Computer Science 1001.py

Lecture 25[†]:
Intro to Error Correction and Detection Codes

Instructors: Daniel Deutch, Amiram Yehudai
Teaching Assistants: Michal Kleinbort, Amir Rubinstein

School of Computer Science
Tel-Aviv University
Spring Semester, 2013-14
http://tau-cs1001-py.wikidot.com

Simple Detection/Correction Codes

- Repetition codes
- Parity check codes
- Hamming codes

Reminder: Hamming Distance

Richard W. Hamming (1915 –1998).

- Let $x, y \in \Sigma^n$ be two length n words over alphabet Σ. The Hamming distance between x, y is the number of coordinates where they differ.
- The Hamming distance satisfies the three usual requirements from a distance function
  1. For every x, $d(x, x) = 0$.
  2. For every x, y, $d(x, y) = d(y, x) \geq 0$, with equality iff $x = y$.
  3. For every x, y, z, $d(x, y) + d(y, z) \geq d(x, z)$ (triangle inequality).

  where $x, y, z \in \Sigma^n$ (same length).
- Examples
  1. $d(00101, 00101) = 0$
  2. $d(00101, 11010) = 5$ (maximum possible for length 5 vectors)
  3. $d(00101, 1101011)$ is undefined (unequal lengths).
  4. $d(\texttt{BEN}, \texttt{RAN}) = 2$

Repetition Codes

In the following code, the original messages are two bits long. The encoder repeats each original bit, two more times.

original		encoded
2 bits		6 bits
00	$\longrightarrow$	000000
01	$\longrightarrow$	000111
10	$\longrightarrow$	111000
11	$\longrightarrow$	111111

Repetition Codes

In the following code, the original messages are two bits long. The encoder repeats each original bit, two more times.

original		encoded
2 bits		6 bits
00	$\longrightarrow$	000000
01	$\longrightarrow$	000111
10	$\longrightarrow$	111000
11	$\longrightarrow$	111111

This code can correct any single error. For example, suppose the decoder recieves 001000. The single flipped bit can only be 001000, leading to the codeword 000000 and original message 00.

Remark: Of course, the channel is not aware of a difference between "regular" (original) bits and "blue" (parity check) bits.

Hamming Distances in the Repetition Code

- The minimum distance of the Repetition Code is 3.
- So the Repetition Code is capable of correcting any single digit error.
- But some combinations of two digit errors can also be corrected: when one of the errors is in the first 3 bits, and one in the last 3 bits.
- Other combinations of two digit errors cannot be detected - they look just like a single error!
- So its best to treat each block of 3 bits separately.
- What is the minimum distance of a Repetition code for a single bit source (3 bit codewords)?

Decoding Repetition Codes

So will decode blocks of 3 bits.

received signal		decoded message
3 bits		1 bit
$000, 100, 010, 001$	$\longrightarrow$	0
$111, 011, 101, 110$	$\longrightarrow$	1

Decoding Repetition Codes

So will decode blocks of 3 bits.

received signal		decoded message
3 bits		1 bit
$000, 100, 010, 001$	$\longrightarrow$	0
$111, 011, 101, 110$	$\longrightarrow$	1

Decoding rule: If the received 3 bit signal is at Hamming distance 0 or 1 from one of the two codewords, the decoded message is the original message encoded by this codeword. This covers all possible cases.

So if the decoder receives 001100. It can identify the codeword as 000000 and the original message is 00.

Decoding the Repetition Code in Python

A simple solution uses table lookup, implemented via a dictionary:

```
decoder={"000":"0","100":"0","010":"0","001":"0",
         "111":"1", "011":"1","101":"1","110":"1"}

# repetition code decoder for 3 bit blocks
```

Decoding the Repetition Code in Python

The decode function can be coded as

```
def decode(word,dictio=decoder):
    if word in decoder:
        return decoder[word]
    else:
        return "error"    # does not occur for this code
>>> decode("000")
'0'
>>> decode("001")
'0'
```

In our case, there are 8 words. Such a size is OK to create a decoding dictionary manually.

For larger codes, manually coding table lookup becomes infeasible.

Parity Check Codes

In the following code, the original messages are two bits long. The encoder xors the two original bit (*i.e.* adds them modulo 2). The resulting bit is appended to the original message.

original	encoded
00	000
01	011
10	101
11	110

Parity Check Codes: Encoding

original	encoded
00	000
01	011
10	101
11	110

This code can detect any single error. But it cannot correct a single error. For example, suppose the decoder receives 001. The single flipped bit could be either 001 (encoded signal 000) or 001 (encoded signal 011) or 001 (encoded signal 101).

Decoding Parity Check Codes

received signal		decoded message
000	$\longrightarrow$	00
001	$\longrightarrow$	"error"
010	$\longrightarrow$	"error"
011	$\longrightarrow$	01
100	$\longrightarrow$	"error"
101	$\longrightarrow$	10
110	$\longrightarrow$	11
111	$\longrightarrow$	"error"

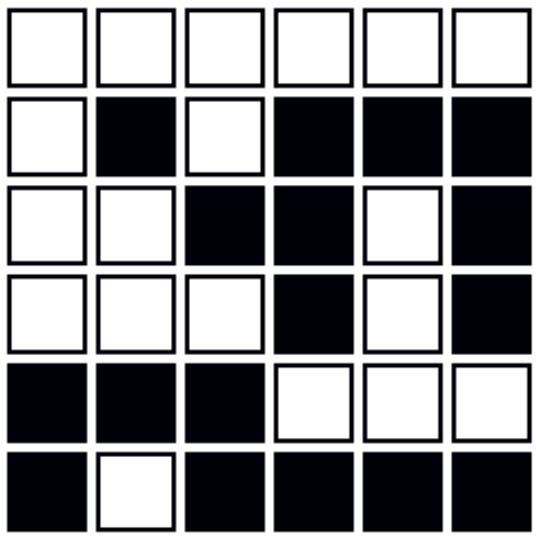
Decoding rule

If the received signal is one of the four codewords, decoded message is the original message encoded by this codeword. Otherwise, return "error".

Decoding the Card Magic Code

The card magic code "resides" in $\{0,1\}^{36}$, and has $\{0,1\}^{25}$ codewords. Decoding by table look-up is infeasible – even storing just the codewords in a dictionary is highly cumbersome, due to the large number of them.

But as we saw in class, there is a simple algorithm that can identify and correct any single error.



Repetition Code and Parity Check

- The repetition code we saw is hardly ever used – it expands messages threefold.
- The parity check code is in use, but it cannot correct even one error.
- Why can't the parity check code correct even a single error?

Repetition Code and Parity Check

- The repetition code we saw is hardly ever used – it expands messages threefold.
- The parity check code is in use, but it cannot correct even one error.
- Why can't the parity check code correct even a single error?
- What is the minimum distance of the parity check code?

- We will see a more effective code, named the Hamming code.
- But before this, we will formalize the role of the Hamming distance in the theory of error correction and detection codes, and generalize our observations.

Definitions and Properties

An encoding, E, from k to n ($k < n$) bits, is a one-to-one mapping $E : \{0,1\}^k \mapsto \{0,1\}^n$. The set of codewords is the set $C = \{y \in \{0,1\}^n \mid \exists x \in \{0,1\}^k, E(x) = y\}$. The set C is often called the code.

Let $\Delta(y,z)$ denote the Hamming distance between y, z.
Let $y \in \{0,1\}^n$. The sphere of radius r around y is the set $B(y,r) = \{z \in \{0,1\}^n \mid \Delta(y,z) \leq r\}$.

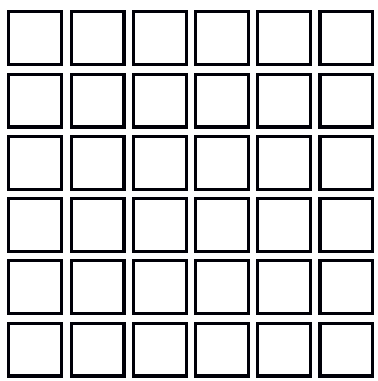
The minimum distance of a code, C, is $\Delta(C) = \min_{y \neq z \in C} \{\Delta(y,z)\}$.

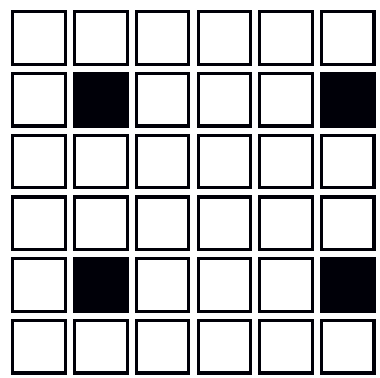
Minimum Distances of Codes

The minimum distance of a code, C, is $\Delta(C) = \min_{y \neq z \in C} \{\Delta(y, z)\}$.

In words: The minimum distance of the code, C, is the minimal Hamming distance over all pairs of codewords in C.

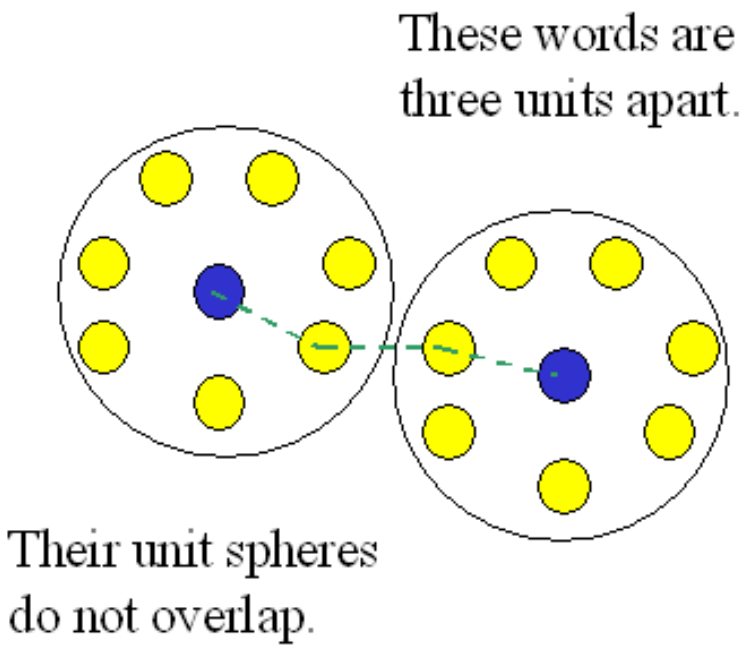
- For the parity check code, $\Delta(C) = 2$.
- For the ID code, $\Delta(C) = 2$
- For the repetition code, $\Delta(C) = 3$.
- For the 6-by-6 cards codes, $\Delta(C) = 4$.





An Important Observation

Proposition: Suppose $d = \Delta(C)$.
Then if the code C is used for error detection only, it is capable of detecting up to $d - 1$ errors.
Alternatively, the code is capable of correcting up to $\lfloor (d-1)/2 \rfloor$ errors.



(figure from course EE 387, by John Gill, Stanford University, 2010.)

Rate and Distance of a Code

Let $E : \{0,1\}^k \mapsto \{0,1\}^n$ be an encoding that gives rise to a code C whose minimum distance equals d.

We say that such C is an (n, k, d) code.

The rate of the code is k/n. It measures the ratio between the number of information bits, k, and transmitted bits, n.

The relative distance of the code is d/n.

The repetition code we saw is a $(6, 2, 3)$ code. Its rate is $2/6 = 1/3$. Its relative distance is $3/6 = 1/2$.
The parity check code is a $(3, 2, 2)$ code. Its rate and relative distance are both $2/3$.

Goals in Code Design

- Large rate, the closer to 1 the better (less communication overhead).
- Large relative distance (related to lower error in decoding).

Goals in Code Design

- Large rate, the closer to 1 the better (less communication overhead).
- Large relative distance (related to lower error in decoding).
- Efficient encoding.
- Efficient decoding.

Many useful codes employ linear encoding, namely the transformation $E : \{0,1\}^k \mapsto \{0,1\}^n$ is linear, and can be computed as a vector-by-matrix (so called generator matrix) multiplication.

Decoding may be hard even if encoding is linear. Algorithmically, minimum codeword decoding may require searching over a large space of codewords, and may thus require time that is exponential in k. This is not a problem only for small values of k. For larger values, it is highly desirable to have efficient decoding as well.

The Volume Bound

- Let C be a (n, k, d) code. Then $2^k \cdot \sum_{\ell=0}^{\lfloor (d-1)/2 \rfloor} \binom{n}{\ell} \leq 2^n$.
- This is called the volume, sphere packing, or Hamming, bound.
- Proof idea: Spheres at radius $\lfloor (d-1)/2 \rfloor$ around the 2^k codewords are all disjoint. Thus the "volume" of their union cannot be greater than the "volume" of the whole space, which is 2^n.
- Codes satisfying Hamming bound with equality are called perfect codes.

The Singleton Bound

- Let C be a (n, k, d) code. Then $d \leq n - k + 1$
- This is called the Singleton bound (R.C. Singleton, 1964),
  - Proof idea: "Project" all 2^k codewords on any $k - 1$ coordinates (say the first). There are fewer combinations than number of codewords, thus at least two codewords must share the same values in all these $k - 1$ coordinates. The two codewords are at distance at least d. Thus the remaining $n - k + 1$ coordinates must have "enough room" for this distance.
- Codes that achieve equality in Singleton bound are called MDS (maximum distance separable) codes.

Hamming Code

The Hamming encoder gets an original message consisting of 4 bits, and produces a 7 bit long codeword. For reasons to be clarified soon, we will number the bits in the original message in a rather unusual manner. For $(x_3, x_5, x_6, x_7) \in Z_2^4$,

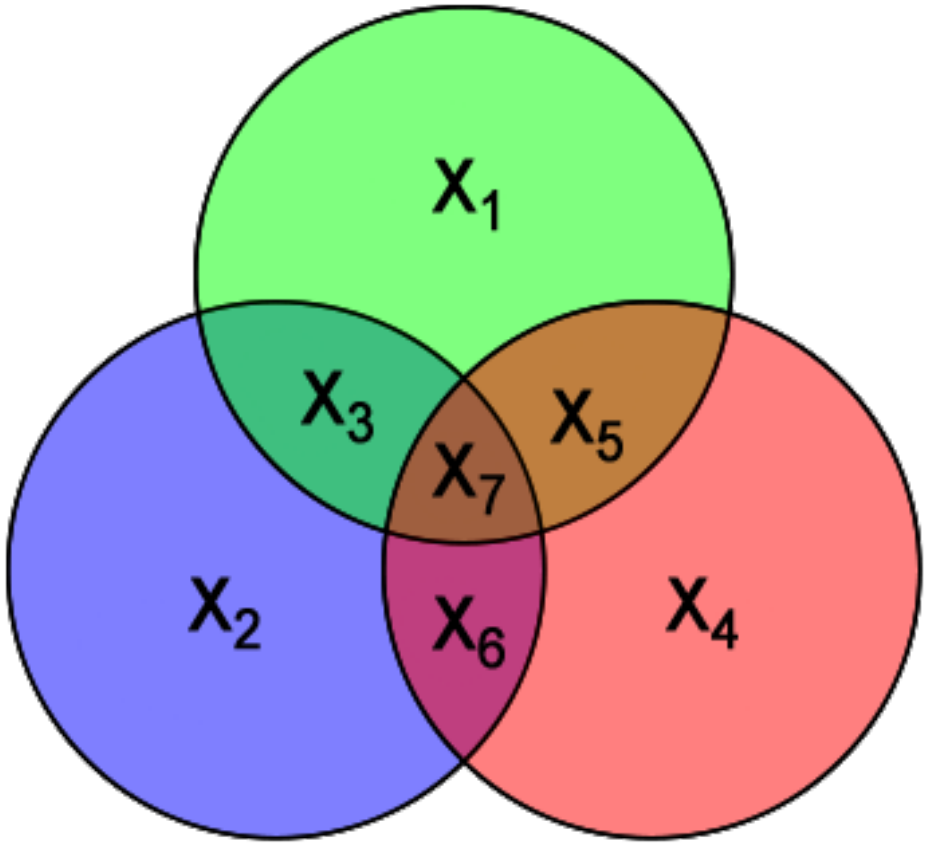
$$(x_3, x_5, x_6, x_7) \longrightarrow \quad (x_1, x_2, x_3, x_4, x_5, x_6, x_7) \ ,$$

where x_1, x_2, x_4 are parity bits, computed (modulo 2) as following:

$$\begin{aligned} x_1 &= x_3 + x_5 + x_7 \\ x_2 &= x_3 + x_6 + x_7 \\ x_4 &= x_5 + x_6 + x_7 \end{aligned}$$

What is n, k, d for this code?

Hamming Code: A Graphical Depiction



(image taken from Wikipedia)

The d_i in the image are data bits, x_3, x_5, x_6, x_7 in our notation.

The p_i in the image are parity check bits, x_1, x_2, x_4 in our notation.

Decoding Hamming Code

Let $(y_1, y_2, y_3, y_4, y_5, y_6, y_7)$ be the 7 bits signal received by the decoder. Assume that at most one error occurred by the channel. This means that the sent message, $(x_1, x_2, x_3, x_4, x_5, x_6, x_7)$ differs from the received signal in at most one location (bit).

Let the bits b_1, b_2, b_3 be defined (modulo 2) as following

$$\begin{aligned} b_1 &= y_1 + y_3 + y_5 + y_7 \\ b_2 &= y_2 + y_3 + y_6 + y_7 \\ b_3 &= y_4 + y_5 + y_6 + y_7 \end{aligned}$$

Decoding Hamming Code

Let $(y_1, y_2, y_3, y_4, y_5, y_6, y_7)$ be the 7 bits signal received by the decoder. Assume that at most one error occurred by the channel. This means that the sent message, $(x_1, x_2, x_3, x_4, x_5, x_6, x_7)$ differs from the received signal in at most one location (bit).

Let the bits b_1, b_2, b_3 be defined (modulo 2) as following

$$\begin{aligned} b_1 &= y_1 + y_3 + y_5 + y_7 \\ b_2 &= y_2 + y_3 + y_6 + y_7 \\ b_3 &= y_4 + y_5 + y_6 + y_7 \end{aligned}$$

Writing the three bits (from left to right) as $b_3 b_2 b_1$, we get the binary representation of an integer ℓ in the range $\{0, 1, 2, \ldots, 7\}$.

Decoding Rule:

- Interpret $\ell = 0$ as "no error" and return bits 3,5,6,7 of signal received.
- Interpret other values as "error in position ℓ ", and flip y_ℓ. Return bits 3,5,6,7 of the result.

Decoding Hamming Code: Why Does It Work?

Recall the bits b_1, b_2, b_3 be defined (modulo 2) as following

$$\begin{aligned} b_1 &= y_1 + y_3 + y_5 + y_7 \\ b_2 &= y_2 + y_3 + y_6 + y_7 \\ b_3 &= y_4 + y_5 + y_6 + y_7 \end{aligned}$$

- If there was no error (for all i, $x_i = y_i$) then $b_3 b_2 b_1$ is zero and we correct nothing.
- If there is an error in one of the parity check bits, say $x_2 \neq y_2$. Then only the corresponding bit is non zero ($b_2 = 1$ in this case). The position ℓ ($010 \to 2$ in this case) points to the bit to be corrected.
- If there is an error in one of the original message bits, say $x_5 \neq y_5$. Then the bits of the binary representation of this location will be non zero ($b_1 = b_3 = 1$ in this case). The position ℓ ($101 \to 5$ in this case) points to the bit to be corrected.

Decoding Hamming Code: Binary Representation

Recall the Hamming enccoding

$$\begin{aligned} x_1 &= x_3 + x_5 + x_7 \\ x_2 &= x_3 + x_6 + x_7 \\ x_4 &= x_5 + x_6 + x_7 \end{aligned}$$

The locations of the parity bits x_1, x_2, x_4, are all powers of two.

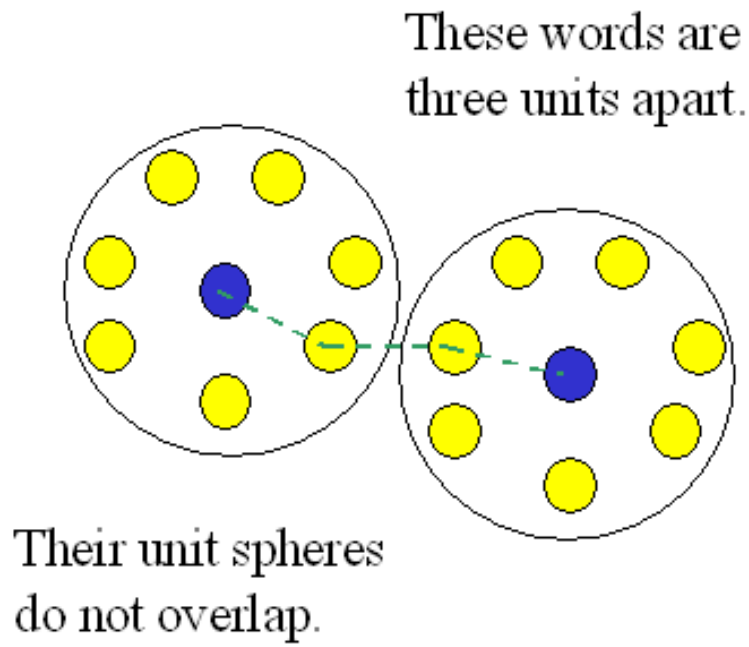
- x_1 corresponds to indices 3,5,7 having a 1 in the first (rightmost) position of their binary representations 011,101,111.
- x_2 corresponds to indices 3,6,7 having a 1 in the second (middle) position of their binary representations 011,110,111.
- x_4 corresponds to indices 5,6,7 having a 1 in the last (leftmost) position of their binary representations 101,110,111.

Geometry of Hamming Code

Let $\mathcal{C}_{\mathcal{H}}$ be the set of $2^4 = 16$ codewords in the Hamming code. A simple computation shows that $\mathcal{C}_{\mathcal{H}}$ equals

$\{(0,0,0,0,0,0,0),(1,1,0,1,0,0,1),(0,1,0,1,0,1,0),(1,0,0,0,0,1,1)$
$(1,0,0,1,1,0,0),(0,1,0,0,1,0,1),(1,1,0,0,1,1,0),(0,0,0,1,1,1,1)$
$(1,1,1,0,0,0,0),(0,0,1,1,0,0,1),(1,0,1,1,0,1,0),(0,1,1,0,0,1,1)$
$(0,1,1,1,1,0,0),(1,0,1,0,1,0,1),(0,0,1,0,1,1,0),(1,1,1,1,1,1,1)\}$.

By inspection, the Hamming distance between two codewords is ≥ 3. Therefore the unit spheres around different codewords do not overlap.



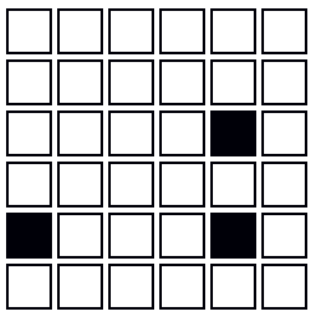
(figure from course EE 387, by John Gill, Stanford University, 2010.)

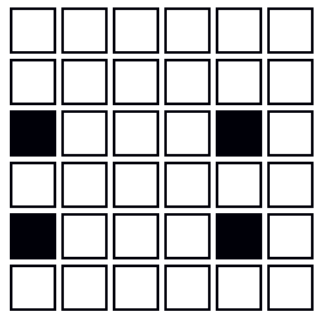
Closest Codeword Decoding

Given a code $C \subset \{0,1\}^n$ and an element $t \in \{0,1\}^n$, the closest codeword decoding, D, maps the element t to a codeword $y \in C$, that minimizes the distance $\Delta(t,z)_{z \in C}$.
If there is more than one $y \in C$ that attains the minimum distance, $D(t)$ announces an error.

Closest Codeword Decoding: Example

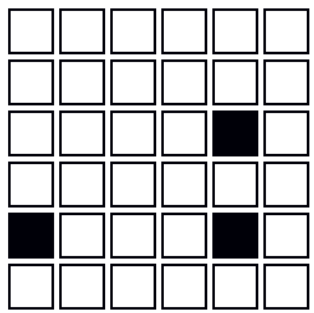
In the card magic code, suppose we receive the following string over $\{0,1\}^{36}$, depicted pictorially as the 6-by-6 black and white matrix on the left. There is a single codeword at distance 1 from this string, depicted to the right.

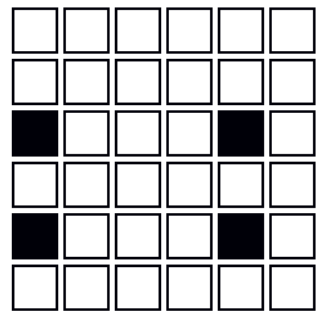




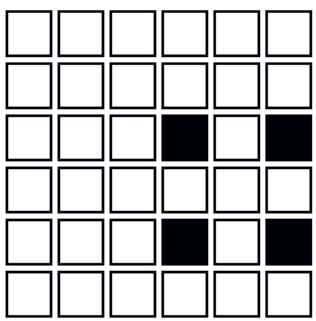
Closest Codeword Decoding: Example

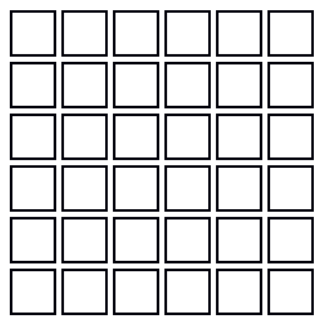
In the card magic code, suppose we receive the following string over $\{0,1\}^{36}$, depicted pictorially as the 6-by-6 black and white matrix on the left. There is a single codeword at distance 1 from this string, depicted to the right.

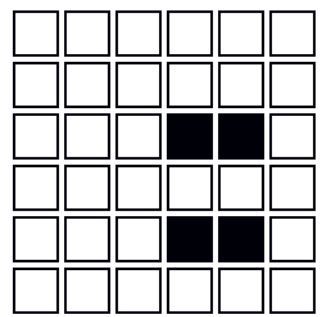




There is no codeword at distance 2 from this string (why?), and many that are at distance 3. Some of those are shown below.

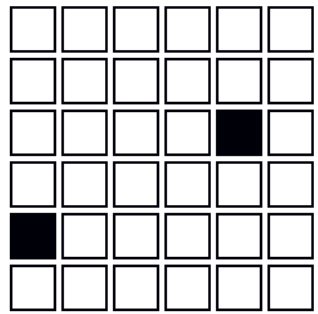






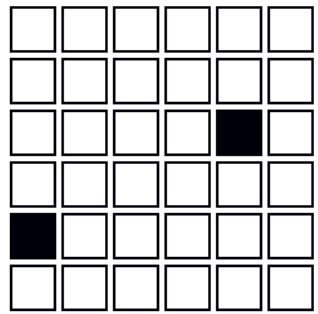
Closest Codeword Decoding: Example2

In the card magic code, suppose we receive the following string over $\{0,1\}^{36}$, depicted pictorially as the 6-by-6 black and white matrix.

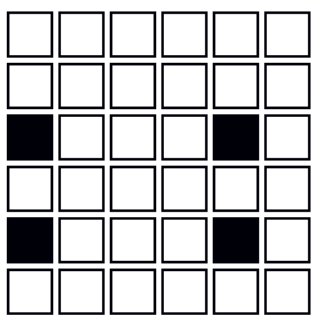


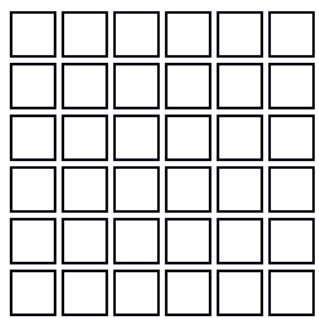
Closest Codeword Decoding: Example2

In the card magic code, suppose we receive the following string over $\{0,1\}^{36}$, depicted pictorially as the 6-by-6 black and white matrix.



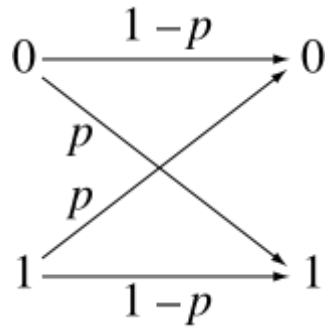
There is no codeword at distance 1 from this string (why?). There are exactly two codewords that are at distance 2 from this string. They are shown below. In such situation, closest codeword decoding announces an error.





Closest Codeword Decoding, cont.

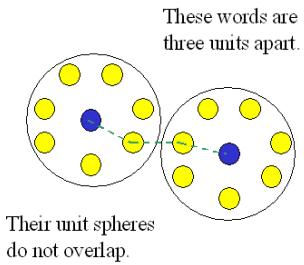
Observation: For the binary symmetric channel ($p < 1/2$), closest codeword decoding of t, if defined, outputs the codeword y that maximizes the likelihood of producing t, namely $Pr(t \text{ received} \mid y \text{ sent})$.



$\forall z \neq y \in C : Pr(t \text{ received} \mid z \text{ sent}) < Pr(t \text{ received} \mid y \text{ sent})$.

Proof: Simple arithmetic, employing independence of errors hitting different bits, and $p < 1/2$.

Closest Codeword Decoding and Hamming Codes



If we transmit a codeword in $\{0,1\}^7$ and the channel changes at most one bit in the transmission (corresponding to a single error), the received word is at distance 1 from the original codeword, and its distance from any other codeword is ≥ 2.

Thus, decoding the received message by taking the closest codeword to it is guaranteed to produce the original message, provided at most one error occurred.

Hamming Decoding in Python

```
def hamming_decode(y1,y2,y3,y4,y5,y6,y7):
    """ Hamming decoding of the 7 bits signal """
    b1= (y1+y3+y5+y7) % 2
    b2= (y2+y3+y6+y7) % 2
    b3= (y4+y5+y6+y7) % 2
    b=4*b3+2*b2+b1 # the integer value
    if b==0 or b==1 or b==2 or b==4:
        return (y3,y5,y6,y7)
    else:
        y=[y1,y2,y3,y4,y5,y6,y7]
        y[b-1]=(y[b-1]+1) % 2  # correct bit b
        return (y[2],y[4],y[5],y[6])

>>> y=hamming_encode(0,0,1,1)
>>> z=list(y); z  # y is a tuple (immutable)
[1, 0, 0, 0, 0, 1, 1]
>>> z[6] ^=1  # ^ is XOR (flip the 7-th bit)
>>> hamming_decode(*z) * unpacks list
(0, 0, 1, 1)

>>> y=hamming_encode(0,0,1,1)
>>> z=list(y)
>>> z[0] ^=1 ; z[6] ^=1  # flip two bits
>>> hamming_decode(*z)
(0, 0, 0, 0)  # code does not correct two errors
```

Highlights

- Using error correction codes to fight noise in communication channels.
- The binary symmetric channel.
- Three specific (families) of codes:
  - Repetition
  - Parity bit
  - Hamming
- Hamming distance and geometry.
- Spheres around codewords.
- Closest codeword decoding.

- Coding theory is a whole discipline.
- There are additional types of errors (erasures, bursts, etc.).
- And highly sophisticated codes, employing combinatorial and algebraic (finite fields) techniques.
- Numerous uses: *E.g.* communication systems, hardware design (DRAM, flash memories, etc.), and computational complexity theory.
- We have hardly scratched the surface.