Path Deletions for Finite Stack-Size Sequential-Type Decoding Algorithms

Chen-Yi Wang

Directed by: Prof. Po-Ning Chen

Department of Communications Engineering,
National Chiao-Tung University

July 9, 2009

Outline

- Introduction
- Preliminaries
- Path Deletion Schemes for Limited Stack-Size Sequential-Type Decoding Algorithm
- A Novel Two-Pass Sequential-Type Decoding Algorithm
- Simulation Results
- Concluding Remark and Future work

Chapter 1 :

Introduction

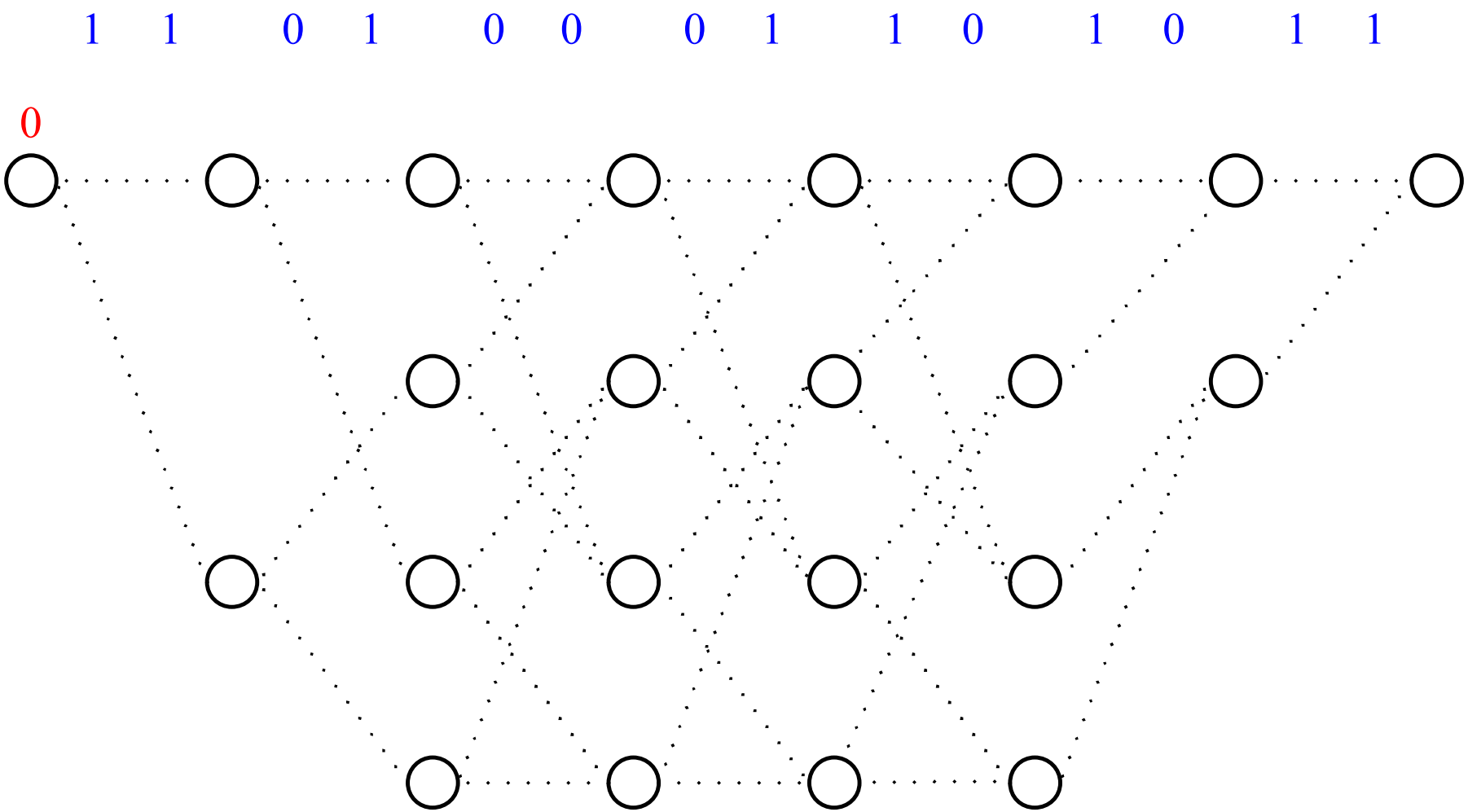
Viterbi Decoding Algorithm

- The most popular decoding algorithm for convolutional codes
  - Decoding complexity increases exponentially with respect to the code constraint length
  - Limited to constraint lengths of 10 or less

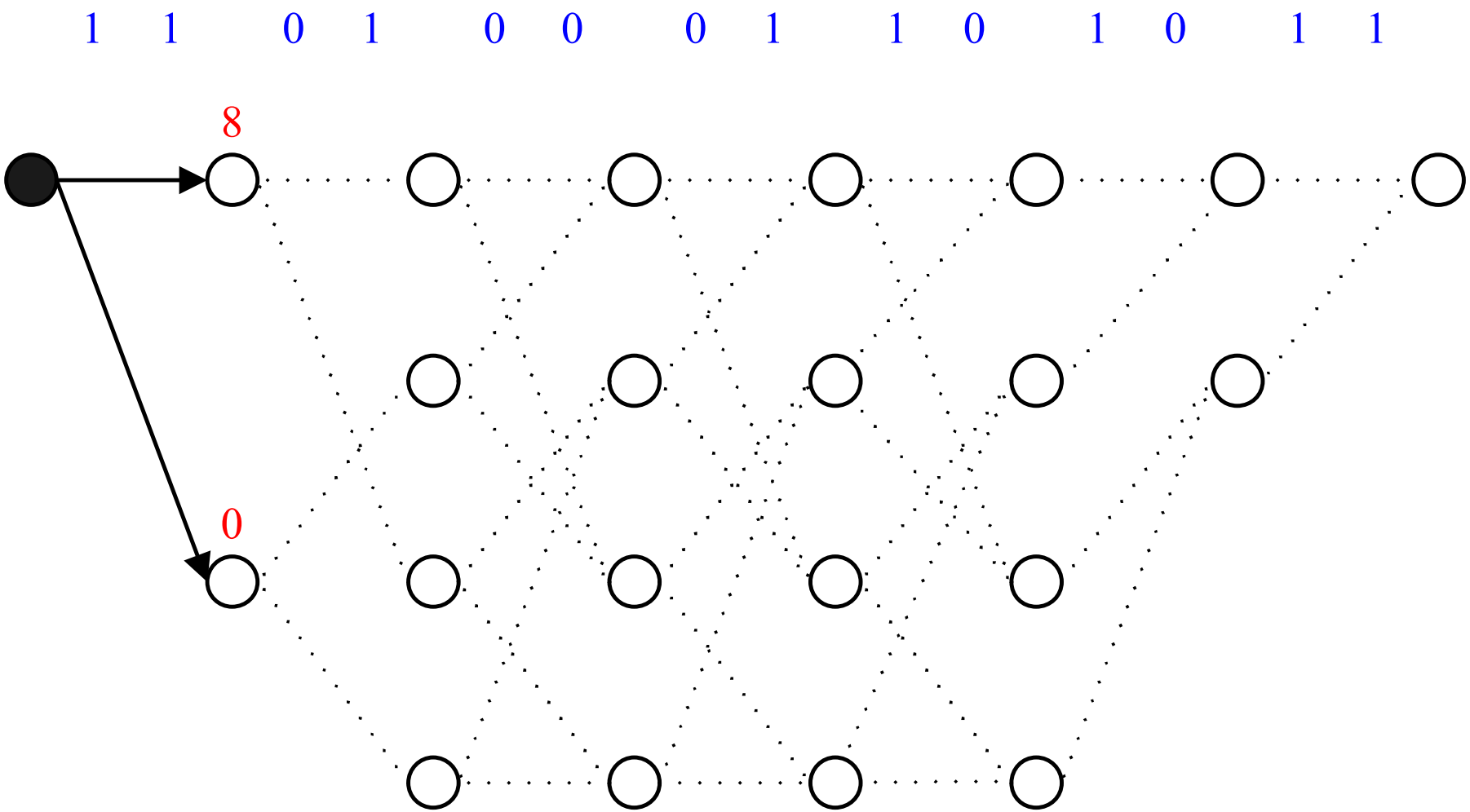
Sequential-Type Decoding Algorithms

- The computational complexity is almost independent of code constraint length
- The most well-known sequential-type decoding algorithm is perhaps the stack algorithm with Fano metric
  - sub-optimal in performance
  - efficient in complexity for medium to high SNRs
- In 2002, Han, Chen, and Wu proposed a new metric based on the Wagner rule :
  - adopting the new metric in place of the Fano metric guarantees maximum-likelihood performance

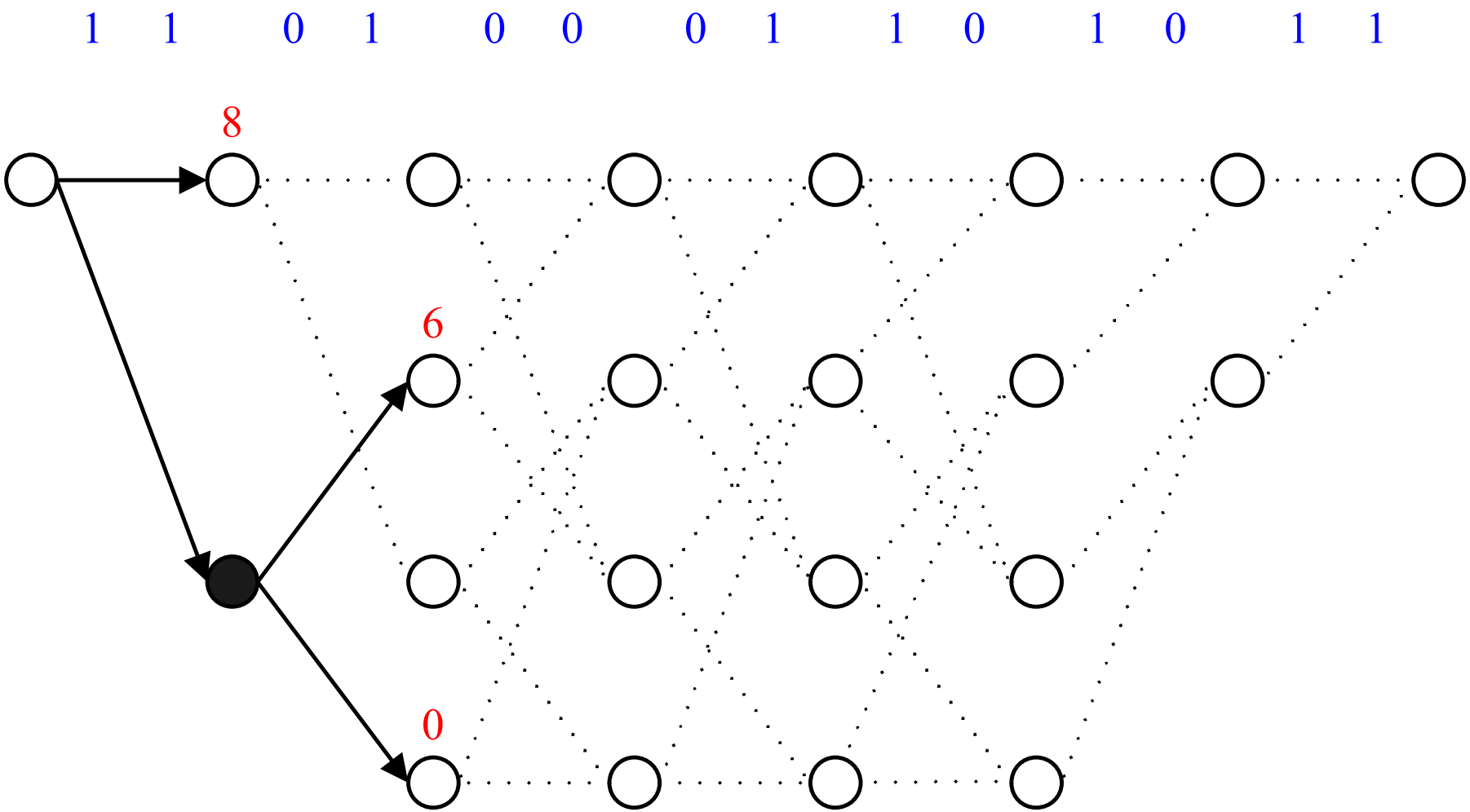
Begin



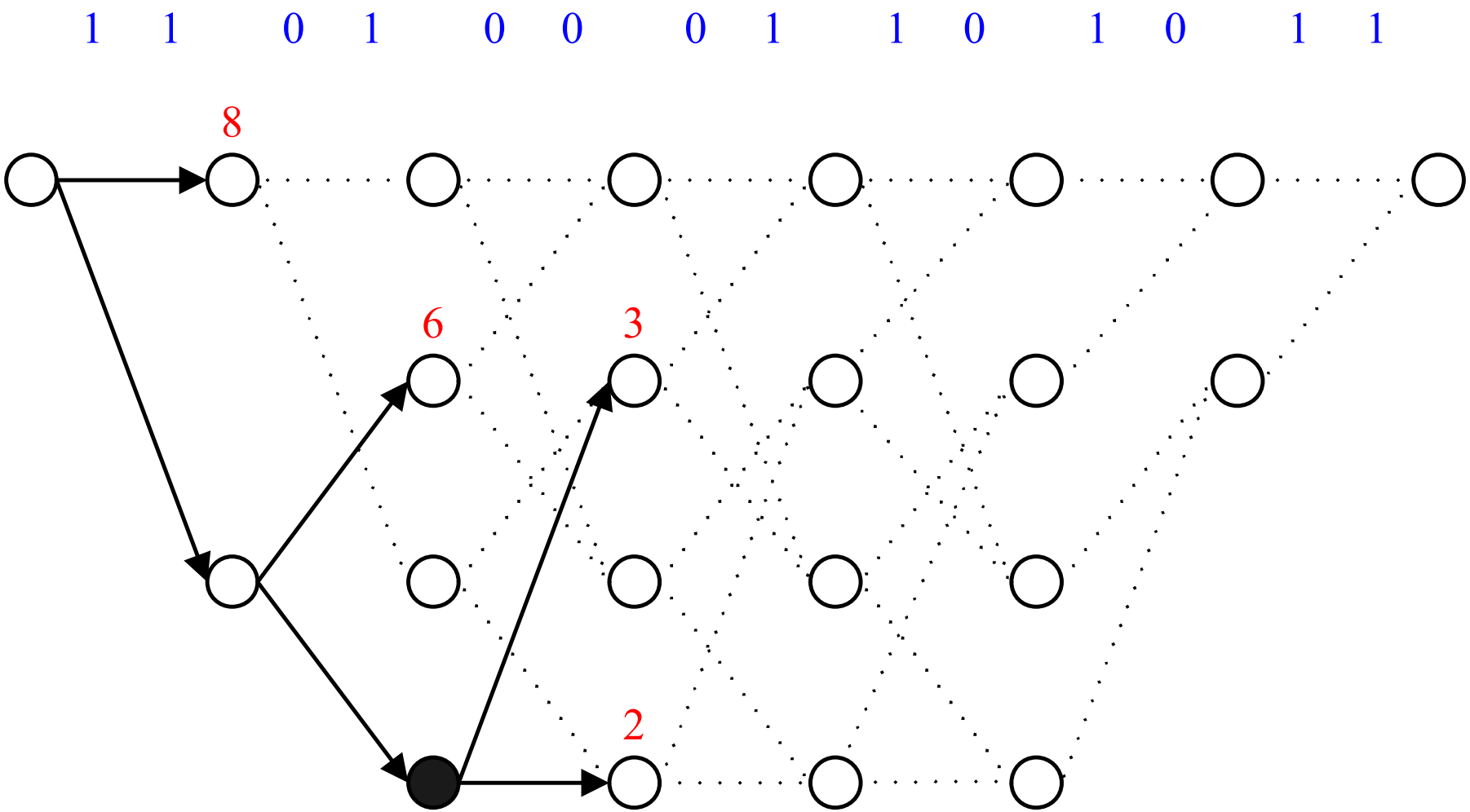
Search step # 1:



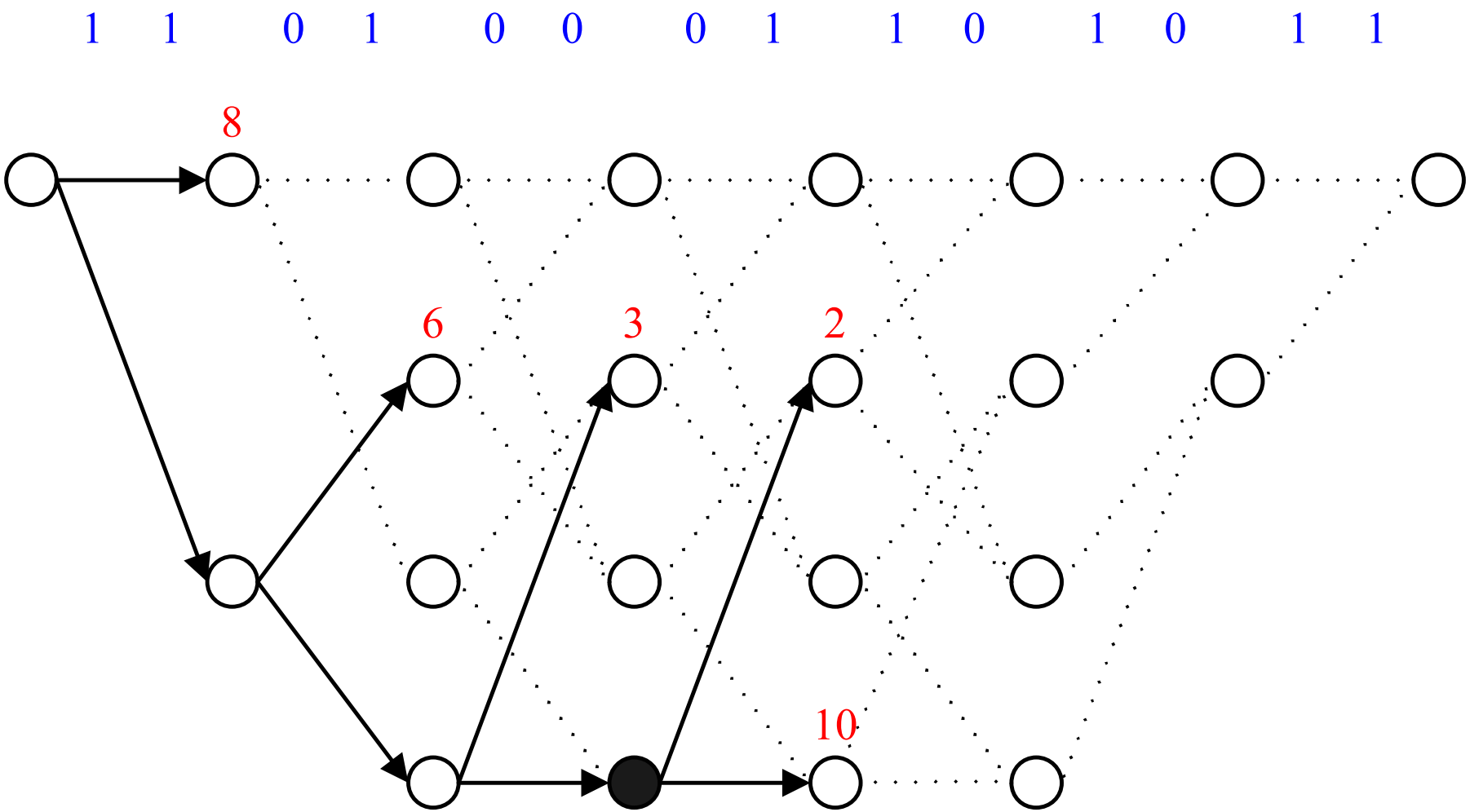
Search Step # 2:



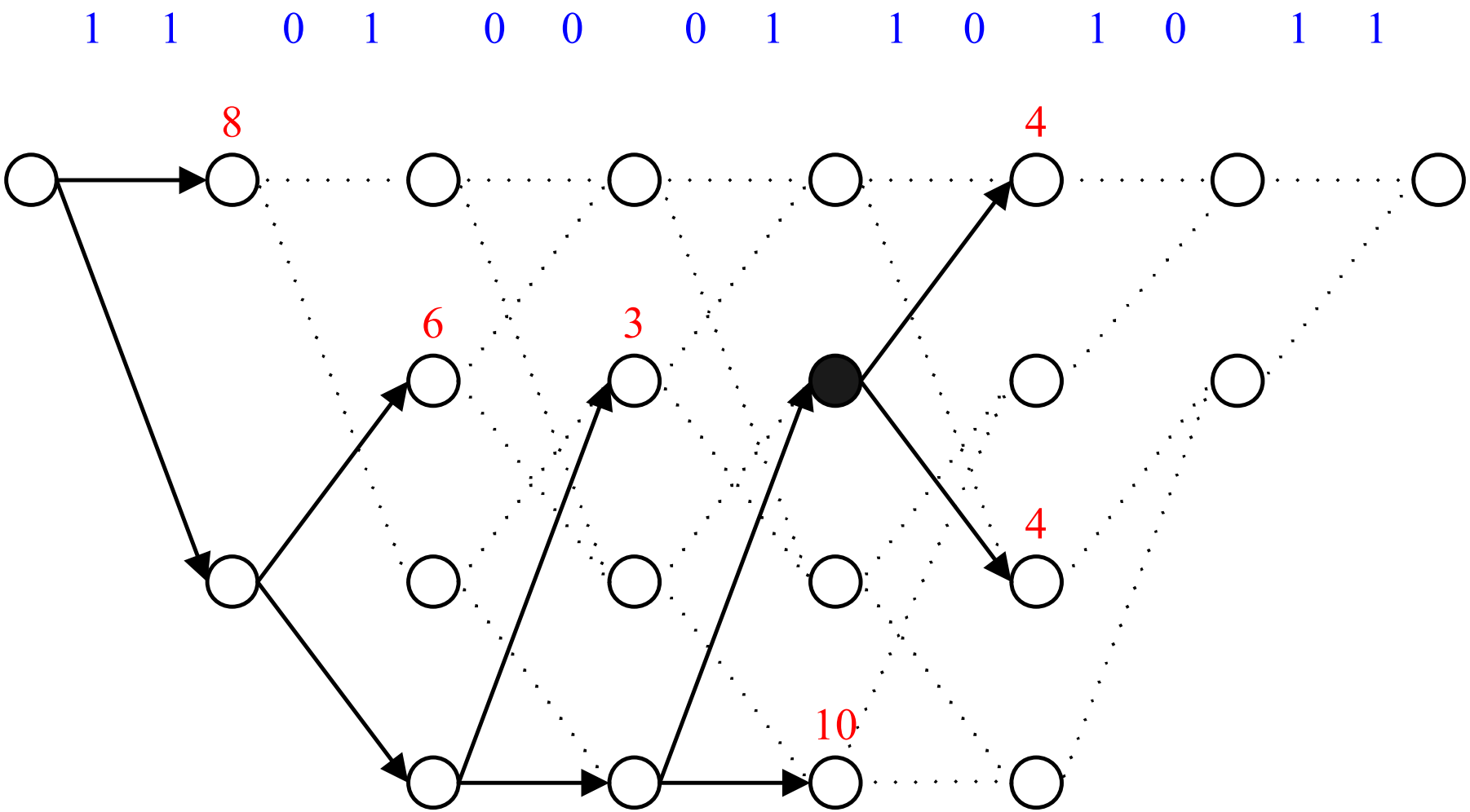
Search step # 3:



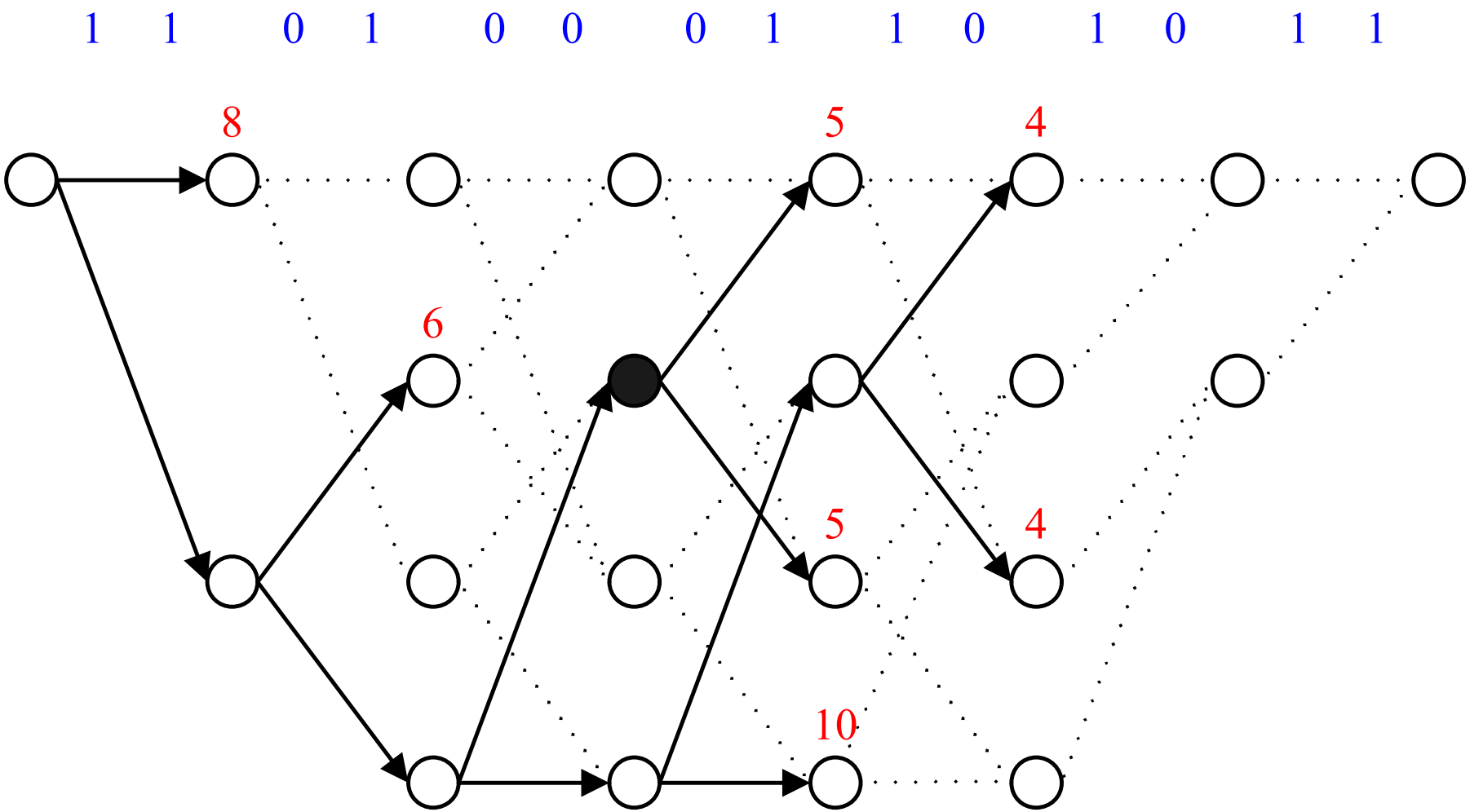
Search step # 4:



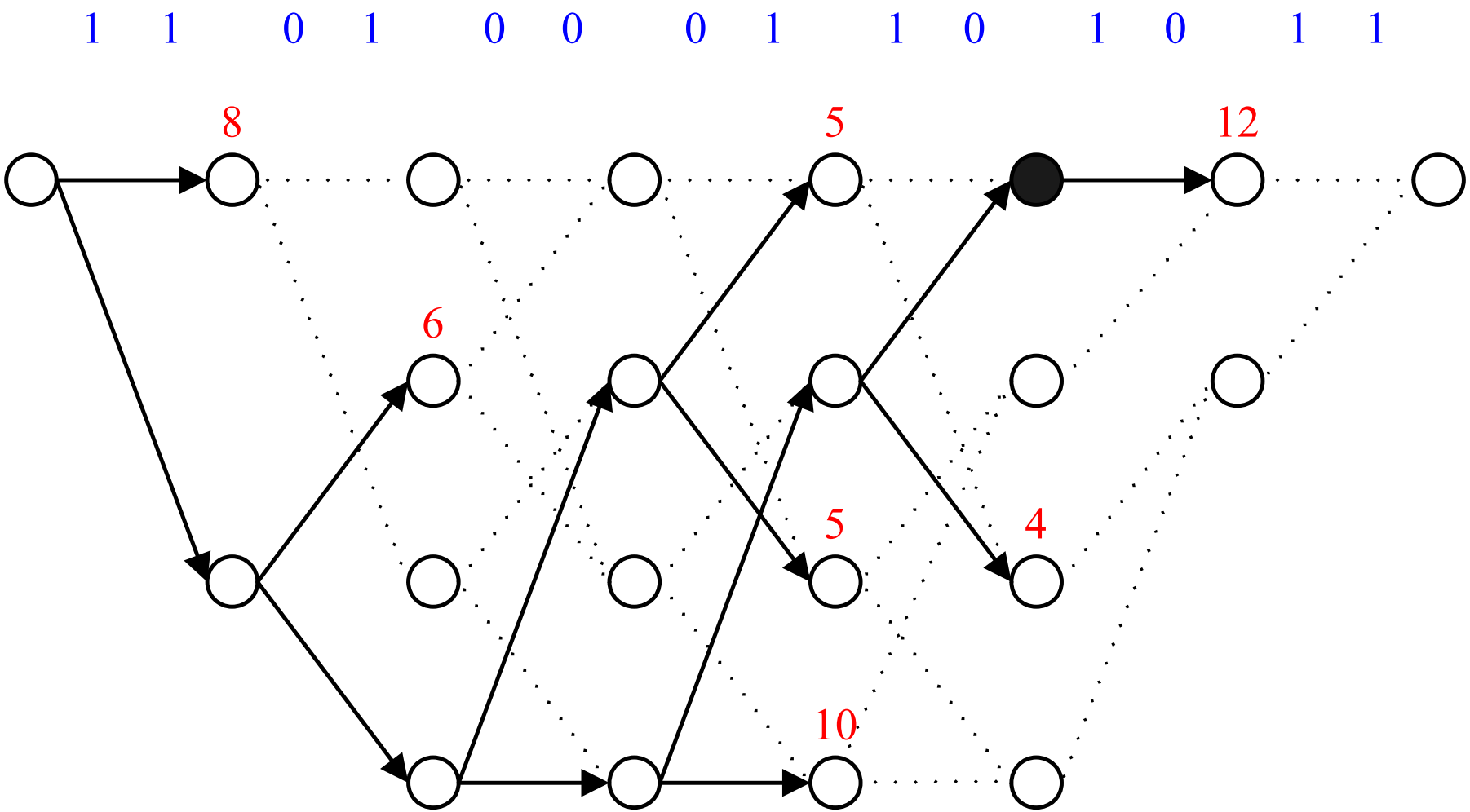
Search step # 5:



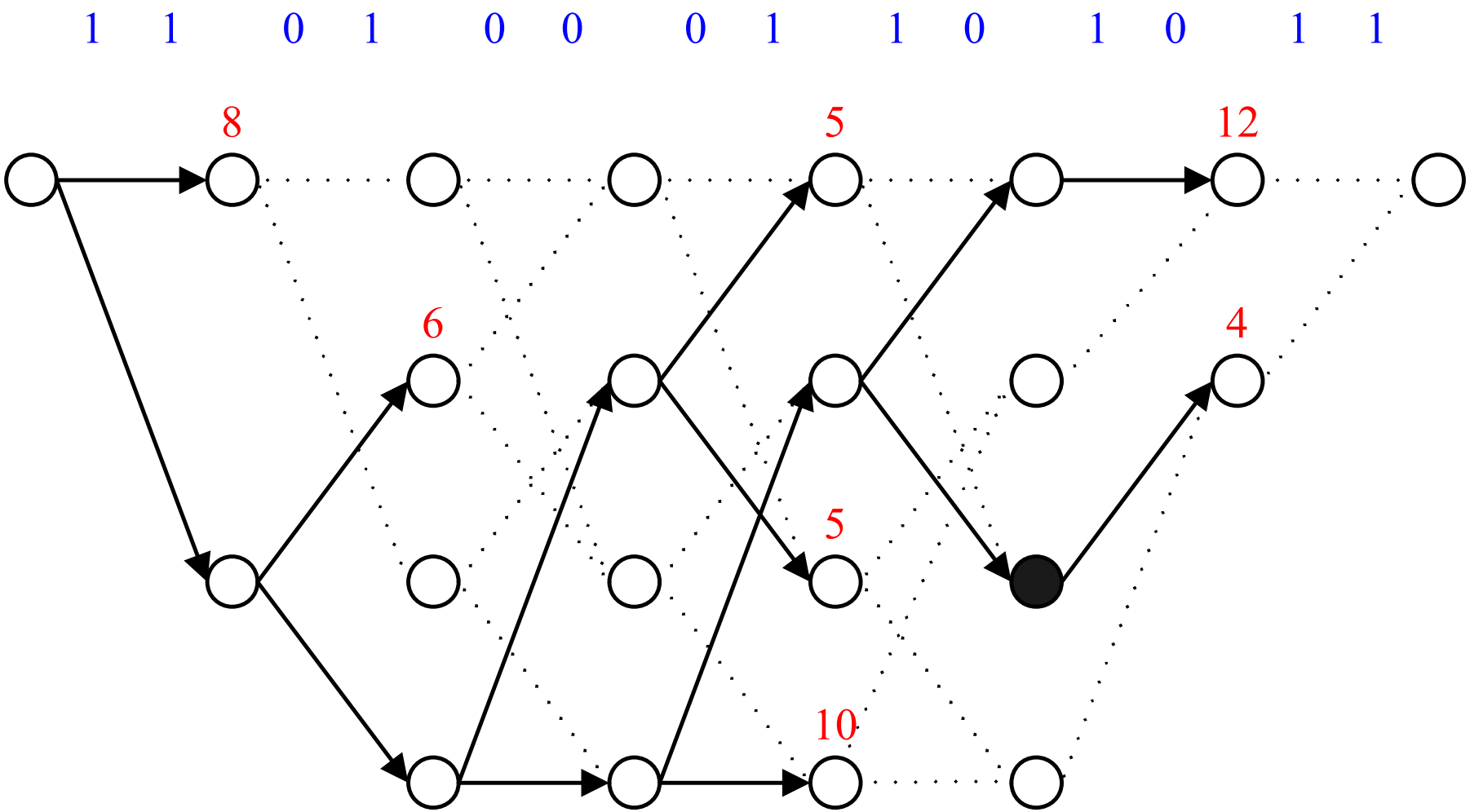
Search step # 6:



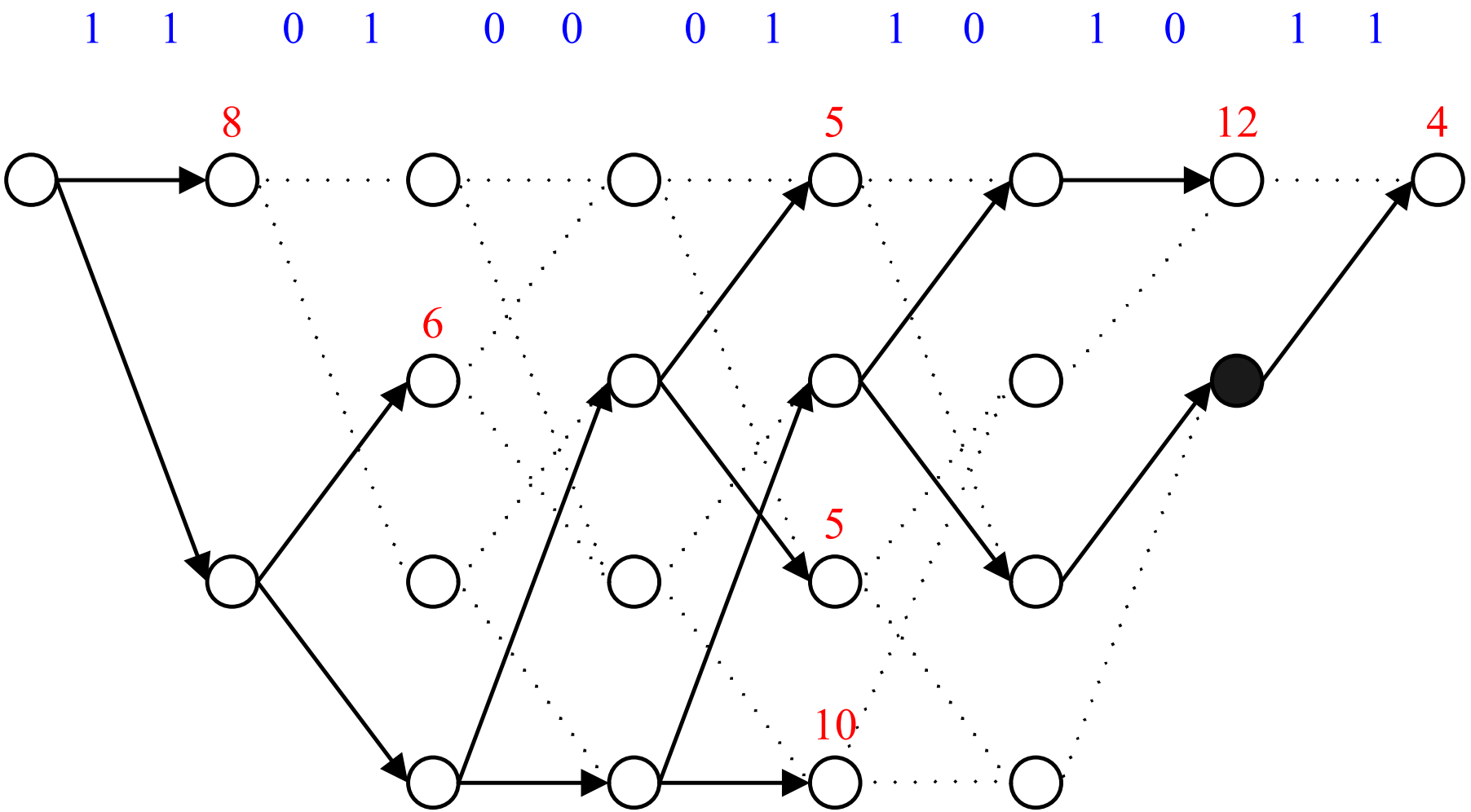
Search step # 7:



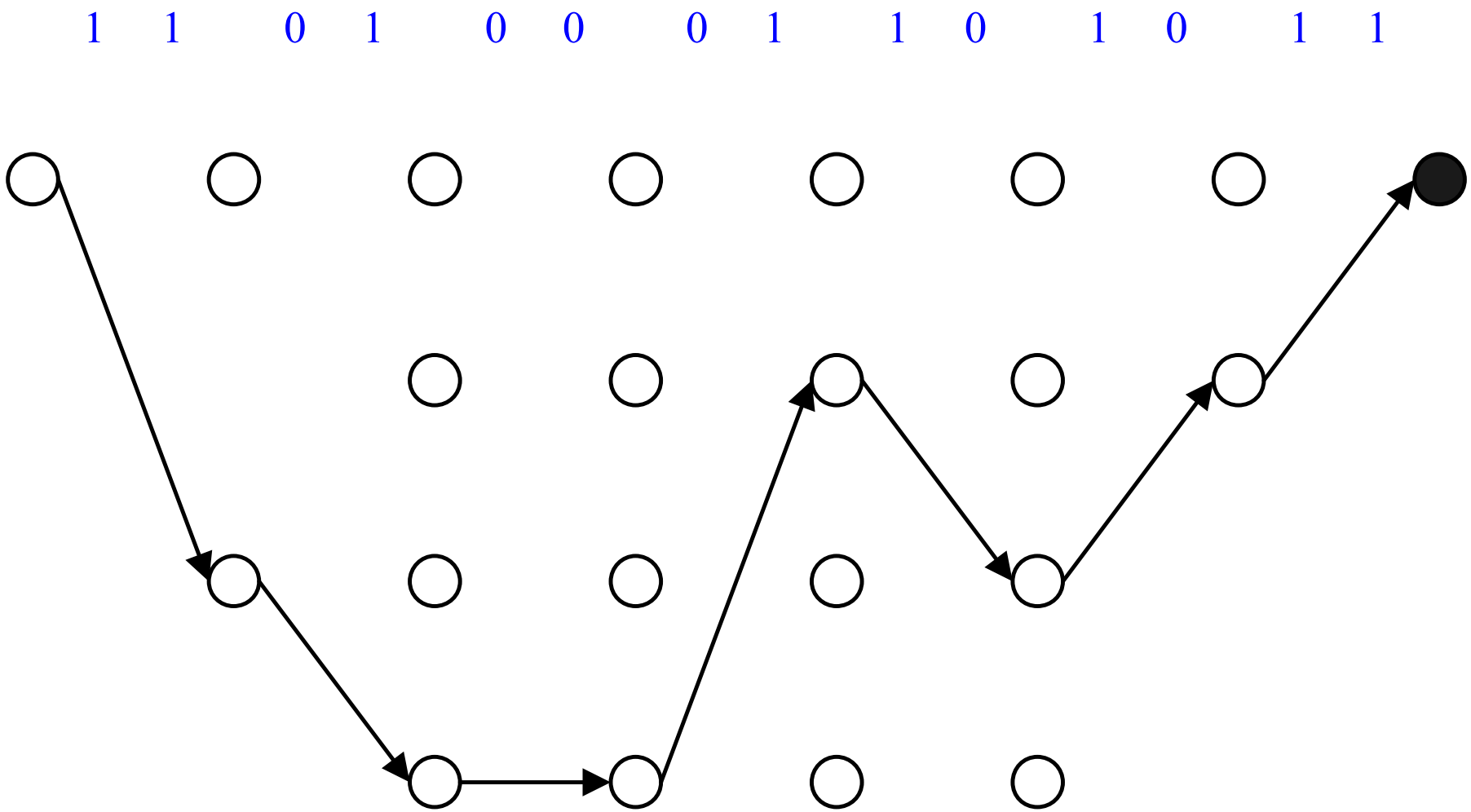
Search step # 8:



Search step # 9:



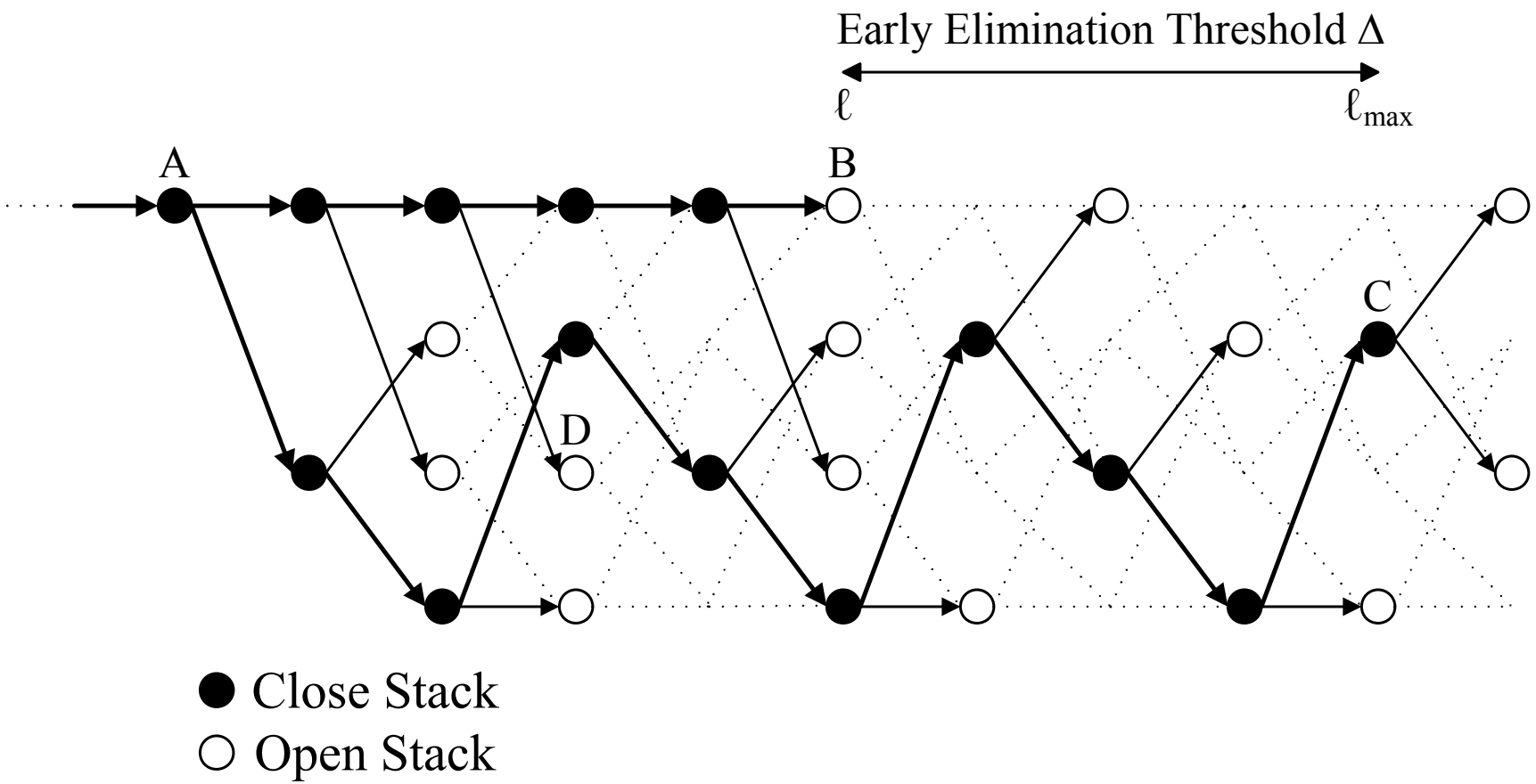
Final Output



Path Deletion Schemes

- A common problem of sequential-type decoding algorithms is
  - memory consumption grows with the length of information sequence
- In 2007, Shieh, Chen and Han proposed a method named "Early Elimination" to alleviate the problem
  - eliminate the top path that is Δ-trellis-level prior to the farthest one among all paths that have been expanded thus far
  - Δ being around three times code constraint length suffices to achieve near optimal performance

An example of Early Elimination



- To attack the problem from different standpoint:
  - assume the memory of the system has an upper limit
  - some paths have to be deleted when the stack is full
  - several path deletion schemes are proposed and examined
- Assume that the decoding process can be launched after the reception of the entire received vector, named **off-line decoding**, a two-pass sequential-type decoding algorithm is also proposed
  - the decoding complexity is much smaller than the stack algorithm with Fano metric, subject to a negligible performance degradation

Algorithm A

- The stack algorithm can be treated as a special case of the "best first" algorithm A for tree search in computer science literature
- Use a cost function μ to guide the search through the graph, $\mu = g + h$
  - g is the accumulated cost
  - h is the heuristic function
- With the help of a proper heuristic function, the decoding complexity can be reduced significantly

Supercode Heuristics of Sikora and Costello

- In 2008, Sikora and Costello proposed a two-pass decoder:
  - backward pass generates proper heuristic function values
  - forward pass sequentially searches the codeword
- Two heuristic functions are designed in their paper

M-algorithm Based Heuristic Estimation Method (MHEM)

- In this thesis, we proposed a new method to generate the heuristic function in backward pass
- Execute the M-algorithm over the backward code tree
- To compare with Sikora and Costello's second heuristic function, memory requirement for the heuristic function is greatly reduced
- Our design outperforms the stack algorithm with Fano metric in decoding complexity for all SNRs simulated

Chapter 2 :

Preliminaries

System models

- A binary (n, k, m) convolutional code $\boldsymbol{C}$ with finite input information sequence of $k \times L$ bits, followed by $k \times m$ zeros to clear the encoder memory
- Denote a codeword of $\boldsymbol{C}$ by $\boldsymbol{v} \triangleq (v_0, v_1, \ldots, v_{N-1})$, $v_j \in \{0, 1\}$, $N \triangleq n(L + m)$
- Transmitted over a binary-input time-discrete memoryless channel with channel output $\boldsymbol{r} \triangleq (r_0, r_1, \ldots, r_{N-1})$

Novel metric with ML performance & low complexity

- Han, Chen, and Wu (2002) derived another metric based on the Wagner rule
- The bit metric is defined as

$$\nu(z_j) \triangleq (y_j \oplus z_j)|\phi_j|.$$

  where ϕ_j is the log-likelihood ratio, y_j is the hard decision and z_j is the encoder output

- This bit metric is **non-negative**; hence, it can be proved that sequential search using this metric yields maximum-likelihood performance
- This metric is adopted if not particularly specified

Algorithm A and Algorithm A*

- The algorithms A and A* are sequential-type graph search algorithms
- separate the decoding metric $\mu(x_\ell)$ associated with node x_ℓ (at level ℓ) into two portions:

$$\mu(x_\ell) = g(x_\ell) + h(x_\ell),$$

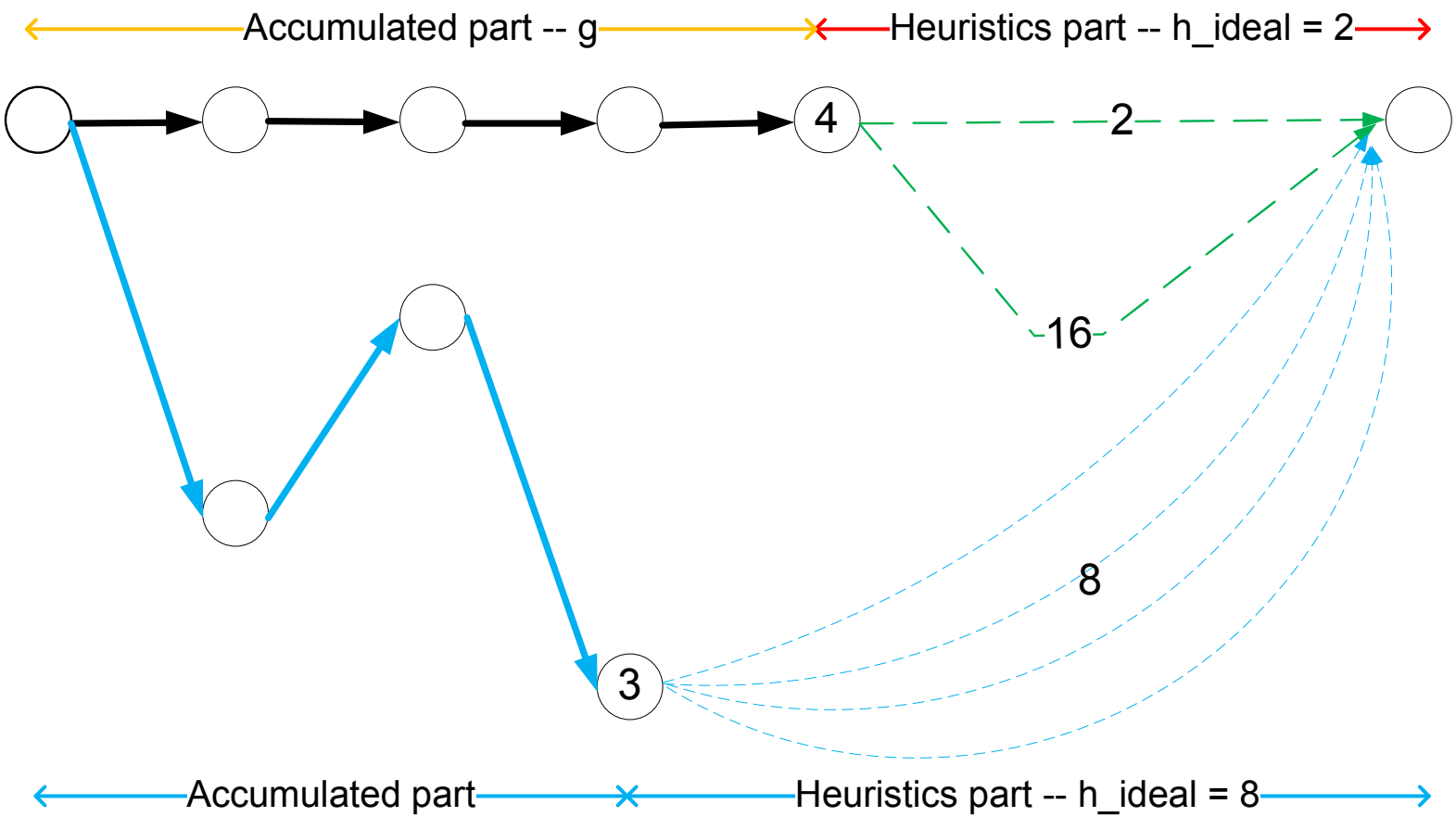
  - $g(x_\ell)$ accumulates the cost from the start node to node x_ℓ
  - $h(x_\ell)$ estimates the remaining cost from node x_ℓ to the terminal node, called the *heuristic function*
- The algorithm A guarantees the finding of the best solution $h_{\text{ideal}}(x_\ell)$ with minimum search complexity. However, such a genie-assisted heuristic function can only be obtained with undue complexity.

- The algorithm A* restricts the selections of heuristic function $h^*(x_\ell)$ to

$$h^*(x_\ell) \leq h_{\text{ideal}}(x_\ell).$$

  - It can be shown that algorithm A* guarantees to find the best path
  - For all heuristic functions satisfying this restriction, a larger heuristic function will expand less number of nodes

An example of Algorithm A



Supercode Heuristics of Sikora and Costello

- In 2008, Sikora and Costello proposed a two-pass decoder:
  - Perform the backward pass not on the actual trellis but on a supercode $\boldsymbol{S}$ to generate proper heuristic function values
  - Forward pass sequentially search the codeword
- Two heuristic functions are designed in their paper
  - **Method 1.** Performing the Viterbi algorithm backwardly on supercode trellis
    - ML performance is guaranteed
    - simpler trellis representation
    - $h_{\boldsymbol{S}}$ satisfies the algorithm A* condition
    - however, the decoding complexity of forward pass is larger than that of the stack algorithm with Fano metric

– **Method 2.** h_S generated in a fashion similar to the backward stage of the well-known BCJR algorithm
  * sub-optmail in performance
  * with negligible performance degradation, the decoding complexity in forward pass is smaller than that of the stack algorithm with Fano metric for all SNRs
  * computational complexity of BCJR-like operations is considerably high
  * high memory requirement to store the heuristic function

Chapter 3 :

Path Deletion Schemes for Limited Stack-Size Sequential-Type Decoding Algorithm

Main Idea

- Assume the stack size for sequential-type decoding algorithm is limited
- Some paths have to be deleted when the stack exceeds its upper limit
- Goal :
  - drop the path that are more unlikely to be the maximum-likelihood one

1. Path Deletion Based on ML Path Metric

- A straightforward path deletion scheme is thus to drop the one with the largest ML path metric
- The decoding procedure with ML metric-based deletion scheme :

  Step 1. Load the stack with the path consisting of the single start node x_0, *whose ML metric* $\mu(x_0) = 0$. *Set* $\ell = 0$.

  Step 2. Extend the top path in the stack to obtain its successor paths. Compute the ML path metric of each successor path. Delete the top path from the stack.

Step 3. Insert the successors into the stack and reorder the stack according to ascending ML path metric values μ. If the stack size exceeds its limit, recursively eliminate the one whose ML metric is the largest until the stack size $\leq$ limit.

Step 4. If the top path in the stack ends at a terminal node in the code tree, the algorithm stops; otherwise go to Step 2.

2. Path Deletion Based on Path Levels

- In 2007, Shieh, Chen and Han proposed a method named "Early Elimination" to reduce the decoding complexity
  - eliminate the top path that is Δ-trellis-level prior to the farthest one among all paths that have been expanded thus far
  - The path with the smallest level among all paths currently in the stack usually tends to accumulate large ML metrics
- For the finite stack size assumption :
  - drop the path with the smallest level when the stack exceeds its limit

3. Path Deletion Based on Fano Metric

- In *Level-based path deletion scheme*, only ending levels are considered to eliminate the path
- Replace the ending level by a metric that includes future route prediction may improve the performance
- Based on Massey's analysis, Fano metric can be regarded as an average of the future branch metrics, which have not been traversed.
- For the finite stack size assumption :
  - drop the path with the smallest Fano metric when the stack exceeds its limit

Summary of the Path Deletion Schmes

- Simulation results show :
  - The Fano metric-based deletion scheme has better performance
  - The level-based deletion scheme has lower complexity

Simulation Results

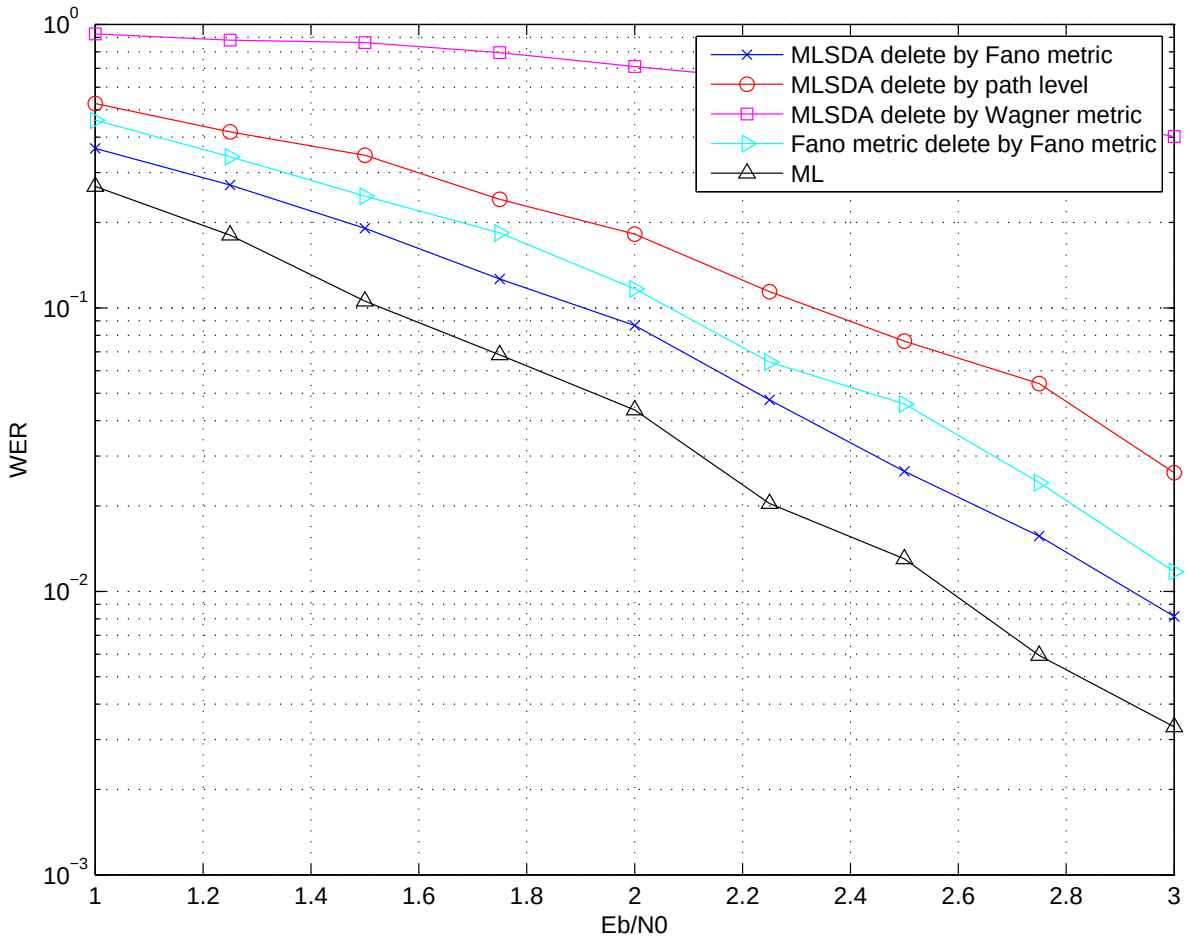


Figure 1: Word error rate (WER) performance of path deletion schemes for (2,1,8) convolutional code with generator polynomial [457,755]. The stack size is $2^6 - 1$, and the information sequence length $L = 100$.

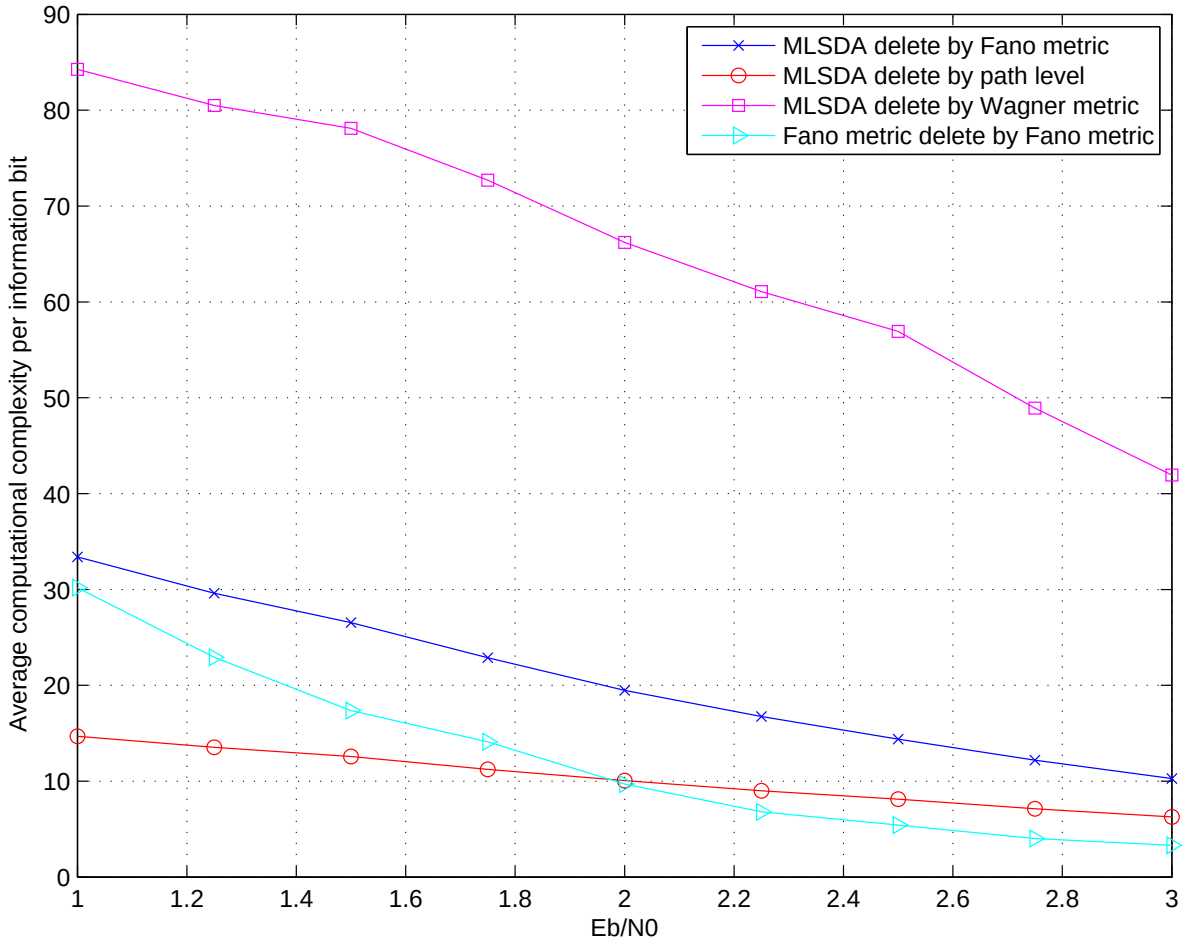


Figure 2: Average computational complexity per information bit of path deletion schemes for (2,1,8) convolutional code with generator polynomial [457,755]. The stack size is $2^6 - 1$, and the information sequence length $L = 100$.

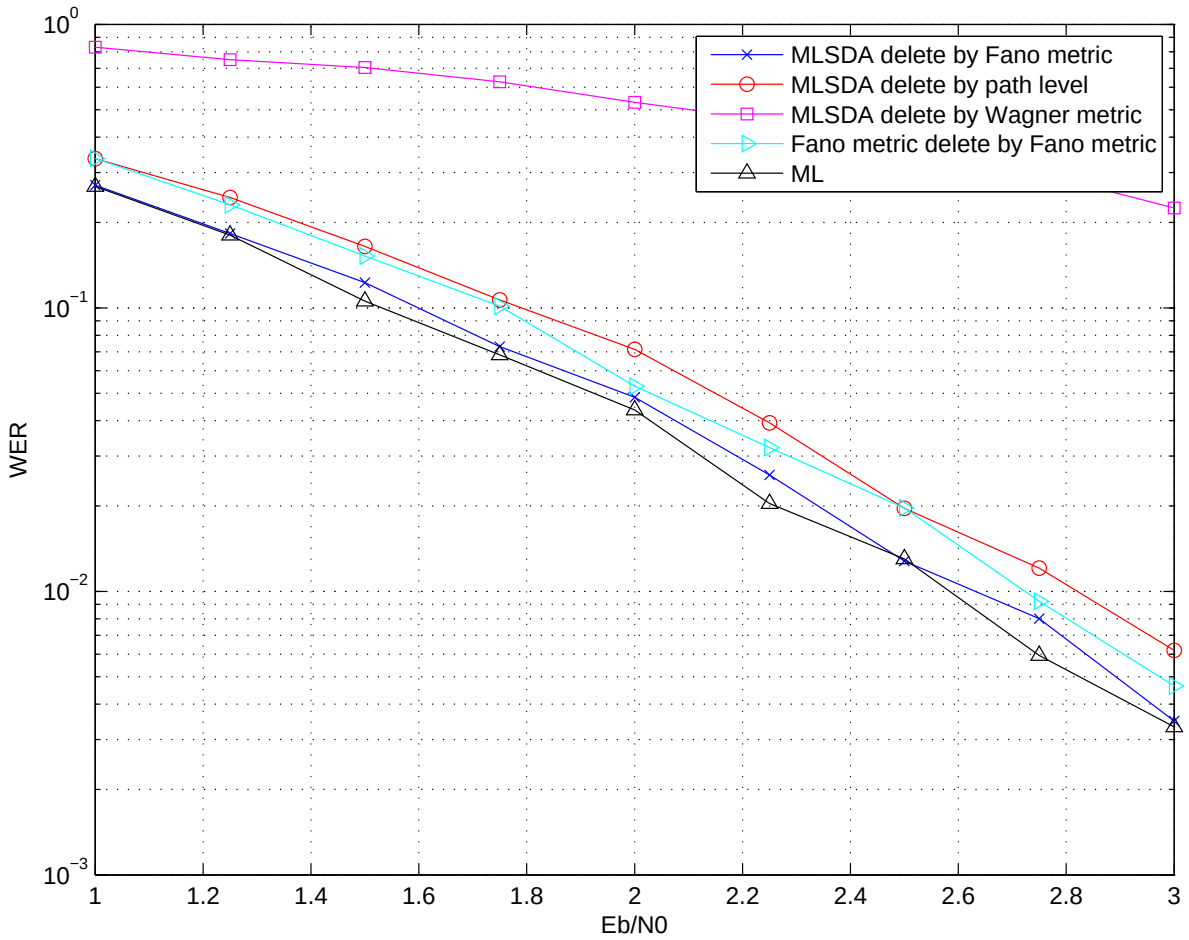


Figure 3: Word error rate (WER) performance of path deletion schemes for (2,1,8) convolutional code with generator polynomial [457,755]. The stack size is $2^8 - 1$, and the information sequence length $L = 100$.

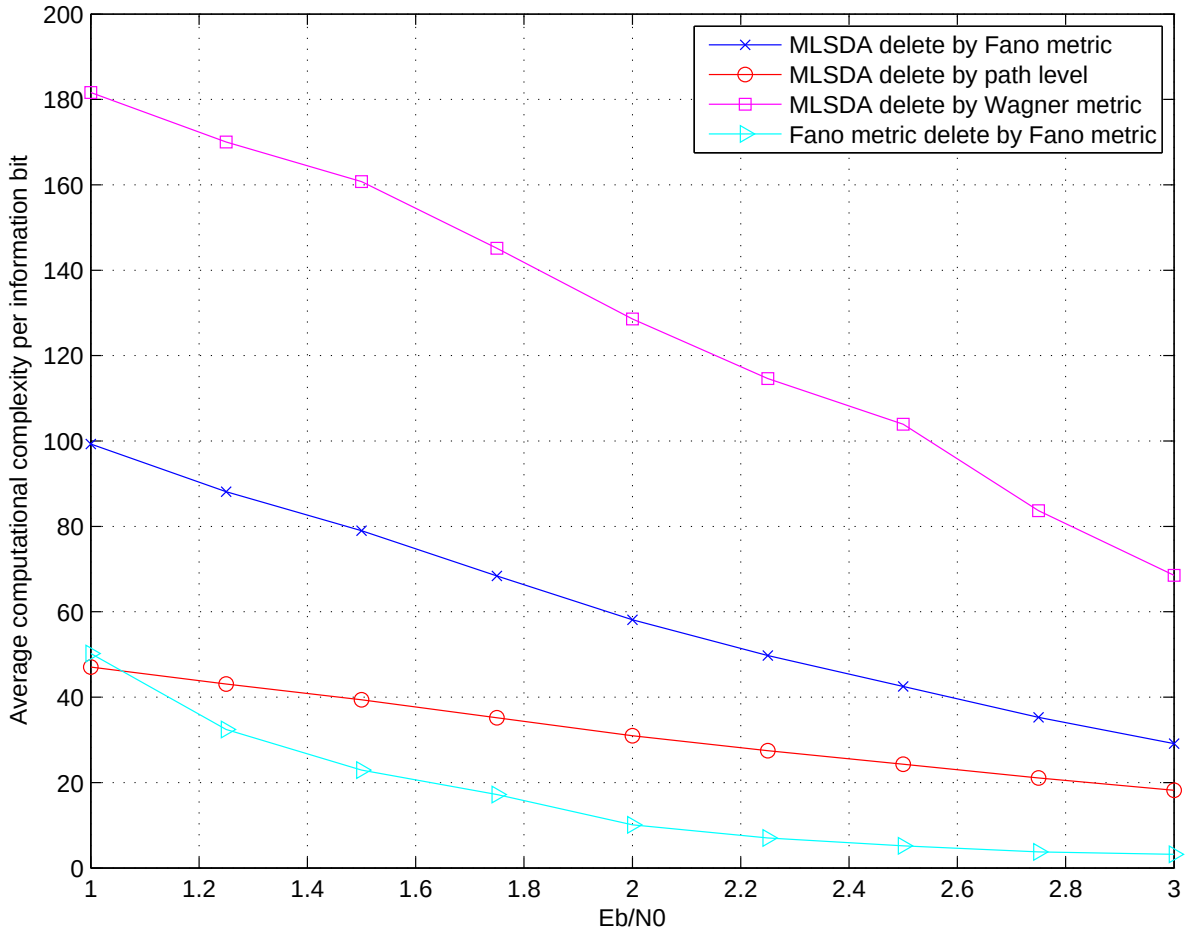


Figure 4: Average computational complexity per information bit of path deletion schemes for (2,1,8) convolutional code with generator polynomial [457,755]. The stack size is $2^8 - 1$, and the information sequence length $L = 100$.

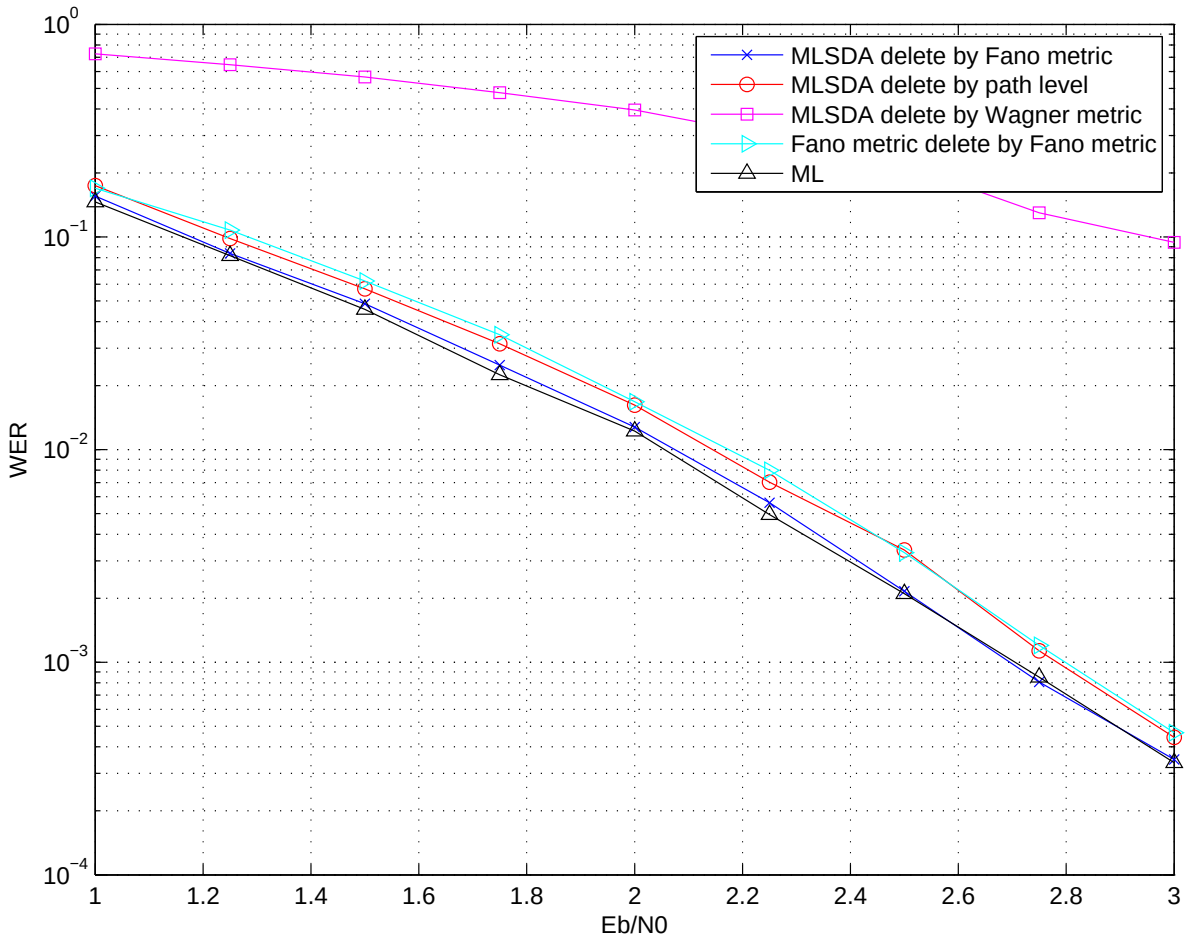


Figure 5: Word error rate (WER) performance of path deletion schemes for (2,1,12) convolutional code with generator polynomial [17663,11271]. The stack size is $2^{12} - 1$, and the information sequence length $L = 100$.

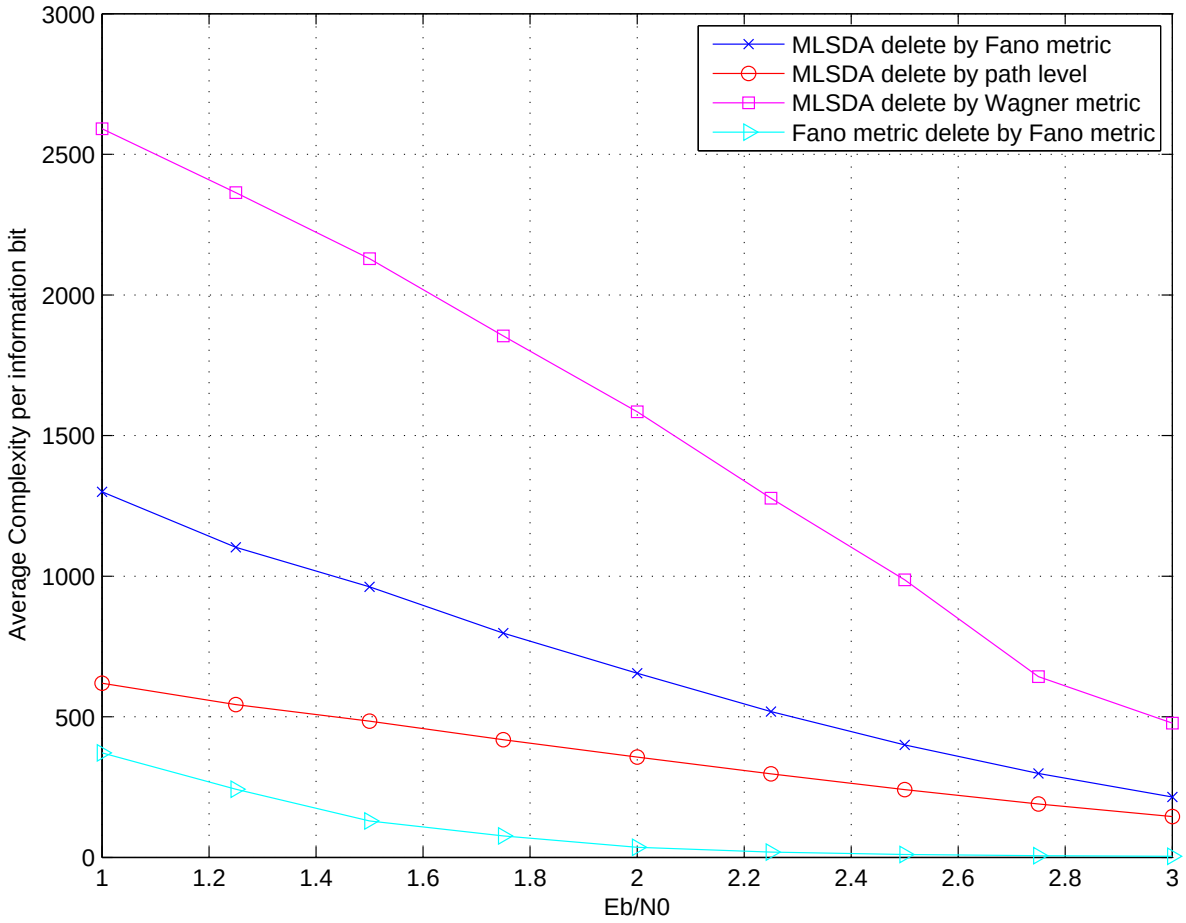


Figure 6: Average computational complexity per information bit of path deletion schemes for (2,1,12) convolutional code with generator polynomial [17663,11271]. The stack size is $2^{12} - 1$, and the information sequence length $L = 100$.

Chapter 4 :

A Novel Two-Pass Sequential-Type Decoding Algorithm

Heuristics Analysis

- For AWGN channel, the ML decision upon the reception of a received vector $\boldsymbol{r} \triangleq (r_0, r_1, \ldots, r_{N-1})$ is given by:

$$\begin{aligned}
\hat{\boldsymbol{v}} &= \arg\min_{\boldsymbol{v}\in\boldsymbol{C}} \sum_{i=0}^{N-1} (-r_i v_i) \\
&= \arg\min_{\boldsymbol{v}\in\boldsymbol{C}} \sum_{i=0}^{N-1} |r_i v_i| - \sum_{i=0}^{N-1} r_i v_i \\
&= \arg\min_{\boldsymbol{v}\in\boldsymbol{C}} \sum_{i=0}^{N-1} (|r_i| - r_i v_i).
\end{aligned}$$

- To maintain the optimality through sequential decoding, the heuristic function should satisfy

$$\mu(\boldsymbol{v}_\ell) = g(\boldsymbol{v}_\ell) + h(\boldsymbol{v}_\ell) \leq \min_{\boldsymbol{v}'_N=(v_0,v_1,\ldots,v_{\ell-1},v'_\ell,v'_{\ell+1},\ldots,v'_{N-1})\in\boldsymbol{C}} [g(\boldsymbol{v}'_N) + h(\boldsymbol{v}'_N)]$$

- Since $h(\boldsymbol{v}'_N) = 0$

$$h(\boldsymbol{v}_\ell) \leq \min_{\boldsymbol{v}'_N=(v_0,v_1,\ldots,v_{\ell-1},v'_\ell,v'_{\ell+1},\ldots,v'_{N-1})\in\boldsymbol{C}} \left[\sum_{i=\ell}^{N-1}(|r_i| - r_i v'_i)\right].$$

- Without the knowledge of the code structure, we obtain:

$$h(\boldsymbol{v}_\ell) \leq \min_{(v'_\ell,v'_{\ell+1},\ldots,v'_{N-1})\in\{\pm 1\}^{N-\ell}} \left[\sum_{i=\ell}^{N-1}(|r_i| - r_i v'_i)\right] = 0.$$

- **Without the knowledge of the code structure, the largest heuristic function is the zero-heuristics function**

- On-the-fly decoding procedure may not be able to improve the complexity without performance loss
- How about off-line decoding algorithms?

A Novel Two-Pass Sequential-Type Decoding Algorithm

- Decode through off-line decoding procedure
- Designed based on the well-known algorithm A
- Alleviate the decoding burden in computational complexity
- In Comparison with Sikora and Costello's decoder, the proposed algorithm improves the computational complexity and memory storage of heuristic function

- Two-pass decoding idea which is similar to Sikora and Costello's decoder
- A new heuristic function is designed
- The proposed algorithm performed on the forward and backward code trees rather than the supercode

M-Algorithm-Based Heuristics Estimate

- Initially, set the heuristic function value of the single terminal node on the backward code tree as $h_{L+m} = 0$
- Then, we execute the M-algorithm from the single terminal node at level $L + m$ back to the 2^L start nodes at level 0
- The heuristic function h_ℓ is the minimum path metric value among the M survivor path metrics at level ℓ
  - The optimal performance is actually guaranteed for $M = 2^L$
  - Near optimal performance can be achieved for small to moderate M

Example of the backward pass

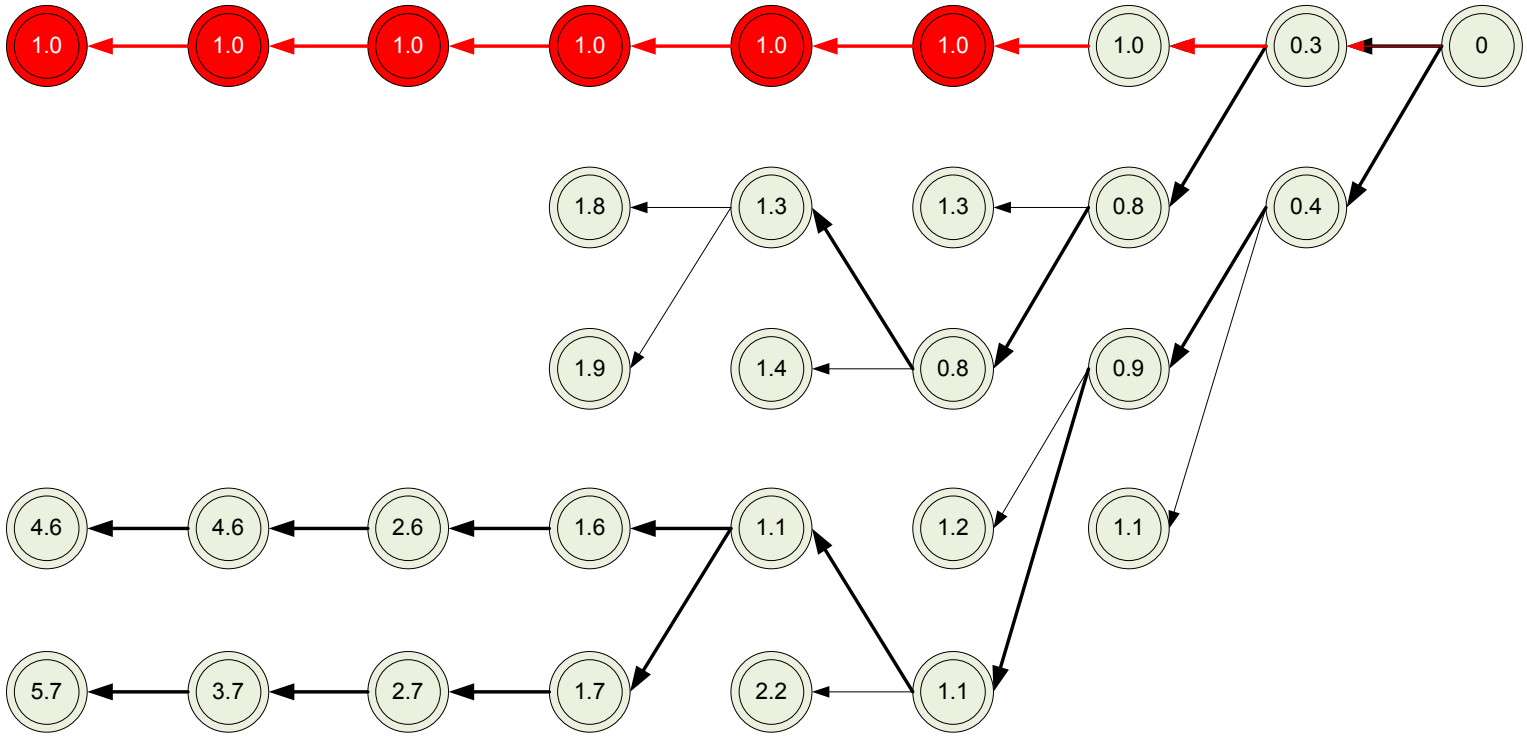


Figure 7: An example of M-algorithm-based heuristic function generation for $(2, 1, 3)$ convolutional code of length $N = n(L + m) = 2(5 + 3) = 16$. The maximum-likelihood code word path is marked in red.

Example of the heuristic function values

Table 1: The heuristic function values resulted from Figure 7.

Level ℓ	0	1	2	3	4	5	6	7	8
$h_\ell(M=2)$	4.6	4.6	2.6	1.6	1.1	0.8	0.8	0.3	0.0
$h_\ell(M=2^L)$	1.0	1.0	1.0	1.0	1.0	0.8	0.8	0.3	0.0

M-Algorithm-Based Heuristics Estimate

- The forward pass performs the usual stack algorithm with metric μ
- Remove the path with worst metric μ when stack size exceeds the limit

Decoding Procedure of the Backward Pass

Step 1. Set $\ell = L + m$ *and* $h_\ell = 0$.

Step 2. Backwardly extend all successors of the survivor paths at level ℓ. *Recursively calculate the path metrics of the successor paths. Set* $h_{\ell-1}$ *to be the minimum path metric among all successor paths.*

Step 3. Retain the M *successor paths with the smallest path metrics as the new survivor paths, and let* $\ell = \ell - 1$.

Step 4. If $\ell = 0$, *the backward pass stops;otherwise go to Step 2.*

Decoding Procedure of the Forward Pass

Step 1. Load the stack with the path consisting of the single start node x_0, whose metric $\mu(x_0) = g(x_0) + h_0 = h_0$. Set $\ell = 0$.

Step 2. Extend the top path in the stack to obtain its successor paths. Compute the path metric of each successor path. Delete the top path from the stack.

Step 3. Insert the successors into the stack and reorder the stack according to ascending path metric values μ. If the stack size exceeds its limit, recursively eliminate the one whose μ metric is the largest until the stack size $\leq$ limit.

Step 4. If the top path in the stack ends at a terminal node in the code tree, the algorithm stops; otherwise go to Step 2.

Simulation Results

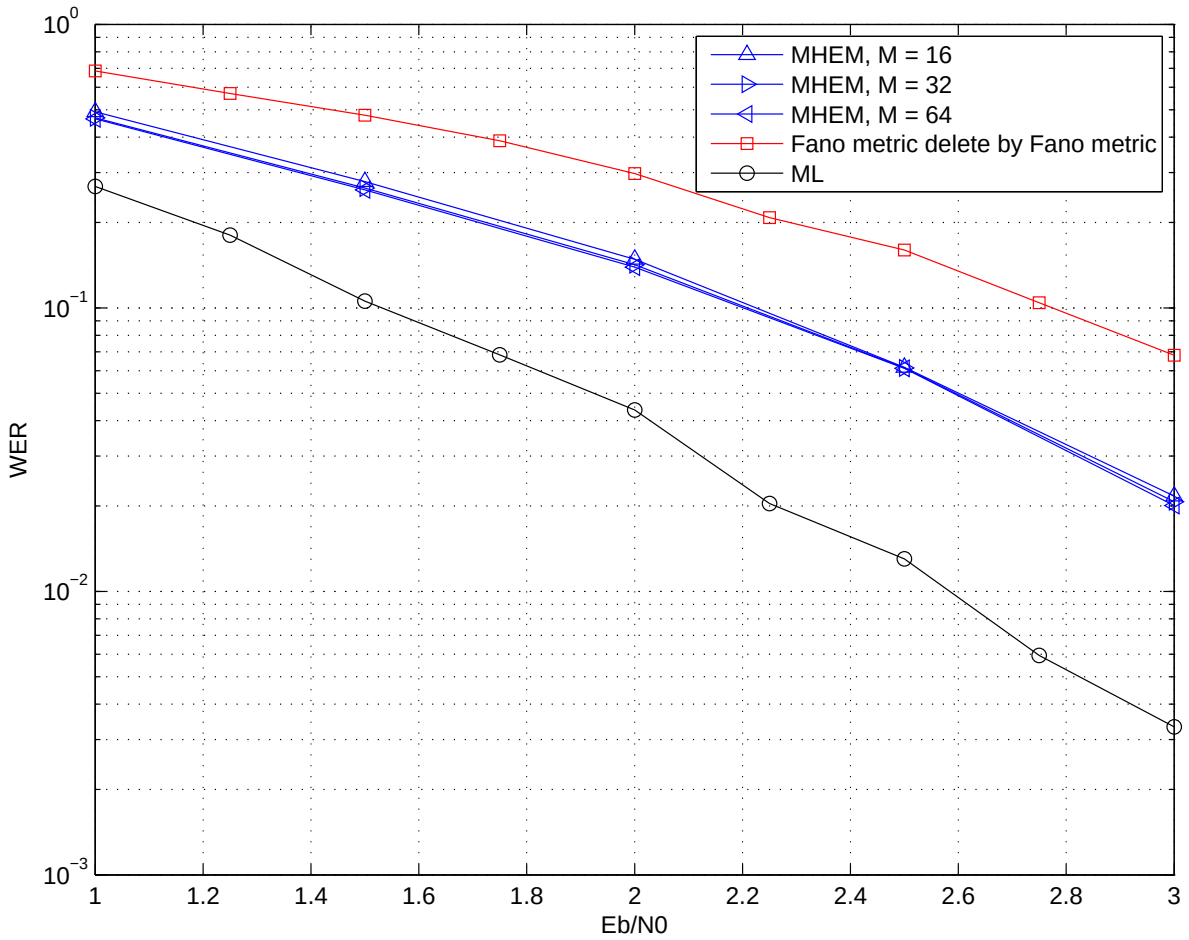


Figure 8: Word error rate (WER) performance of MHEM-enhanced two-pass decoder for (2,1,8) convolutional code with generator polynomial [457,755]. The stack size is $= 2^4 - 1$, and the information sequence length $L = 100$.

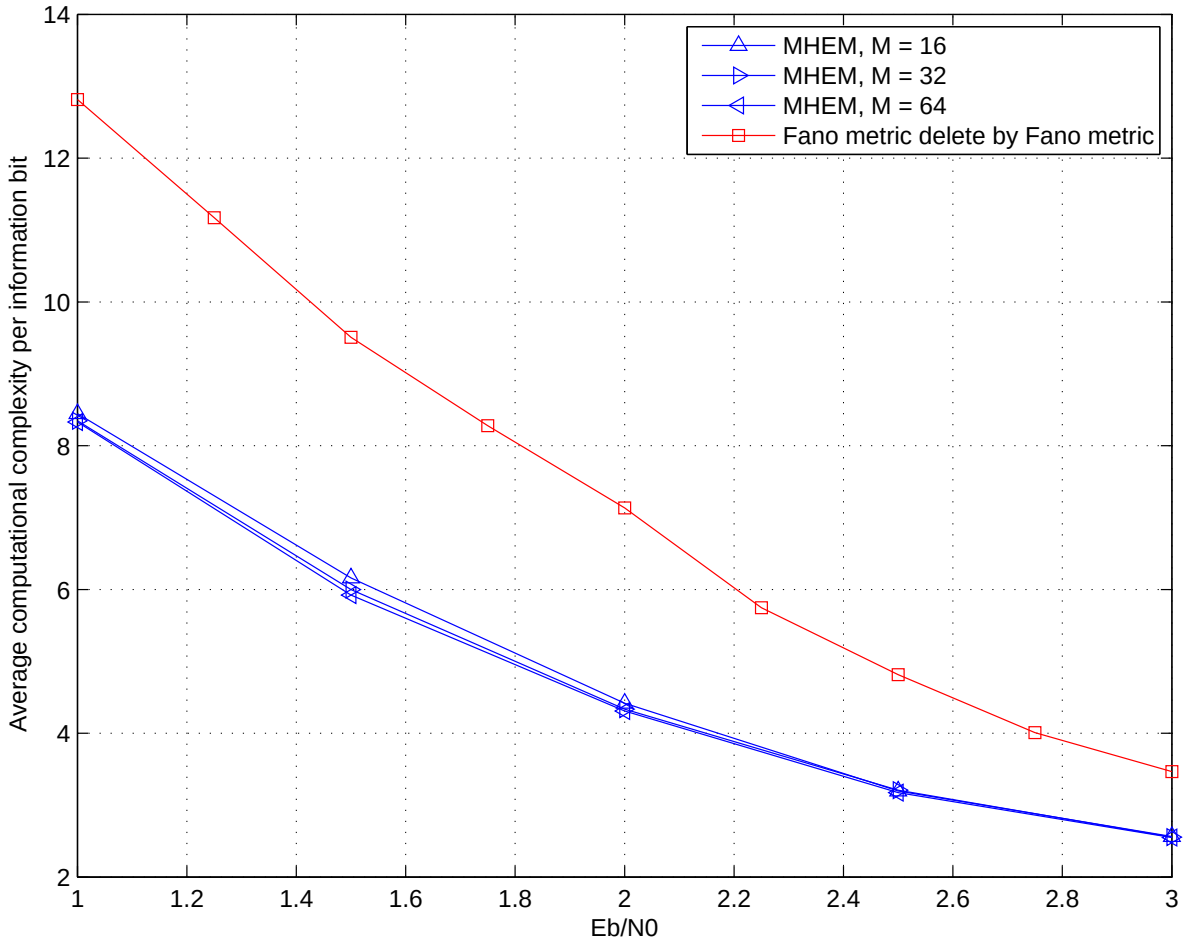


Figure 9: Average computational complexity per information bit of MHEM-enhanced two-pass decoder for (2,1,8) convolutional code with generator polynomial [457,755]. The stack size is $2^4 - 1$, and the information sequence length $L = 100$.

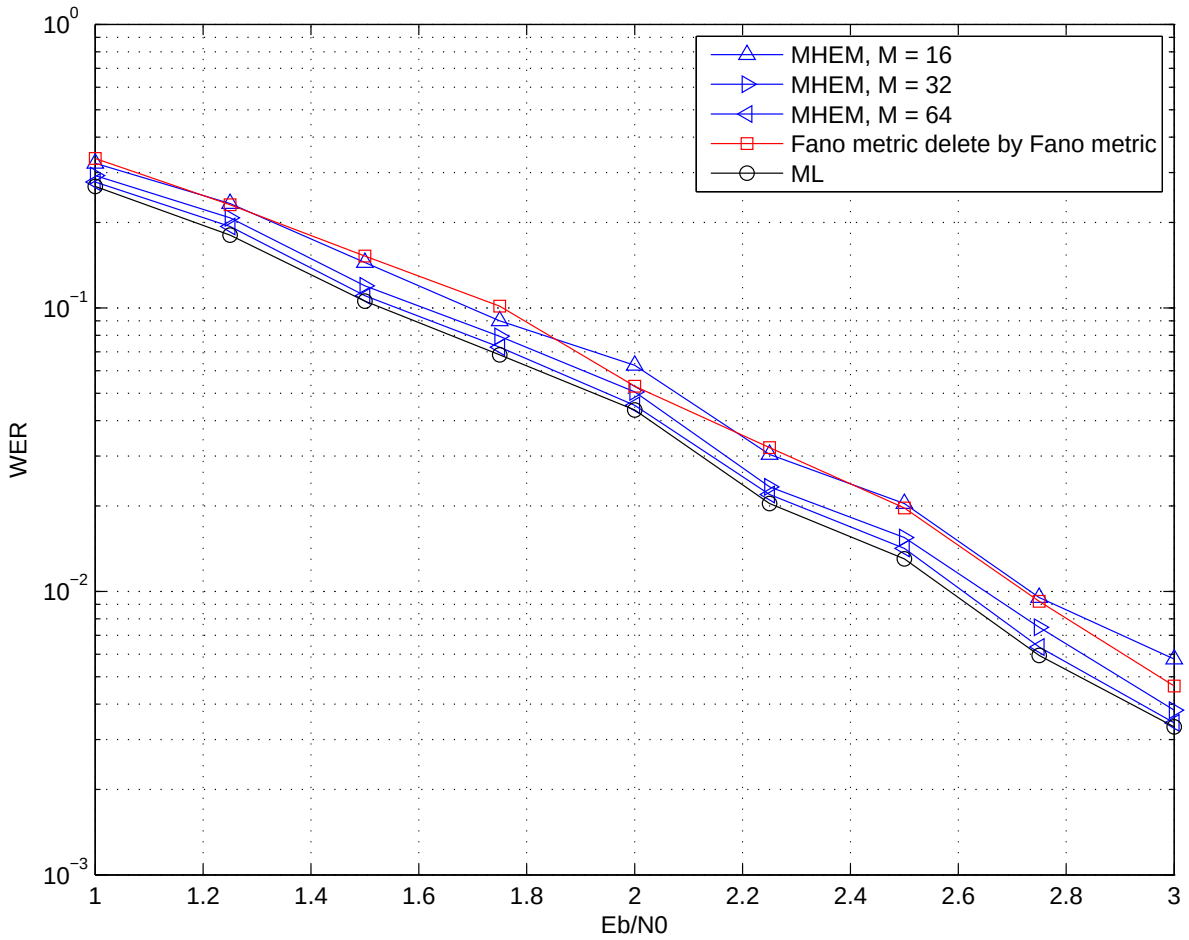


Figure 10: Word error rate (WER) performance of MHEM-enhanced two-pass decoder for (2,1,8) convolutional code with generator polynomial [457,755]. The stack size is $= 2^8 - 1$, and the information sequence length $L = 100$.

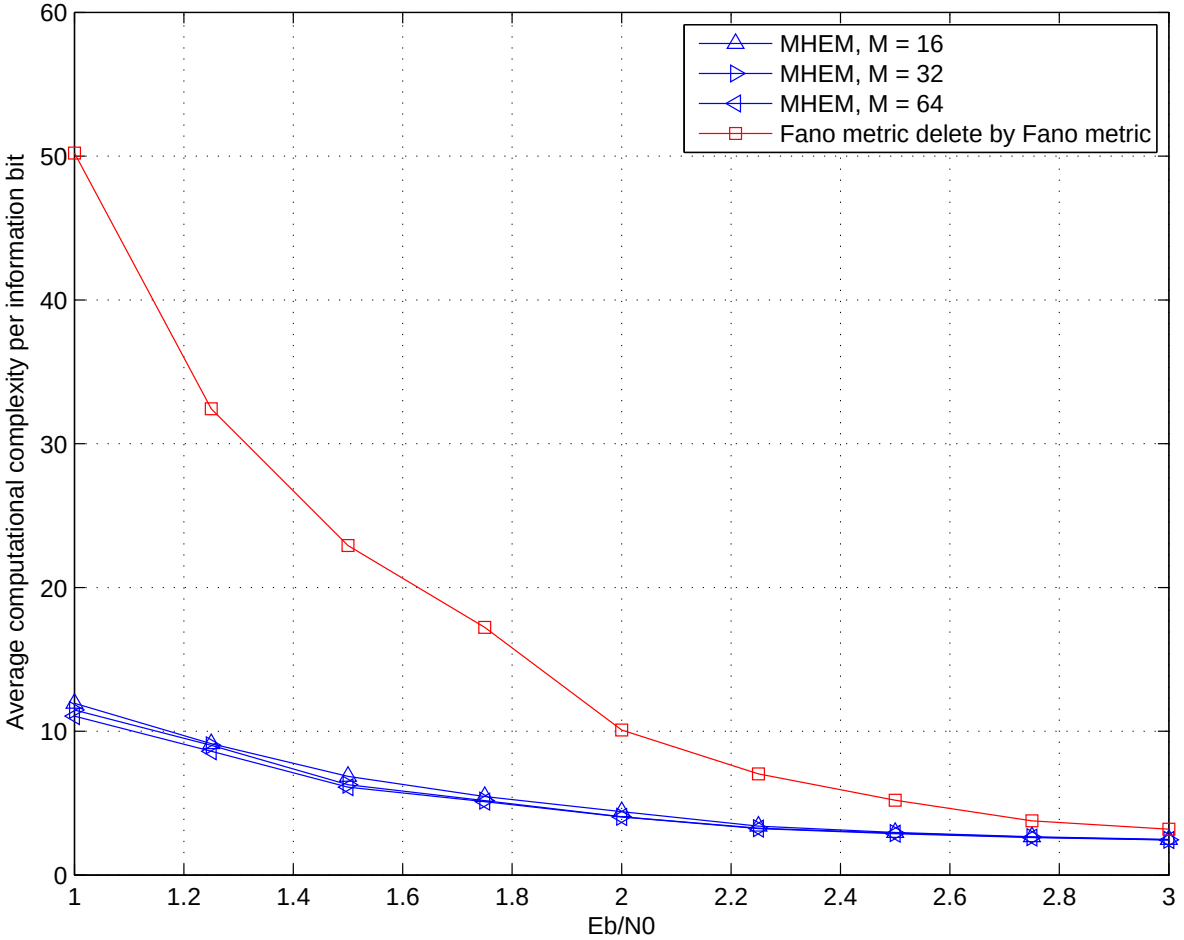


Figure 11: Average computational complexity per information bit of MHEM-enhanced two-pass decoder for (2,1,8) convolutional code with generator polynomial [457,755]. The stack size is $2^8 - 1$, and the information sequence length $L = 100$.

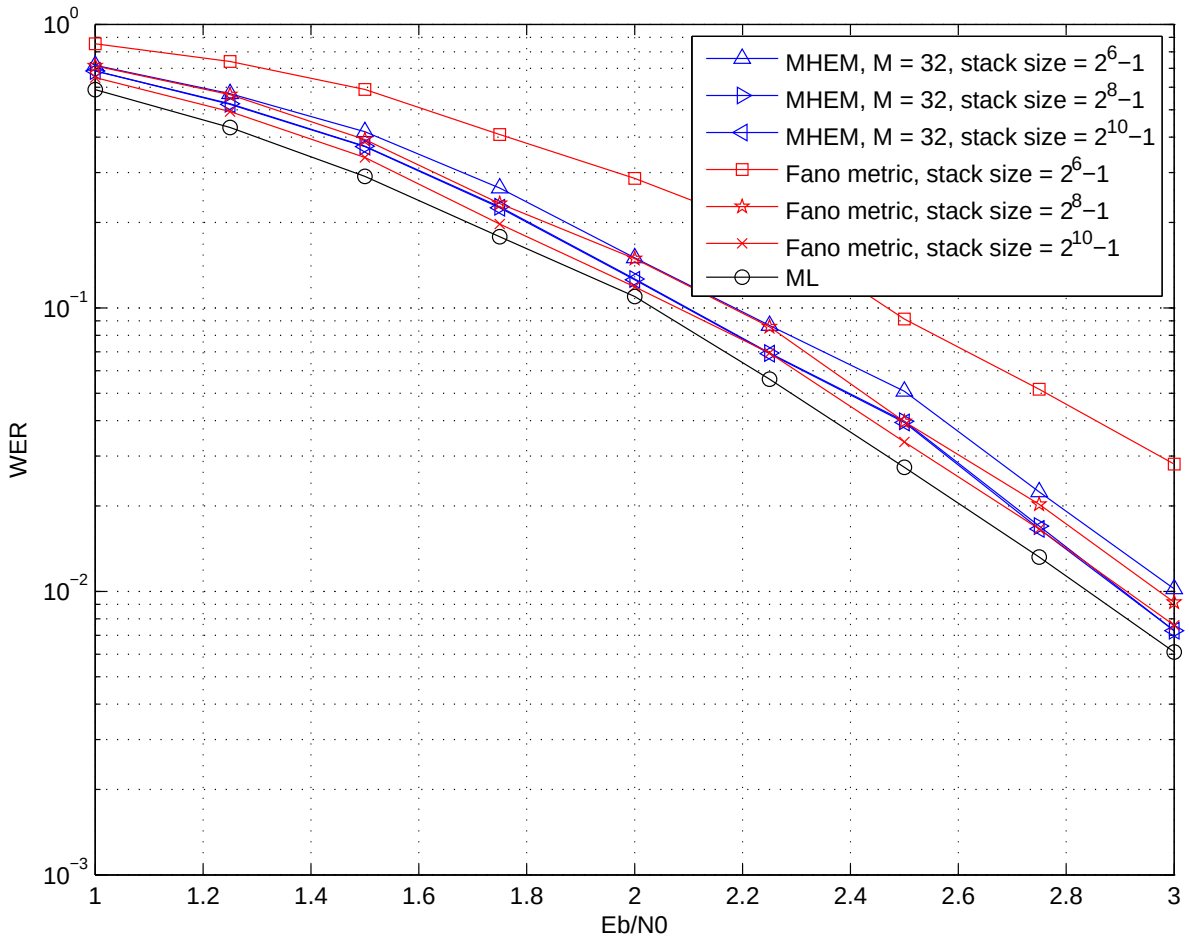


Figure 12: Word error rate (WER) performance of MHEM-enhanced two pass decoder for (2,1,8) convolutional code with generator polynomial [457,755]. Here, $M = 32$ and the information sequence length $L = 400$.

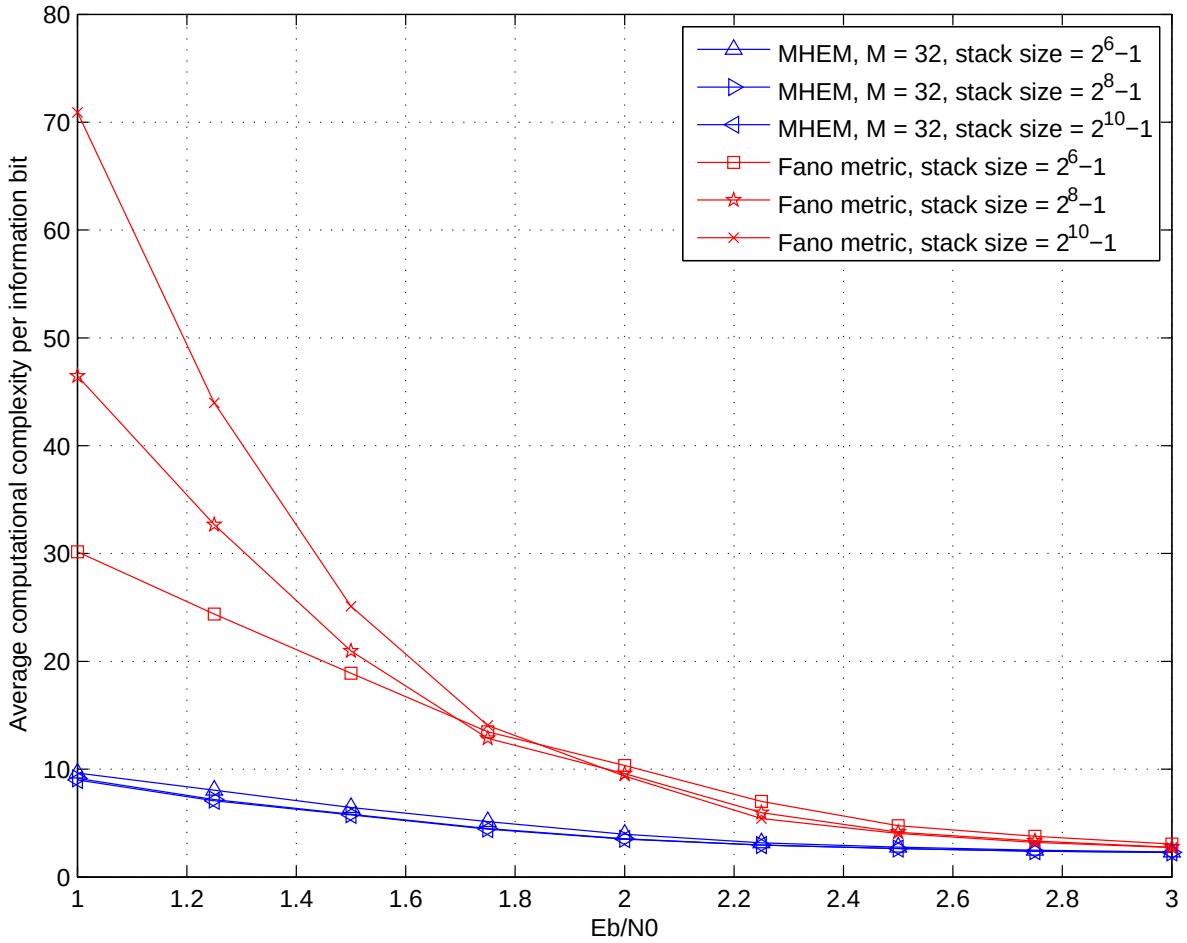


Figure 13: Average computational complexity per information bit of MHEM-enhanced two-pass decoder for (2,1,8) convolutional code with generator polynomial [457,755]. Here, $M = 32$, and the information sequence length $L = 400$.

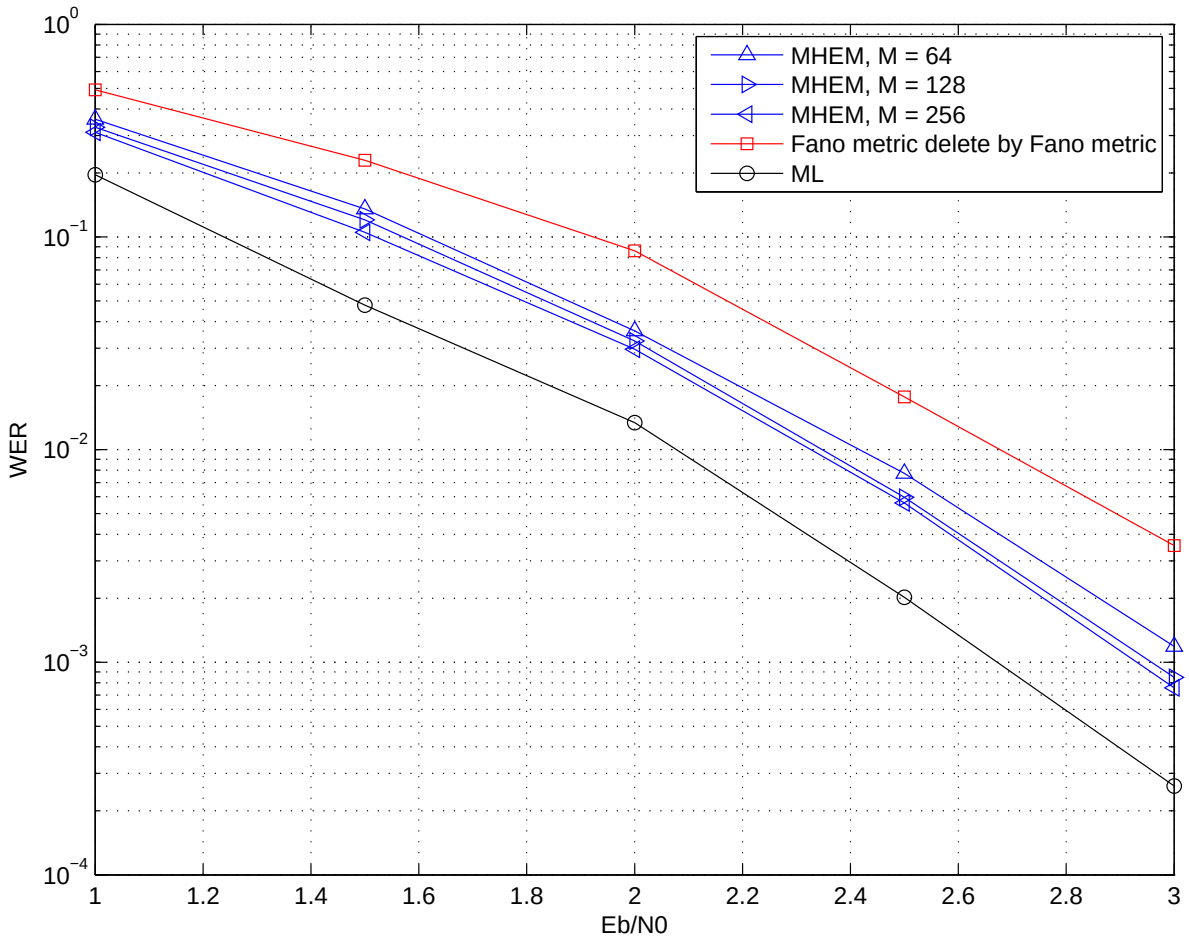


Figure 14: Word error rate (WER) performance of MHEM-enhanced two-pass decoder for (2,1,12) convolutional code with generator polynomial [17663,11271]. The stack size is $2^8 - 1$, and the information sequence length $L = 200$.

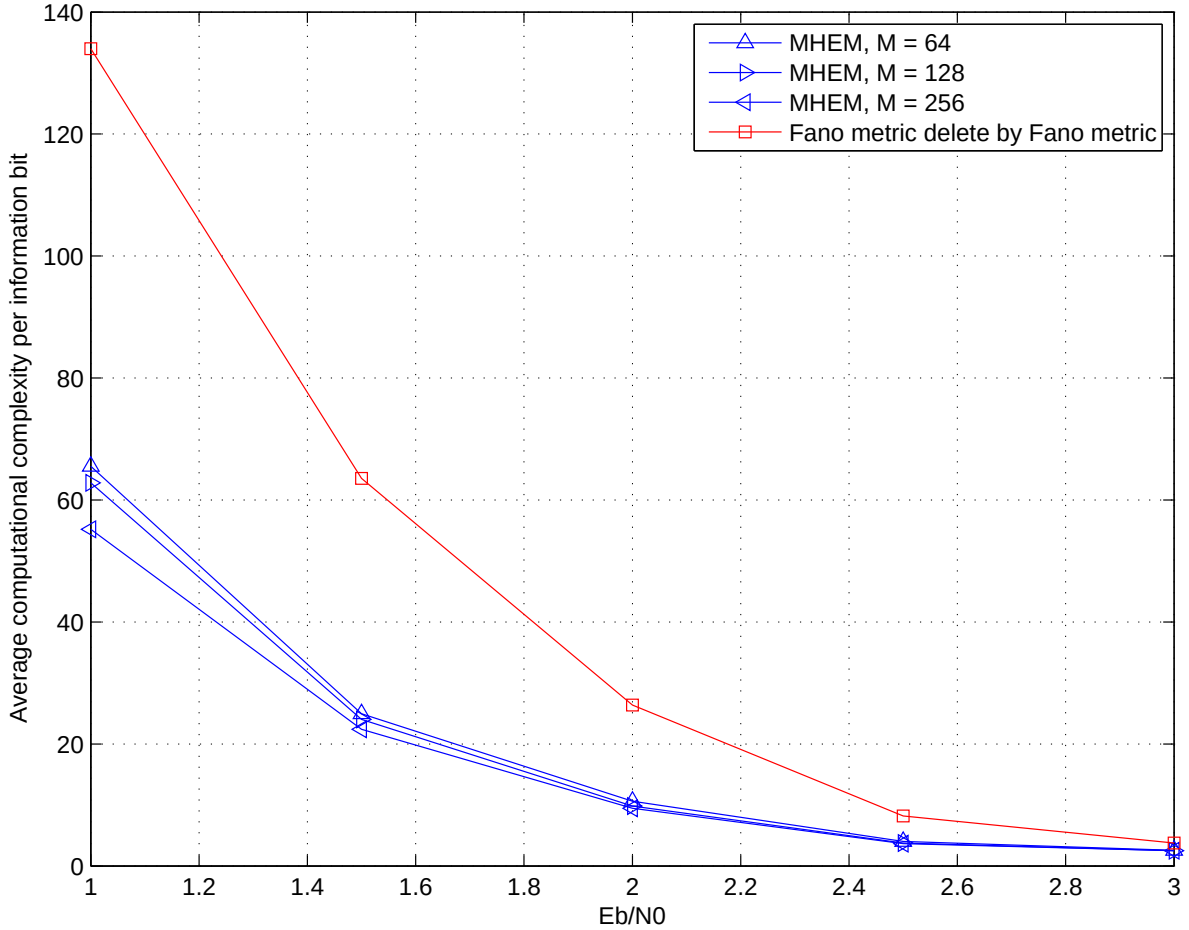


Figure 15: Average computational complexity per information bit of MHEM-enhanced two-pass decoder for (2,1,12) convolutional code with generator polynomial [17663,11271]. The stack size is $2^8 - 1$, and the information sequence length $L = 200$.

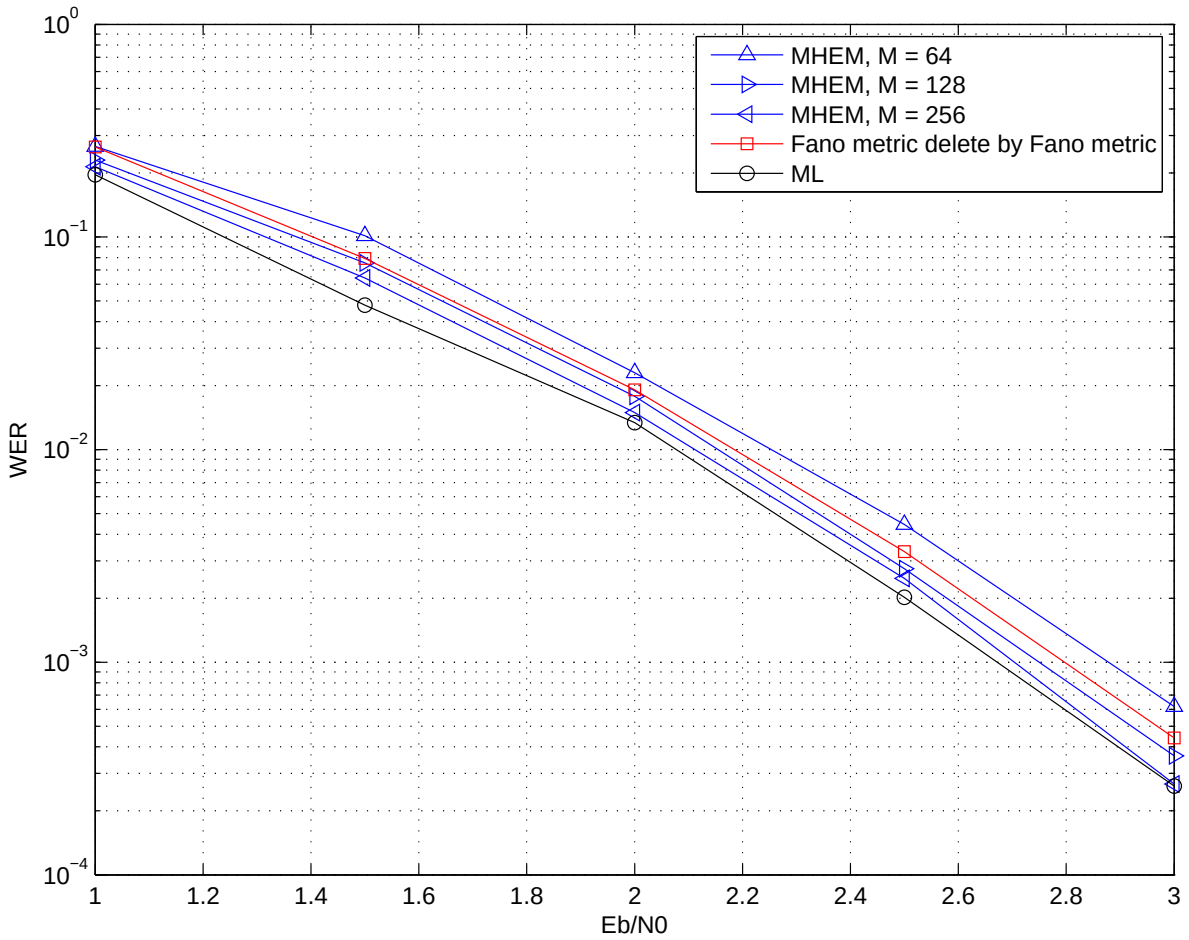


Figure 16: Word error rate (WER) performance of MHEM-enhanced two-pass decoder for (2,1,12) convolutional code with generator polynomial [17663,11271]. The stack size is $2^{12} - 1$, and the information sequence length $L = 200$.

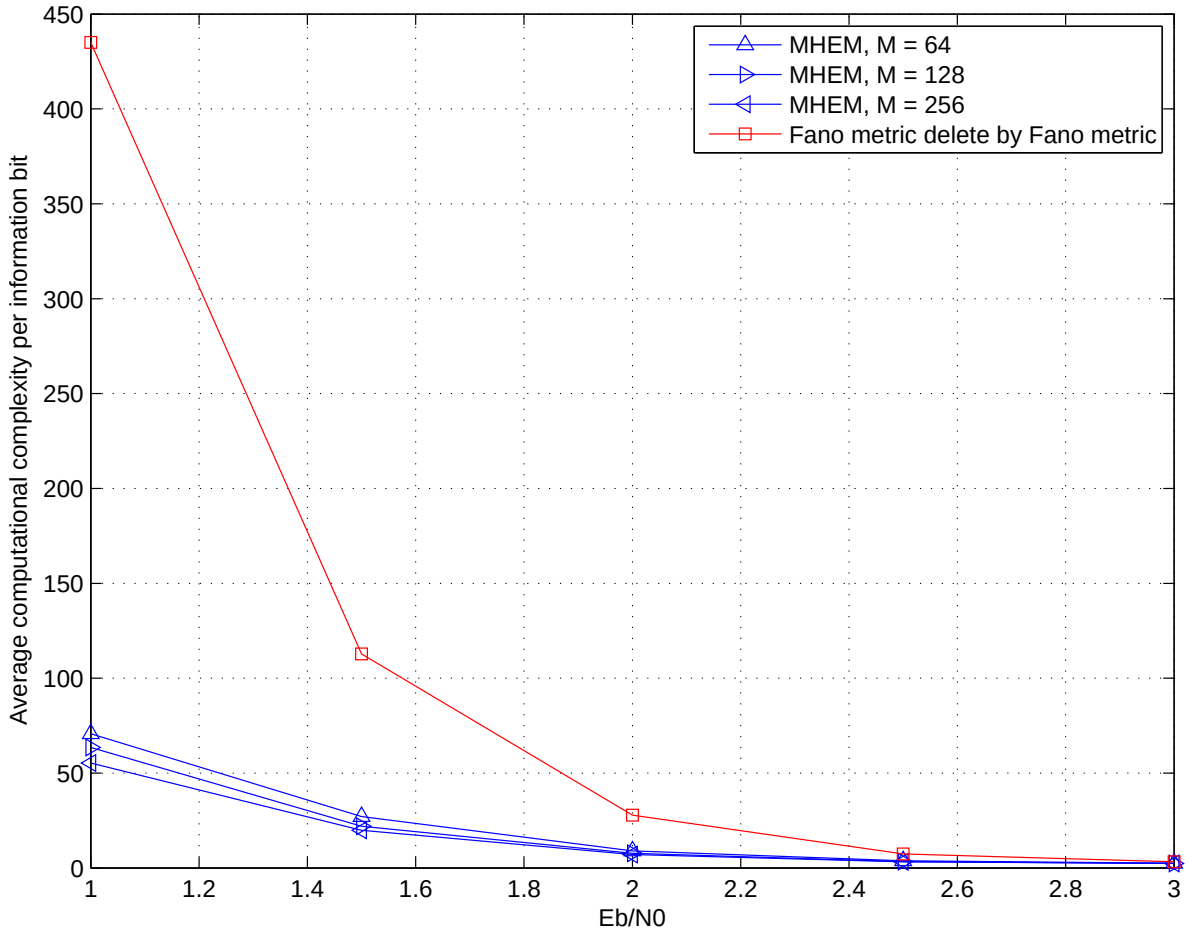


Figure 17: Average computational complexity per information bit of MHEM-enhanced two-pass decoder for (2,1,12) convolutional code with generator polynomial [17663,11271]. The stack size is $2^{12} - 1$, and the information sequence length $L = 200$.

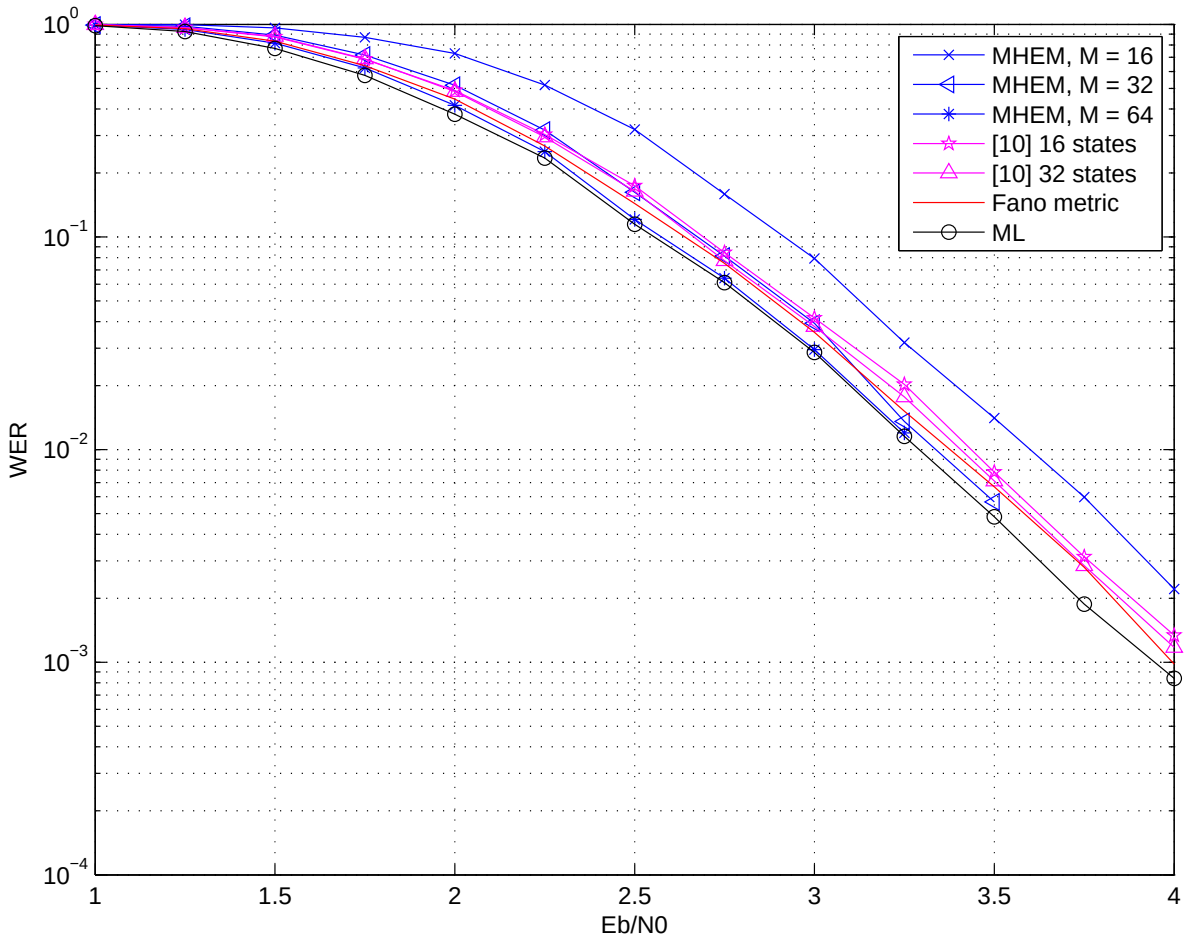


Figure 18: Word error rate (WER) performance of MHEM-enhanced two-pass decoder, and low-complexity suboptimal decoding algorithm in Sikora and Costello's work for (2,1,8) convolutional code with generator polynomial [457,755]. The stack size is $2^{18} - 1$, and the information sequence length $L = 2048$.

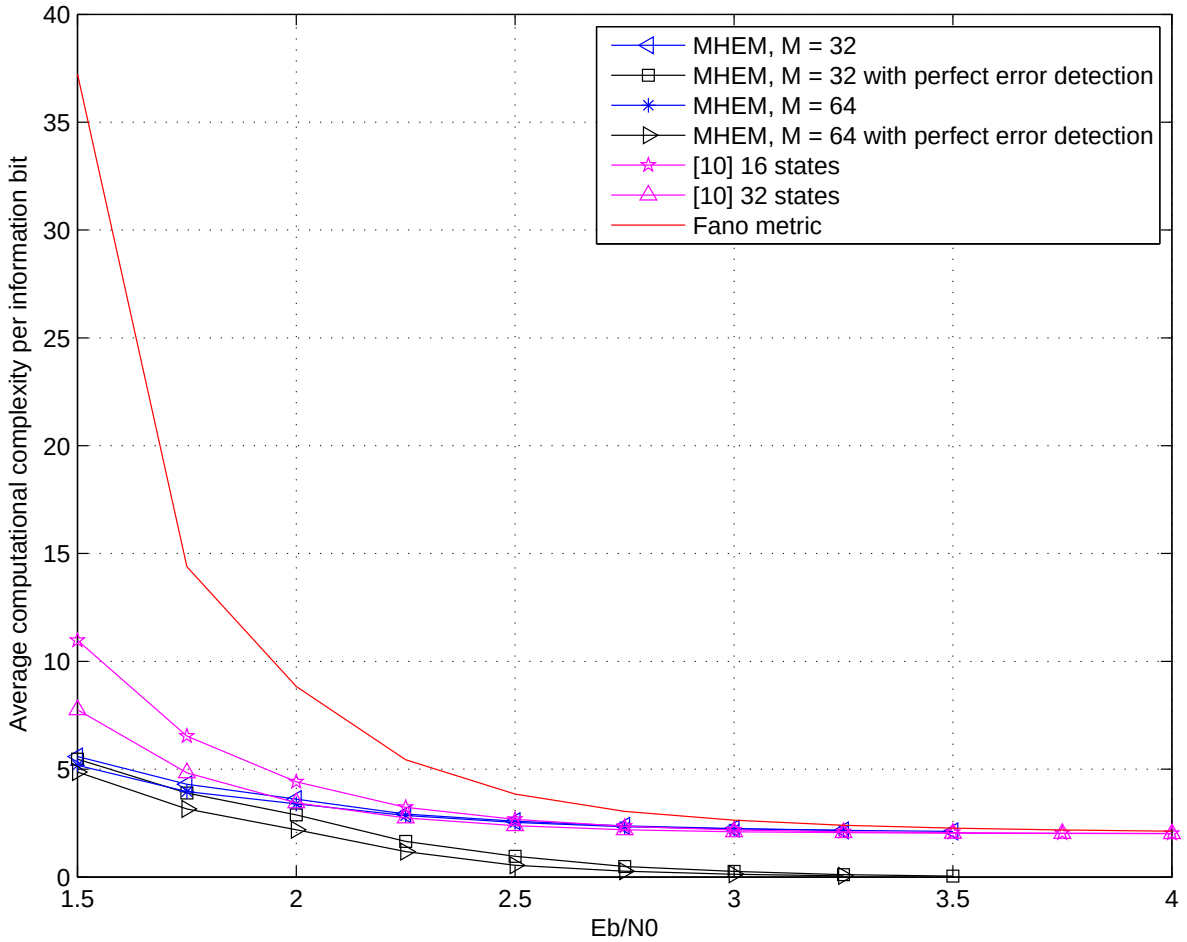


Figure 19: Average computational complexity per information bit of MHEM-enhanced two-pass decoder, and low-complexity suboptimal decoding algorithm in Sikora and Costello's work for (2,1,8) convolutional code with generator polynomial [457,755]. The stack size is $2^{18} - 1$, and the information sequence length $L = 400$.

Chapter 6 :

Concluding Remark and Future work

Concluding Remark and Future work

- For finite stack size consideration, several path deletion schemes are proposed
  - The Fano metric-based deletion scheme has better performance
  - The level-based deletion scheme has lower complexity
- Consider of the possibility of off-line decoding, a two-pass decoding structure is proposed
  - The computational complexity of the forward pass is not only better the stack algorithm with Fano metric, but also is smaller than that of the two-pass supercode decoder
- The appropriate M is still obtained through simulations

Thanks for listening