Lecture 8: Polar Codes

March 8, 2019

Polar Codes

1. Introduction
2. Coding
3. Decoding

1. Introduction

Polar codes, a class of codes which allows to

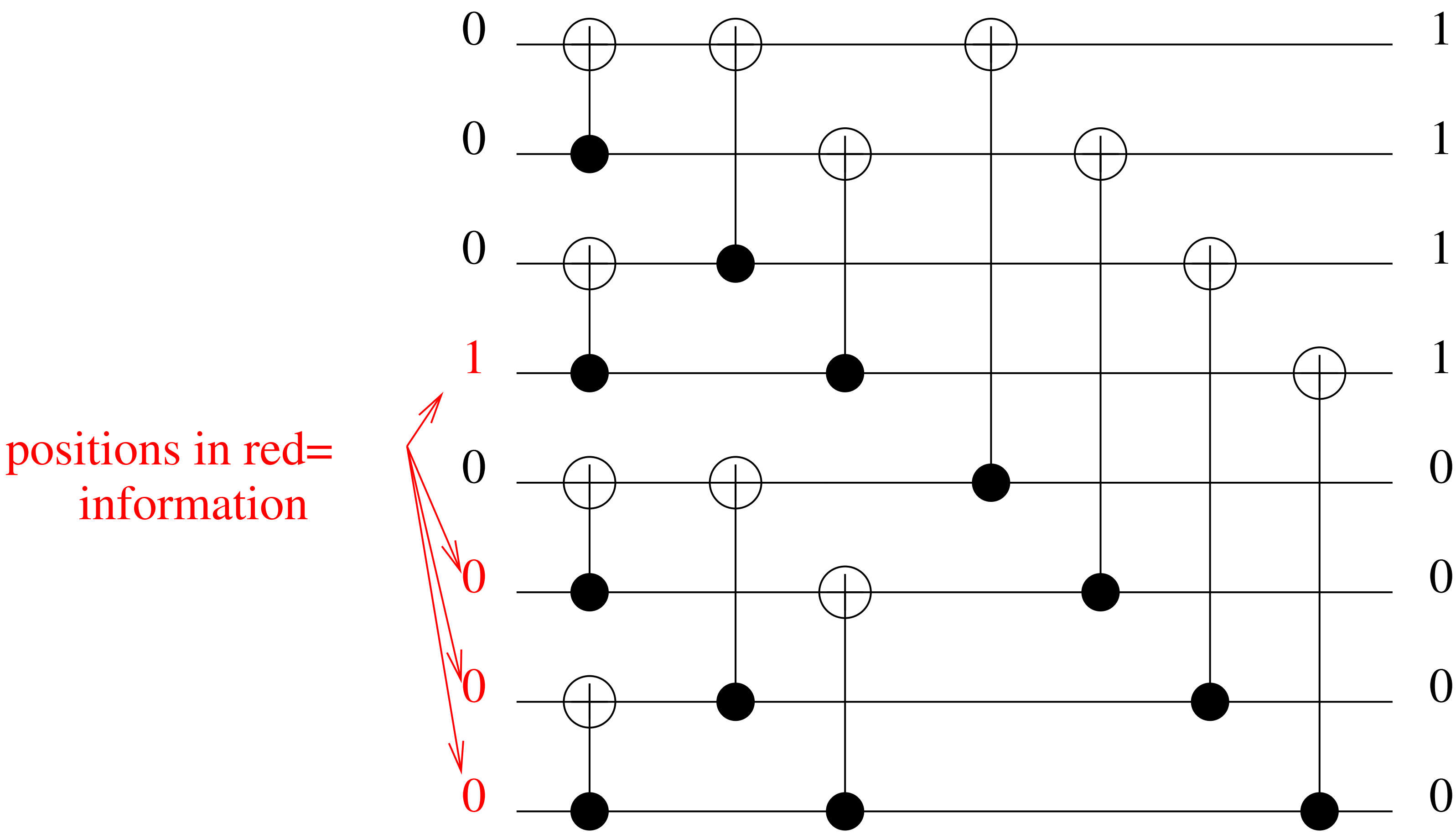
1. attain the capacity of all symmetric memoryless channels (=those for which the capacity is attained for a uniform input distribution),
2. with an encoding algorithm of complexity $O(N \log N)$ ($N =$ code length),
3. with a decoding algorithm of complexity $O(N \log N)$.

This decoding algorithm borrows many ideas from the decoding algorithm used for LDPC codes.

Polar Codes

1. a coding architecture based on the Fast Fourier Transform by fixing some bits to 0,
2. a code construction based on recursive $(U + V|V)$ codes.
3. a (suboptimal) decoding algorithm which computes the probability that the input bits are equal to 0 given the previous input bits and the probabilities of the output bits.

Encoding : example



Polar Code : linear code

In the previous case, it is a code of generator matrix

$$G = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

2. Polar codes= recursive $(U+V|V)$ codes

- They can be encoded/decoded by viewing them as recursive $(U+V|V)$-codes
- A $(U+V|V)$-code is a simple way of constructing a code of length $2n$ from two codes of length n that keeps/improves the distance properties of the two codes.

Definition[$(U+V|V)$-code] Let U and V be two linear codes of length n over $\mathbb{F}_q$. The $(U+V|V)$-code associated to U and V is defined by

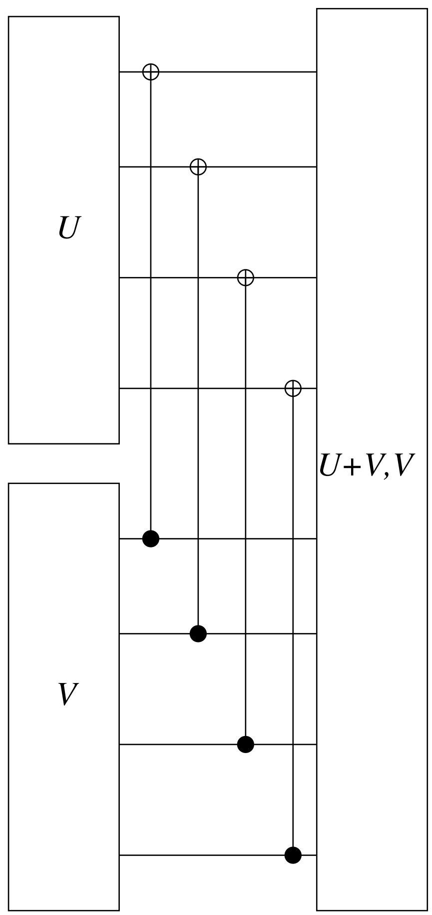
$$(U+V|V) \stackrel{\text{def}}{=} \{(\mathbf{u}+\mathbf{v},\mathbf{v}) : \mathbf{u} \in U, \mathbf{v} \in V\}.$$

Properties

Proposition 1.

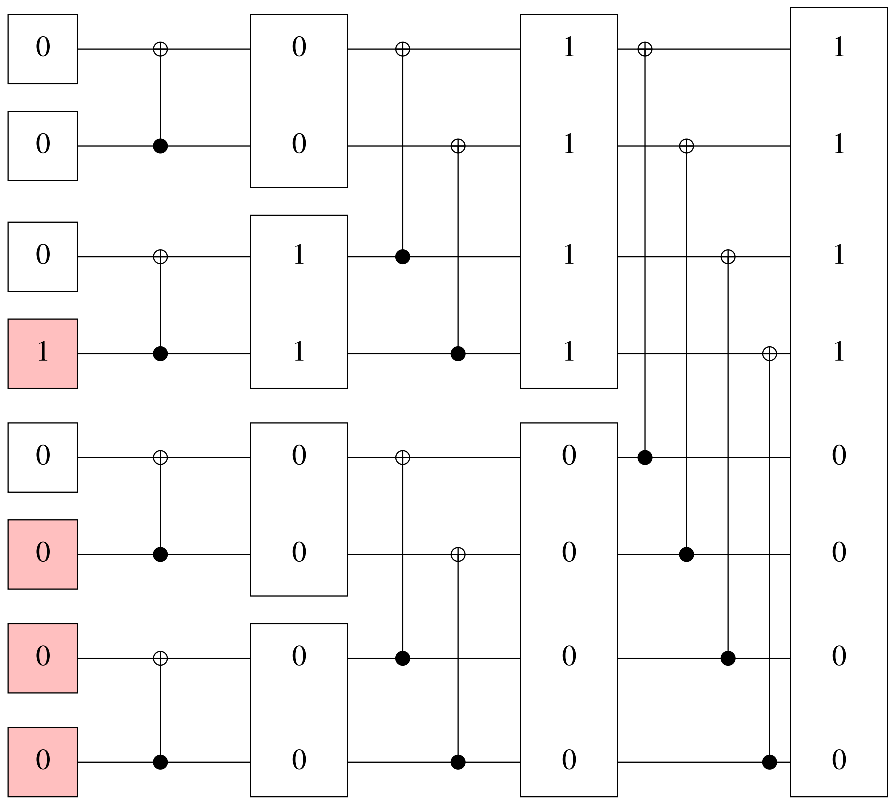
1. $\dim(U+V|V) = \dim U + \dim V$
2. $d_{min}(U+V|V) = \min(d_{min}(U), 2d_{min}(V))$.

Encoding circuit



Encoding circuit of a recursive $(U+V|V)$-code

- Basic codes = bits



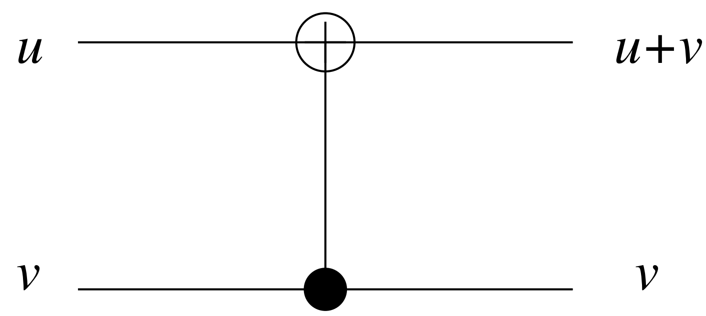
Decoding of polar codes

- Based on using soft information i.e. $\mathbf{Prob}(x_i = 1|y_i)$ where x_i is a bit that was sent through the channel and y_i is the received symbol.

$$(\mathbf{u}+\mathbf{v},\mathbf{v}) \overset{\text{channel}}{\rightsquigarrow} (\mathbf{y}_1,\mathbf{y}_2)$$

- Strategy :
  1. Decode the $\mathbf{u}$ part first by using a decoder for the U code and $\mathbf{y}_1$, $\mathbf{y}_2$
  2. Decode the $\mathbf{v}$ part by using a decoder for the V-code based on the knowledge of $\mathbf{y}_1$, $\mathbf{y}_2$ and $\mathbf{u}$
- Recursive decoding of a recursive $(U+V|V)$-code, where the end-operation is just decoding a bit x knowing its probability p to be equal to 1, deciding $x=1$ if $p \geq \frac{1}{2}$ and 0 otherwise.

The basic operation on the erasure channel

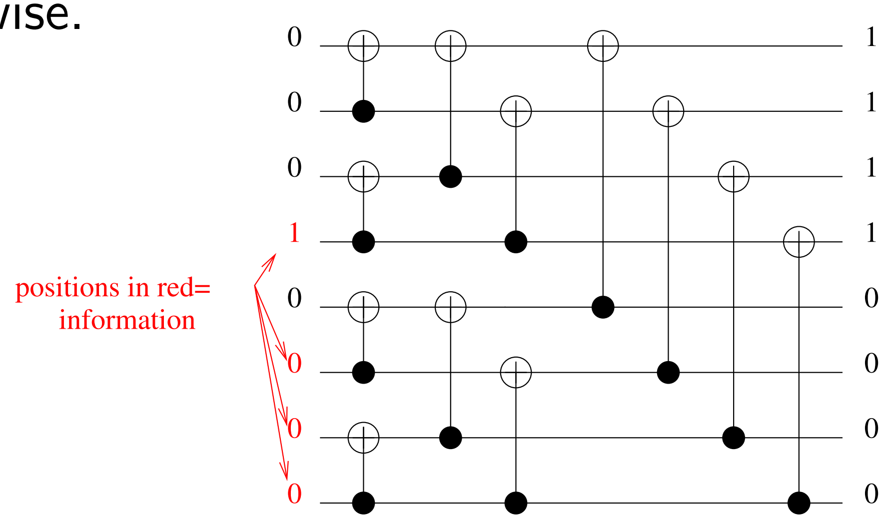


$$\begin{aligned} u+v, v &\in \{0,1,?\} \\ u &= ? \text{ if } u+v \text{ or } v = ? \\ &= (u+v)+v \text{ otherwise} \\ v &= ? \text{ if } u+v \text{ and } v = ? \\ &= v \text{ or } (u+v)+u \end{aligned}$$

- u is not recovered when either $u+v$ or v are erased
- under the assumption that u was recovered v is not recovered when $u+v$ and v were erased

The erasure decoder

- Computes recursively the $\mathbf{u}$ or the $\mathbf{v}$ part, based on the $+$ operation over $\mathcal{A} \stackrel{\text{def}}{=} \{0, 1, ?\}$, where $x+? = ?+x = ?$ for any $x \in \mathcal{A}$ and is the standard addition over $\mathbb{F}_2$ otherwise.
- Polar code of length $N = 2^n$ with n layers of recursion as described before.
- Input vector $\mathbf{z} \in \mathcal{A}^N$ specifying what is known from the decoder about the encoding process, i.e. $z_i = 0$ if the i-th input bit is fixed to 0 and $z_i = ?$ otherwise.



$\boldsymbol{z} = (0,0,0,?,0,?,?,?)$

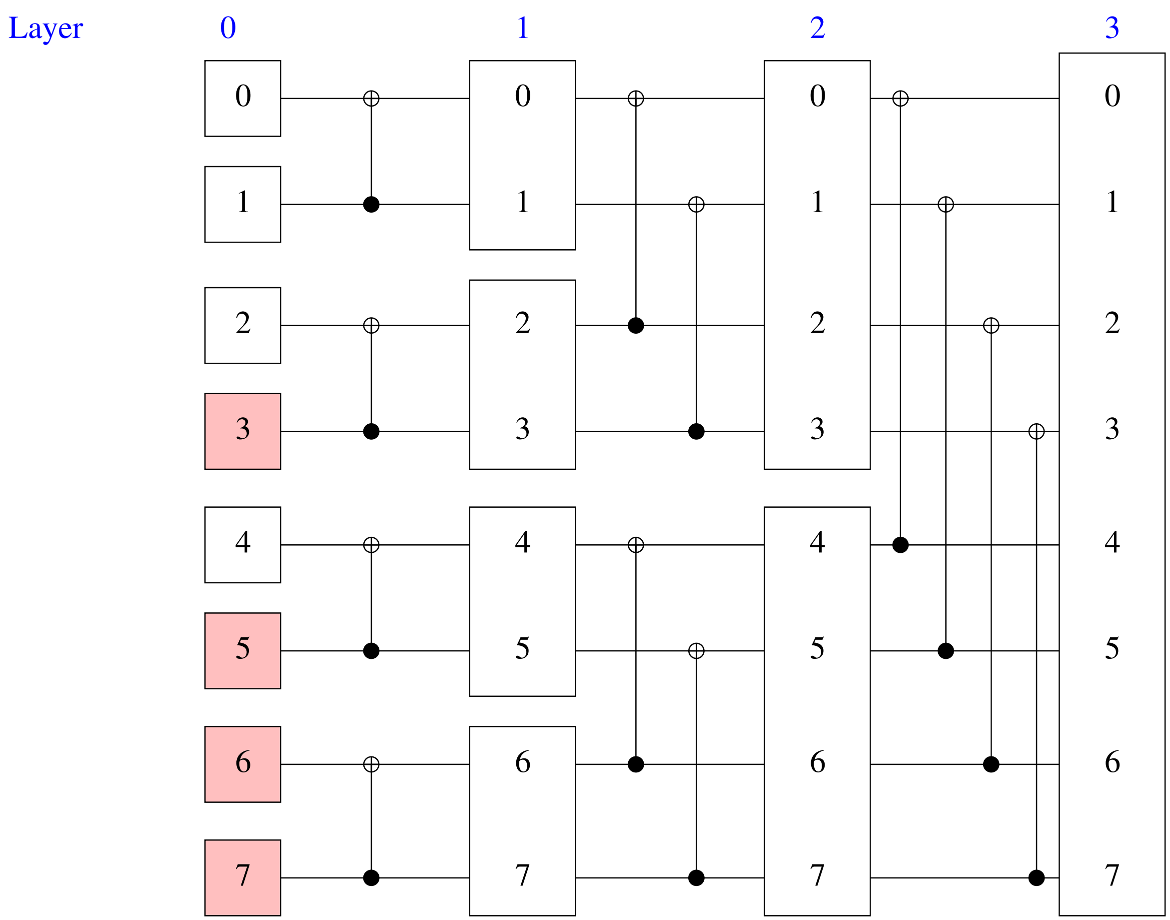
Decoding : notation

- Code positions numbered from 0 to $2^n - 1$, layers from 0 to n (last one corresponding to the polar code positions)
- At layer $n-1$ the positions from 0 to $2^{n-1} - 1$ are positions from a U code, positions from 2^{n-1} to $2^n - 1$ those of a V code
- In general if at layer t the positions from a $(U+V|V)$ code are in $[i \cdots j[$, the positions of the corresponding U code at layer $t-1$ are in $[i \cdots \frac{i+j}{2}[$ and those of the V code in $[\frac{i+j}{2} \cdots j[$.
- matrix $\mathbf{y}[i][j]$, $i \in [0 \cdots n]$, $j \in [0 \cdots 2^n[$

$$\begin{aligned} \mathbf{y}[n][i] &\stackrel{\text{def}}{=} \text{received symbol for the } i\text{-th code position} \\ \mathbf{y}[t][i] &= \text{decoded } i\text{-th bit at layer } t, \text{ for } t < n \end{aligned}$$

- Decoding is performed by calling $\texttt{Decode}(0, 2^n, n)$

Numbering



Algorithm

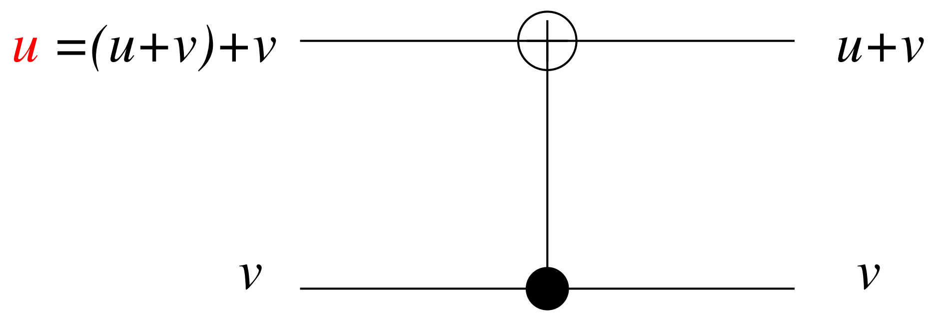
function $\textsc{Decode}(i, j, t)$
  if $t = 0$ **then**
    $\textsc{DecodeDirectly}(i)$
  else
    $m \leftarrow \frac{i+j}{2}$
    $\textsc{UpdateU}(i, m, t-1)$ ▷ computes soft information for the U code
    $\textsc{Decode}(i, m, t-1)$ ▷ finds $\mathbf{u}$
    $\textsc{UpdateV}(m, j, t-1)$ ▷ computes soft information for the V code
    $\textsc{Decode}(m, j, t-1)$ ▷ finds $\mathbf{v}$
    $\textsc{SetPositionsUV}(i, j, t)$ ▷ computes the $(\mathbf{u}+\mathbf{v}, \mathbf{v})$-codeword

Decoding the last layer

- In the last layer the bit $\mathbf{y}[0][i]$ takes the value $\mathbf{z}[i]$ if it was known and takes a random value in $\{0, 1\}$ if $\mathbf{y}[0][i] = ?$

function $\textsc{DecodeDirectly}(i)$
    if $\mathbf{z}[i] \neq ?$ **then**
        $\mathbf{y}[0][i] \leftarrow \mathbf{z}[i]$
    else
        if $\mathbf{y}[0][i] = ?$ **then** $\mathbf{y}[0][i] \leftarrow \textsc{RandomCoin}$

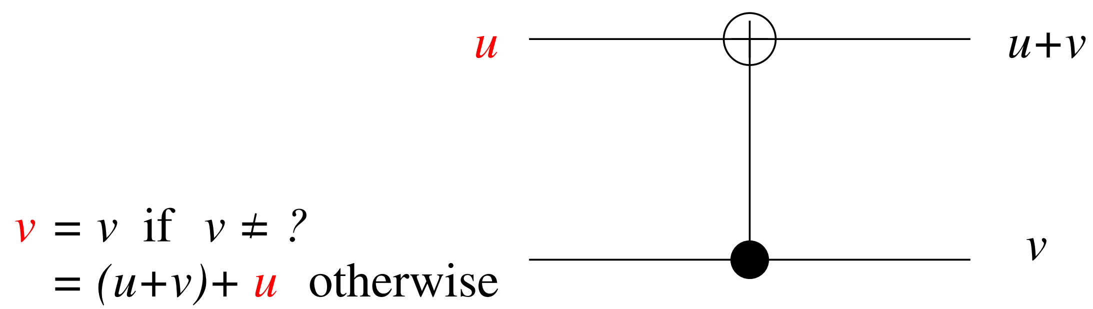
Updating the positions of the U code



- $x+? =? + x =?$ and $+$ is the addition on $\mathbb{F}_2$ otherwise

function $\textsc{UpdateU}(i, j, t)$
  for $\ell = i$ to $j - 1$ **do**
    $\mathbf{y}[t][\ell] \leftarrow \mathbf{y}[t+1][\ell] + \mathbf{y}[t+1][\ell + 2^t]$

Updating the positions of the V code



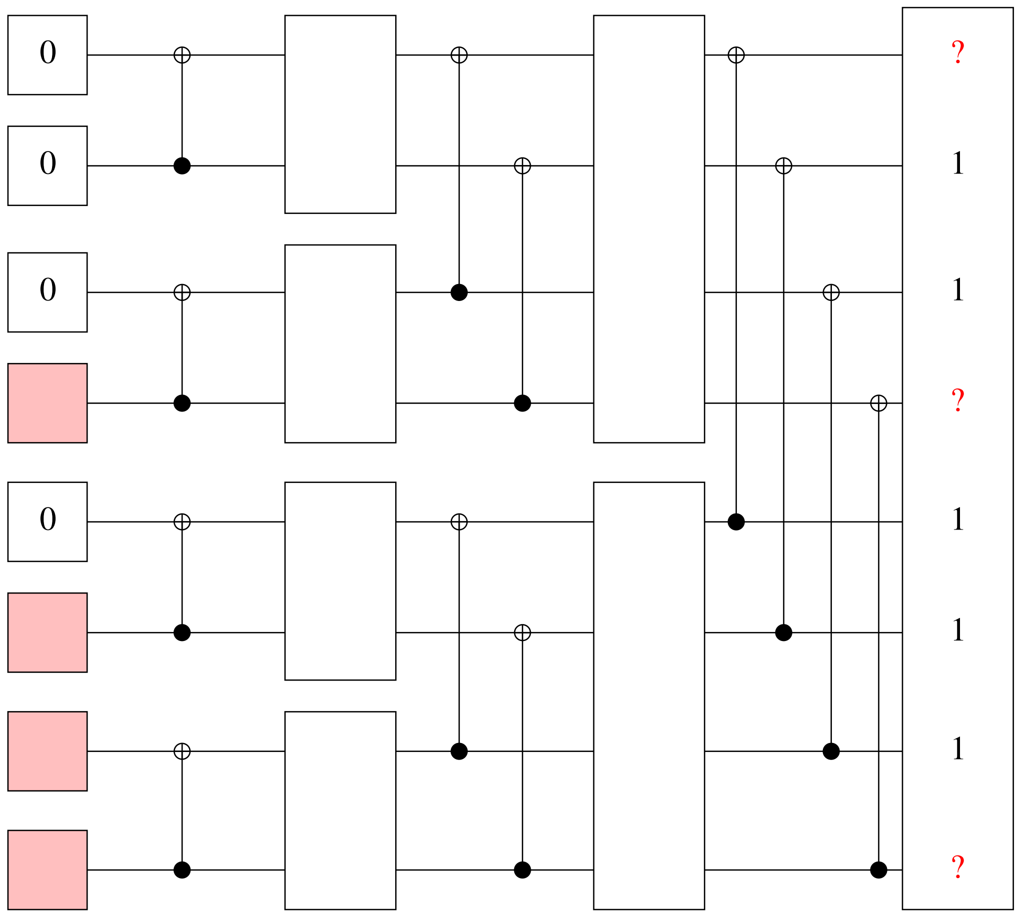
function $\textsc{UpdateV}(i, j, t)$
  for $\ell = i$ to $j - 1$ **do**
    if $\mathbf{y}[t+1][\ell] \neq ?$ **then**
      $\mathbf{y}[t][\ell] \leftarrow \mathbf{y}[t+1][\ell]$
    else
      $\mathbf{y}[t][\ell] \leftarrow \mathbf{y}[t+1][\ell - 2^t] + \mathbf{y}[t+1][\ell]$

Encoding

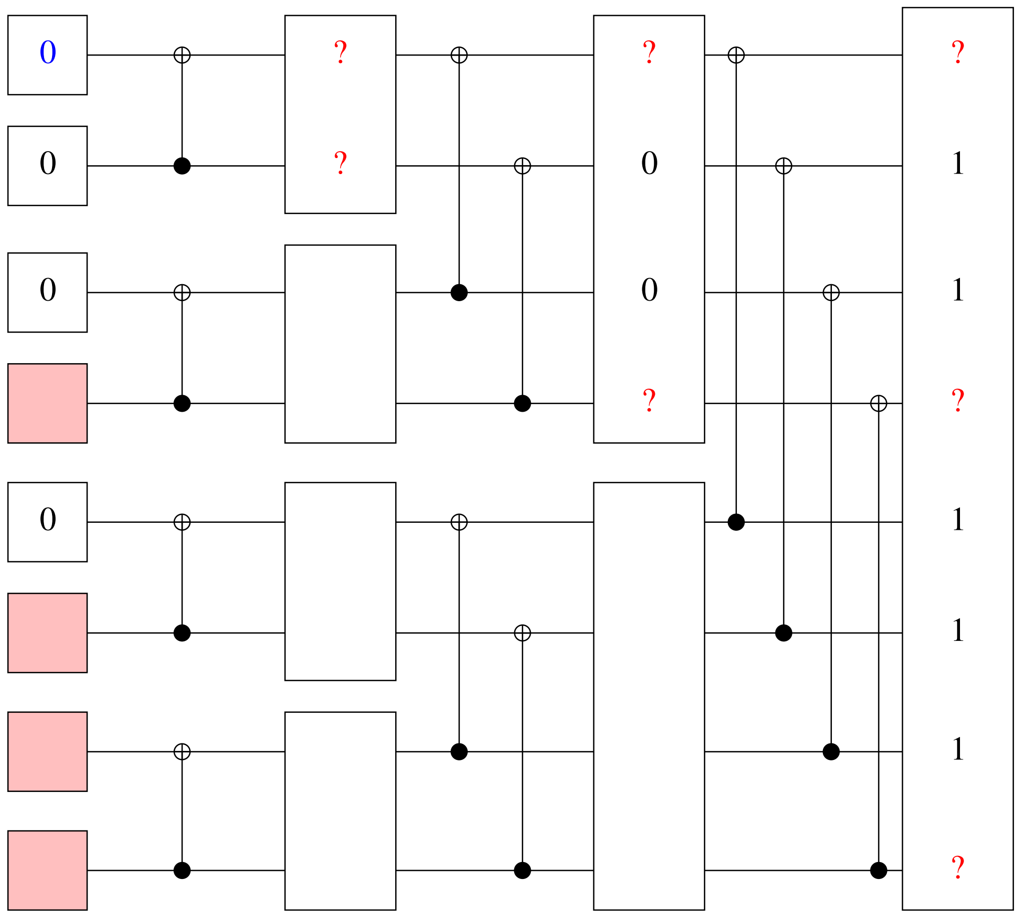
- Encodes the t-layer of the $(U+V|V)$-code as $(\mathbf{u}+\mathbf{v},\mathbf{v})$ once the $\mathbf{u}$ part and the $\mathbf{v}$ part is recovered

function $\textsc{SetPositions}\text{UV}(i,j,t)$
  for $\ell = i$ to $\frac{i+j}{2} - 1$ **do**
    $\mathbf{y}[t][\ell] \leftarrow \mathbf{y}[t-1][\ell] + \mathbf{y}[t-1][\ell + 2^{t-1}]$
  for $\ell = \frac{i+j}{2}$ to $j-1$ **do**
    $\mathbf{y}[t][\ell] \leftarrow \mathbf{y}[t-1][\ell]$

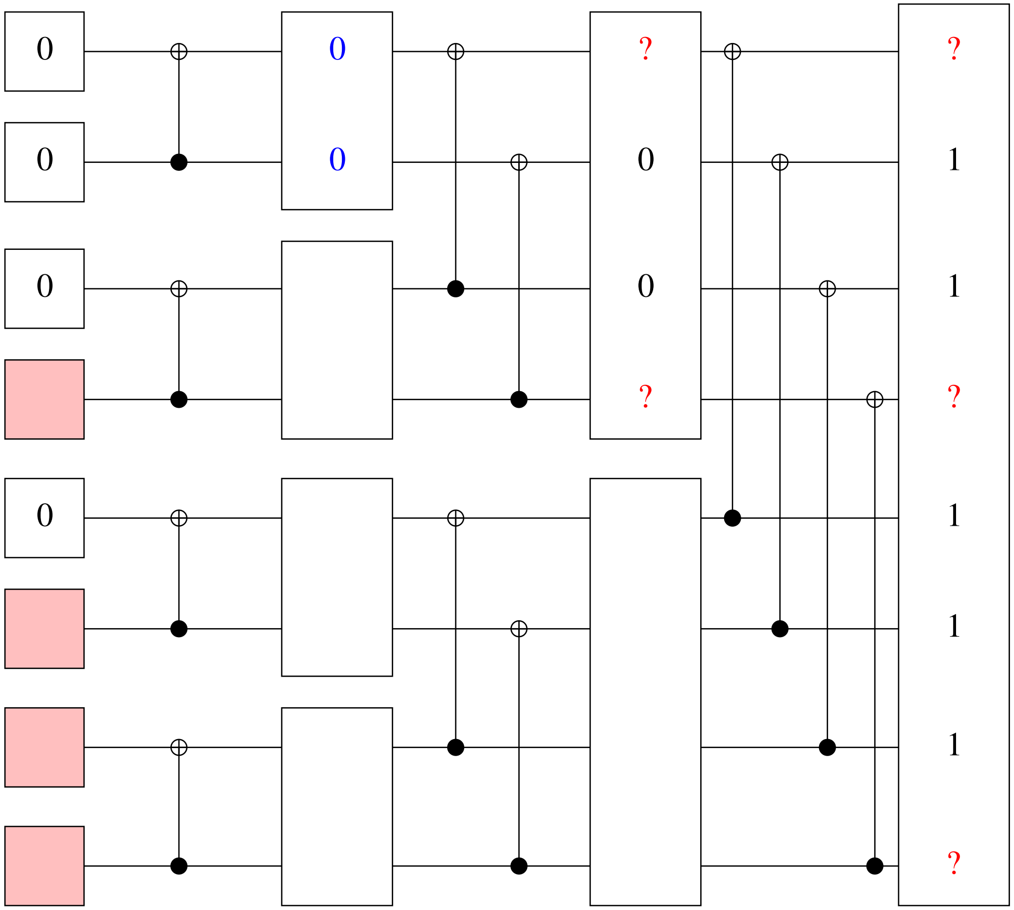
An example on the erasure channel



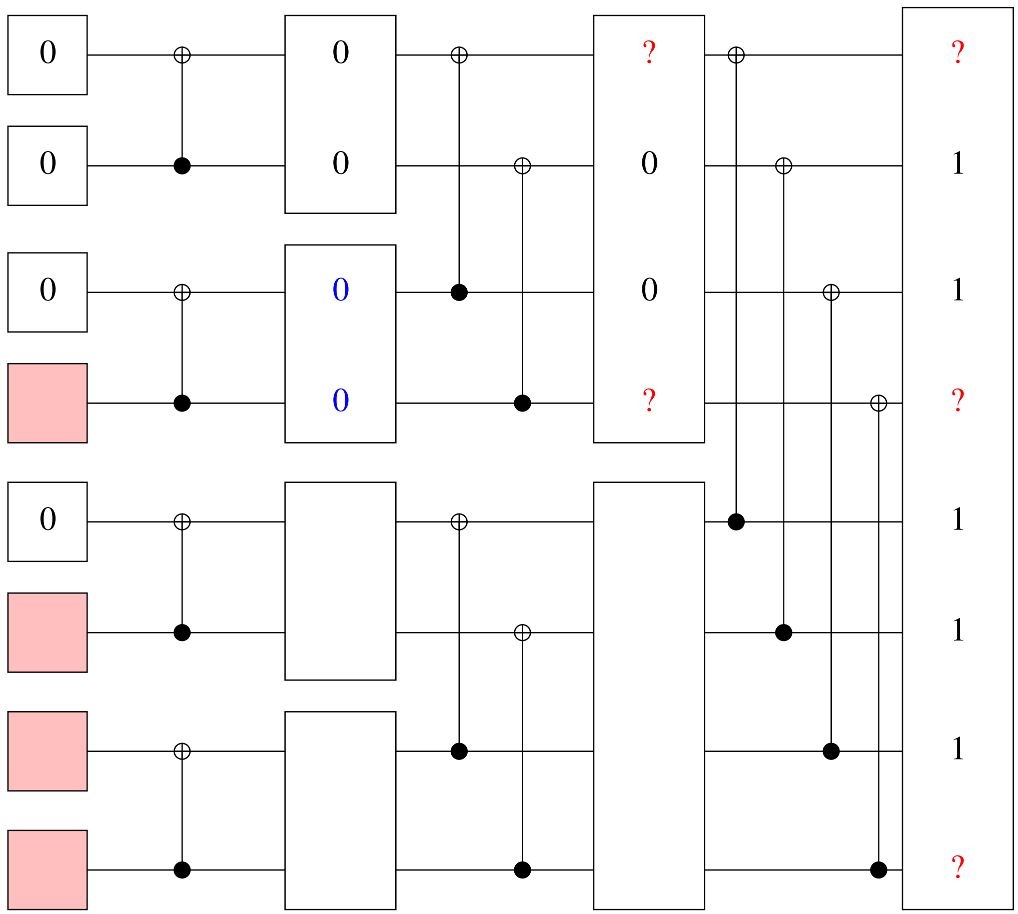
DecodeDirectly(0)



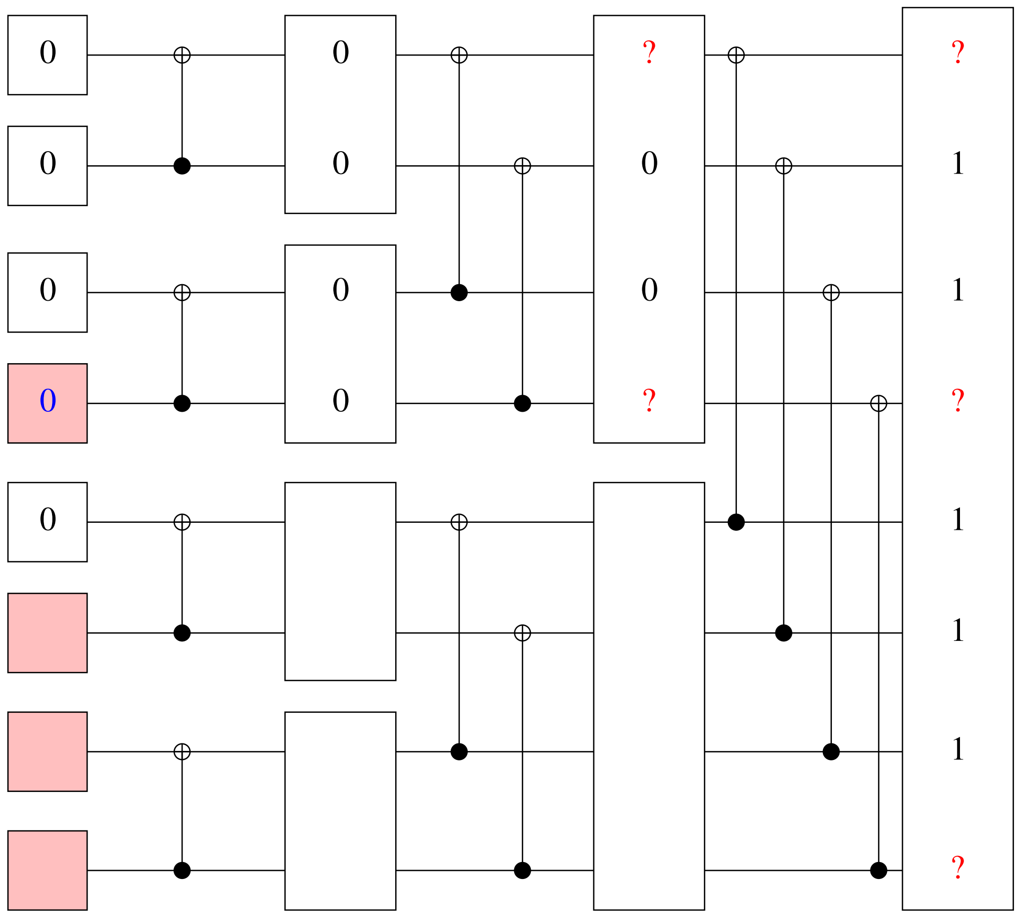
EncodeUV$(0,1,1)$



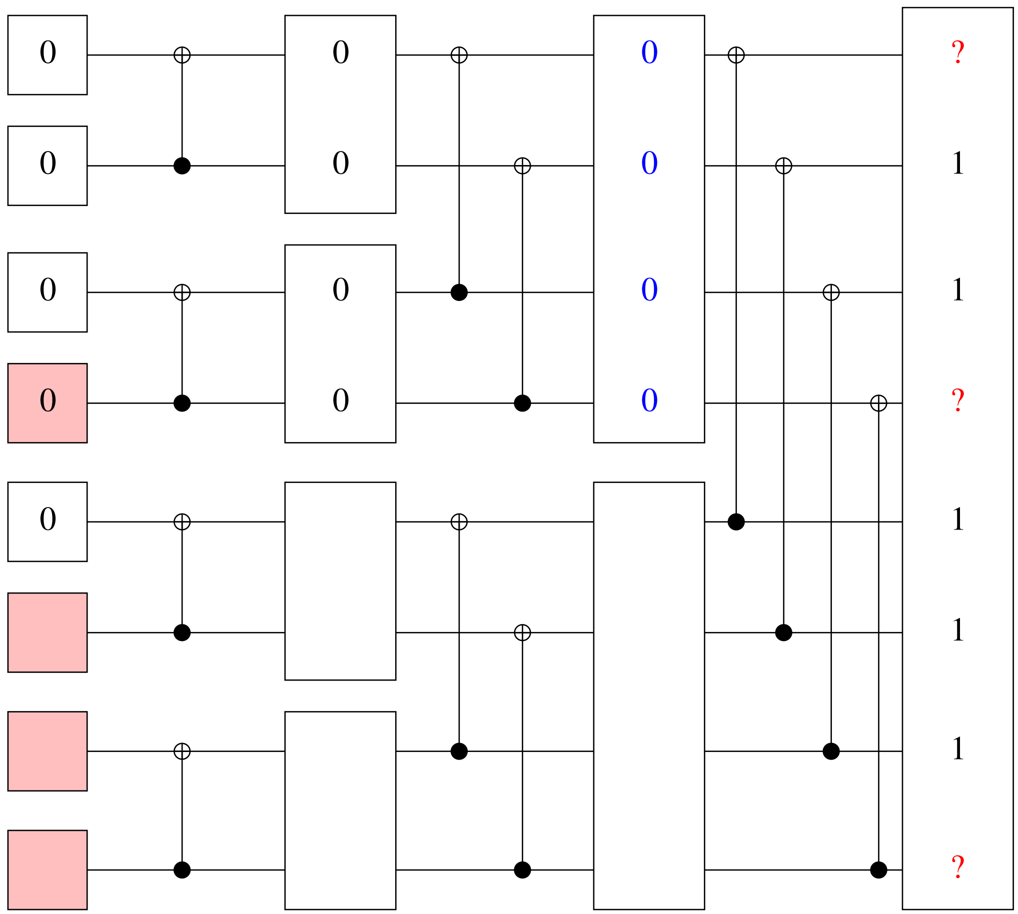
UpdateV$(1, 3, 1)$



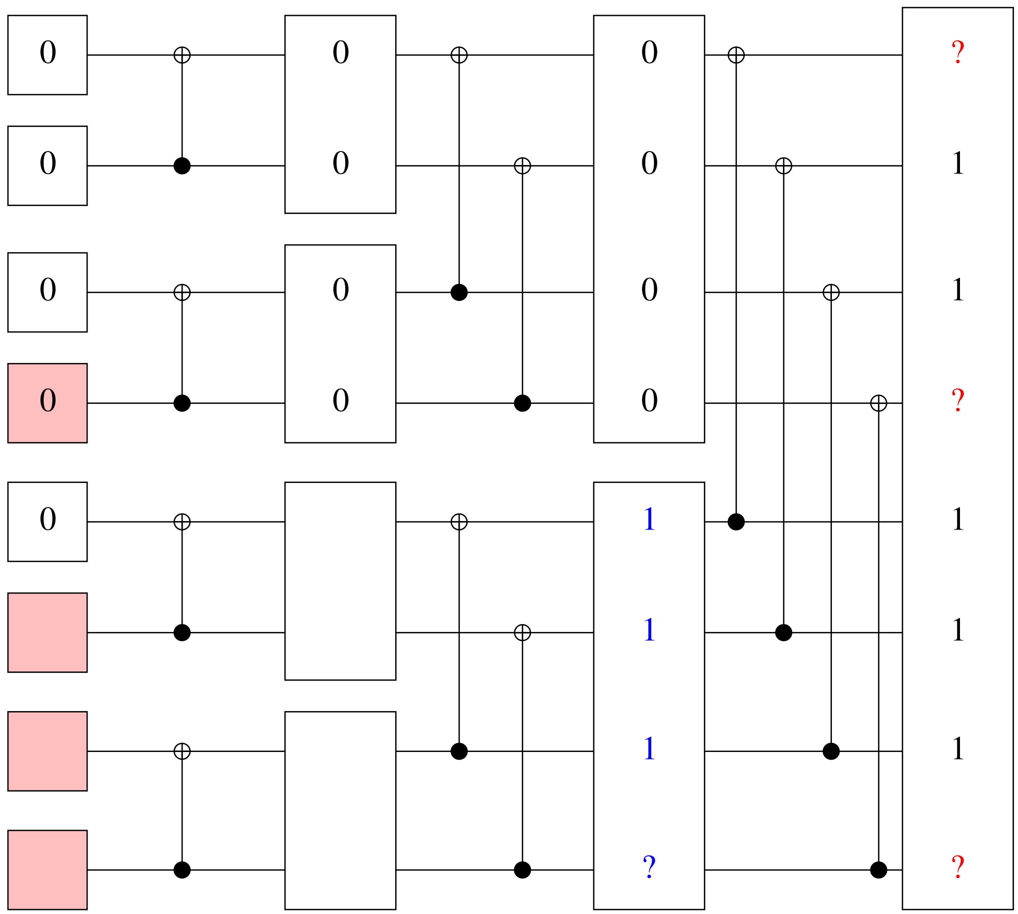
UpdateU$(3, 4, 0)$



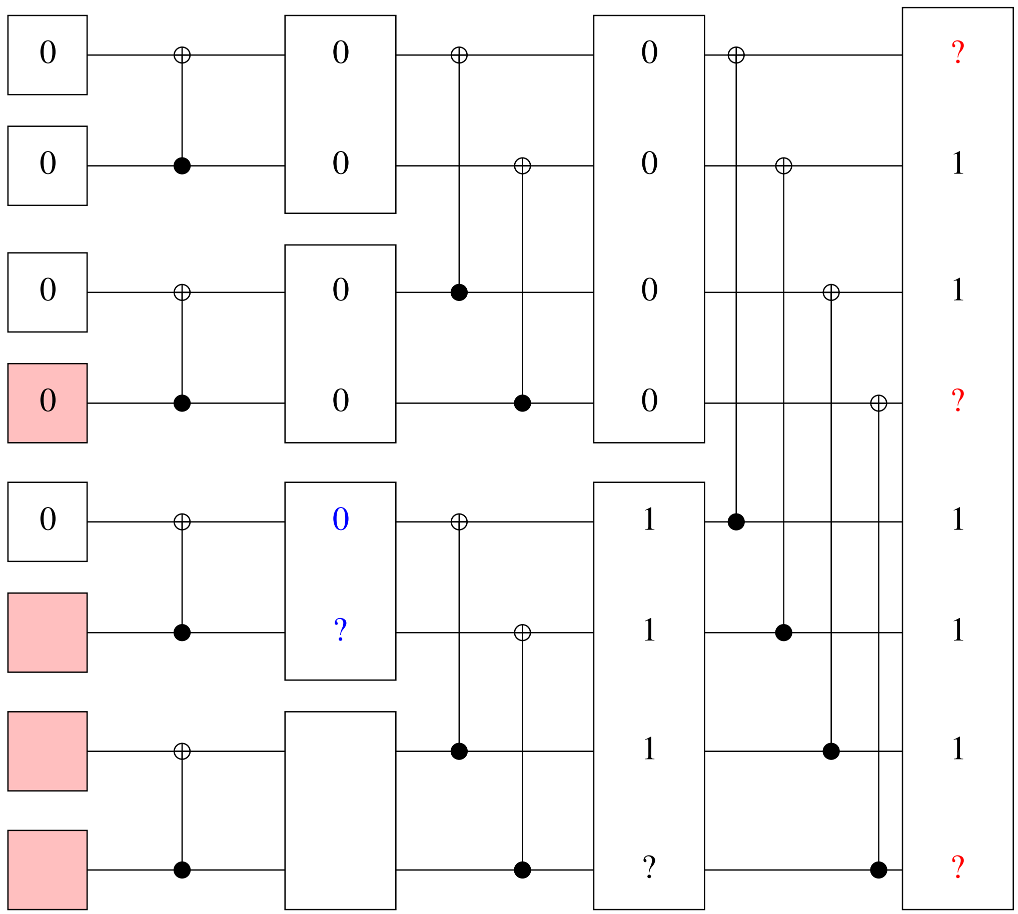
EncodeUV$(0, 4, 2)$



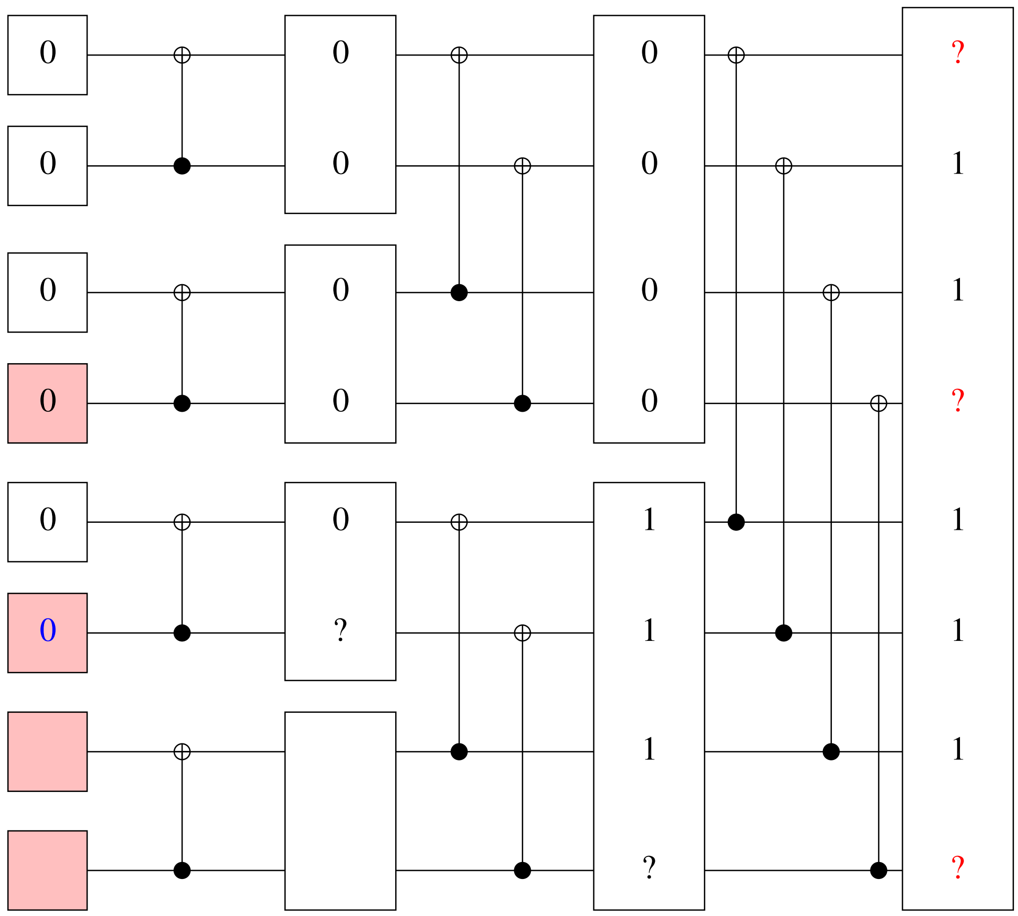
UpdateV$(4, 8, 2)$



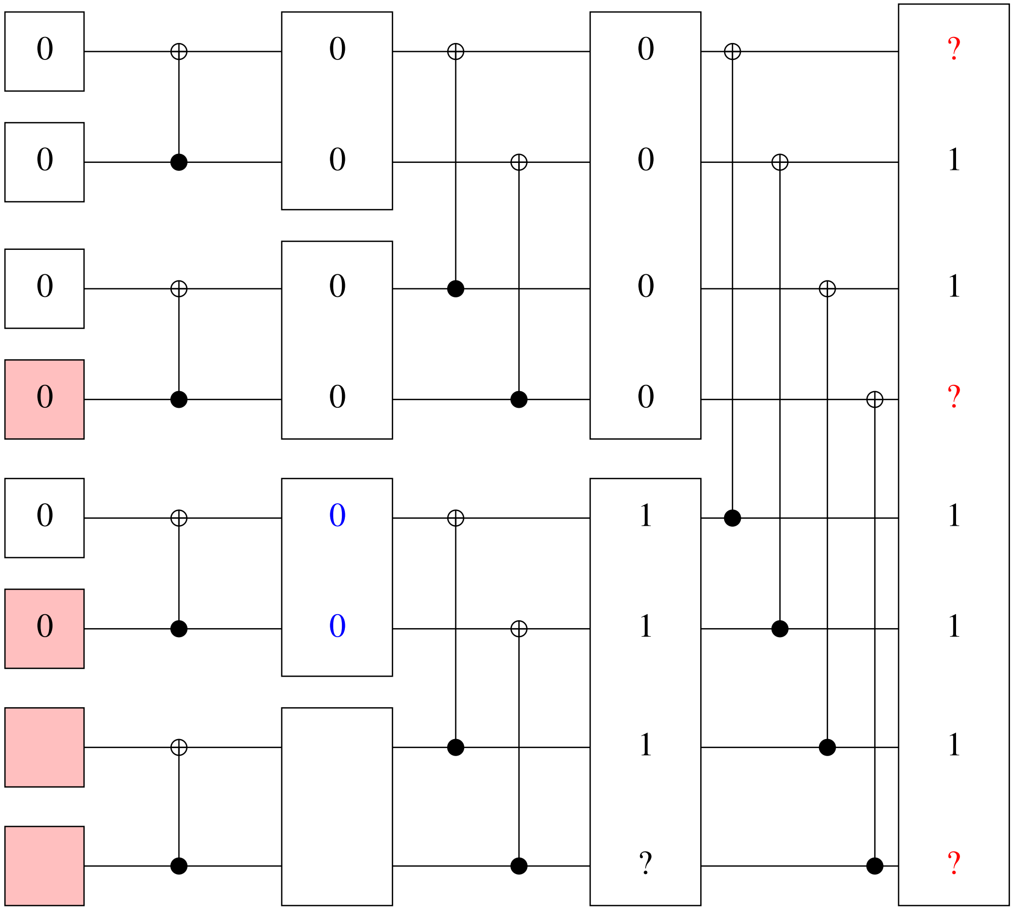
UpdateU$(4, 6, 1)$



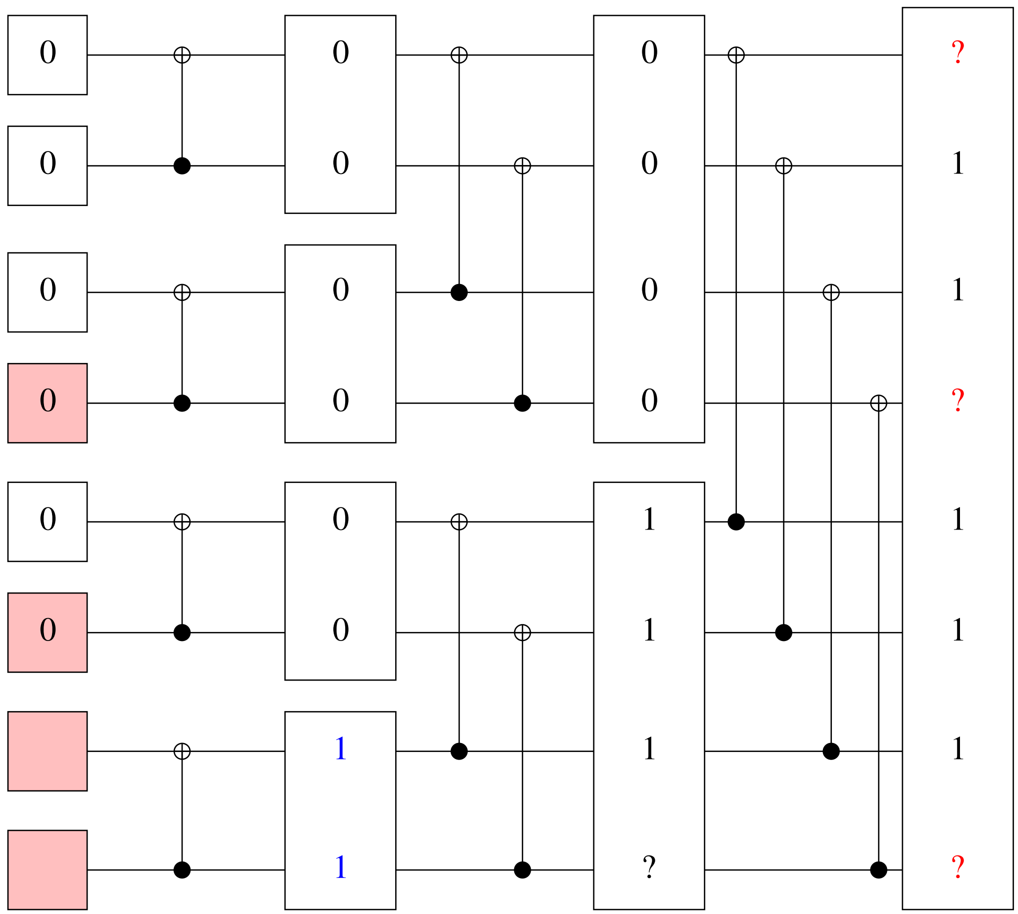
UpdateV$(5, 6, 0)$



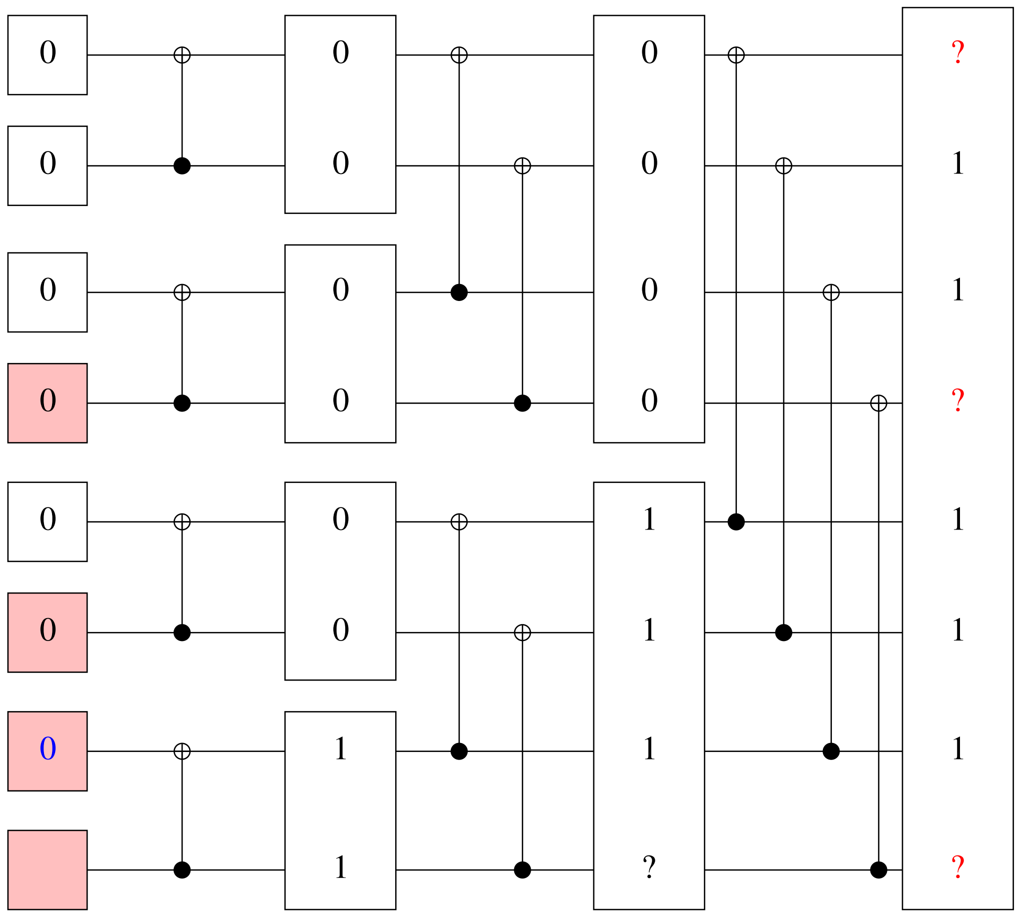
EncodeUV$(4, 6, 1)$



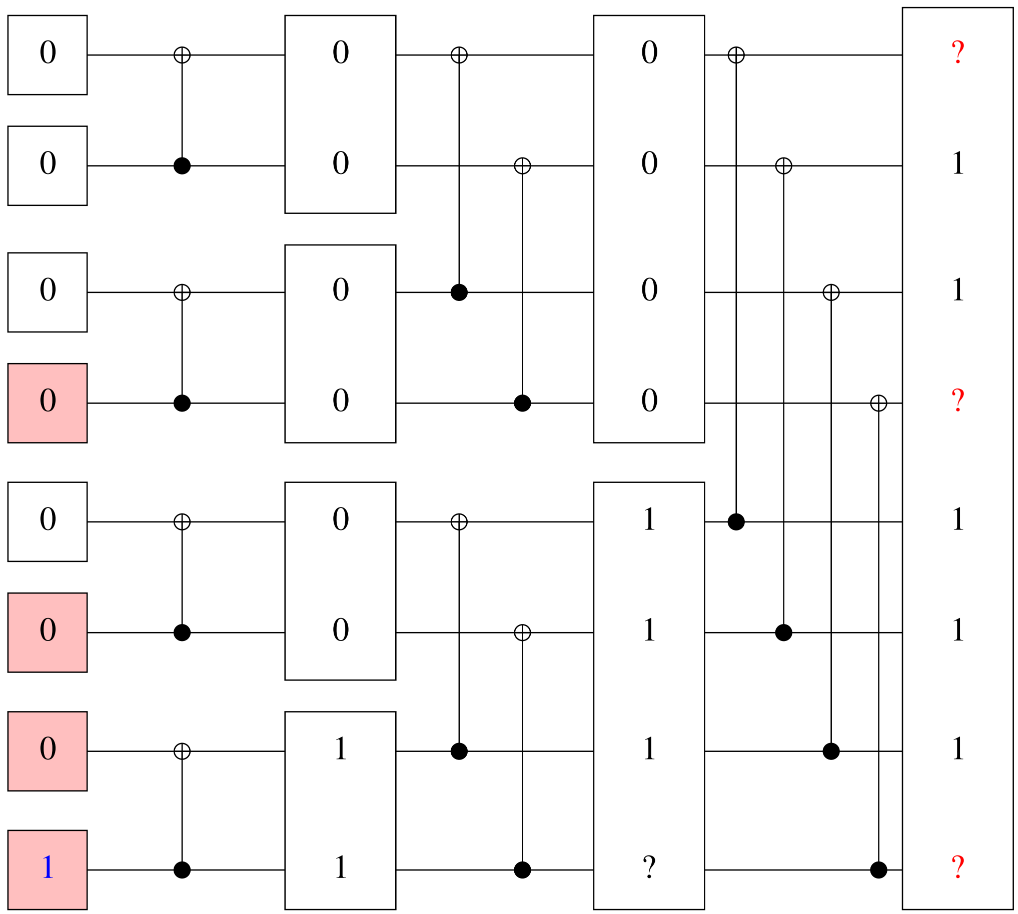
UpdateV$(6, 8, 1)$



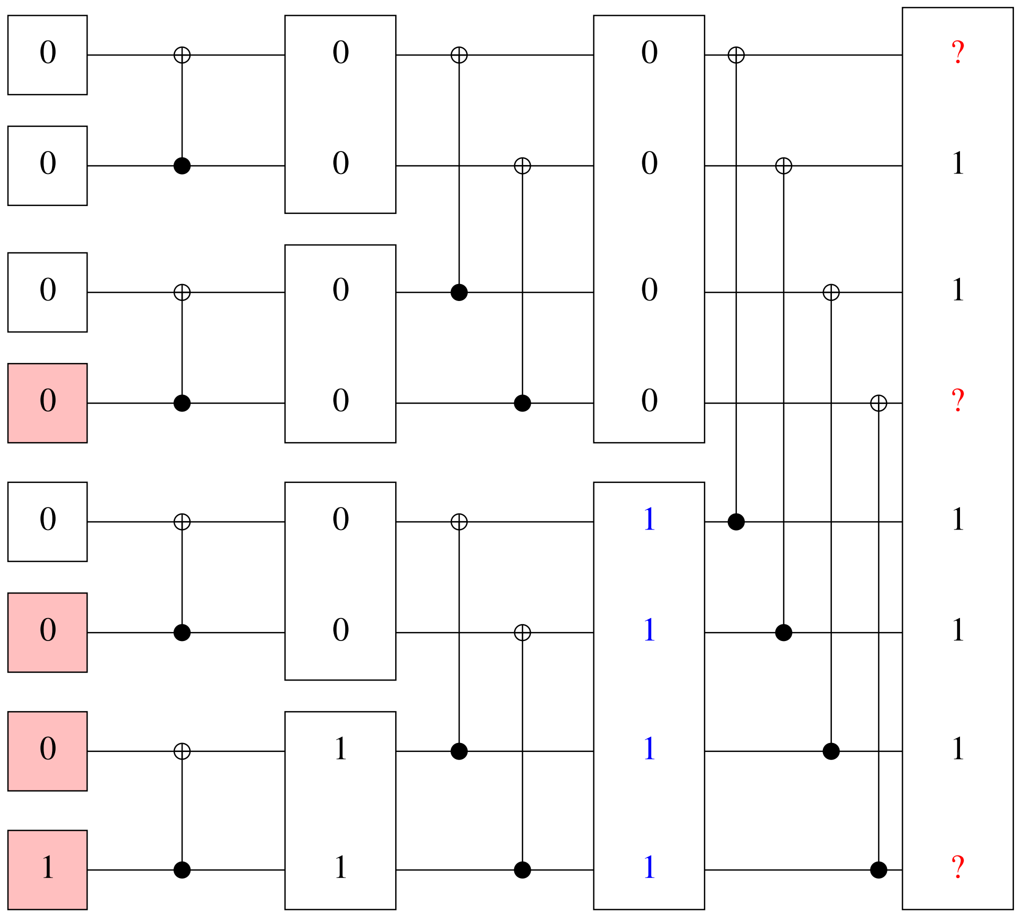
UpdateU$(6, 7, 0)$



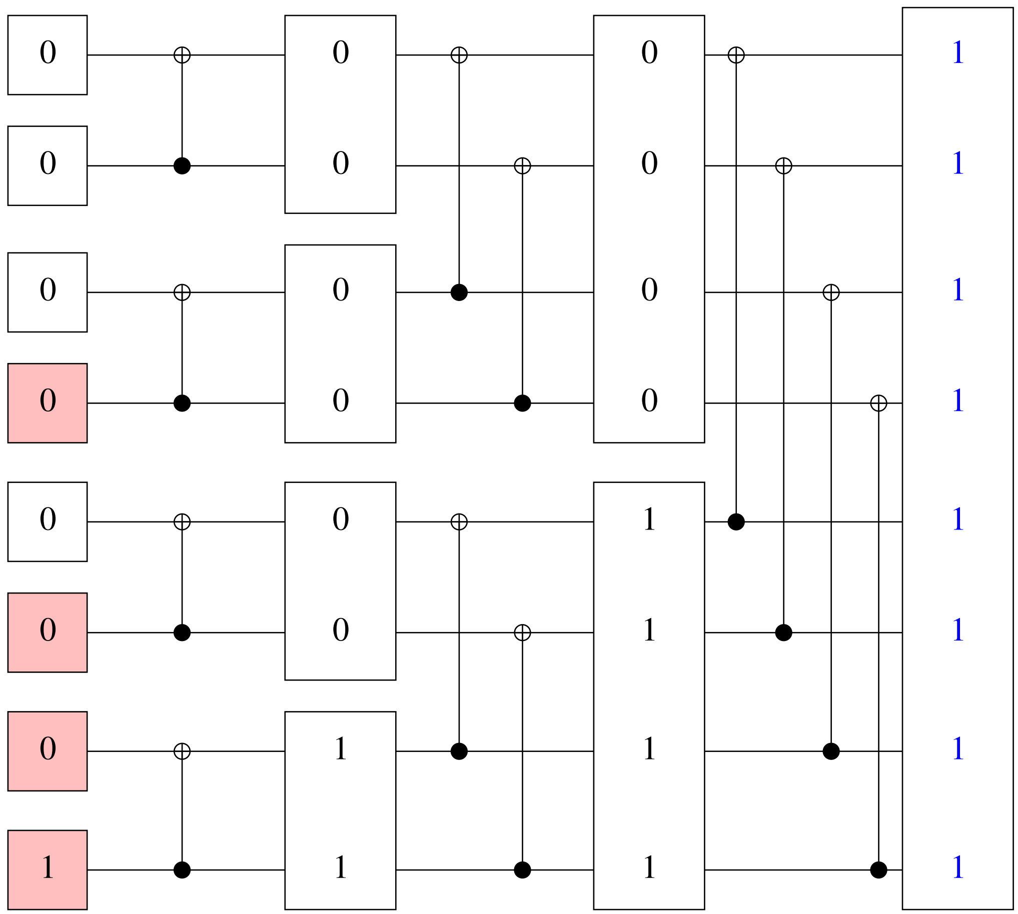
UpdateV$(7, 8, 0)$



EncodeUV$(4, 8, 2)$



EncodeUV$(0, 8, 3)$



Why does this work ?

- How to choose the positions where the input is fixed to 0, i.e. i such that $z_i = 0$?
- Why does this procedure work at all ?
- Generalizing this to other channels

Modeling the decoder

$$
\begin{array}{ccccc}
u & \overset{\text{coding}}{\rightarrow} & u+v & \overset{\text{channel}}{\rightsquigarrow} & y_1 \\
v & \overset{\text{coding}}{\rightarrow} & v & \overset{\text{channel}}{\rightsquigarrow} & y_2
\end{array}
$$

Decoding of the base code can be modeled by the following transmission over two different channels :

$$
\begin{array}{ccc}
u & \overset{\text{channel 1}}{\rightsquigarrow} & y_1, y_2 \\
v & \overset{\text{channel 2}}{\rightsquigarrow} & u, y_1, y_2
\end{array}
$$

and we know the channels, they provide $\mathbf{Prob}(u = 1|y_1, y_2)$ and $\mathbf{Prob}(v = 1|u_1, y_1, y_2)$.

Case of an erasure channel

Assume that

$$x_1 = u + v \quad \overset{\text{erasure channel of prob. } p_1}{\rightsquigarrow} \quad y_1$$

$$x_2 = v \quad \overset{\text{erasure channel of prob. } p_2}{\rightsquigarrow} \quad y_2$$

$$\begin{aligned}
\mathbf{Prob}(u \text{ stays erased after decoding }) &= \mathbf{Prob}(x_1 \oplus x_2 \text{ erased }) \\
&= \mathbf{Prob}(x_1 \text{ or } x_2 \text{ erased }) \\
&= 1 - (1 - p_1)(1 - p_2) \\
&= p_1 + p_2 - p_1 p_2 \\
\mathbf{Prob}(v \text{ stays erased after decoding}) &= \mathbf{Prob}(x_1 \text{ and } x_2 \text{ erased }) \\
&= p_1 p_2
\end{aligned}$$

Induced channel model in the case of the erasure channel

$$y_1, y_2 \overset{\text{decoding}}{\rightsquigarrow} \hat{u}$$

$$u, y_1, y_2 \overset{\text{decoding}}{\rightsquigarrow} \hat{v}$$

Induced channel model :

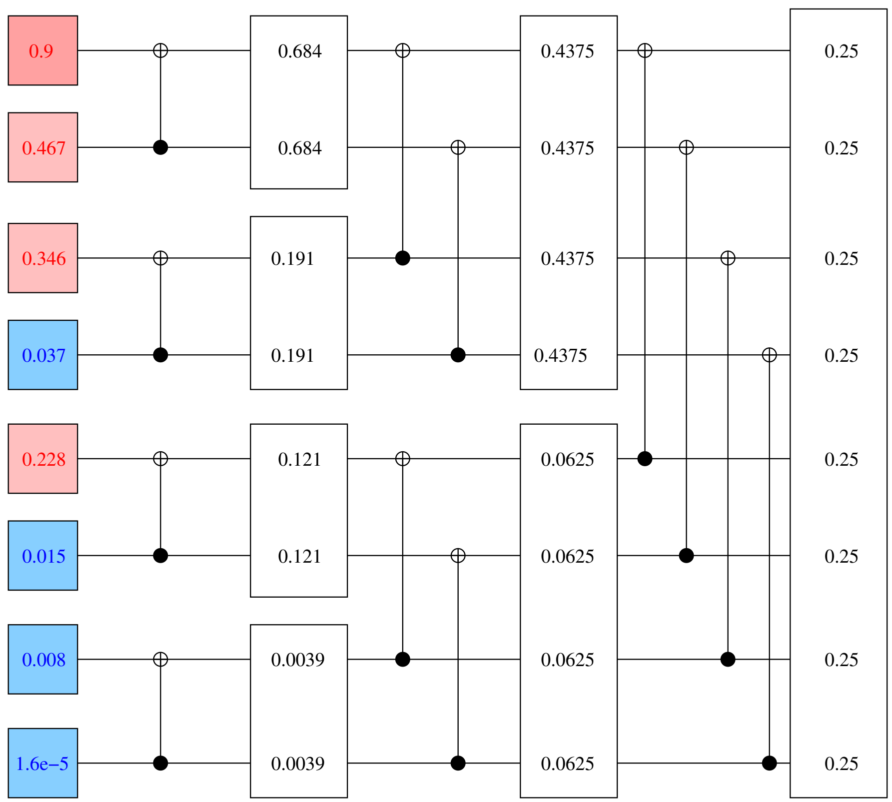
$$u \overset{\text{channel 1 of eras. prob. } p_1+p_2-p_1p_2}{\rightsquigarrow} \hat{u}$$

$$v \overset{\text{channel 2 of eras. prob. } p_1p_2}{\rightsquigarrow} \hat{v}$$

If we denote by $C(p)$ the capacity of the erasure channel of probability p ($C(p) = 1 - p$) then

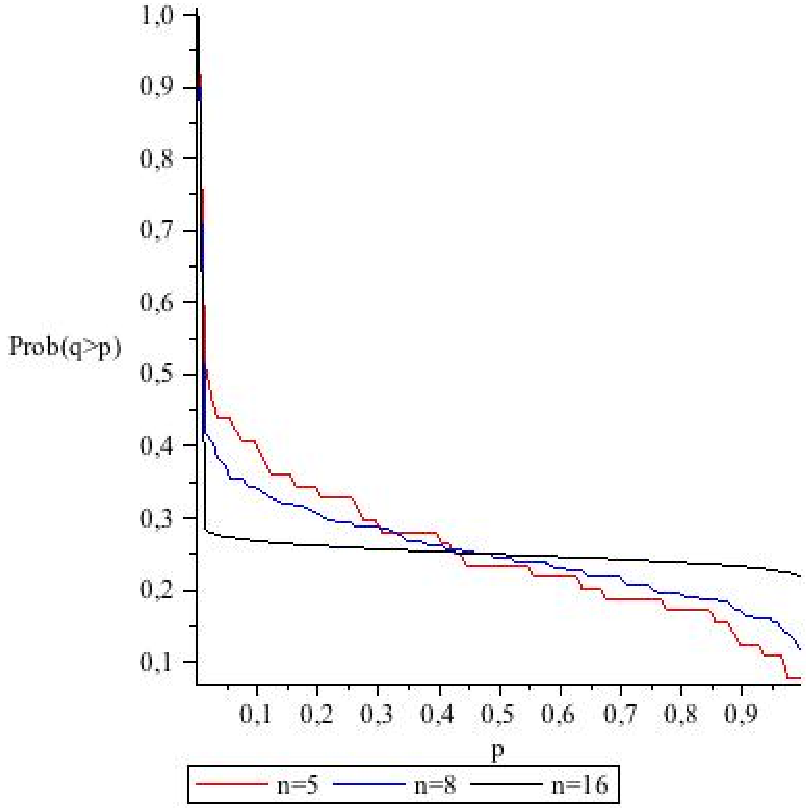
$$C(p_1) + C(p_2) = C(p_1 + p_2 - p_1p_2) + C(p_1p_2). \tag{1}$$

Equivalent models for $p = 0.25$ and $n = 3$



We choose the positions in red to be fixed to 0.

Equivalent models for $n \in \{5, 8, 16\}$



$$\text{"}\mathbf{Prob}(q > p)\text{"} = \frac{\#\{i : q > p\}}{2^n}$$

Why the whole scheme works and attains the capacity of the erasure channel

Point 1 : The equivalent channels "polarize", the erasure probability is either close to 0 or 1.

Point 2 : The "conservation law" (1) $C(p_1)+C(p_2) = C(p_1+p_2-p_1p_2)+C(p_1p_2)$ ensures that

$$\sum_{i=0}^{N-1} C(q_i) = \sum_{i=0}^{N-1} C(p_i) = NC(p)$$

with $q_i =$ capacity of the $i-$th equivalent channel at the input and $p_i =$ capacity of the $i-$th output channel.

Point 3 : Since either $C(q_i) \approx 0$ or $C(q_i) \approx 1$,

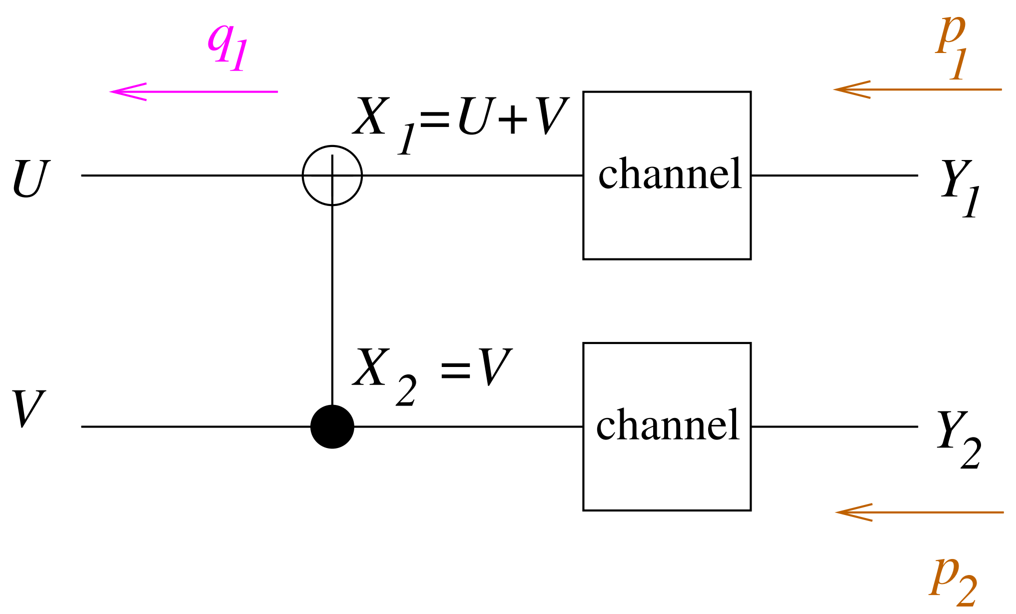
$$k \stackrel{\text{def}}{=} N - |\mathcal{F}| \stackrel{\text{def}}{=} \#\{i : C(q_i) \approx 1\} \approx NC(p)$$

Generalizing the procedure to other channels

$$(\mathbf{u}+\mathbf{v},\mathbf{v}) \overset{\text{channel}}{\rightsquigarrow} (\mathbf{y}_1,\mathbf{y}_2)$$

- $\textsc{UpdateU}$ computes the probability that the bits of $\mathbf{u}$ are equal to 1 knowing $\mathbf{y}_1$ and $\mathbf{y}_2$ respectively, whereas $\textsc{UpdateV}$ computes the probabilities of the symbols of $\mathbf{v}$ knowing $\mathbf{y}_1$, $\mathbf{y}_2$ and $\mathbf{u}$.

A simple computation



We know $p_1 = \mathbf{Prob}(x_1 = 1 | y_1)$ and $p_2 = \mathbf{Prob}(x_2 = 1 | y_2)$. We compute

$$q_1 = \mathbf{Prob}(u = 1 | y_1, y_2).$$

The formula

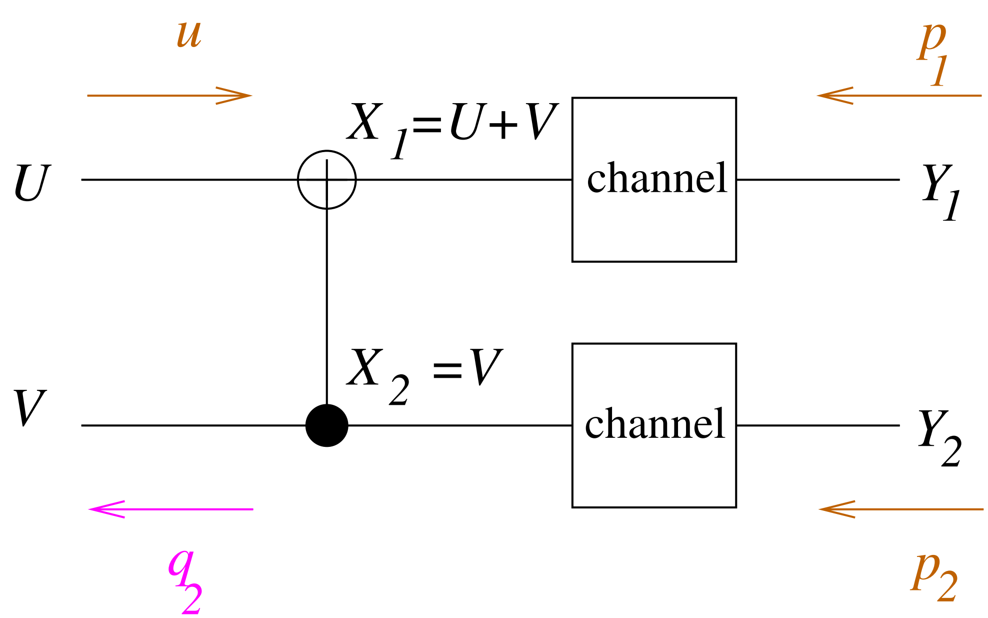
Lemma 1. *Let* X_1 *and* X_2 *be two independent binary random variables and denote by* $r_i \stackrel{\text{def}}{=} \mathbf{Prob}(X_1 = 1)$, *then*

$$\mathbf{Prob}(Z_1 \oplus Z_2 = 1) = \frac{1 - (1 - 2r_1)(1 - 2r_2)}{2}$$

Application :

$$q_1 = \frac{1 - (1 - 2p_1)(1 - 2p_2)}{2}$$

Another simple computation



We know $p_1 = \mathbf{Prob}(x_1 = 1|y_1)$, $p_2 = \mathbf{Prob}(x_2 = 1|y_2)$ and u_1. We compute $q_2 = \mathbf{Prob}(v = 1|u, y_1, y_2)$.

The formula

Lemma 2. *A uniformly distributed random bit B is sent through two memoryless channels, y_1 and y_2 are the corresponding outputs. If we denote by $r_i = \mathbf{Prob}(B = 1|y_i)$, then*

$$\mathbf{Prob}(B = 1|y_1, y_2) = \frac{r_1 r_2}{r_1 r_2 + (1 - r_1)(1 - r_2)}.$$

Application :

$$\begin{aligned} q_2 &= \frac{p_1 p_2}{p_1 p_2 + (1 - p_1)(1 - p_2)} \text{ if } u = 0 \\ q_2 &= \frac{(1 - p_1) p_2}{(1 - p_1) p_2 + p_1(1 - p_2)} \text{ if } u = 1 \end{aligned}$$

Decoding algorithm on general channels

- Sent codeword $\mathbf{x} = (x_i)_{0 \leq i < 2^n}$, received word $\mathbf{y} = (y_i)_{0 \leq i < 2^n}$
- matrix $\mathbf{p}[i][j]$, $i \in [0 \cdots n]$, $j \in [0 \cdots 2^n[$

$$\begin{aligned} \mathbf{p}[n][i] &\stackrel{\text{def}}{=} \mathbf{Prob}(x_i = 1 | y_1) \\ \mathbf{p}[t][i] &= \text{probability } i\text{-th bit at layer } t = 1, \text{ for } t < n \end{aligned}$$

Decoding algorithm

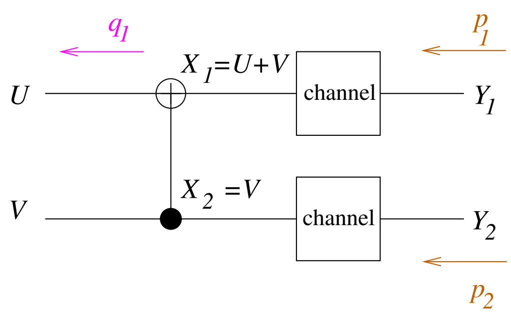
function $\textsc{Decode}(i, j, t)$
  if $t = 0$ **then**
    $\textsc{DecodeDirectly}(i)$
  else
    $m \leftarrow \frac{i+j}{2}$
    $\textsc{UpdateU}(i, m, t-1)$
    $\textsc{Decode}(i, m, t-1)$
    $\textsc{UpdateV}(m, j, t-1)$
    $\textsc{Decode}(m, j, t-1)$
    $\textsc{SetPositionsUV}(i, j, t)$

Decoding the last layer

function $\textsc{DecodeDirectly}(i)$
    if $\mathbf{z}[i] \neq ?$ **then**
        $\mathbf{p}[0][i] = \mathbf{z}[i]$
    else
        if $\mathbf{p}[0][i] < \frac{1}{2}$ **then** $\mathbf{p}[0][i] \leftarrow 0$
        else $\mathbf{p}[0][i] \leftarrow 1$

- The bits are fixed by fixing the probabilities either to 0 and 1

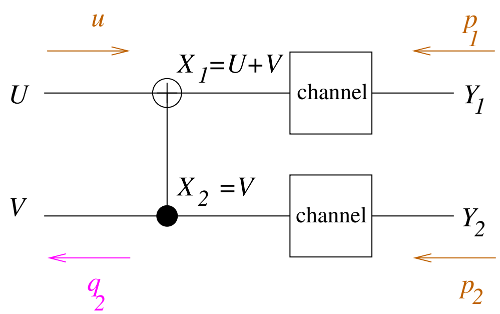
Updating the positions of the U code



$$q_1 = \frac{1-(1-2p_1)(1-2p_2)}{2}$$

function $\textsc{UpdateU}(i, j, t)$
  for $\ell = i$ to $j - 1$ **do**
    $\mathbf{p}[t][\ell] \leftarrow \frac{1-(1-2\mathbf{p}[t+1][\ell])(1-2p[t+1][\ell+2^t])}{2}$

Updating the positions of the V code

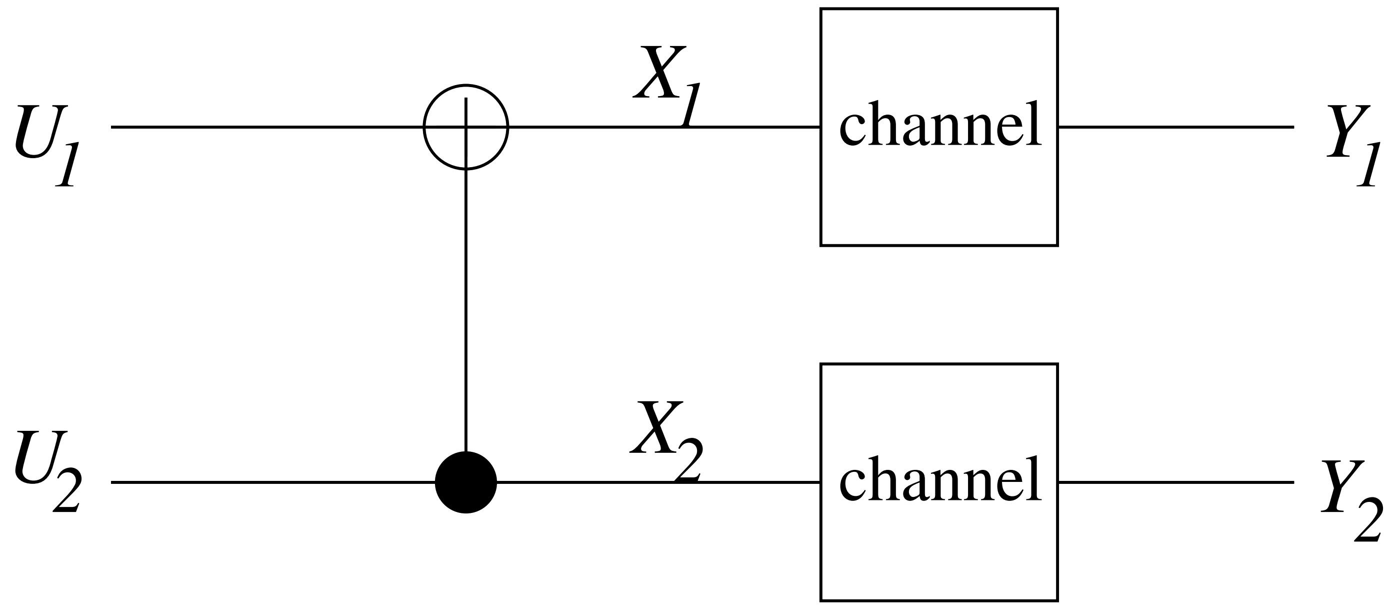


$$q_2 = \frac{p_1p_2}{p_1p_2+(1-p_1)(1-p_2)} \text{ if } u=0$$
$$= \frac{(1-p_1)p_2}{(1-p_1)p_2+p_1(1-p_2)} \text{ if } u=1$$

function $\textsc{UpdateV}(i,j,t)$
 for $\ell = i$ to $j-1$ **do**
  $p_1 \leftarrow \mathbf{p}[t+1][\ell - 2^t]$
  $p_2 \leftarrow \mathbf{p}[t+1][\ell]$
  if $\mathbf{p}[t][\ell-2^t] = 0$ **then**
   $\mathbf{p}[t][\ell] \leftarrow \frac{p_1p_2}{p_1p_2+(1-p_1)(1-p_2)}$
  else
   $\mathbf{p}[t][\ell] \leftarrow \frac{(1-p_1)p_2}{(1-p_1)p_2+p_1(1-p_2)}$

The general case : the basic scheme

Assumption : U_1 and U_2 independent and uniformly distributed in $\{0, 1\}$.



Same "conservation law" as for the erasure channel :

Theorem 1.

$$I(U_1; Y_1, Y_2) + I(U_2; U_1, Y_1, Y_2) = I(X_1; Y_1) + I(X_2; Y_2).$$

A lemma on the independence of random variables

Lemma 3. *U_1 and U_2 independent and uniformly distributed,*
$\Rightarrow X_1$ and X_2 independent and uniformly distributed
$\Rightarrow Y_1$ and Y_2 independent.

proof : X_1 and X_2 independent and uniformly distributed (obvious).

Proof (cont'd)

$$\begin{aligned}
\mathbf{P}(Y_1 = a, Y_2 = b) &= \sum_{c,d} \mathbf{P}(Y_1 = a, Y_2 = b | X_1 = c, X_2 = d)\mathbf{P}(X_1 = c, X_2 = d) \\
&= \sum_{c,d} \mathbf{P}(Y_1 = a | X_2 = c)\mathbf{P}(Y_2 = b | X_2 = d)\mathbf{P}(X_1 = c)\mathbf{P}(X_2 = d) \\
&= S_1 S_2 \quad \text{with}
\end{aligned}$$

$$\begin{aligned}
S_1 &= \sum_{c} \mathbf{P}(Y_1 = a | X_1 = c)\mathbf{P}(X_1 = c) = P(Y_1 = a) \\
S_2 &= \sum_{d} \mathbf{P}(Y_2 = b | X_2 = d)\mathbf{P}(X_2 = d) = P(Y_2 = b)
\end{aligned}$$

Hence

$$\mathbf{P}(Y_1 = a, Y_2 = b) = P(Y_1 = a)P(Y_2 = b)$$

An important lemma in information theory

Lemma 4. *If Y_i is the output corresponding to X_i after transmission through a memoryless channel*

$$I(X_1, X_2; Y_1, Y_2) \leq I(X_1; Y_1) + I(X_2; Y_2).$$

If Y_1 and Y_2 are independent

$$I(X_1, X_2; Y_1, Y_2) = I(X_1; Y_1) + I(X_2; Y_2).$$

Proof

$$
\begin{aligned}
I(X_1, X_2; Y_1, Y_2) &= H(Y_1, Y_2) - H(Y_1, Y_2 | X_1, X_2) \\
&\quad \text{(definition of mutual information)} \\
&= H(Y_1) + H(Y_2) - H(Y_1 | X_1, X_2) - H(Y_2 | X_1, X_2, Y_1) \\
&\quad \text{(independence of the } Y_i\text{'s)} \\
&= H(Y_1) + H(Y_2) - H(Y_1 | X_1) - H(Y_2 | X_2) \\
&\quad \text{(memoryless channel)} \\
&= I(X_1; Y_1) + I(X_2; Y_2) \\
&\quad \text{(definition of mutual information)}
\end{aligned}
$$

Proof of Theorem 1

$$
\begin{aligned}
I(X_1;Y_1) + I(X_2;Y_2) &= I(X_1, X_2; Y_1, Y_2) \\
&= I(U_1, U_2; Y_1, Y_2) \\
&= H(U_1, U_2) - H(U_1, U_2|Y_1, Y_2) \\
&= H(U_1) + H(U_2) - H(U_1|Y_1, Y_2) - H(U_2|U_1, Y_1, Y_2) \\
&= I(U_1; Y_1, Y_2) + I(U_2; U_1, Y_1, Y_2)
\end{aligned}
$$