LE/EECS 4214 Fall 2015

# L13: Sequence Detection

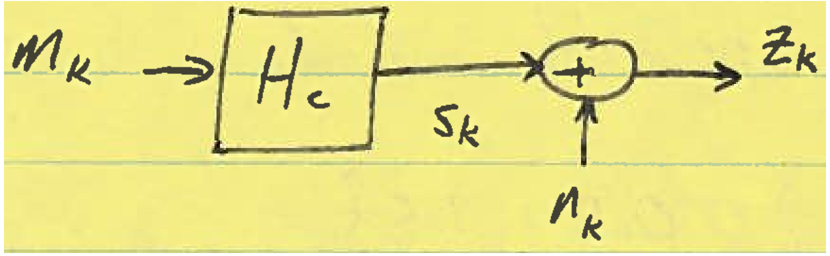
Sebastian Magierowski
York University

# Outline

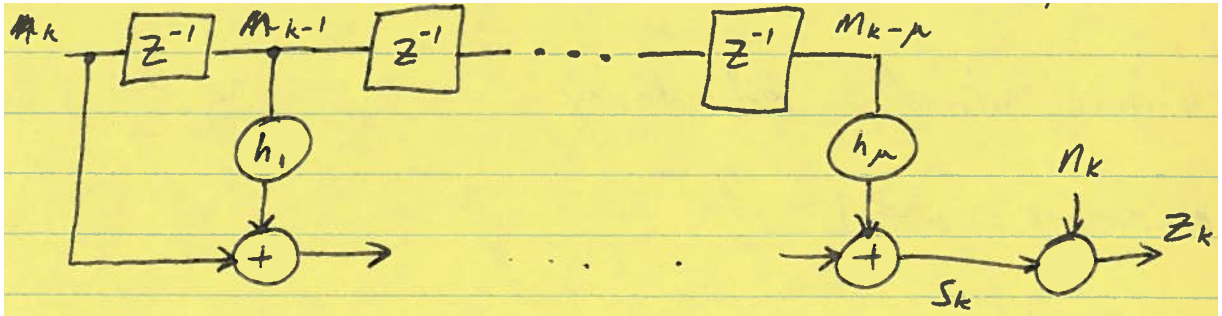
- Channel response in general
- Finite state machines
- Trellis diagram
- Path detection
- Viterbi algorithm

# 13.1 Channel Response

- Generally, channel response is…



- …system with memory



  - s = h * m (convolution)

$$s_k = \sum_{l=0}^{\mu} h_l m_{k-l} = \sum_{l=0}^{\mu} h_{k-l} m_l$$

# Channel Response Generalities

- Effectively output depends on…
  - current input
  - current state

$$s_k = f(m_k, \bar{\psi}_k)$$

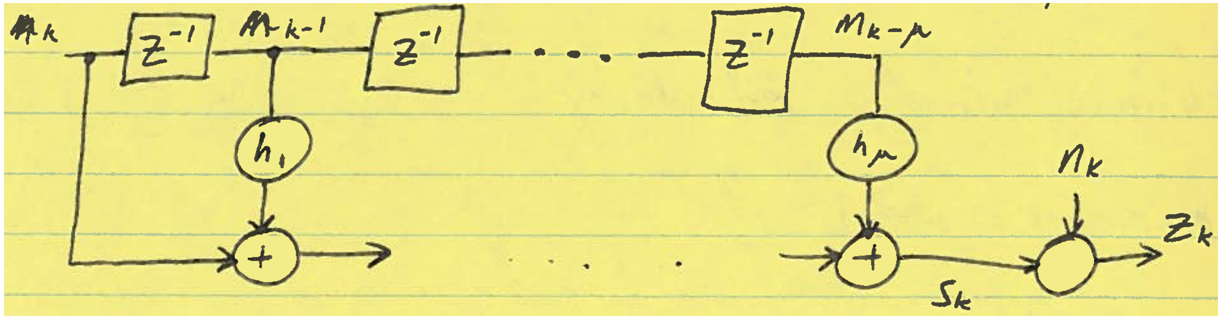
current input

current state of channel

$$\bar{\psi}_k = [m_{k-1}, m_{k-2}, \ldots m_{k-\mu}]$$

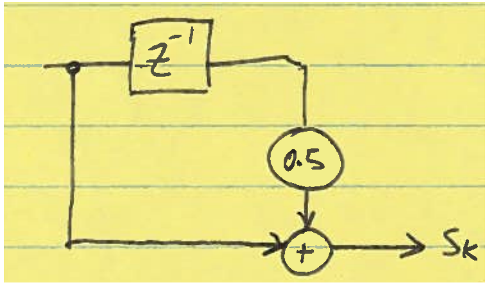
- Or similarly…
  - current state
  - upcoming state

$s_k = g(\bar{\psi}_k, \bar{\psi}_{k+1})$ ← o/p is fn. of a state transition



# 2-State Channel Example

- What comes out of…
  - Input alphabet = {0,1}, M = 2



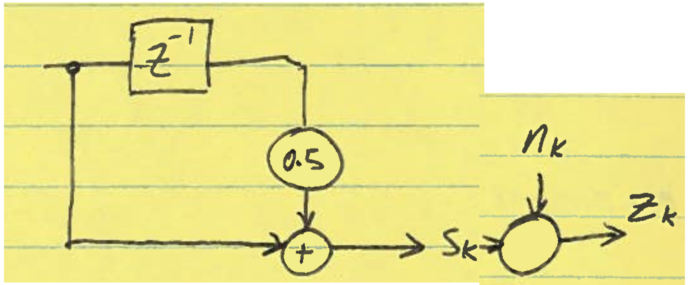
$$s_k = \left\{ s_k^{00},\ s_k^{01},\ s_k^{10},\ s_k^{\overset{m_k \searrow\ \swarrow \psi_k}{11}} \right\} = \{0, 0.5, 1, 1.5\}$$

- 1-bit of info comes in
  - 4-levels come out
- Channel introduces ISI
  - memory, μ = 1
  - in CODING this is done on purpose (REDUNDANCY in ENCODER)

# A Key Question

- What if you observed some output sequence…
  - $z_k$ = 0.2, 0.6, 0.9, 0.1
    - This has both ISI and noise on it
      - Possible mk = {0,1}
      - Possible sk = {0, 0.5,1,1.5}

$m_k$

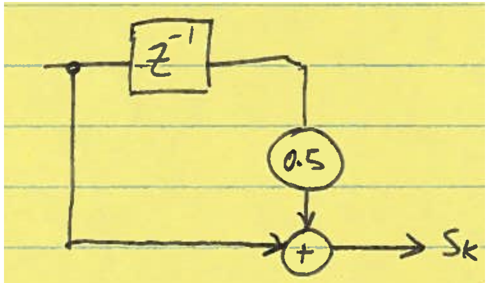


$z_k$

- …what was the original $m_k$ that generated this?
  - Normally your EQ + DET figure this out
  - We'll look at a means of doing it in one block

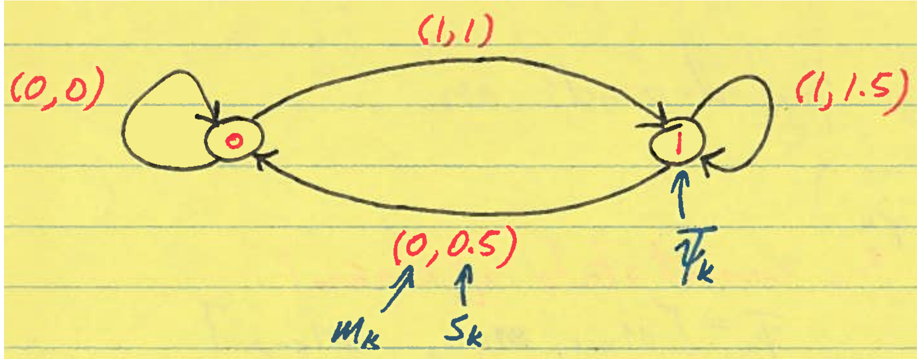
# 13.2 Finite State Machines

- State transition diagrams
  - generic representation of channel



$\{0, 0.5, 1, 1.5\}$

=



$$s_k = f(m_k, \bar{\psi}_k)$$

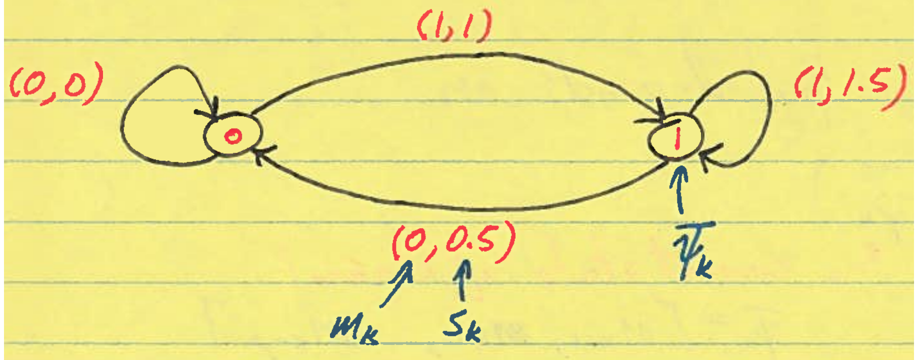
current input

current state of channel

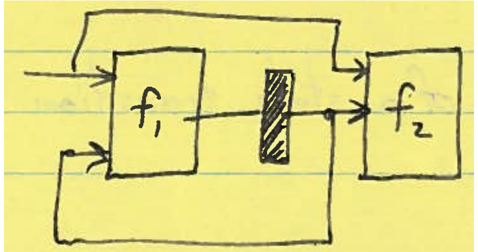
$$\bar{\psi}_k = [m_{k-1}, m_{k-2}, \dots m_{k-\mu}]$$

# Transition Diagram FSM Representation

- Transition diagrams can be represented with FSMs

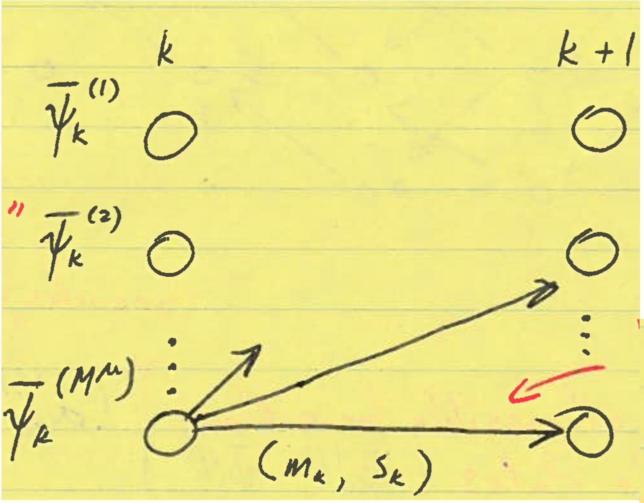


=



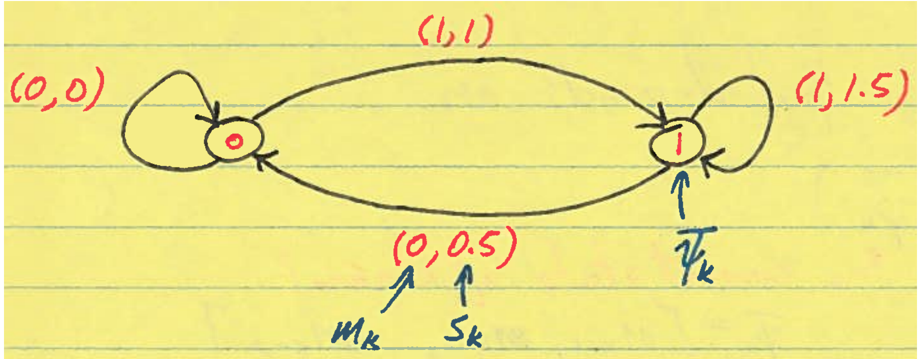
# 13.3 Trellis Diagram

- Unravel state transition diagram over time
  - represent states with circles
    - "nodes"
  - $M^\mu$ nodes at each moment of time k
    - alphabet size: M
    - memory: μ



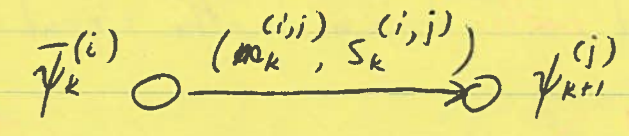
"branch": arcs between states correspond to a particulal state transition at a particular time

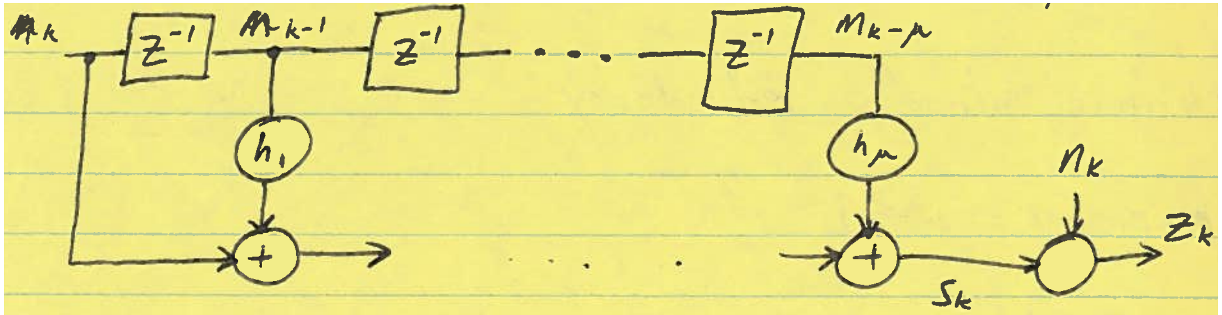
M branches per state



# Trellis Diagram

- General transitions
  - from state i
  - to state j
  - at time k
  - with input $m_k$
  - and output $s_k$

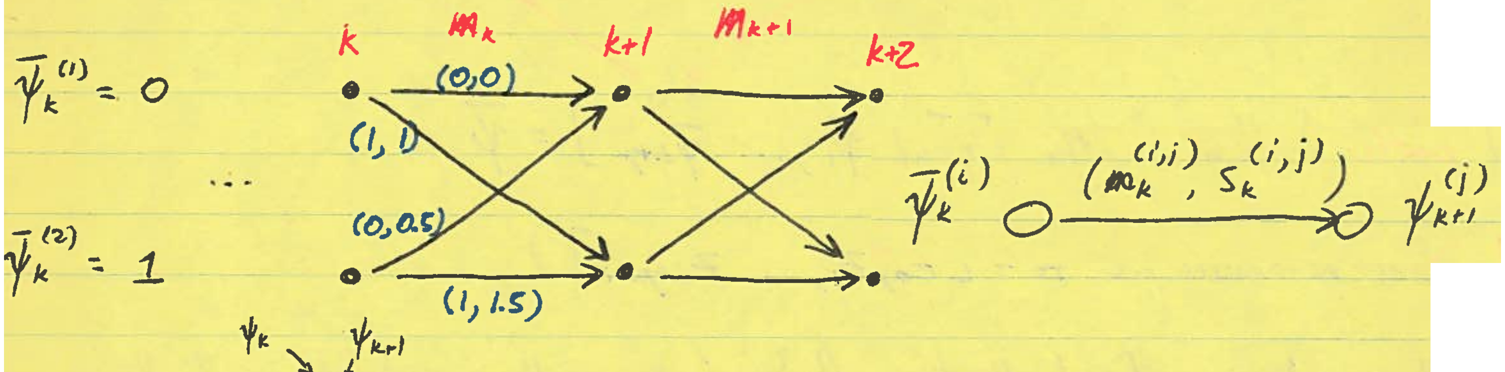




$$s_k = f(m_k, \bar{\psi}_k)$$

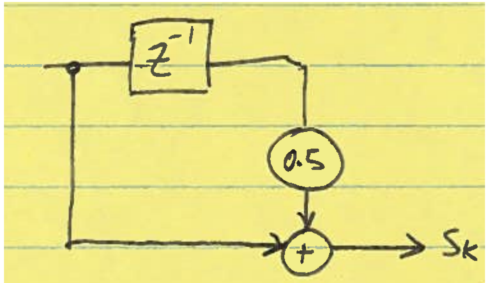
$$\bar{\psi}_k = [m_{k-1}, m_{k-2}, \ldots m_{k-\mu}]$$

# Trellis for our Basic Example



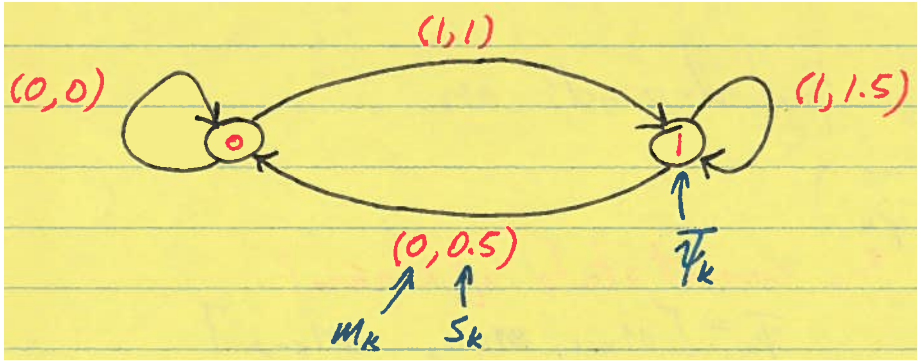
$m^{(0,0)} = 0, \quad m^{(0,1)} = 1, \quad m^{(1,0)} = 0, \quad m^{(1,1)} = 1$

$s^{(0,0)} = 0 \quad s^{(0,1)} = 1, \quad s^{(1,0)} = 0.5, \quad s^{(1,1)} = 1.5$



$\{0, 0.5, 1, 1.5\}$

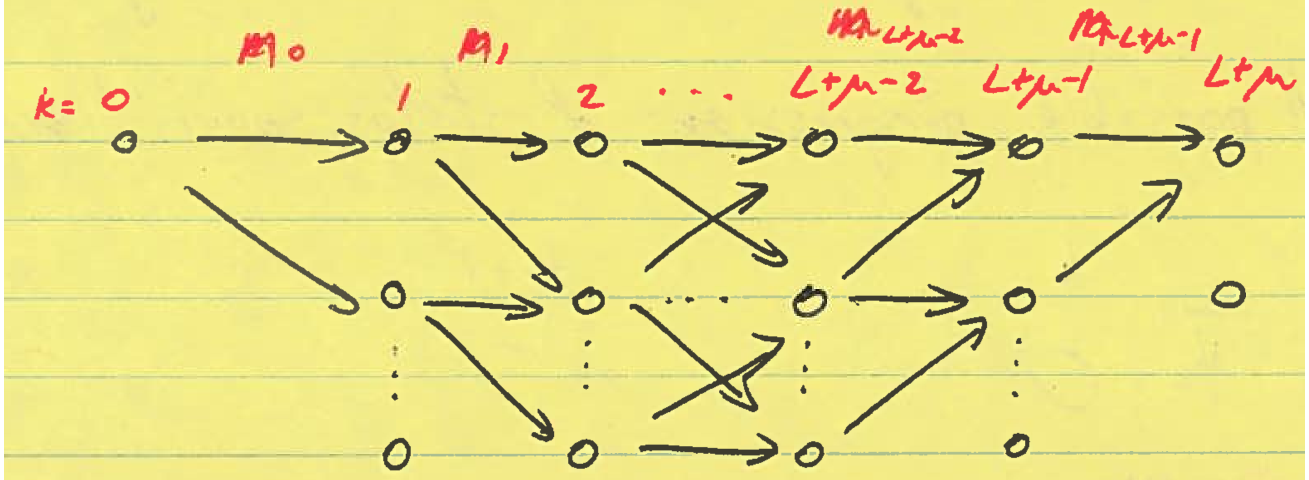
=



$\bar{\psi}_k = [m_{k-1}, m_{k-2}, \ldots m_{k-\mu}]$

# Trellis with Constrained Bounds

- See this often
  - Trellis starting and stopping at some point



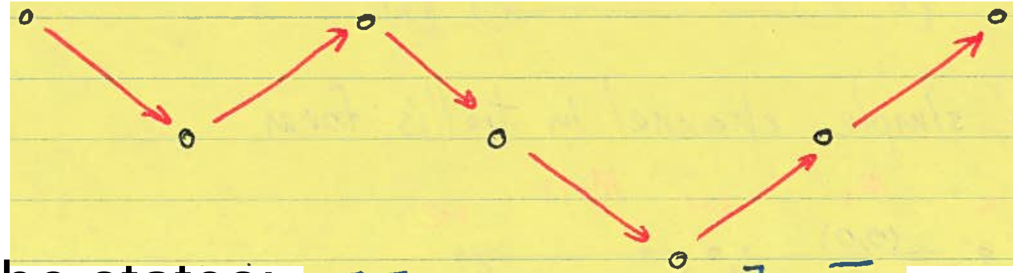
- How?
  - Start in know state
  - Send in random sequence of L bits, followed by μ zeros

# 13.4 Path Detection

- Trellis shows
  - all possible transitions (branches)
    - between…
  - all possible states (nodes)

$$\bar{\psi}_k^{(i)} \;\circ\!\xrightarrow{\left(m_k^{(i,j)},\; s_k^{(i,j)}\right)}\!\circ\; \psi_{k+1}^{(j)}$$

- But we want to know what path the signal actually takes!

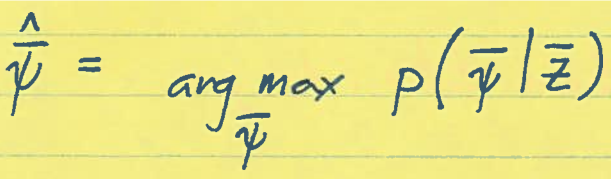


- Through the states: $[\bar{\psi}_0, \bar{\psi}_1, \ldots, \bar{\psi}_{L+\mu}] = \bar{\psi}$

- How to find this path?

# Path Detection

- Use MAP!



$$\hat{\bar{\psi}} = \arg\max_{\bar{\psi}} p(\bar{\psi}|\bar{z})$$

  - see notes for derivation
- To find…

# Branch Weights and Path Metric

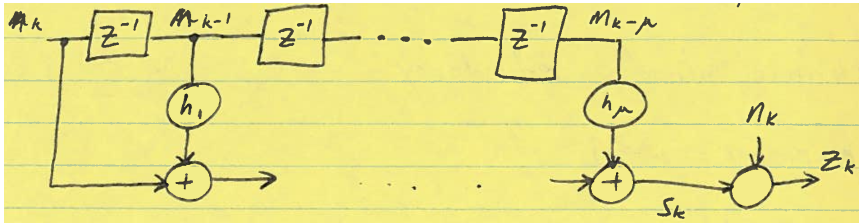
- To find…
  - BRANCH WEIGHT

$$B_k(i,j)\Big|_{ML} = |z_k - s^{(i,j)}|^2$$

$$\psi_k^{(i)} \; \circ \xrightarrow{\left(m_k^{(i,j)},\; s_k^{(i,j)}\right)} \circ \; \psi_{k+1}^{(j)}$$

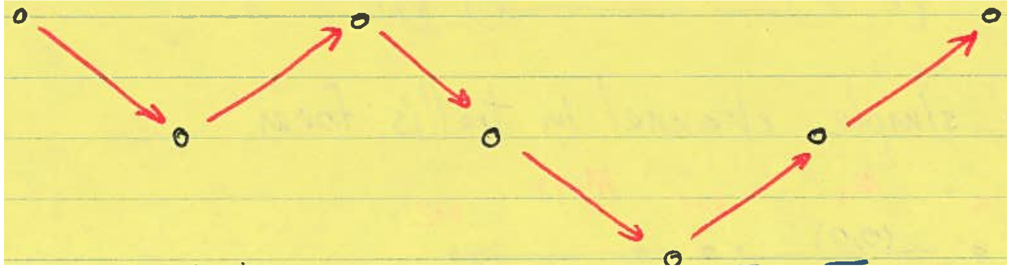
- proportional to
  - logarithm of
  - 1/probability of going from i to j
- PATH METRIC
  - proportional to
    - logarithm of
    - 1/probability of any path through the trellis

$$P\Big|_{ML} = \sum_{k=0}^{L+\mu-1} |z_k - s^{(i,j)}|^2$$



# Finding the Most Likely Path

- Find the most probable path
- Find the sequence that minimizes



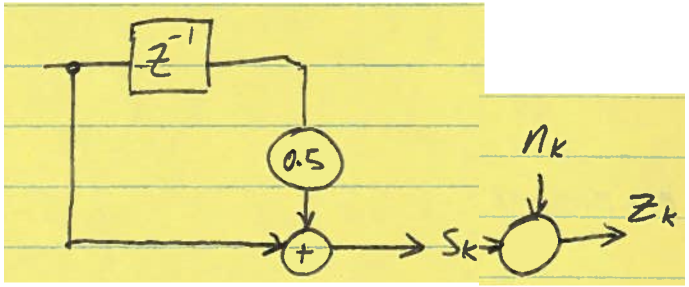
$$P\big|_{ML} = \sum_{k=0}^{L+\mu-1} \left| z_k - s^{(i,j)} \right|^2$$

- But so many paths to consider
  - signal alphabet size: M
  - channel taps: $\mu+1$
  - possible states: $M^{\mu+1}$
  - time steps: L
  - possible number of paths: $(M^{\mu+1})^L$

# 13.5 Viterbi Algorithm

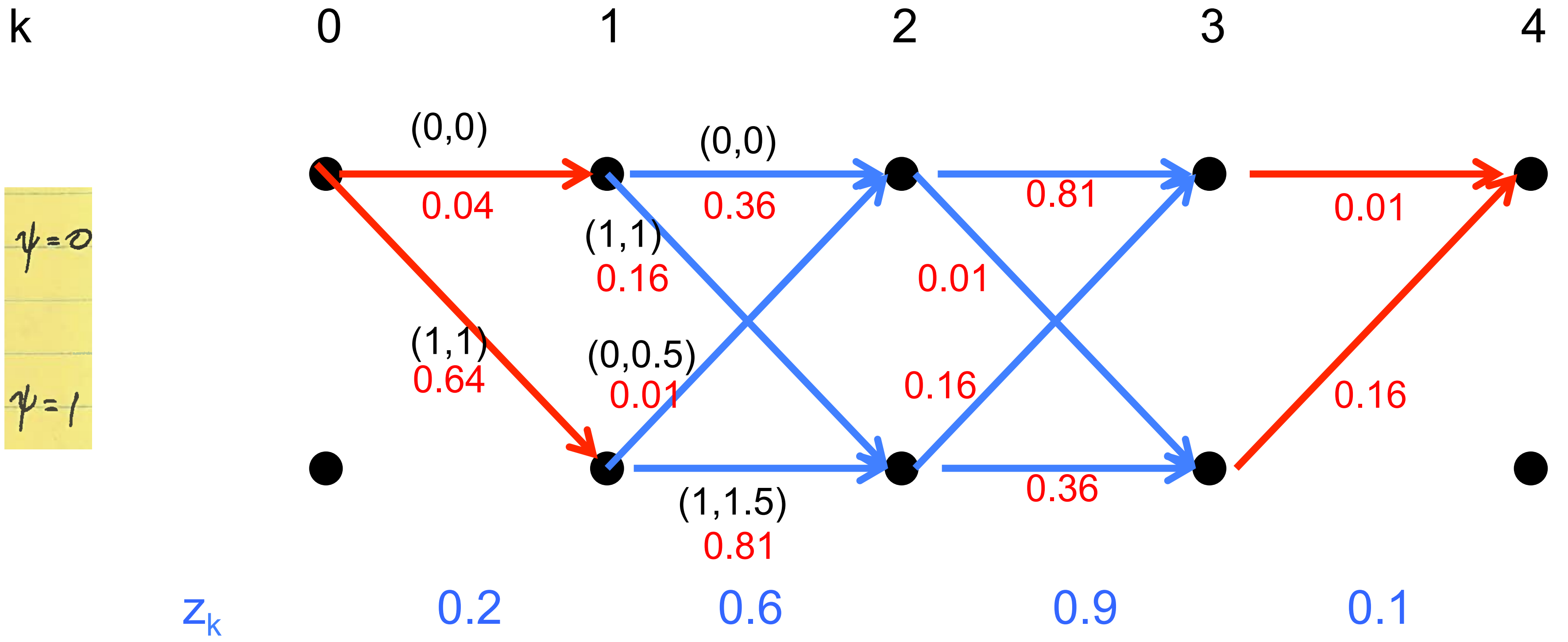
- Luckily can handle this with dynamic programming
  - An optimal sequence can be found one sequential step at a time
    - No need to consider all paths in one go
- Back to starting question:
  - What is $m_k$?
- Assume
  - Starting state is 0
  - Last $m_k$ is 0

$m_k$



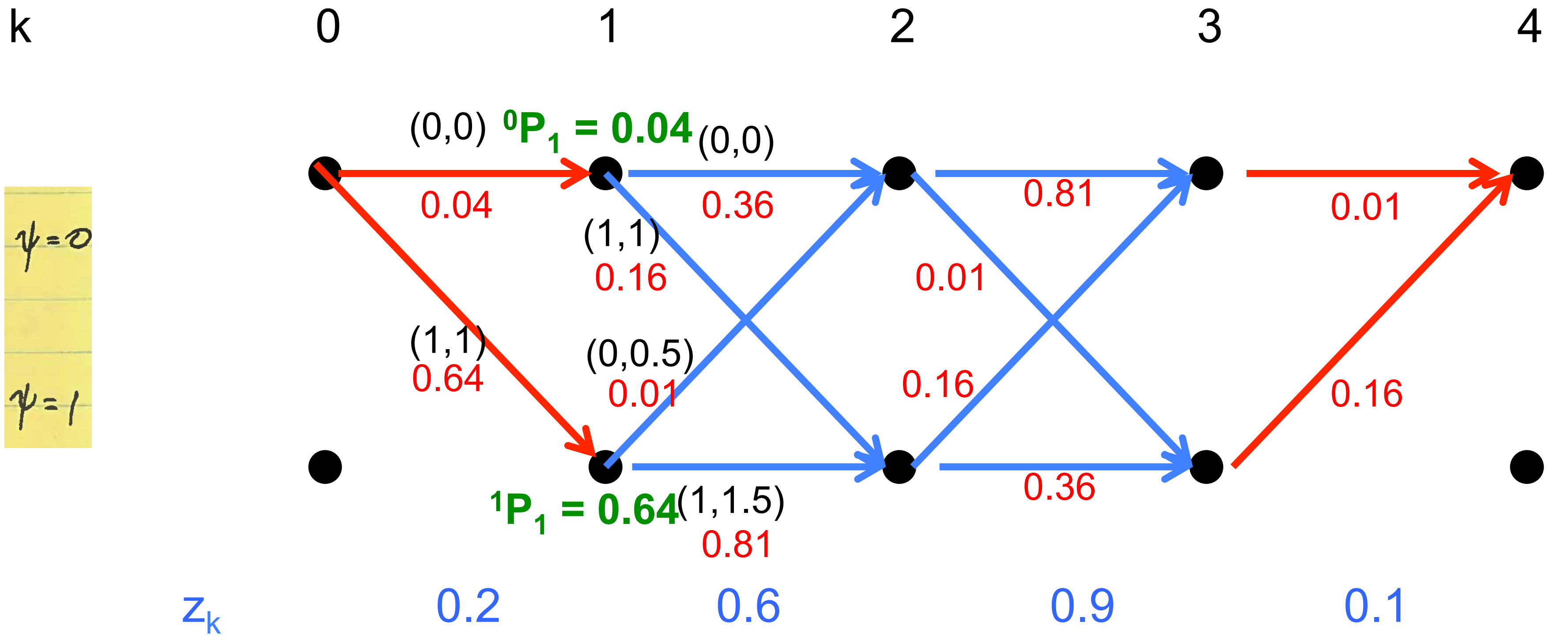
$z_k$ = 0.2, 0.6, 0.9, 0.1

# Trellis: Branch Weights



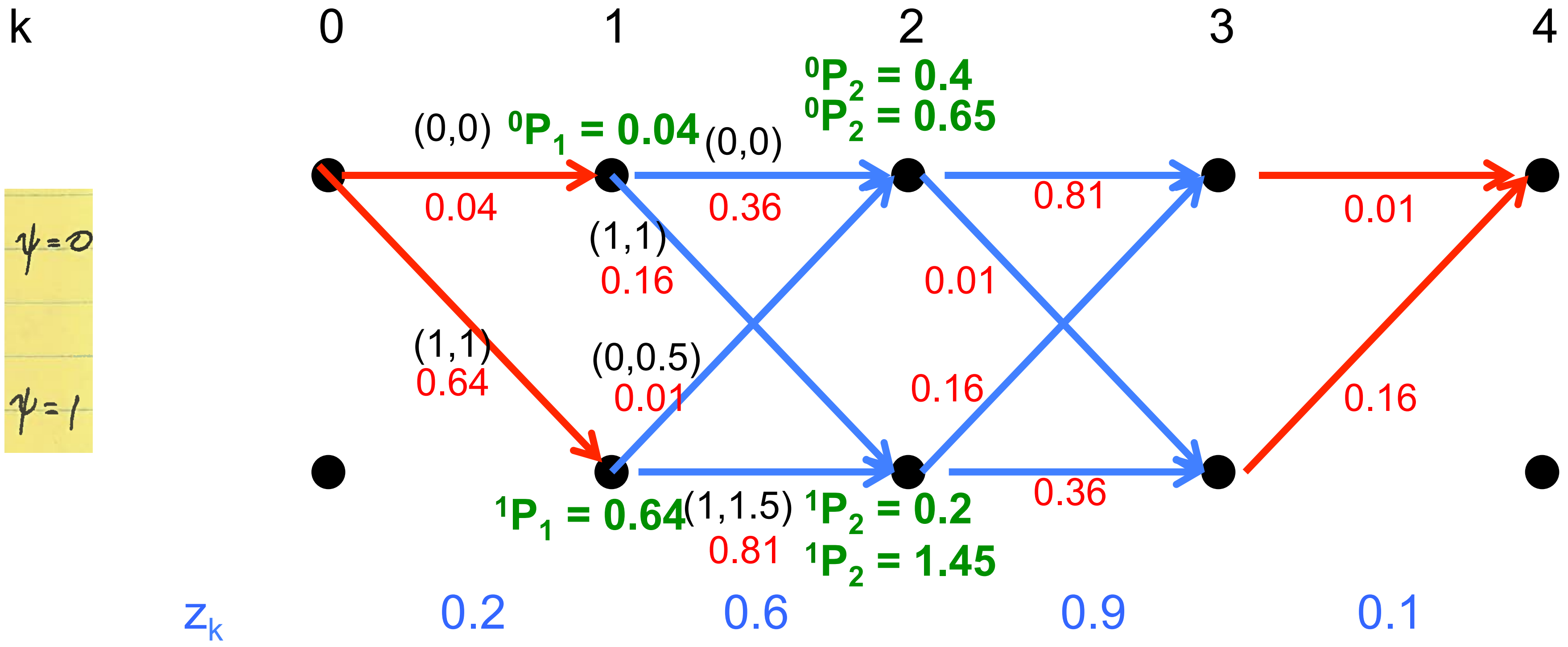
$$B_k(i,j)\Big|_{ML} = |z_k - s^{(i,j)}|^2$$

# Trellis: Path Metric



- Calculate path metrics into each node
  - One step at a time, thus creating partial paths
  - No problem at k = 1

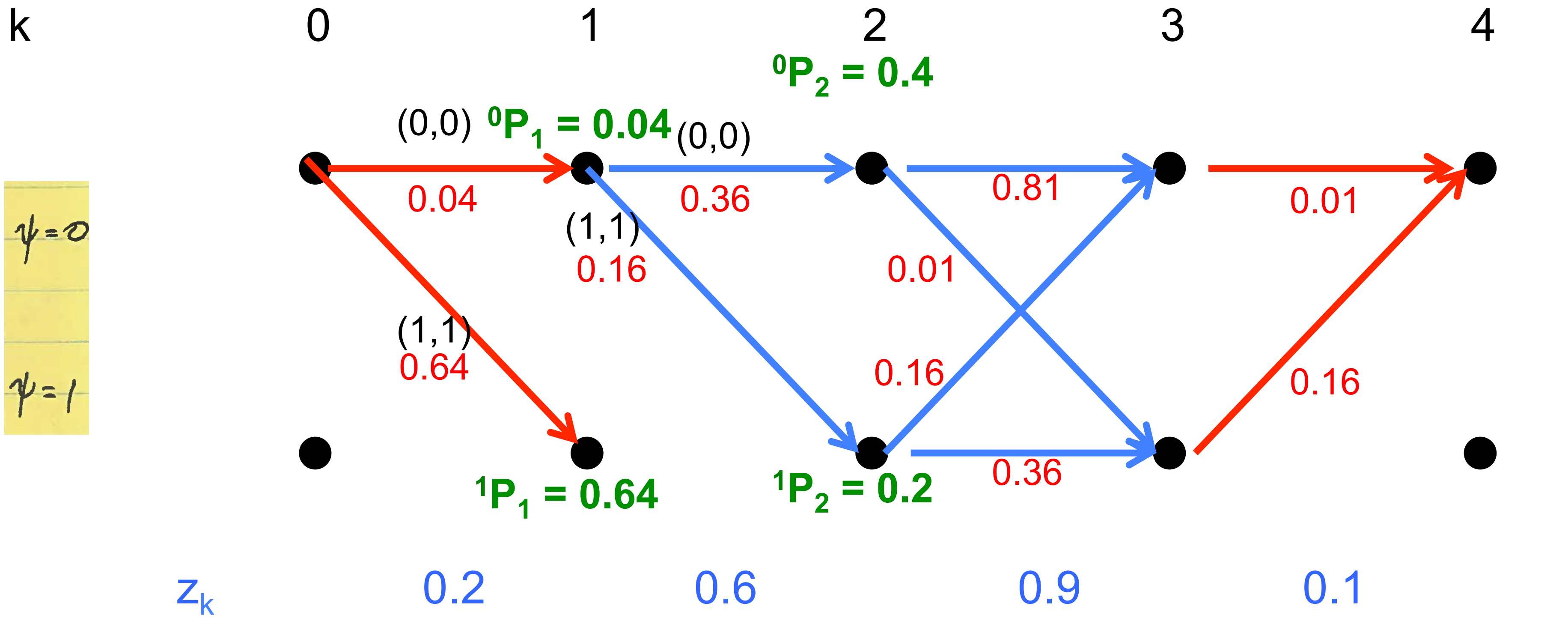
# Trellis: Path Metric



- At k = 2
  - Two possibilities at each node

$$P'^{(j)}_{k+1} = \{P_k^{(i)} + \underbrace{B_k(i,j)}\}$$

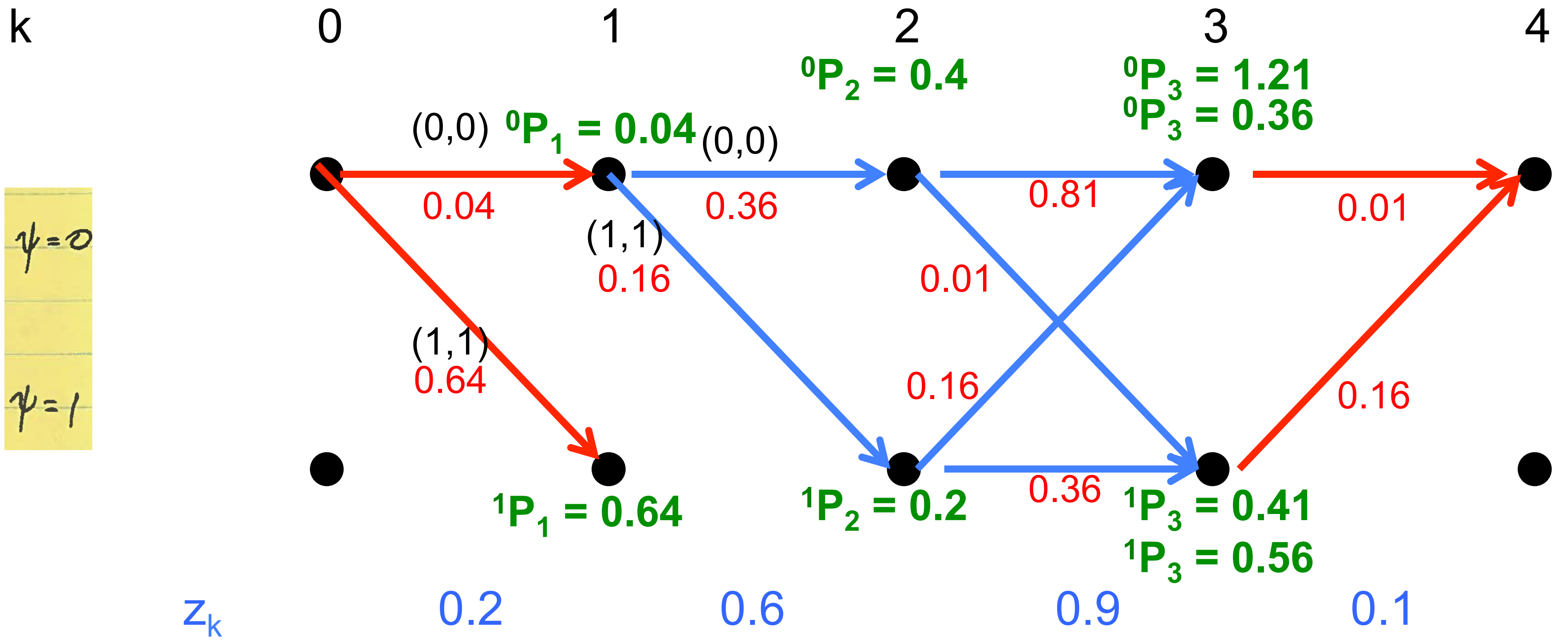
# Trellis: Path Metric



- At k = 2
  - Two possibilities at each node
  - Retain the path that corresponds to the minimum
    - survivor path

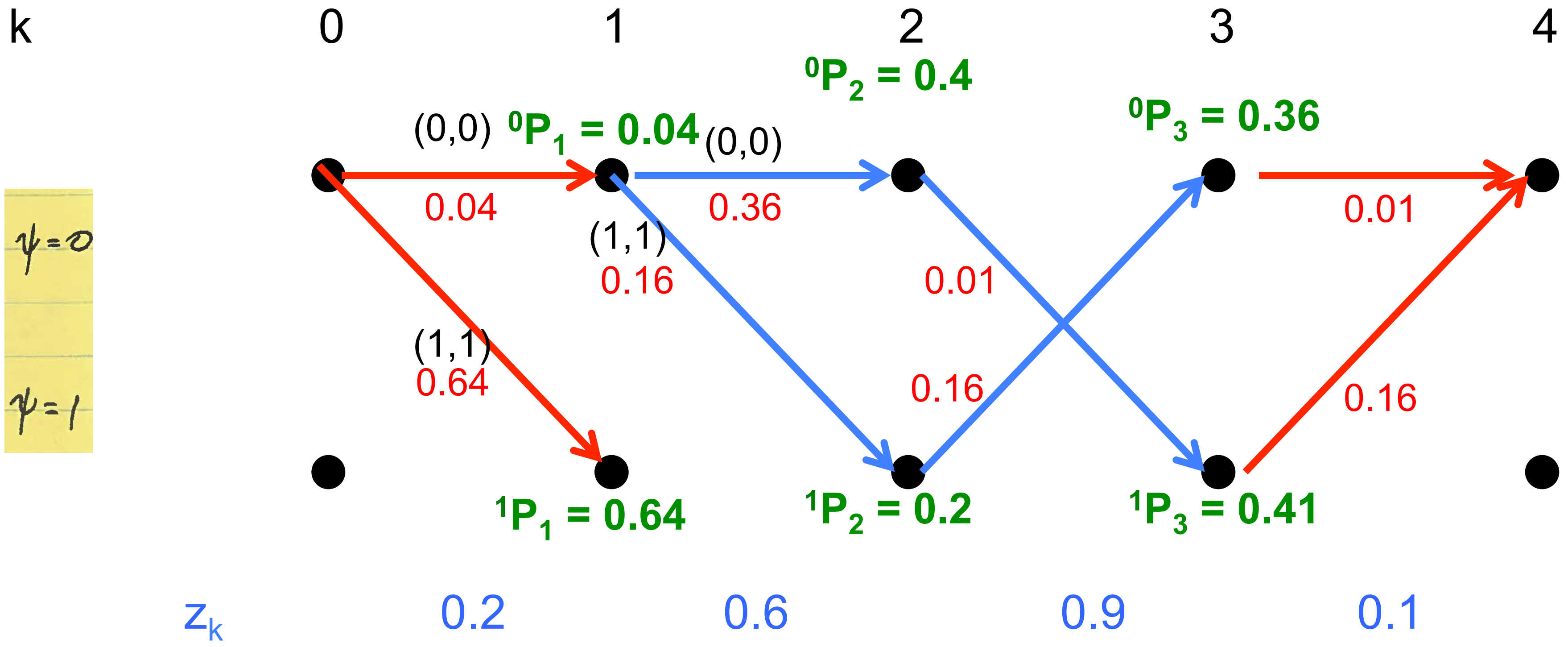
$$P_{k+1}^{(j)} = \min\left\{P_k^{(i)} + B_k(i,j)\right\}$$

# Trellis: Path Metric



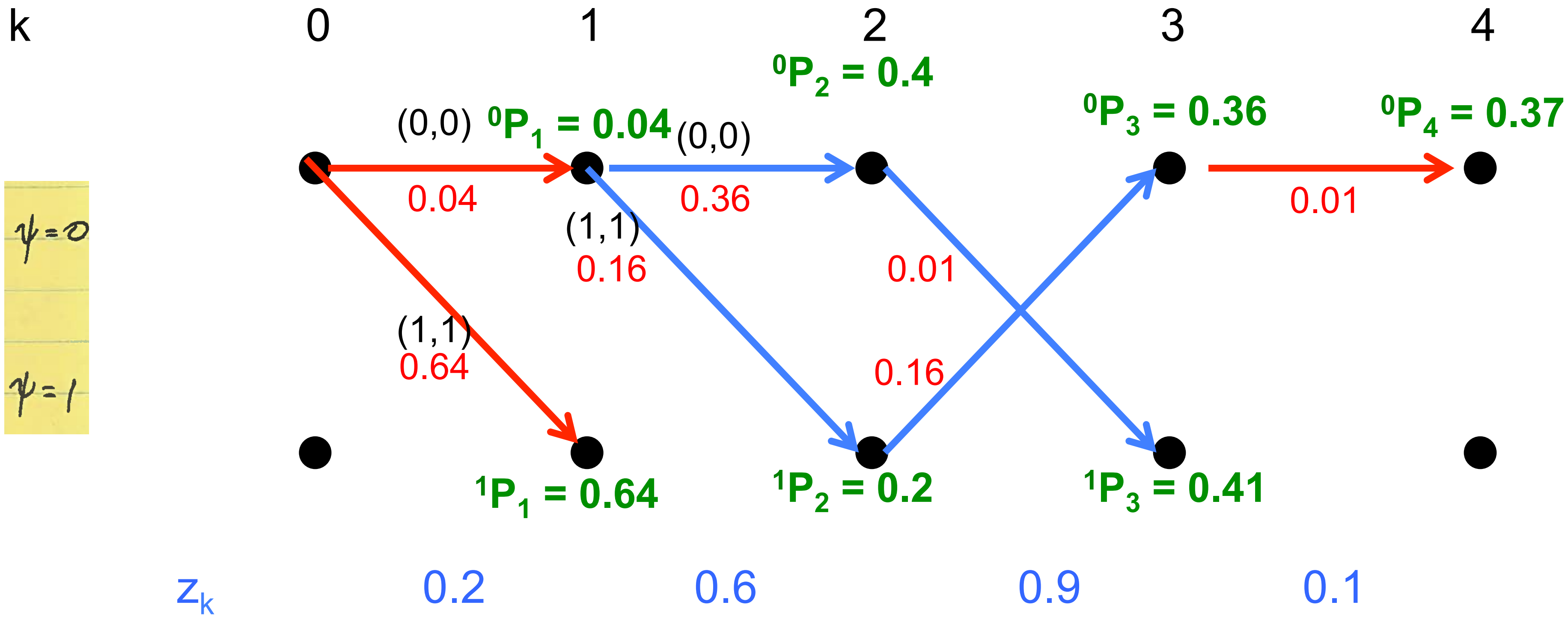
- At k = 3
  - Same idea as before
    - Find your path metrics up to k=3…

# Trellis: Path Metric



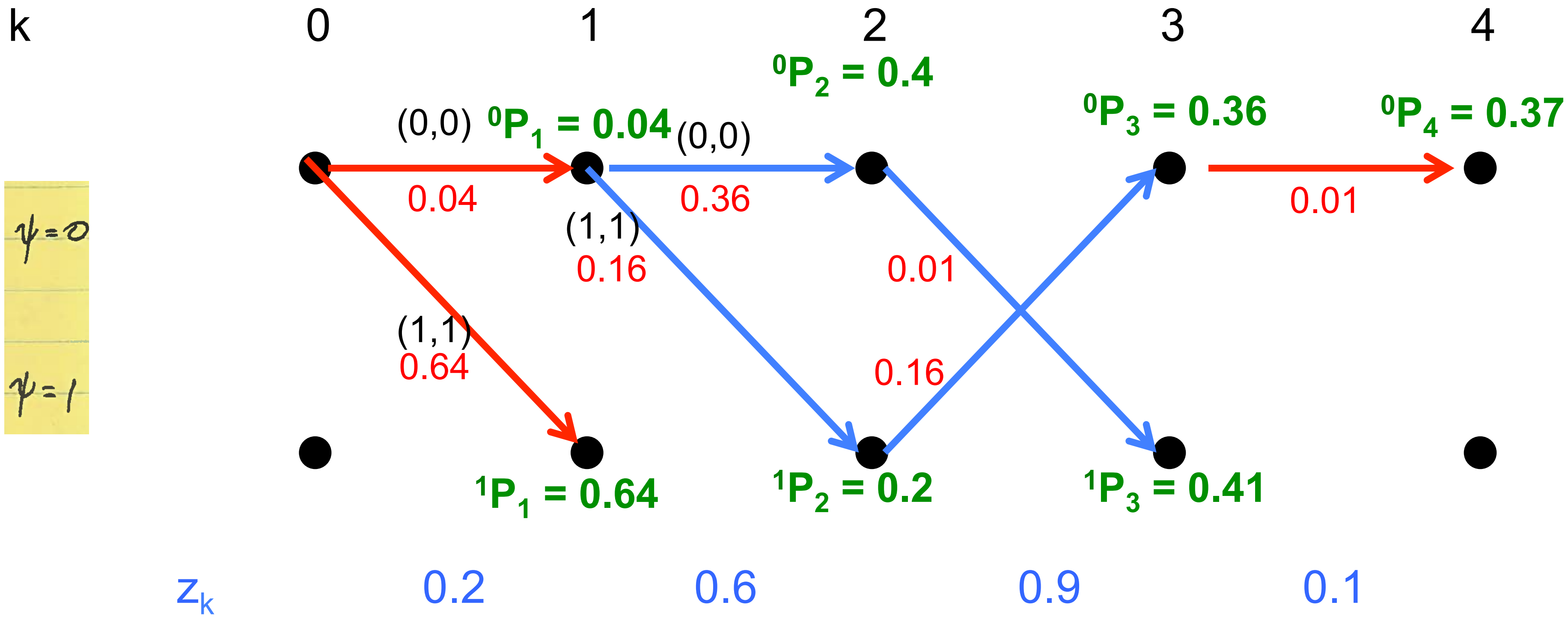
- At k = 3
  - Same idea as before
    - Find your path metrics up to k=3…
    - …and retain survivors, thus trimming down partial paths

# Trellis: Path Metric



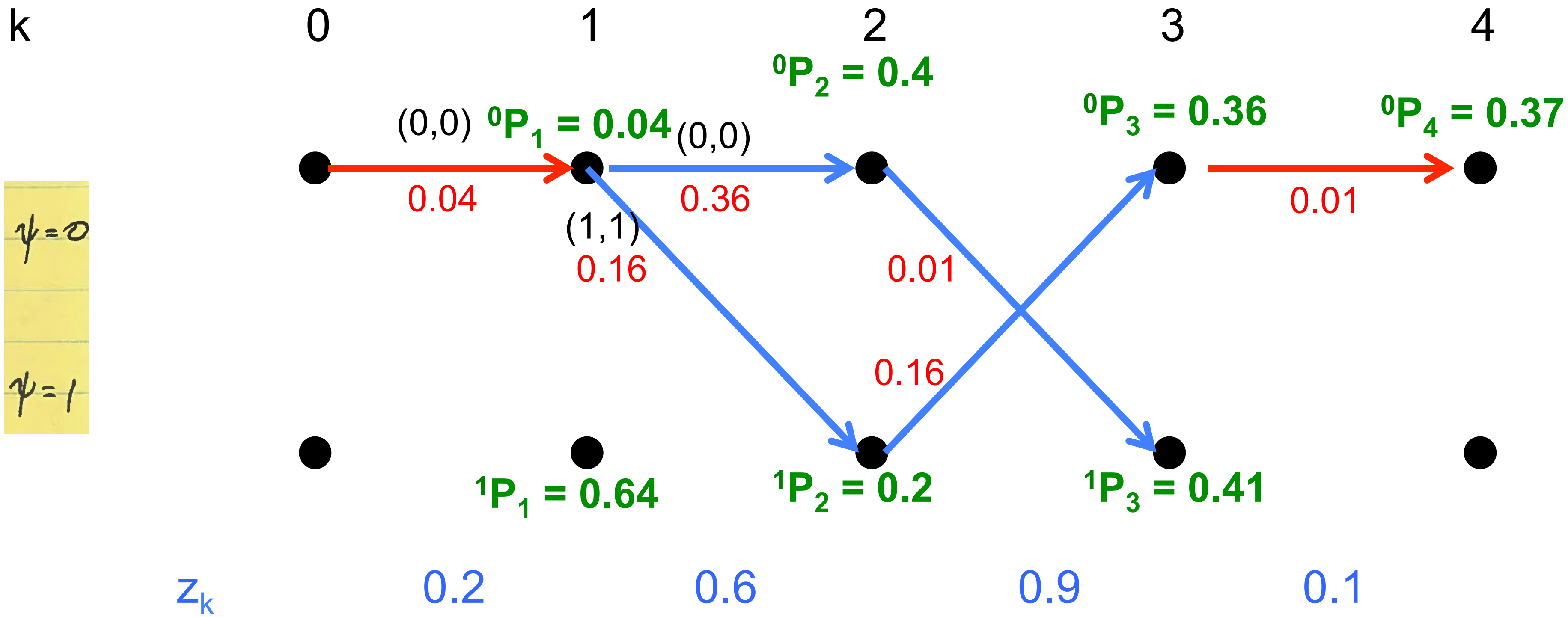
- At k = 4

# Trellis: Path Metric



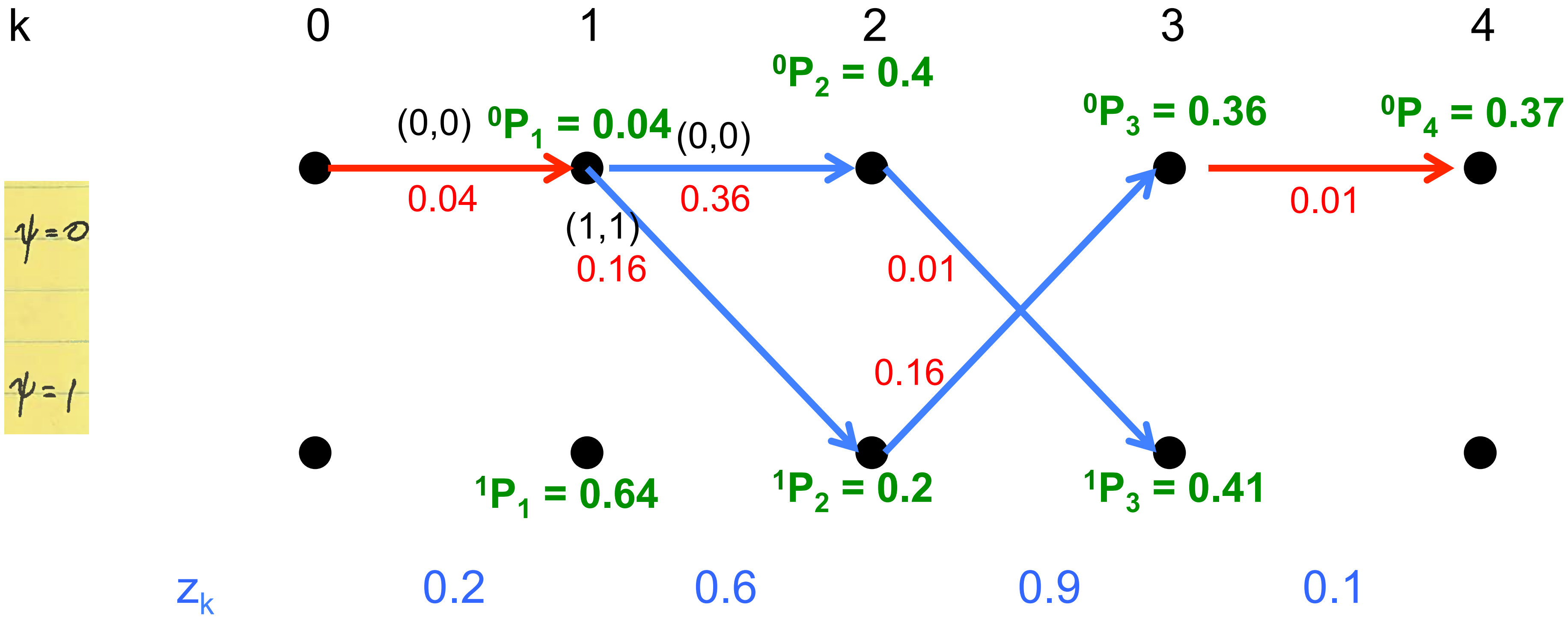
- Note that at k = 2 all the survivors coincide at k =1
  - i.e. partial paths are merged at…
    - depth d = 2 – 1 = 1 (i.e. k = 2 minus k = 1 where the merge happens)

# Trellis: Path Metric



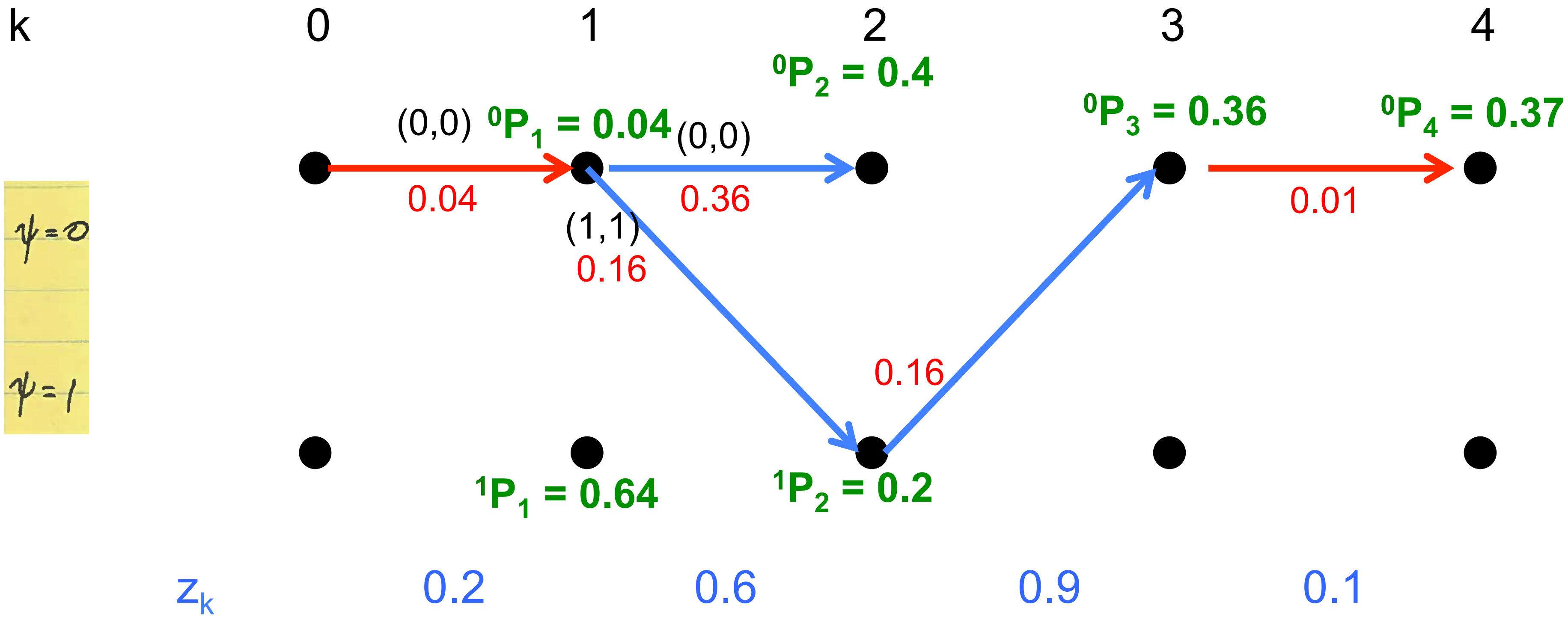
- Note that at k = 2 all the survivors coincide at k =1
  - i.e. partial paths are merged at…
    - depth d = 2 – 1 = 1 (i.e. k = 2 minus k = 1 where the merge happens)
  - So we don't even have to think about the dead-end branch

# Trellis: Most Likely Path



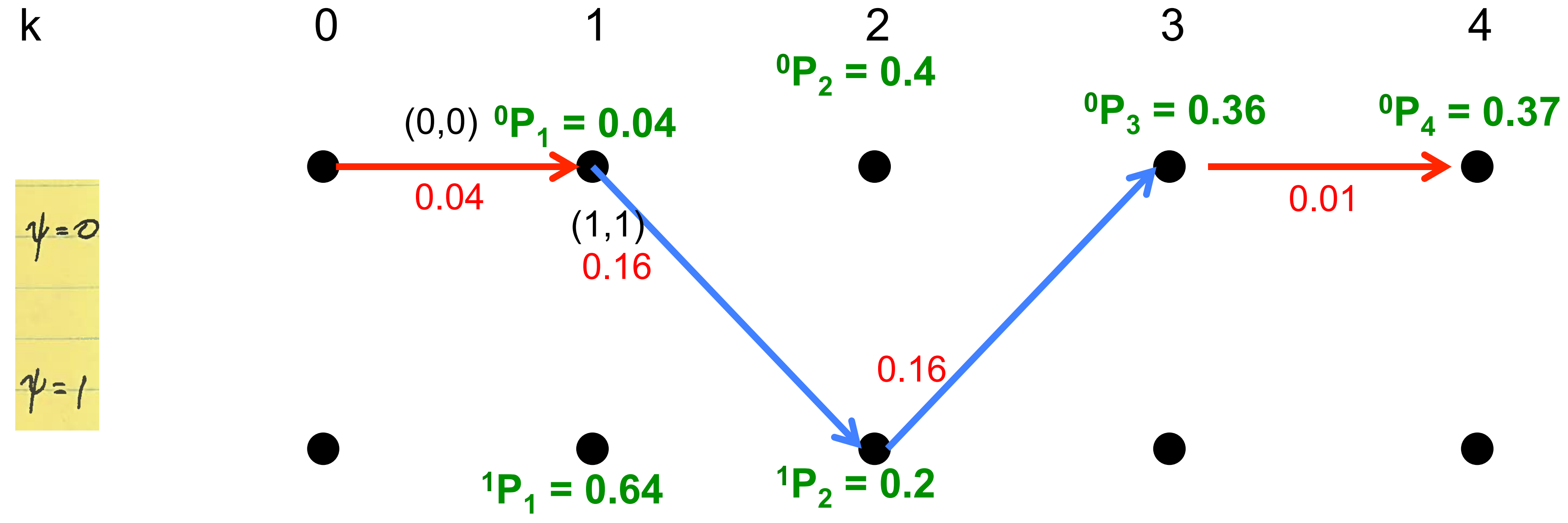
- Now just need to traceback to find the most likely path

# Trellis: Most Likely Path



- Now just need to traceback to find the most likely path

# Trellis: Most Likely Path



$z_k$ 0.2 0.6 0.9 0.1

- Now just need to traceback to find the most likely path
- And the predicted message is…

$m_k$ 0 1 0 0