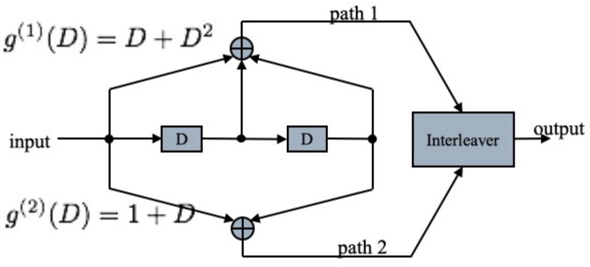
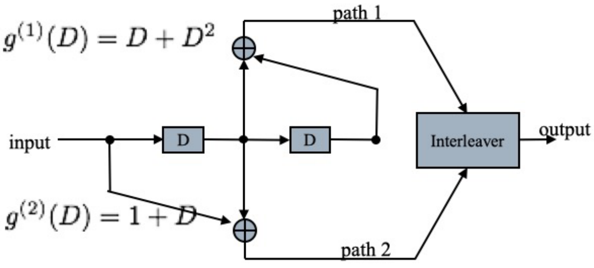
Corrections:

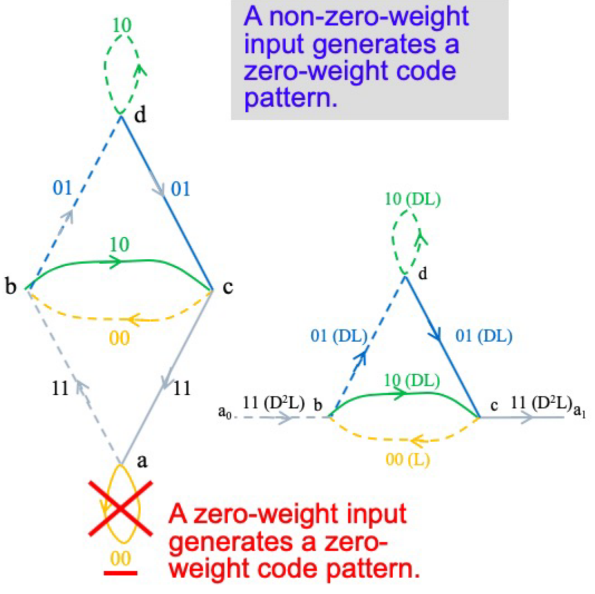
- Slide IDC7-84: (The corrections in the "Sample Problems for Quiz 11" have been again corrected as follows.)



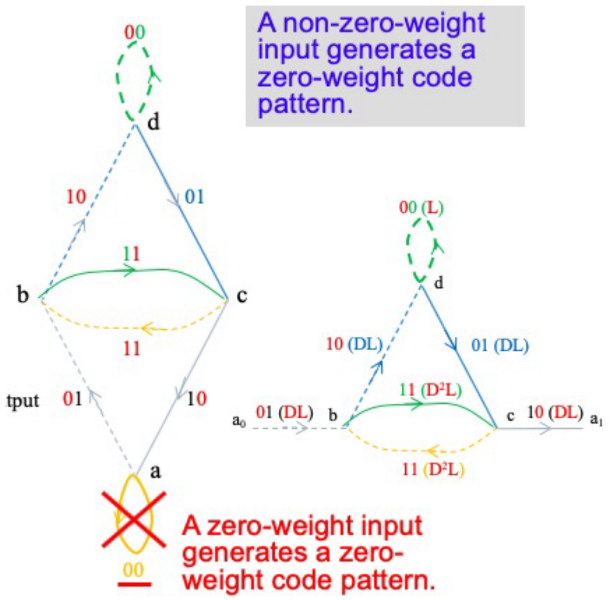
should be



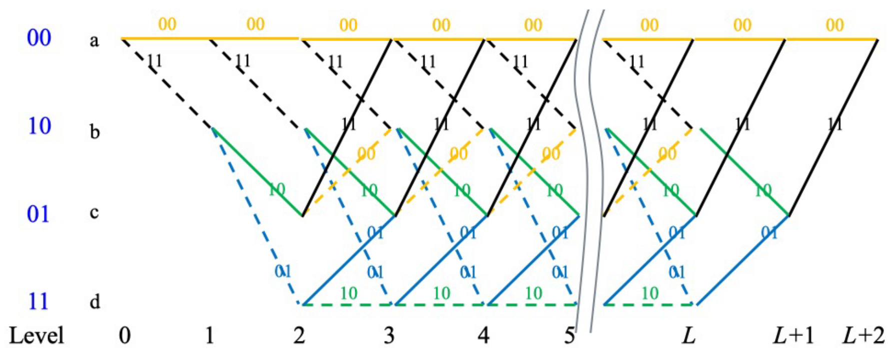
- Slide IDC7-84:



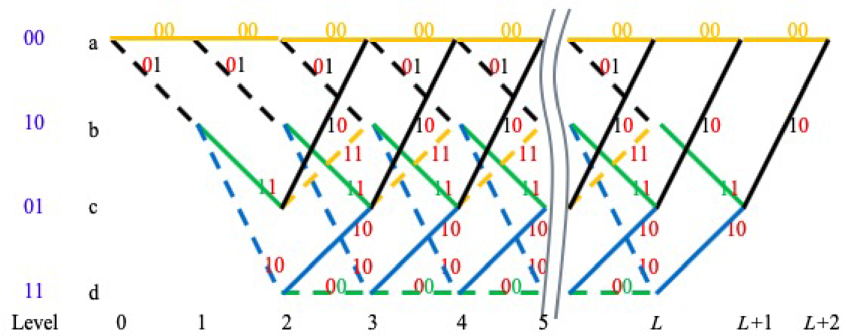
should be



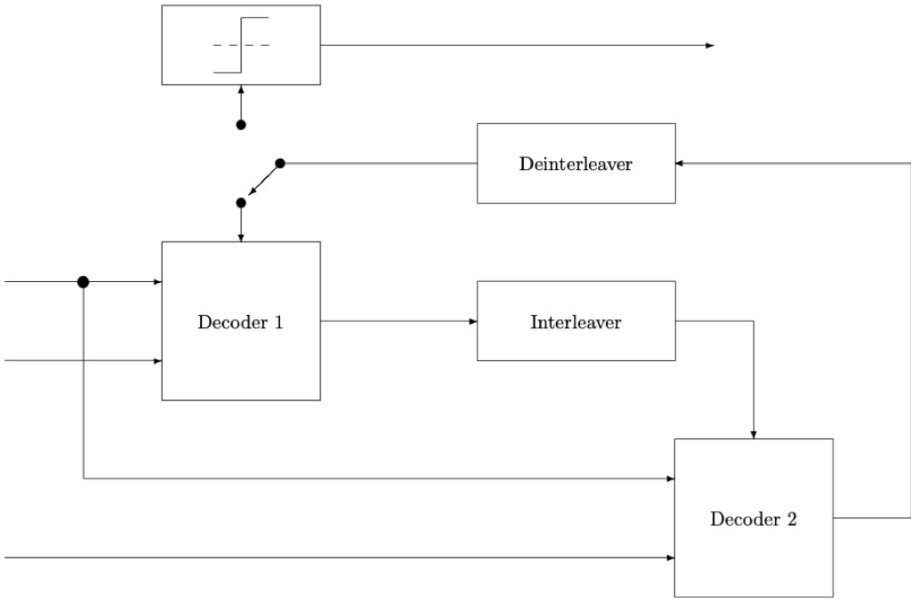
- Slide IDC7-84:



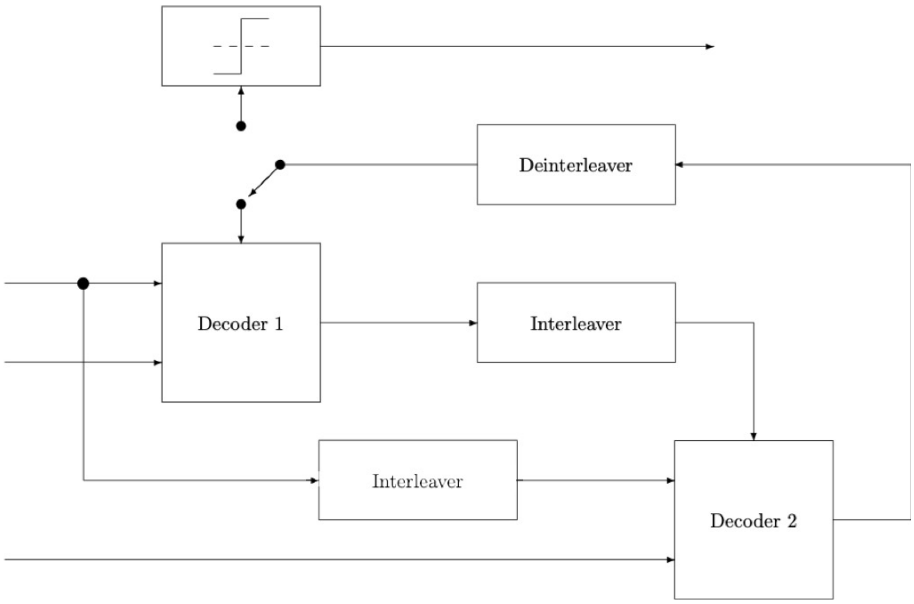
should be



- IDC 8-28:


Deinterleaver
Decoder 1
Interleaver
Decoder 2

should be


Deinterleaver
Decoder 1
Interleaver
Interleaver
Decoder 2

Additional sample problems for the final exam

1. Show that for cyclic codes, we can obtain the syndrome through

$$r(X)h(X)$$

where $h(X)$ is the parity check polynomial of the cyclic code, and $r(X)$ is the received vector polynomial.

Solution. From Slide IDC 7-49, we learn that given the error pattern polynomial $e(X)$, we have

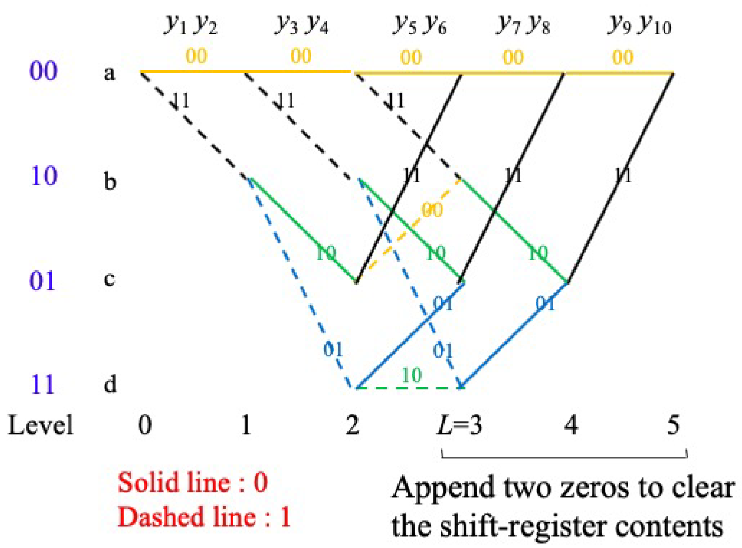
$$\begin{aligned} r(X)h(X) &= [c(X)+e(X)]h(X) \\ &= a(X)(X^n+1)+e(X)h(X) \end{aligned}$$

Hence, the coefficients of $r(X)h(X)$ for degrees k to $n-1$ can be obtained via

$$\sum_{i=j}^{j+k} r_i h_{k+j-i} \text{ for } 0 \le j \le n-k-1$$

which from Slide IDC 7-50 is exactly the syndrome of the cyclic code.

2. Consider the convolutional code trellis with $L = 3$ as shown below:



Suppose 0 and 1 are transmitted using $\sqrt{E}$ and $-\sqrt{E}$, respectively (i.e., antipodal transmission). Denote the transmitted signals as $\boldsymbol{s} = (s_1, s_2, \ldots, s_{10})$. Assume

$$y_i = s_i + n_i,$$

where $\{n_i\}_{i=1}^{10}$ are zero-mean i.i.d. Gaussian distributed with variance σ^2.

(a) How many paths this code trellis has?

(b) List all the codewords.

(c) (Hard-decision decoding) Suppose $E = 1$ and the information sequence 000 is transmitted. The receiver receives

$$\boldsymbol{y} = (y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8, y_9, y_{10}) = (3.2, -0.5, -0.1, -0.3, 1.2, 1.0, 2.1, 0.8, -0.4, -0.1)$$

From Slides IDC 7-71 and IDC 7-72, we learn that the *hard-decision* decoding should follow

$$\hat{\boldsymbol{c}} = \min_{\boldsymbol{c} \in \mathcal{C}} d_{\mathrm{H}}(\boldsymbol{r}, \boldsymbol{c})$$

where $d_{\mathrm{H}}(\cdot, \cdot)$ denotes the Hamming distance, $\boldsymbol{c}$ is one of the codewords listed in (b), and

$$r_i = \begin{cases} 0, & y_i > 0 \\ 1, & y_i < 0 \end{cases}$$

Noting that

$$\underbrace{d_{\mathrm{H}}(\boldsymbol{r}, \boldsymbol{c})}_{\substack{\text{path} \\ \text{metric}}} = \underbrace{d_{\mathrm{H}}(r_1 r_2, c_1 c_2)}_{\substack{\text{branch} \\ \text{metric}}} + d_{\mathrm{H}}(r_3 r_4, c_3 c_4) + d_{\mathrm{H}}(r_5 r_6, c_5 c_6) + d_{\mathrm{H}}(r_7 r_8, c_7 c_8) + d_{\mathrm{H}}(r_9 r_{10}, c_9 c_{10}),$$

we found the path metric $d_{\mathrm{H}}(\boldsymbol{r}, \boldsymbol{c})$ is the sum of five branch metrics. Please mark the Hamming branch metric on each branch, and use the Viterbi algorithm to determine the path that minimizes the Hamming path metric.

(d) (Soft-decision decoding) Continue from (c). From Slides IDC 7-71 and IDC 7-72, we learn that the *soft-decision* decoding follows

$$\hat{\boldsymbol{c}} = \min_{\boldsymbol{c} \in \mathcal{C}} d^2_{\mathrm{E}}(\boldsymbol{y}, (-1)^{\boldsymbol{c}})$$

where $d^2_{\mathrm{E}}(\cdot, \cdot)$ denotes the Euclidean distance square, $\boldsymbol{c}$ is one of the codewords listed in (b), and

$$(-1)^{\boldsymbol{c}} = ((-1)^{c_1}, (-1)^{c_2}, \ldots, (-1)^{c_{10}}).$$

Noting that

$$\underbrace{d^2_{\mathrm{E}}(\boldsymbol{y}, (-1)^{\boldsymbol{c}})}_{\substack{\text{path} \\ \text{metric}}} = \underbrace{(y_1 - (-1)^{c_1})^2 + (y_2 - (-1)^{c_2})^2}_{\substack{\text{branch} \\ \text{metric}}} + \cdots + \underbrace{(y_9 - (-1)^{c_9})^2 + (y_{10} - (-1)^{c_{10}})^2}_{\substack{\text{path} \\ \text{metric}}},$$

we found the path metric $d^2_{\mathrm{E}}(\boldsymbol{y}, (-1)^{\boldsymbol{c}})$ is the sum of five branch metrics. Please mark the branch metric of the Euclidean distance square on each branch, and use the Viterbi algorithm to determine the path that minimizes the Euclidean path metric.

(e) The decision rule of the *soft-decision* decoding in (d) can be rewritten as

$$\hat{\boldsymbol{c}} = \min_{\boldsymbol{c} \in \mathcal{C}} d_{\mathrm{E}}(\boldsymbol{y}, (-1)^{\boldsymbol{c}})$$

i.e., in the form of Euclidean distance, instead of Euclidean distance square. Why we cannot use Euclidean distance as the path metric for the Viterbi algorithm?

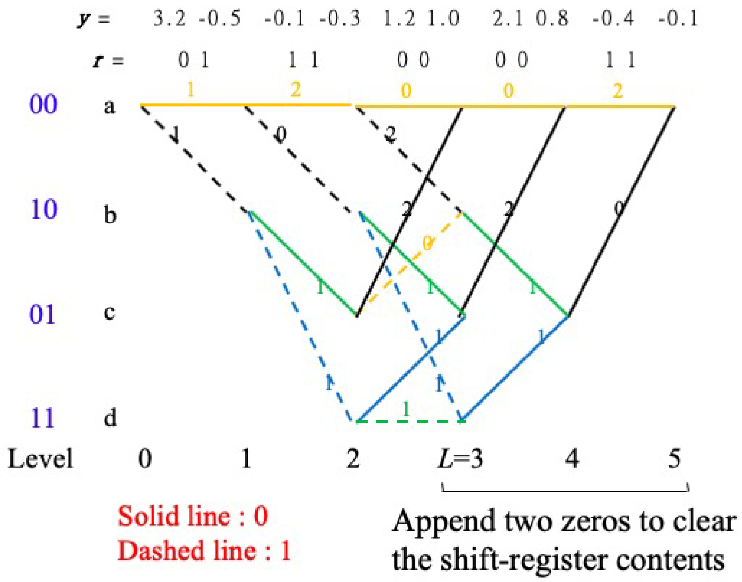
Solution.

(a) There are $2^L = 2^3 = 8$ code paths in this code trellis.

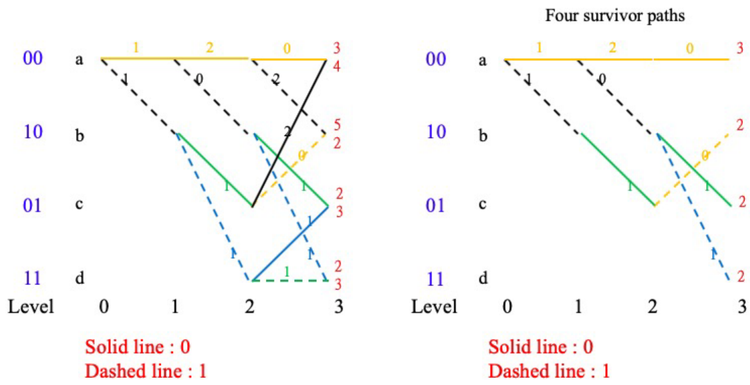
(b)

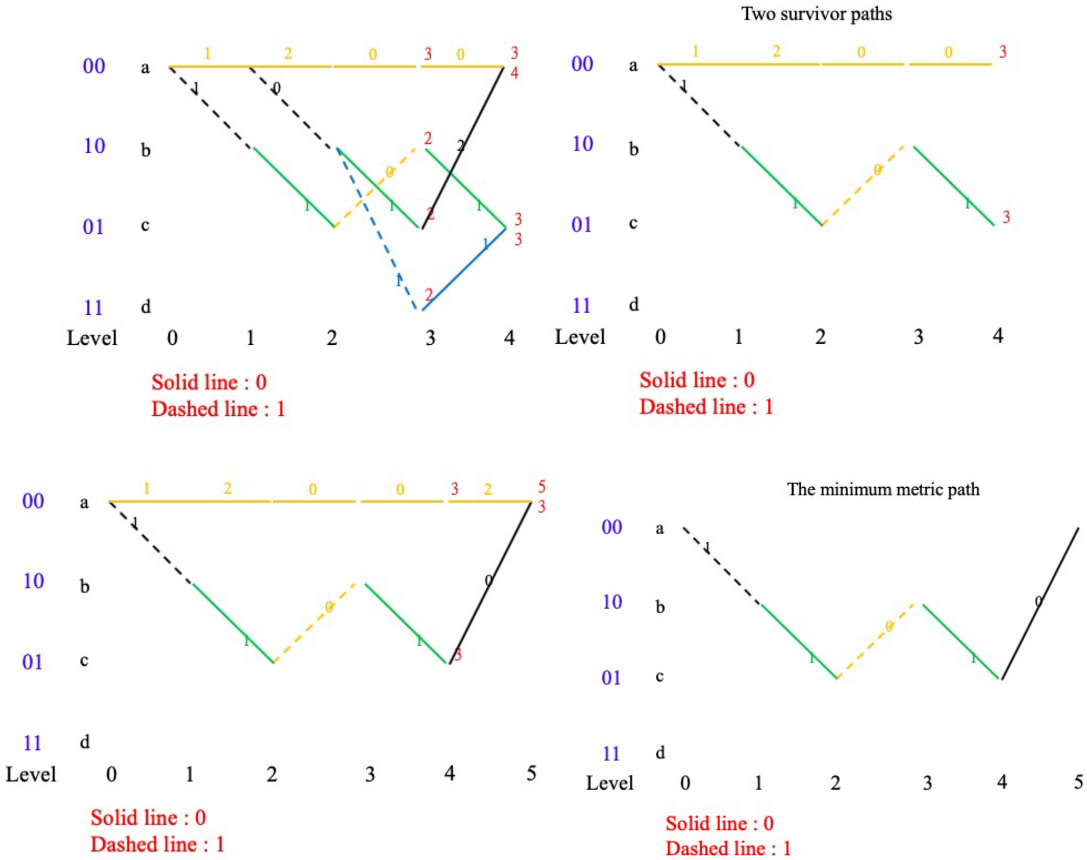
message	codeword			
000	00	00	00	0000
001	00	00	11	1011
010	00	11	10	1100
011	00	11	01	0111
100	11	10	11	0000
101	11	10	00	1011
110	11	01	01	1100
111	11	01	10	0111

(c) The Hamming branch metric for each branch is given as follows.

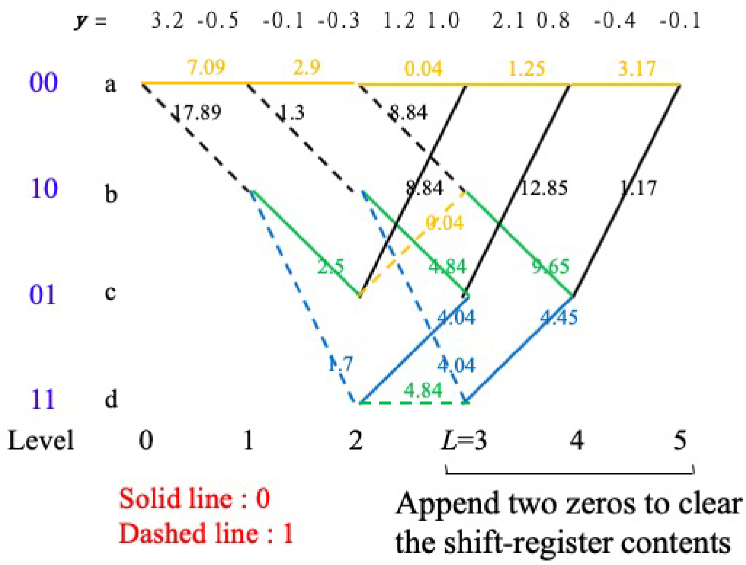


The determination of the path with the minimum Hamming path metric follows the procedure below, which outputs 101 as an estimate of the information sequence.

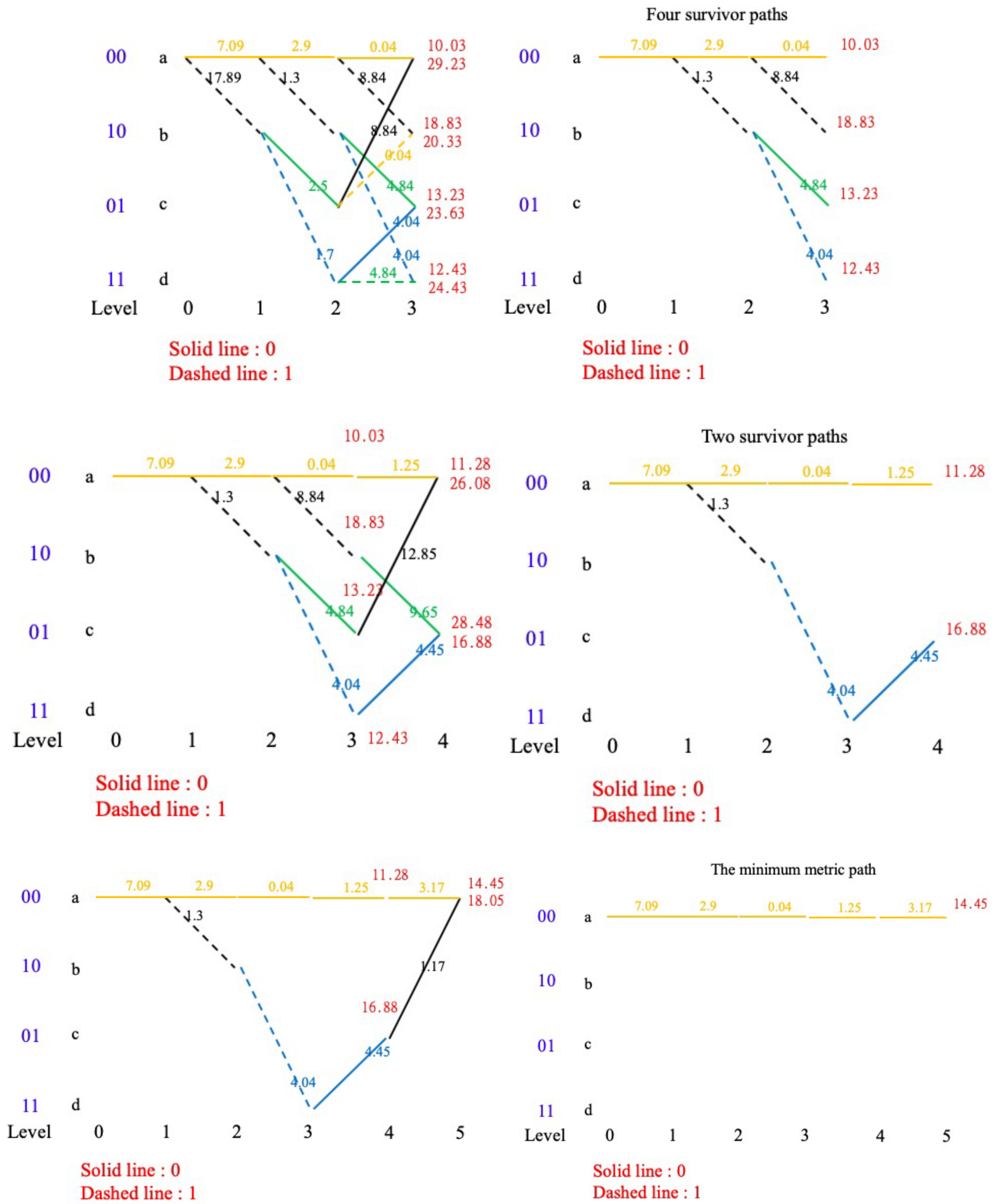




(d) The branch metric of the Euclidean distance square for each branch is given as follows.



The determination of the path with the minimum Hamming path metric follows the procedure below, which outputs 000 as an estimate of the information sequence.



Note: It can be observed from (c) and (d), the hard-decision decoding may force the decision to 0 or 1 due to a small perturbation such as -0.1. The soft-decision decoding, however, can compensate these small perturbations by other reliable receptions such as 3.2 or 2.1.

(e) Because the Euclidean distance is not **additive**, which is a property of the path metric that is required by the Viterbi algorithm.