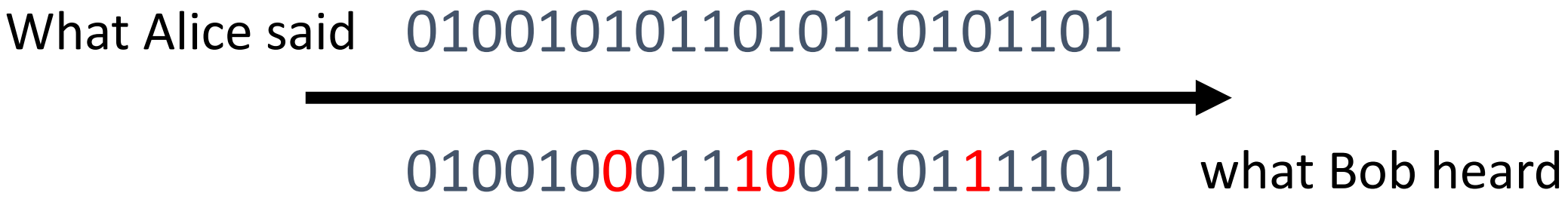
Error Detection & Correction

The scenario

- Alice & Bob communicate over a noisy channel
- Each bit in msg flipped with probability $p \ll 1$



- The idea: add bits for resiliency

The setup

- ***Encoding*** function $C: \{0,1\}^k \rightarrow \{0,1\}^n$
- A ***message*** $m \in \{0,1\}^k$ mapped to a ***codeword*** $x = C(m) \in \{0,1\}^n$
- The set of codewords is: $Im(C) \subset \{0,1\}^n$
  - $|Im(C)| = 2^k$
- Receiver gets ***noisy codeword*** $\widetilde{C(m)}$
- ***Decoding*** function $D: \{0,1\}^n \rightarrow \{0,1\}^k$
- Obviously: $D\big(C(m)\big) = m$
- Hopefully: $D\big(\widetilde{C(m)}\big) = m$
- Is this hope reasonable?

Hamming distance

- $D\left(\widetilde{C(m)}\right) = m$ makes sense only if $\widetilde{C(m)}$ has few errors
- I.e., the ***word*** $\widetilde{C(m)} \in \{0,1\}^n$ is close to some ***codeword*** $x \in Im(C)$
- Closeness via ***Hamming distance***: $\Delta(x, y) = |\{i : x_i \neq y_i\}|$
- ***Distance*** of a code: $d = \Delta(C) = \min_{x \neq y \in \mathrm{Im}(C)} \Delta(x, y)$

The goal

- ***Distance*** of a code: $d = \Delta(C) = \min_{x \neq y \in \mathrm{Im}(C)} \Delta(x, y)$
- Why is this an interesting measure?
- Given k (length of message), we wish to:
  - Maximize d – codewords are "**far apart**"
  - Minimize n – codewords are "**short**"
- Contradictoy. The Singleton bound says that $d \leq n - k + 1$

Basic Codes: Rep_t

- **Repeat** each bit in the message t times
- $Rep_t(x) = x_1x_1 \dots x_1 \circ x_2x_2 \dots x_2 \circ \cdots \circ x_kx_k \dots x_k$
- $n = ?\,, \Delta(Rep_t) = ?$
- Decoding: Majority vote per block

Basic Codes: $ParCode$

- Add a ***parity check*** (or parity bit)
- $par(x) = \sum x_i \bmod 2$
- Equivalently $par(x) = x_1 \oplus x_2 \oplus \cdots \oplus x_n$ where $\oplus$ is xor.
- $ParCode(x) = x \circ par(x)$
- $n = ?\,, \Delta(ParCode) = ?$
- Decoding?

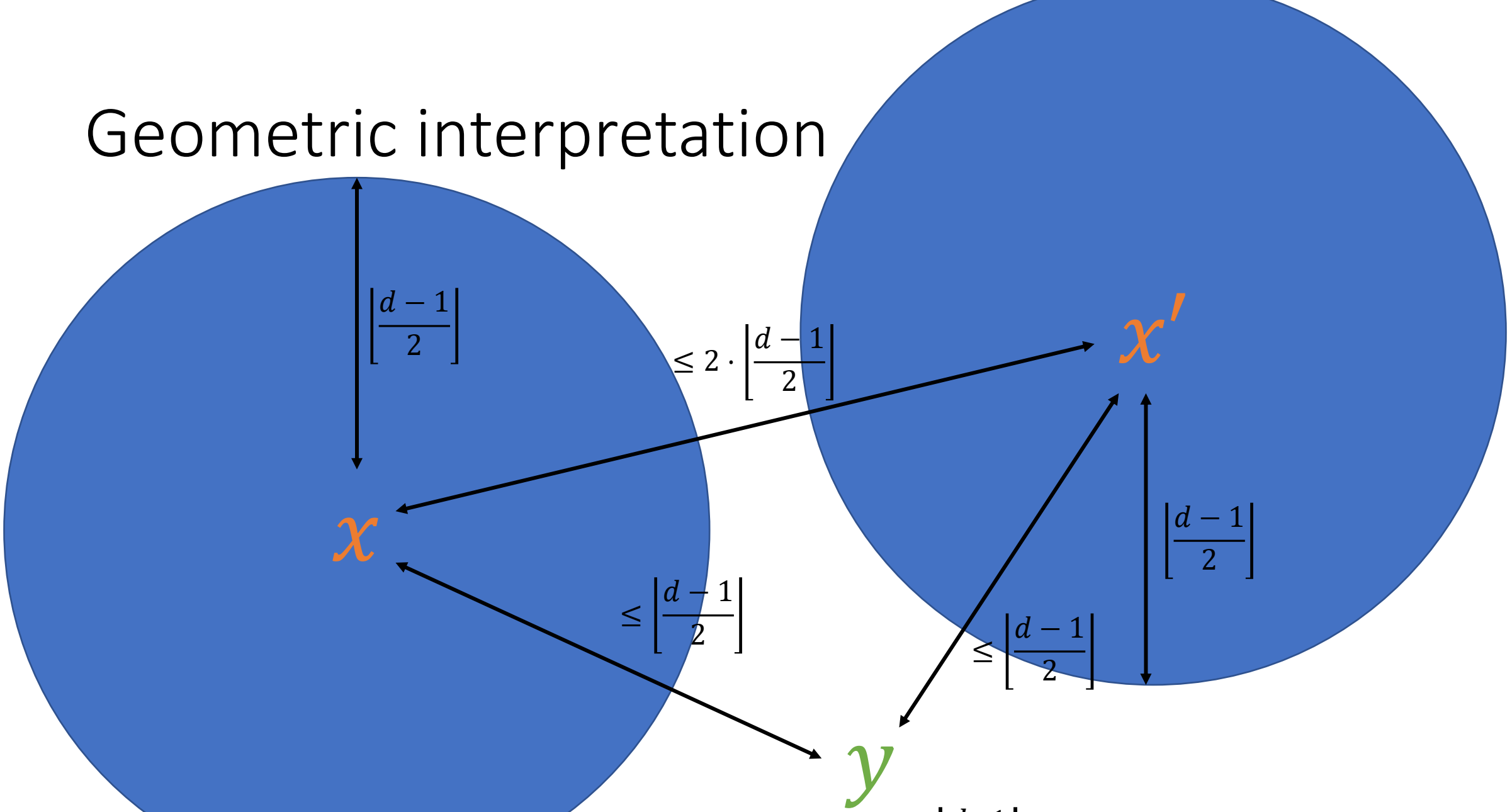
Detecting errors

- Detect: Decide if a reception is a codeword or not
- Claim: If C has distance d one can ***detect*** up to $d - 1$ errors
- Pf: By definition of distance

Correcting errors

- Correct: Find unique closest codeword to reception
- Claim: If C has distance d one can ***correct*** up to $\left\lfloor \frac{d-1}{2} \right\rfloor$ errors
- Pf: Balls of radius $\left\lfloor \frac{d-1}{2} \right\rfloor$ around codewords are disjoint

Geometric interpretation



"Balls" in radius $\left\lfloor \frac{d-1}{2} \right\rfloor$ around codewords are disjoint

Detection vs. correction

- Reception via $Par(\cdot)$
- $y = \widetilde{C(m)} = 011010101$
- Let $wt(y) = \#1's$ in y
- Clearly this is not a codeword since $\text{wt}(y) = 5$
- But! Any flip would change y to a valid codeword!
- So what was the original message?

- Reception via $Rep_3(\cdot)$
- $y = \widetilde{C(m)} = 000101111$
- Easy to see: $msg = 011$
- $\Delta(000101111, 000111111) = 1$
- Any other codeword is at distance ≥ 2

Actual detection/correction

- Important: detection/correction discussed above deals with existence of solution and not with an algorithm for the task
- For the algorithmic side, error detection is usually easy (i.e., we can efficiently verify a reception is valid/invalid)
- On the other hand, error correction (i.e., finding the nearest legal codeword) is a computationally hard task in general
- There is always a trivial correction algorithm. How?

Index Code

Index code (IC) construction

- Pick $k = 2^{\ell} - 1$ for some ℓ
- Encode a message $x = x_1 \dots x_k$ as follows:
  - For every i such that $x_i = 1$, take $bin(i)$ (note: $|bin(i)| = \ell$).
  - Compute bitwise xor ($\oplus$) over all such i, call it $EC(x)$
  - Transmit $x \circ EC(x)$
- Example: $x = 0110110$ (for $\ell = 3$)
  - $2 = 010 \oplus$
  - $3 = 011 \oplus$
  - $5 = 101 \oplus$
  - $6 = 110$
  - $EC = 010$
- Transmit 0110110010

Reminder:
- For bits: $i \oplus j = 1$ if $i \neq j$
- For words: $x \oplus y = x_1 \oplus y_1 \cdots x_n \oplus y_n$

Detour - why is $\oplus$ the best?

- Commutative: $x \oplus y = y \oplus x$
- Associative: $(x \oplus y) \oplus z = x \oplus (y \oplus z)$
- Nilpotent: $x \oplus y = 0 \leftrightarrow x = y$
- Easy to compute: $x_1 \oplus x_2 \oplus \cdots \oplus x_n = \sum_i x_i \bmod 2$
- Many other “nice” properties, but this is enough for us

Index code properties

- Length? $|x \circ EC(x)| = k + \ell = k + O(\log k)$
- Distance?
- To compute distance of code we need:
  - Lower bound ($\Delta(IC) \geq d$): any two codewords differ in **at least** d coordinates
  - Upper bound ($\Delta(IC) \leq d$): there exist two codewords that differ in **exactly** d coordinates
- Claim: $\Delta(IC) = 2$
- We will show both upper and lower matching bounds

$\Delta(IC) \leq 2$, upper bound

- Take two codewords that differ in one EC coordinate
- msg1: $x = 0000000$
- Msg2: $x' = 1000000$
- $EC(x) = 000$
- $EC(x') = 001$
- $\Delta\big(x \circ EC(x), x' \circ EC(x')\big) = \Delta(x, x') + \Delta\big(EC(x), EC(x')\big) = 2$

$\Delta(IC) \geq 2$, lower bound

- Let x, x' be two messages
- If $\Delta(x, x') \geq 2$ we are done since:
  - $\Delta\big(x \circ EC(x), x' \circ EC(x')\big) = \Delta(x, x') + \Delta\big(EC(x), EC(x')\big) \geq \Delta(x, x') \geq 2$
- So we only need to show that if $\Delta(x, x') = 1, EC(x) \neq EC(x')$
  - Why is this enough?
- Let $j \in \{1, \dots, k\}$ be an index such that (wlog) $x_j = 1, x'_j = 0$
  - All other "on" indices in x, x' are the same: $i_1, i_2, \dots, i_t$
- $EC(x) = i_1 \oplus i_2 \oplus \cdots \oplus i_t \oplus j$
- $EC(x') = i_1 \oplus i_2 \oplus \cdots \oplus i_t$
- $EC(x) \oplus EC(x') = j \neq 0 \leftrightarrow EC(x) \neq EC(x')$

$\Delta(IC) = 2$, consequences

- Since $\Delta(IC) = 2$, we can detect 1 error and fix none
- Can we do better? Indeed.
- First improvement – transmit EC twice

First improvement: IC_2

- Given a message x compute $EC(x)$ and transmit :
$$IC_2(x) = x \circ EC_1(x) \circ EC_2(x)$$
- Length still $k + O(\log k)$
- Distance?
- First note $\Delta(IC_2) \geq 2$ (why?)
- We will prove that $\Delta(IC_2) = 3$

$\Delta(IC_2) \leq 3$, upper bound

- Same as before
- $x = 0000000$
- $x' = 1000000$
- $EC(x) = 000$
- $EC(x') = 001$
- $\Delta(x \circ EC(x) \circ EC(x), x' \circ EC(x') \circ EC(x'))$
$= \Delta(x, x') + 2 \cdot \Delta\big(EC(x), EC(x')\big) = 3$

$\Delta(IC_2) \geq 3$, lower bound

- If $\Delta(x, x') \geq 3$ we are again done (why?)
- If $\Delta(x, x') = 1$ we are also done (why?)
- Only case left: $\Delta(x, x') = 2$
- Let $j, j' \in \{1, \ldots, k\}$ be indices where x, x' differ
  - All other "on" indices in x, x' are the same: $i_1, i_2, \ldots, i_t$
- $EC(x) \oplus EC(x') = j \oplus j' \neq 0$ (why???)
- So if $\Delta(x, x') = 2$ encoded distance is at least 4 (why?)

$\Delta(IC_2) = 3$, consequences

- Since $\Delta(IC_2) = 3$, we can detect 2 errors and fix 1
- We now show that we can also fix 1 error efficiently!
- We give an efficient algorithm:
- Let y be a reception containing **at most** 1 error

  we can compute the original message x such that $IC_2(x) = y$

IC_2 error correction algorithm

- Input: $y = x, EC_1, EC_2$
  - Compute $EC' = EC(x)$
  - If $EC' = EC_1 = EC_2$:
    - Return x
  - If $EC_1 \neq EC_2$:
    - Return x
  - $j = EC' \oplus EC_1$
  - Return $(x_1 \dots x_{j-1}) \circ \overline{x_j} \circ (x_{j+1} \dots x_k)$
  - That is – flip the jth bit of x and return the message

C_2 error correction algorithm- Example

- Alice wants to send $x = 0110110$
- She sends the codeword: 0110110010010
- Bob receives the noisy word: 0110010010010 (x | EC_1 | EC_2)
- $EC' = EC(0110010) = 111$
- $j = EC' \oplus EC_1 = 111 \oplus 010 = 101 \Rightarrow \quad j = 5$
- Return $0110\bar{0}10 = 0110110$

- Input: $y = x, EC_1, EC_2$
  - Compute $EC' = EC(x)$
  - If $EC' = EC_1 = EC_2$:
    - Return x
  - If $EC_1 \neq EC_2$:
    - Return x
  - $j = EC' \oplus EC_1$
  - Return $(x_1 \dots x_{j-1}) \circ \bar{x_j} \circ (x_{j+1} \dots x_k)$

IC_2 algorithm correctness

- Three cases to deal with:
  - No errors in transmission
  - One error in EC_1 or EC_2
  - One error in x
- Easy:
  - If no errors then $EC(x) = EC_1 = EC_2$ and we return x
  - If error is in EC_1 or in EC_2 then $EC_1 \neq EC_2$ and we return x

IC_2 algorithm correctness

- Only case left: single error in x
- Let j be the index in x that was flipped (wlog $x_j = 0$)
- Let $i_1, \ldots, i_t$ be the indices of all other "on" bits in x
- First: $EC_1 = i_1 \oplus i_2 \oplus \cdots \oplus i_t \oplus j$
- Second: $EC' = i_1 \oplus i_2 \oplus \cdots \oplus i_t$
- So: $EC' \oplus EC_1 = j$
- What if $x_j = 1$?

Second improvement: IC_3

- One more improvement – add parity bit
- So $IC_3(x) = IC_2(x) \circ par\big(IC_2(x)\big)$

$$= x \circ EC(x) \circ EC(x) \circ par(x \circ EC(x) \circ EC(x))$$

$$= x \circ EC(x) \circ EC(x) \circ par(x)$$

- Claim: $\Delta(IC_3) = 4$
- We will actually prove something **stronger**

Useful claim

- Let $x, y \in \{0,1\}^n$ be two binary strings
- Claim: $\Delta(x, y)$ is even iff $par(x) = par(y)$
- Proof: Denote $i_1, \ldots, i_t$ the indices in which $x_{i_j} \neq y_{i_j}$
- Note that $t = \Delta(x, y)$
- Suppose we take x and "flip" its $j'th$ bit, for example:
  $$x = 1001 \rightarrow 1101 = x'$$
  Then the parity of x is also flipped (i.e. $par(x) = 1 - par(x')$)
- We get from x to y by flipping each of the bits x_{i_1},…,x_{i_t}, thus:
- If $t = \Delta(x, y)$ is even, we flip the parity of x an even number of times, and we get $par(x) = par(y)$
- If $t = \Delta(x, y)$ is odd, we flip the parity of x an odd number of times, and we get $par(x) \neq par(y)$

Adding parity to odd distance codes

- Let C be a code with odd distance d
- Define $C'(x) = C(x) \circ par(C(x))$
- Claim: C' has distance $d + 1$
- Pf:
  - Enough to show for $x, y \in \mathrm{Im}(C)$ such that $\Delta(x, y) = d$ (why?)
  - Using the previous claim $\Delta(x, y) = d$ is odd $\Rightarrow par(x) \neq par(y)$
  - Overall $\Delta(C'(x), C'(y)) = d + 1$

Adding parity to even distance codes

- Let C be a code with even distance d
- Define $C'(x) = C(x) \circ par(C(x))$
- Claim: C' has distance d (no improvement)
- Proof is similar to the odd case

שאלה 4 (20 נק')

השאלה עוסקת בגרסה של קוד אינדקס לאיתור ותיקון שגיאות אותה ראינו בכיתה.

תזכורת: אנו רוצים לשלוח הודעה $msg = b_1 b_2 \ldots b_k$, כאשר $b_i \in \{0,1\}$ ו- $k = 2^m - 1$ עבור m טבעי כלשהו.
נסתכל על הביטים ה"דולקים" ($b_i = 1$, עבור $1 \le i \le k$), ונחשב xor בין האינדקסים שלהם (בכתיב בינארי).
נסמן את תוצאת החישוב ב- ec. אם אין בכלל ביטים "דולקים" אז ec=000. נוסיף ל- msg פעמיים את ec
(נסמנם ec1 ו- ec2), ונוסיף עוד ביט אחד של זוגיות (parity bit) בסוף (נסמנו p). את השדר כולו נסמן trans.

דוגמה, עבור m=3:

```
msg        0110110              (k=2^3-1)
indices    1234567

        2=  010  xor
        3=  011  xor
        5=  101  xor
        6=  110
       ec=  010

trans      0110110 010 010 0
            msg    ec1 ec2 p
```

בכל אחד מהסעיפים א', ב', ג' מתואר מצב בו נפלו 3 שגיאות בשדר (0 שהפך ל- 1 או להיפך). עליכם להחליט האם המצב המתואר אפשרי או לא. אם לדעתכם כן, רישמו בטבלה דוגמה לשדר חוקי מתאים (לפני השגיאות), וסמנו ב- X את מיקומי 3 הביטים בהם נפלו השגיאות. אם לא – רישמו "לא ייתכן" על הטבלה והסבירו. בכל הסעיפים $m=3$.

Part (a)

א. (4 נק') נפלו 3 שגיאות בשדר, ומילת הקוד הקרובה ביותר היא במרחק 1.

- $C(x) = 0000000\ 000\ 000\ 0$
- $\widetilde{C(x)} = 1000000\ 001\ 001\ 0$, $\widetilde{C(x)} \notin \mathrm{Im}(C)$
- $\Delta\left(C(x), \widetilde{C(x)}\right) = 3$

- $C(y) = 1000000\ 001\ 001\ 1$
- $\Delta\left(C(y), \widetilde{C(x)}\right) = 1$

Part (b)

ב. (4 נק') נפלו 3 שגיאות בשדר, ומילת הקוד הקרובה ביותר היא במרחק 2.

- We disprove a more general claim: we cannot have **<u>two codewords</u>** with **<u>odd</u>** and **<u>even</u>** distances from any message
- Suppose $C(y)$ and $C(z)$ have (resp.) even and odd distance from $\widetilde{C(x)}$
- Code has parity bits → all codewords have parity 0
- So $par(C(y)) = par(C(z))$
- From the previous lemma, $par(C(y)) = par(\widetilde{C(x)})$ (even distance)
- Also, $par(C(z)) \neq par(\widetilde{C(x)})$ (odd distance)
- Together: $par(\widetilde{C(x)}) \neq par(C(z)) = par(C(y)) = par(\widetilde{C(x)})$
- Note: claim does not rely on $\widetilde{C(x)}$ coming from some codeword

Part (c)

ג. (4 נק') נפלו 3 שגיאות בשדר, ומילת הקוד הקרובה ביותר היא במרחק 3.

- $C(x) = 0000000\ 000\ 000\ 0$
- $\widetilde{C(x)} = 0000000\ 111\ 000\ 0$, $\widetilde{C(x)} \notin \mathrm{Im}(C)$
- $\Delta\left(C(x), \widetilde{C(x)}\right) = 3$

- There is no codeword at distance 1 from $\widetilde{C(x)}$ (why?)
- There is no codeword at distance 2 from $\widetilde{C(x)}$ (why?)
- $C(x)$ is the closest codeword

Berger Codes

- Given $k = 2^{\ell} - 1$, define $Berger: \{0,1\}^k \to \{0,1\}^n$ as follows:
- $Berger(x) = x \circ bin(x.count(0))$
- That is – append to x the binary representation of the number of 0s bits in x
- Example: $x = 1000011 \to 1000011 \circ 100$
- What is the length and distance?
- $n = k + \log k$
- $d = 2$ (why?)

Berger Codes (performance)

- Note that we cannot correct errors: $1111110 \circ 000$ is the same distance from the encoding of 1111111 and 1111110
- How many can we detect? According to our definition, only 1
- Seems like parity code, with worse parameters

Berger Codes (unidirectional error)

- Say a codeword only underwent errors of the type $0 \rightarrow 1$
- How many unidirectional errors can $Berger$ detect?
- Write $Berger(x) = x \circ EC(x)$
- If all errors are in x then $EC(x)$ will be wrong
- If all errors are in $EC(x)$ then x will be wrong

Berger Codes (unidirectional error)

- Errors must be combined: both in x and in $EC(x)$, but:
- Any error in x reduced the number of 0 entries (so $EC(x)$ should decrease)
- Any error in $EC(x)$ only increases the supposed number of zero entries in x
- Conclusion: we can detect any number of such flips!
- What about only $1 \to 0$ errors?

Exam Question (2018ba)

קוד חזרה (repetition code) אותו למדנו בכיתה, הוא דוגמה לקוד לתיקון שגיאות. בכיתה ראינו דוגמה בה ההודעות הן באורך 2 , וכל ביט מההודעה משודר 3 פעמים. זהו כמובן מקרה פרטי של קוד חזרה עבור הודעות באורך k כלשהו, כאשר כל ביט משודר m פעמים (והשֶׁדָר לכן באורך כולל של $n = mk$).

להלן קוד אחר לגילוי ותיקון שגיאות, שמוגדר ע"י הפונקציה encode. הפונקציה מקבלת הודעה msg שהיא מחרוזת בינארית באורך כלשהו, ומספר חיובי שלם m.

```
def encode(msg, m):
      trans = ''
      for i in range(1,m+1):
            for c in msg:
                  trans += c*i
      return trans
```

דוגמת הרצה:

```
>>> encode('10', 3)
>>> '101100111000'
```

א. מהו המרחק המינימלי של הקוד כתלות בפרמטרים m, k (אחד מהם או שניהם)? $\frac{(m+1)m}{2}$

ב. תנו דוגמה לשדר trans, שאינו מילת קוד עבורו יש לפחות שתי מילות קוד קרובות ביותר. בתשובתכם כתבו מהם m, k, השדר ושתי מילות הקוד.

m = 3

k = 1

trans = 000111

מילות הקוד: 000000, 111111

Exam Question (2018ba)

$d = 3 \rightarrow \left\lfloor \frac{d-1}{2} \right\rfloor = 1$

ג. נתון k=2, m=2. נסמן על ידי d את המרחק המינימלי של קוד זה. תנו דוגמה לשדר trans, שאינו מילת קוד, עבורו יש מילת קוד יחידה קרובה ביותר, שמרחקה מ-trans גדול ממש מ-$\left\lfloor \frac{d-1}{2} \right\rfloor$. בתשובתכם כתבו מהו השדר, מהו d, ומהי מילת הקוד היחידה הקרובה ביותר לשדר.

trans = 010110

המרחק המינימלי של קוד זה = $d = 3$

מילת הקוד, ומרחקה מ-trans: 010011, מרחק 2