdots + g_6X^6 + X^7)}_{g(X)}$$

for some $g(X)$. By this definition, we know when $a_1 = a_2 = 0$, $c(X) = g(X)$. Apparently from the table in (a), when $a_0a_1a_2 = m_0m_1m_2 = 100$, $c(X) = 1 + X + X^2 + X^4 + X^5$ does not fulfill the requirement.

Note: Note that we require $g_0 = g_7 = 1$ for $g(X)$. Thus, reordering $a_0a_1a_2 = m_2m_1m_0 = 100$ or even setting $a_0a_1a_2 = m_1m_0m_2 = 100$ also cannot result in a legitimate $g(X)$.

(d) From (a), the minimum weight (i.e., the number of 1s) of non-zero codewords is 5.

(e) Continueing from (d), we derive

$$\begin{aligned} d^2_{\text{E,min}} &= \|\sqrt{E}(-1)^{0000000000}, \sqrt{E}(-1)^{0000111011}\|^2 \\ &= d_{\text{H,min}}\left(\sqrt{E} - (-\sqrt{E})\right)^2 \\ &= 4E\, d_{\text{H,min}} \\ &= 20E \end{aligned}$$

Note: From Slide IDC 7-91, we have an upper error bound for the solf-decision decoding as

$$e^{-d_{\text{H,free}}\frac{E}{N_0}} = e^{-\frac{d^2_{\text{E,free}}}{4N_0}}$$

where the equality follows from $d^2_{\text{E,min}} = 4E\, d_{\text{H,min}}$. This formula is exactly what we obtain in Slide IDC 8-10.

2. Continue from Problem 1. For a linear code, the optimal error performance can be obtained by assuming that the all-zero codeword is transmitted, i.e.,

$$P_{\text{e}} = \sum_{\boldsymbol{c}\in\mathcal{C}} \Pr(\boldsymbol{c}) \Pr(\hat{\boldsymbol{c}} \neq \boldsymbol{c}|\boldsymbol{c}) = \Pr(\hat{\boldsymbol{c}} \neq 0000000000|\boldsymbol{c} = 0000000000),$$

where $\hat{\boldsymbol{c}}$ is the optimal decoding decision when $\boldsymbol{c}$ is transmitted. Index the codewords, as well as the corresponding coded signals, as follows:

$\boldsymbol{c}_1$	00 00 00 00 00	$\boldsymbol{s}_1$	$+\sqrt{E},+\sqrt{E},+\sqrt{E},+\sqrt{E},+\sqrt{E},+\sqrt{E},+\sqrt{E},+\sqrt{E},+\sqrt{E},+\sqrt{E}$
$\boldsymbol{c}_2$	00 00 11 10 11	$\boldsymbol{s}_2$	$+\sqrt{E},+\sqrt{E},+\sqrt{E},+\sqrt{E},-\sqrt{E},-\sqrt{E},-\sqrt{E},+\sqrt{E},-\sqrt{E},-\sqrt{E}$
$\boldsymbol{c}_3$	00 11 10 11 00	$\boldsymbol{s}_3$	$+\sqrt{E},+\sqrt{E},-\sqrt{E},-\sqrt{E},-\sqrt{E},+\sqrt{E},-\sqrt{E},-\sqrt{E},+\sqrt{E},+\sqrt{E}$
$\boldsymbol{c}_4$	00 11 01 01 11	$\boldsymbol{s}_4$	$+\sqrt{E},+\sqrt{E},-\sqrt{E},-\sqrt{E},+\sqrt{E},-\sqrt{E},+\sqrt{E},-\sqrt{E},-\sqrt{E},-\sqrt{E}$
$\boldsymbol{c}_5$	11 10 11 00 00	$\boldsymbol{s}_5$	$-\sqrt{E},-\sqrt{E},-\sqrt{E},+\sqrt{E},-\sqrt{E},-\sqrt{E},+\sqrt{E},+\sqrt{E},+\sqrt{E},+\sqrt{E}$
$\boldsymbol{c}_6$	11 10 00 10 11	$\boldsymbol{s}_6$	$-\sqrt{E},-\sqrt{E},-\sqrt{E},+\sqrt{E},+\sqrt{E},+\sqrt{E},-\sqrt{E},+\sqrt{E},-\sqrt{E},-\sqrt{E}$
$\boldsymbol{c}_7$	11 01 01 11 00	$\boldsymbol{s}_7$	$-\sqrt{E},-\sqrt{E},+\sqrt{E},-\sqrt{E},+\sqrt{E},-\sqrt{E},-\sqrt{E},-\sqrt{E},+\sqrt{E},+\sqrt{E}$
$\boldsymbol{c}_8$	11 01 10 01 11	$\boldsymbol{s}_8$	$-\sqrt{E},-\sqrt{E},+\sqrt{E},-\sqrt{E},-\sqrt{E},+\sqrt{E},+\sqrt{E},-\sqrt{E},-\sqrt{E},-\sqrt{E}$

From the union bound in Slide IDC 2-59, we have

$$\begin{aligned} \Pr(\hat{\boldsymbol{s}} = \boldsymbol{s}_2|\boldsymbol{s} = \boldsymbol{s}_1) &\leq \Pr(\hat{\boldsymbol{s}} \neq \boldsymbol{s}_1|\boldsymbol{s} = \boldsymbol{s}_1) \\ &= \Pr(\hat{\boldsymbol{s}} = \boldsymbol{s}_2 \text{ or } \boldsymbol{s}_3 \text{ or } \cdots \text{ or } \boldsymbol{s}_8|\boldsymbol{s} = \boldsymbol{s}_1) \\ &\leq \sum_{i=2}^{8} \Pr(\hat{\boldsymbol{s}} = \boldsymbol{s}_i|\boldsymbol{s} = \boldsymbol{s}_1) \end{aligned}$$

Represent the lower and upper bound in terms of the cdf of the standard normal $\Phi(\cdot)$.

Hint: From Slides IDC 1-60 (or Slide IDC 7-91 and Problem 1(e)),

$$\Pr(\hat{\boldsymbol{s}} = \boldsymbol{s}_i | \boldsymbol{s} = \boldsymbol{s}_1) = \Phi\left(-\sqrt{\frac{\|\boldsymbol{s}_i - \boldsymbol{s}_1\|^2}{2N_0}}\right).$$

Solution. The code contain four non-zero codewords of weight 5, two non-zero codewords of weight 6, and one non-zero codeword of weight 7. Thus,

$$\begin{aligned}
\Pr(\hat{\boldsymbol{s}} = \boldsymbol{s}_2 | \boldsymbol{s} = \boldsymbol{s}_1) = \Phi\left(-\sqrt{10\frac{E}{N_0}}\right) &\leq P_{\rm e} \\
&\leq \sum_{i=2}^{8} \Pr(\hat{\boldsymbol{s}} = \boldsymbol{s}_i | \boldsymbol{s} = \boldsymbol{s}_1) \\
&= 4\Phi\left(-\sqrt{10\frac{E}{N_0}}\right) + 2\Phi\left(-\sqrt{12\frac{E}{N_0}}\right) + \Phi\left(-\sqrt{14\frac{E}{N_0}}\right)
\end{aligned}$$

Note: The dominant pairwise error derived in Slide IDC 7-91 is actually a lower bound of the optimal error probability.

3. Determine $d_{\rm E,free}$ via the signal diagram in Slide IDC 8-13.

   Solution. For the determination of $d_{\rm E,free}$, we only need to consider the smaller distance between the two on a branch. For example, $D^{d_2^2-d_0^2}; D^{d_0^2}$ can be simplified to $D^{d_0^2}$ since

$$d_0^2 = 2 - \sqrt{2} < d_2^2 - d_0^2 = 4 - (2 - \sqrt{2}) = 2 + \sqrt{2}.$$

<u>**Note that the output 000 due to input 00 should be ignored.**</u>

We then list the four equations to be solved jointly.

$$\begin{cases}
b = D^{d_1^2}L \cdot a_0 + L \cdot c \\
c = D^{d_0^2}L \cdot b + D^{d_0^2}L \cdot d \\
d = D^{d_0^2}L \cdot b + D^{d_0^2}L \cdot d \\
a_1 = D^{d_1^2}L \cdot c + \underbrace{D^{d_2^2}L \cdot a_0}_{\text{'1' should be ignored!}}
\end{cases}$$

With $c = d$, the four equations are reduced to

$$\begin{cases}
b = D^{d_1^2}L \cdot a_0 + L \cdot c \\
c = D^{d_0^2}L \cdot b + D^{d_0^2}L \cdot c = D^{d_0^2}L \cdot \left(D^{d_1^2}L \cdot a_0 + L \cdot c\right) + D^{d_0^2}L \cdot c \\
a_1 = D^{d_1^2}L \cdot c + D^{d_2^2}L \cdot a_0
\end{cases}$$

From the equation in the middle, we get

$$c = \frac{D^{d_1^2 + d_0^2}L^2}{1 - D^{d_0^2}L^2 - D^{d_0^2}L} \cdot a_0$$

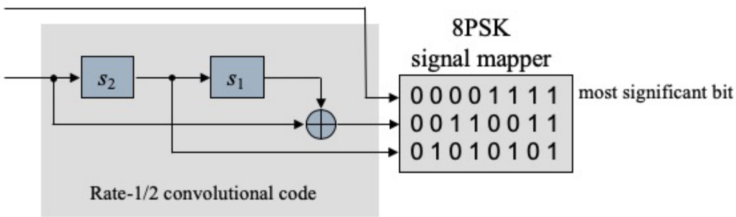
Consequently,

$$
\begin{aligned}
a_1 &= D^{d_1^2} L \cdot c + D^{d_2^2} L \cdot a_0 \\
&= D^{d_1^2} L \cdot \left(\frac{D^{d_1^2+d_0^2} L^2}{1 - D^{d_0^2} L^2 - D^{d_0^2} L} \cdot a_0 \right) + D^{d_2^2} L \cdot a_0 \\
&= \left(\frac{D^{2d_1^2+d_0^2} L^2}{1 - D^{d_0^2}(L^2 + L)} + D^{d_2^2} \right) L \cdot a_0 \\
&= \left(D^{2d_1^2+d_0^2} L^2 \left[1 + D^{d_0^2}(L^2+L) + D^{2d_0^2}(L^2+L)^2 + \cdots \right] + D^{d_2^2} \right) L \cdot a_0 .
\end{aligned}
$$

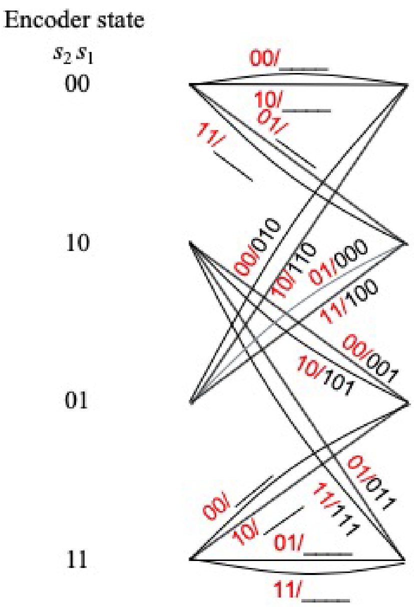
The lowest order term $D^{d_2^2} L$ indicates that there exists $d_{\text{E,free}} = d_2$ with one branch transition.

Note: From the derivation, we also know that the second largest "distance deviation" from the all-zero coded signal is $d = \sqrt{2d_1^2 + d_0^2}$ that can be achieved with three branch transitions.

4. Consider the 4-state Ungerboeck trellis coded modulation below.

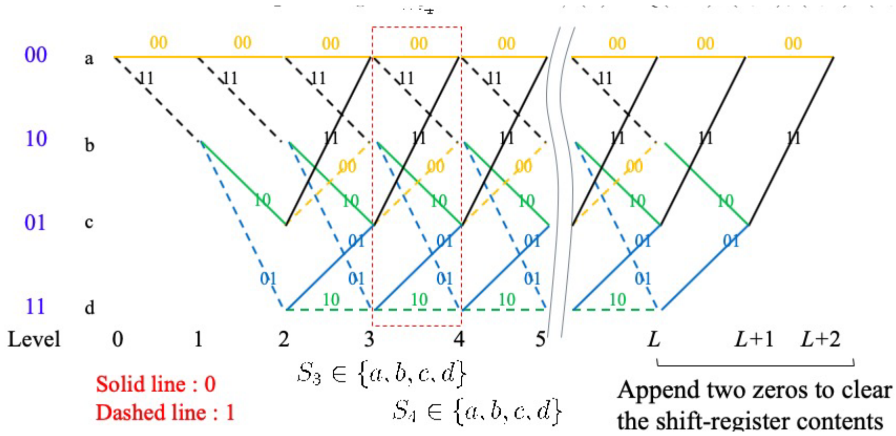


Fill in the **eight** output 8-PSK signals corresponding to the inputs indicated.



Solution. See Slide IDC 8-7.

5. (a) In the dotted box below, what is the set of transitions $\mathcal{B}_{3,4}(0) \subset \mathcal{S}_3 \times \mathcal{S}_4$ corresponding to symbol 0?



(b) The BCJR algorithm intends to minimize a particular bit error such as the one from level 3 to level 4. The decision is based on the log-likelihood ratio

$$l(4) = \log \frac{\sum_{(S_3,S_4)\in\mathcal{B}_{3,4}(1)} P(S_3, S_4, \mathbf{r})}{\sum_{(S_3,S_4)\in\mathcal{B}_{3,4}(0)} P(S_3, S_4, \mathbf{r})}$$

The technique is to decompose $P(S_3, S_4, \mathbf{r})$ into the product of $\alpha(S_3)$, $\beta(S_4)$ and $\gamma(S_3, S_4)$, where $\alpha(S_3)$ is only a function of the portion of the received vector before level 3, $\beta(S_4)$ is only a function of the portion of the received vector after level 4, and $\gamma(S_3, S_4)$ is only a function of the portion of the received vector between levels 3 and 4. Please complete the following derivation.

$$\begin{aligned}
P(S_3, S_4, r_1^N) &= P(S_3, S_4, \underbrace{r_1^6}_{\text{past}}, \underbrace{r_7^8}_{\text{now}}, \underbrace{r_9^N}_{\text{future}}) \\
&= P(r_9^N|\cdot,\cdot,\cdot,\cdot)P(\cdot,\cdot,\cdot,\cdot) \\
&= P(r_9^N|\cdot)P(\cdot,\cdot,\cdot,\cdot) \\
&\quad \text{(Explain why we can simplify the first term.)} \\
&= P(r_9^N|\cdot)P(\cdot,\cdot)P(S_4, r_7^8|\cdot,\cdot) \\
&= P(r_9^N|\cdot)P(\cdot,\cdot)P(S_4, r_7^8|\cdot) \\
&\quad \text{(Explain why we can simplify the last term.)}
\end{aligned}$$

Indicate which part the α-function is, which part the β-function is, and which part the γ-function is.

(c) The computation of α and β functions can be done recursively, provided that γ function is given. Please complete the recursive derivation below.

$$\begin{aligned}
\alpha(S_3) &= P(S_3, r_1^6) \\
&= \sum_{S_2\in\{a,b,c,d\}} P(S_2, S_3, r_1^4, r_5^6) \\
&= \sum_{S_2\in\{a,b,c,d\}} P(S_2, \cdot)P(S_3, \cdot|S_2, \cdot) \\
&= \sum_{S_2\in\{a,b,c,d\}} P(S_2, \cdot)P(S_3, \cdot|S_2) \\
&= \sum_{S_2\in\{a,b,c,d\}} \alpha(S_2)\gamma(S_2, S_3)
\end{aligned}$$

and

$$\begin{aligned}
\beta(S_4) &= P(r_9^N|S_4) \\
&= \sum_{S_5\in\{a,b,c,d\}} P(S_5, r_9^{10}, r_{11}^N|S_4) \\
&= \sum_{S_5\in\{a,b,c,d\}} P(\cdot|S_4, S_5, \cdot)P(S_5, \cdot|S_4) \\
&= \sum_{S_5\in\{a,b,c,d\}} P(\cdot|S_5)P(S_5, \cdot|S_4) \\
&= \sum_{S_5\in\{a,b,c,d\}} \beta(S_5)\gamma(S_4, S_5)
\end{aligned}$$

Solution.

(a) $\mathcal{B}_{3,4}(0) = \{(a,a), (b,c), (c,a), (d,c)\}$

(b)

$$\begin{aligned}
P(S_3, S_4, r_1^N) &= P(S_3, S_4, \underbrace{r_1^6}_{\text{past}}, \underbrace{r_7^8}_{\text{now}}, \underbrace{r_9^N}_{\text{future}}) \\
&= P(r_9^N|S_3, S_4, r_1^6, r_7^8)P(S_3, S_4, r_1^6, r_7^8) \\
&= P(r_9^N|S_4)P(S_3, S_4, r_1^6, r_7^8)
\end{aligned}$$

(Because $(S_3, r_1^6, r_7^8) \to S_4 \to r_9^N$ forms a Markov chain.)
($P(r_9^N|S_4)$ is only a function of the "future" received vector r_9^N and state S_4; thus, it can be served as the desired β-function.)

$$\begin{aligned}
&= P(r_9^N|S_4)P(S_3, r_1^6)P(S_4, r_7^8|S_3, r_1^6) \\
&= P(r_9^N|S_4)P(S_3, r_1^6)P(S_4, r_7^8|S_3)
\end{aligned}$$

(Because $r_1^6 \to S_3 \to (S_4, r_7^8)$ forms a Markov chain.)
($P(S_3, r_1^6)$ is only a function of "past" r_1^6 and state S_3; thus, we obtain the desired α-function.)
($P(S_4, r_7^8|S_3)$ is only a function of "now" r_7^8 and states S_3, S_4; thus, we obtain the desired γ-function.)

(c) See Slides IDC 8-32 and IDC 8-33.

Note: The decomposition of γ-function into the product of "systematic" part, "parity" part and "prior" part is also important, and shall be included in your preparation of the final exam.

6. (a) What is the code rate of an (n, t_c, t_r) regular low-density parity-check code?

(b) For a given parity-check matrix

$$H_{(n-k)\times n} = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}_{4\times 6}$$

draw Forney's factor graph (or bipartite graph) with six variable nodes and four check nodes.

(c) The Gallager Sum-Product Algorithm can be described as follows.

Step 1. Initialization. Set $P_{i,j}^1 = p_j^1 = P(c_j = 1|r_j)$ for $0 \le i \le 2$.

Step 2. Check Node Update (Horizontal Step). Perform

$$Q_{i,j}^1 = \frac{1}{2}\left(1 + \prod_{k\in\text{Check}(i)\setminus\{j\}} (2P_{i,k}^1 - 1)\right)$$

where Check(i) is the set of indices of variable nodes that connect to check node i.

Step 3. Variable Node Update (Vertical Step). Perform

$$P_{i,j}^1 = \frac{p_j^1 \prod_{k\in\text{Bit}(j)\setminus\{i\}} Q_{k,j}^1}{p_j^1 \prod_{k\in\text{Bit}(j)\setminus\{i\}} Q_{k,j}^1 + (1-p_j^1) \prod_{k\in\text{Bit}(j)\setminus\{i\}} (1 - Q_{k,j}^1)}$$

where Bit(j) is the set of indices of check nodes that connect to variable node j.

Step 4. Decision. Compute

$$P_j^1 = \frac{p_j^1 \prod_{k\in\text{Bit}(j)} Q_{k,j}^1}{p_j^1 \prod_{k\in\text{Bit}(j)} Q_{k,j}^1 + (1-p_j^1) \prod_{k\in\text{Bit}(j)} (1 - Q_{k,j}^1)}$$

and

$$\hat{c}_j = \begin{cases} 1, & \text{if } P_j^1 > \frac{1}{2} \\ 0, & \text{if } P_j^1 < \frac{1}{2} \end{cases}$$

Step 5. Termination. If $\hat{\boldsymbol{c}}H^T = \mathbf{0}$, the algorithm stops;
else set $p_j^1 = P_j^1$ and go to Step 2.

Suppose the all-zero codeword is transmitted (where we map $\{0,1\}$ to $\{1,-1\}$) and initially, we have

$$p_j^1 = P((-1)^{c_j} = 1|r_j) = 0.9 \text{ for } 1 \le j \le 5$$

and

$$p_0^1 = P((-1)^{c_0} = 1|r_0) = 0.1.$$

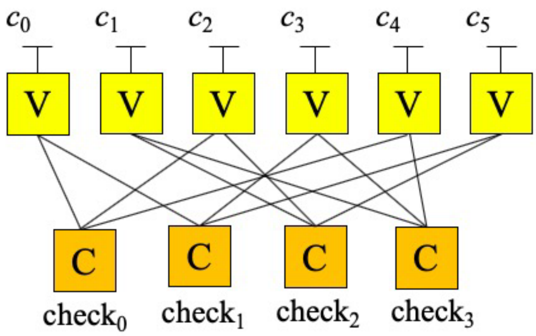
What are the values of $Q_{0,0}^1$, $Q_{1,0}^1$ and $Q_{2,0}^1$, $Q_{3,0}^1$?

(d) Perform the decision step to find P_0^1 using the result in (c).

Solution.

(a) $\frac{k}{n} = 1 - \frac{t_c}{t_r}$

(b)



(c) Initially, we have

$$\begin{bmatrix} P_{0,0}^1 & P_{1,0}^1 & P_{2,0}^1 & P_{3,0}^1 \\ P_{0,1}^1 & P_{1,1}^1 & P_{2,1}^1 & P_{3,1}^1 \\ P_{0,2}^1 & P_{1,2}^1 & P_{2,2}^1 & P_{3,2}^1 \\ P_{0,3}^1 & P_{1,3}^1 & P_{2,3}^1 & P_{3,3}^1 \\ P_{0,4}^1 & P_{1,4}^1 & P_{2,4}^1 & P_{3,4}^1 \\ P_{0,5}^1 & P_{1,5}^1 & P_{2,5}^1 & P_{3,5}^1 \end{bmatrix} = \begin{bmatrix} 0.1 & 0.1 & 0.1 & 0.1 \\ 0.9 & 0.9 & 0.9 & 0.9 \\ 0.9 & 0.9 & 0.9 & 0.9 \\ 0.9 & 0.9 & 0.9 & 0.9 \\ 0.9 & 0.9 & 0.9 & 0.9 \\ 0.9 & 0.9 & 0.9 & 0.9 \end{bmatrix}$$

Also, we know Check(0)=$\{0,2,4\}$, Check(1)=$\{0,3,5\}$, Check(2)=$\{1,2,5\}$ and Check(3)=$\{1,3,4\}$. This gives

$$Q_{0,0}^1 = \frac{1}{2}\left(1 + \prod_{k\in\text{Check}(0)\setminus\{0\}} (2P_{0,k}^1 - 1)\right) = \frac{1}{2}\left(1 + \prod_{k\in\{2,4\}} (2P_{0,k}^1 - 1)\right) = 0.82$$

$$Q_{1,0}^1 = \frac{1}{2}\left(1 + \prod_{k\in\text{Check}(1)\setminus\{0\}} (2P_{1,k}^1 - 1)\right) = 0.82$$

$$Q_{2,0}^1 = \frac{1}{2}\left(1 + \prod_{k\in\text{Check}(2)\setminus\{0\}} (2P_{2,k}^1 - 1)\right) = 0.756$$

and

$$Q_{3,0}^1 = \frac{1}{2}\left(1 + \prod_{k\in\text{Check}(3)\setminus\{0\}} (2P_{2,k}^1 - 1)\right) = 0.756$$

Note: $Q^1_{0,0}$ represents the probability that check 0 is satisfied (i.e., $\text{check}_0 = 1$) given $(-1)^{c_0} = 1$.

(d) With Bit(0)=$\{0, 1\}$, we have

$$\begin{aligned} P^1_0 &= \frac{p^1_0 \prod_{k \in \text{Bit}(0)} Q^1_{k,0}}{p^1_0 \prod_{k \in \text{Bit}(0)} Q^1_{k,0} + (1 - p^1_0) \prod_{k \in \text{Bit}(0)} (1 - Q^1_{k,0})} \\ &= \frac{0.1 \cdot 0.82 \cdot 0.82}{0.1 \cdot 0.82 \cdot 0.82 + 0.9 \cdot 0.18 \cdot 0.18} \\ &\approx 0.698 \end{aligned}$$

Note: Initially, we have $p^1_0 = 0.1$; after the first iteration, this probability is raised to 0.698 (refined based on the probabilistic messages or beliefs from the other nodes).