+1/1/60+

EPFL

1

Teacher : Prof. M. Gastpar
COM-102 (Advanced ICC II) - MA
June 20, 2024
Duration : 180 minutes

Student 1

SCIPER : **999000**

Do not turn the page before the start of the exam. This document is double-sided, has 16 pages, the last ones possibly blank. Do not unstaple.

- Place your student card on your table.
- **No other paper materials** are allowed to be used during the exam.
- Using a **calculator** or **any electronic device** is not permitted during the exam.
- The exam has **4 parts.** The fourth part is a single, comprehensive open problem (containing three sub-problems).
- For each question there is **exactly one** correct answer. We assign **negative points** to the **wrong answers** in such a way that a person who chooses a wrong answer loses **25 %** of the points given for that question.
- Use a **black or dark blue ballpen** and clearly erase with **correction fluid** if necessary.
- If a question is wrong, the teacher may decide to nullify it.

Respectez les consignes suivantes | Observe this guidelines | Beachten Sie bitte die unten stehenden Richtlinien		
choisir une réponse | select an answer Antwort auswählen	ne PAS choisir une réponse | NOT select an answer NICHT Antwort auswählen	Corriger une réponse | Correct an answer Antwort korrigieren
ce qu'il ne faut **PAS** faire | what should **NOT** be done | was man **NICHT** tun sollte		

Pour votre examen, imprimez de préférence les documents compilés à l'aide de auto-multiple-choice.

First part: Entropy and Source Coding

Question 1

[2 Points] In a binary Huffman code, symbols with equal probability always have codewords of equal length.

☐ TRUE ■ FALSE

Question 2:

[6 Points] About a binary source code for a source whose alphabet has 5 letters, we only know the codeword lengths: $\ell_1 = 1, \ell_2 = 2, \ell_3 = 4, \ell_4 = 5, \ell_5 = 6$.
Answer the following true/false questions.

The average codeword length of this source code could be equal to the entropy of the source.

☐ TRUE ■ FALSE

Solution: This could only happen if the symbol probabilities were $p_i = 2^{-\ell_i}$. But for the proposed codeword lengths, this does not sum to one. So, there is no way of assigning probabilities to the symbols such that the average codeword length for this code attains exactly the entropy.

It is possible that this source code is uniquely decodable.

■ TRUE ☐ FALSE

Solution: Kraft inequality.

This source code could be a Huffman code.

☐ TRUE ■ FALSE

Solution: Draw the tree. Code can obviously be improved.

Question 3

[4 Points] Consider a random variable $X \in \mathcal{X}$ and denote its alphabet size by $M = |\mathcal{X}|$. The symbol probabilities are not known. A binary Huffman code has been produced for this source. Now, suppose we observe a string of M letters : The string contains *each symbol of the source alphabet exactly once.* We compress the string with the given binary Huffman code. What is the maximum number of bits the binary Huffman compressor could produce for the entire string? Note: $\lceil x \rceil$ is rounding x up to the nearest integer.

- ☐ $2^{M\lceil \log_2 M \rceil}$
- ☐ $\lceil M \log_2 M \rceil$
- ☐ We do not have enough information.
- ■ $\frac{1}{2}M(M+1)-1$
- ☐ $M\lceil \log_2 M \rceil - 1$

Solution: The best approach is to start with small M. Start with $M = 3$. Here, there is only one Huffman tree, and hence, the Huffman code will produce $1+2+2 = 5$ bits. Here, $\lceil \log_2 M \rceil = 2$ and $\lceil M \log_2 M \rceil = 5$. Therefore, the only option that can be excluded is $2^{M\lceil \log_2 M \rceil}$.
Next, $M = 4$. Here, there are only two possible binary Huffman trees: the complete binary tree of depth 2, leading to 8 bits, and the unbalanced tree leading to $1+2+3+3 = 9$ bits. Hence, the right answer here is 9. Here, $\lceil M \log_2 M \rceil = 8$ and $M\lceil \log_2 M \rceil - 1 = 7$, so the only answer left in the running is $\frac{1}{2}M(M+1)-1$. Could this be the correct answer?
Extrapolating from $M = 4$, we should probably construct a very "long" tree. The longest tree is obtained if at every step, we extend, say, the "rightmost" side of the tree. In this way, the tree depth will be $M-1$ (we have two leaves at the last stage), and the codeword lengths are $1+2+3+\ldots+(M-1)+(M-1)$. This, of course, gives exactly the claimed formula. It should be clear that this is indeed a valid Huffman tree for some probabilities (left for the reader: construct an explicit example!). Now, could a different tree lead to something even longer? The answer is no, which can be seen a bit like in our proof of the optimality of Huffman: If you move any leaf in a "Huffman-compatible" fashion, it will have to be moved to a shorter length.

Question 4

[3 Points] Alice and Bob are playing an outfit guessing game. Alice uniformly randomly chooses a top, a bottom and a pair of shoes from her wardrobe. Alice's wardrobe has the following unique items:

- Tops: 3 Shirts, 4 Pullovers
- Bottom: 4 Jeans, 2 Skirts
- Shoes: 2 Pairs of Sneakers (sometimes *baskets* in French), 1 Pair of Boots

Bob needs to find out Alice's outfit precisely, including which of the 4 pullovers or which of the 3 shirts she's wearing, and so on. What is the minimum required number of Yes/No questions such that Bob is guaranteed to find out Alice's outfit? Note that arbitrarily complex questions are allowed — no restrictions — but the answer can only be either Yes or No.

- ☐ 6
- ☐ 9
- ☐ 4
- ☐ 5
- ■ 7
- ☐ 8

Solution: Let us start by calculating the total number of outfits:

$$(3+4)\cdot(4+2)\cdot(2+1) = 7\cdot 6\cdot 3 = 126.$$

Hence, a binary question strategy will require at least $\log_2(7\cdot 6\cdot 3) = \log_2(7)+1+2\log_2(3) \approx 2.8074+1+3.16$, just a little below 7 (where we took the numbers from the official formula collection). That is, we would need at least 7 binary questions.

Question 5

[3 Points] Continued from previous question: Alice wants to give the following clue to reduce the number of questions that Bobs needs to ask:

"If I am wearing a pullover, I am also wearing a skirt and boots."

Given the clue, what is the minimum required number of Yes/No questions such that Bob is guaranteed to find out Alice's outfit?

☐ 7	☐ 4	☐ 8
☐ 9	■ 6	☐ 5

Solution: We can remove any combination that involves pullover-jeans-any shoes or pullover-skirt-sneakers. How many? $4 \cdot 4 \cdot 3 + 4 \cdot 2 \cdot 2 = 64$. Then, the total number of outfits is $126 - 64 = 62$.
Hence, a binary question strategy will require at least $\log_2(2 \cdot 31) = 1 + \log_2(31) \leq 1 + \log_2(32) = 6$ just a little below 6. That is, we would need at least 6 binary questions.

Question 6

[2 Points] Consider a certain source symbol $x \in \mathcal{X}$. Claim: the length of the codeword corresponding to x in an optimal binary code (optimal in the sense of average codeword length) must be smaller or equal to the length of the codeword corresponding to x in the binary Shannon-Fano code.

☐ TRUE ■ FALSE

Solution: This is false. Consider a source with 5 symbols $\{a, b, c, d, e\}$ with probabilities $\frac{1}{8}(3, 2, 1, 1, 1)$. There are several Huffman codes here (non-unique). It's quick to see that one Huffman code will assign codeword lengths $1, 3, 3, 3, 3$. That is, it assigns a codeword length of 3 to a symbol with probability $1/4$. The Shannon-Fano code will assign a codeword of length 2. Hence, the claim is false.

Second part: Cryptography

Question 7

[4 Points] How many $x \in \{0, 1, 2, \ldots, 34\}$ satisfy the equation $x^2 - 5x + 4 \mod 35 = 0$?

- [x] 4
- [ ] 0
- [ ] 1
- [ ] 2

Solution: Probably the fastest is to go via the Chinese Remainder Theorem. Write $35 = 5 \cdot 7$. Recall that $x^2 - 5x + 4 \mod 35 = 0$ if and only if $x^2 - 5x + 4 \mod 5 = 0$ and $x^2 - 5x + 4 \mod 7 = 0$.
The $\mod 5$ part is particularly easy: $x^2 - 5x + 4 \equiv x^2 + 4 \mod 5$. Moreover, $4 \equiv -1 \mod 5$. So, clearly, we have two solutions here: $x = 1$ and $x = 4$.
For the $\mod 7$ part, it is perhaps clever to observe that $x^2 - 5x + 4 = (x - 1)(x - 4)$. (This holds always, and of course also $\mod 7$.) So we can see that $x = 1$ and $x = 4$ both work.
To combine, thinking about the grid representation of the Chinese Remainder Theorem, we have 4 solutions (two choices for the row and two choices for the column in the grid representation). Namely, $(1, 1), (1, 4), (4, 1), (4, 4)$.
You are not asked to find the actual solutions. You could find those by the usual inversion formula for the Chinese Remainder Theorem. Numbers in this example are so small that we can probably guess the solutions... The first solution is a number x such that $x \mod 5 = 1$ and $x \mod 7 = 1$. This is simply $x = 1$. Next, we need a number x such that $x \mod 5 = 4$ and $x \mod 7 = 4$. This is simply $x = 4$. Now it gets more interesting: Another solution is x such that $x \mod 5 = 1$ and $x \mod 7 = 4$. A moment's reflection gets us to $x = 11$. And finally, we are looking for x such that $x \mod 5 = 4$ and $x \mod 7 = 4$. Here, we have to perhaps scratch our head a little longer (or actually draw up the Chinese Remainder Theorem table). Then, we find $x = 29$.
So, the four solutions are $x \in \{1, 4, 11, 29\}$. (It is not hard to verify that these four numbers are indeed solutions to the original equation.)

Question 8:

[8 points] Alice and Bob are studying cryptography for the first time. They each did a project and each got a score. The score is out of 30 points. They represent it with 5 bits, using the natural binary representation. They want to tell their project scores to their friend Charlie, using the one-time pad. Charlie has a single uniformly sampled binary string K of length 5. He sends this *same* string to both Alice and Bob. Nobody else ever gets to know K.
Class policy dictates that students get a grade of 6 on the project if they score more than 27 points and a grade of 5.75 if they score more than 23 points.
Several people have partial information and try to infer Alice's and Bob's grades:

- David intercepts only Bob's transmission.
- Eve intercepts both Alice's and Bob's transmissions.
- Frank does not intercept anything, but he saw Bob very happy on finding his project score. So Frank knows that Bob received a grade of 6.

David can tell if Bob got a grade of 6 on the project

☐ TRUE ■ FALSE

Solution: Since Bob's message was encrypted using the one-time-pad, the ciphertext is independent of the plaintext; David cannot guess anything about Bob's score using only Bob's transmission.

Eve can tell for sure if either of them received full points.

☐ TRUE ■ FALSE

Solution: Using both the messages, Eve can find $c_1 \oplus c_2 = s_1 \oplus K \oplus s_2 \oplus K = s_1 \oplus s_2$. However, this does not indicate if either of the messages was 11110, which is the binary representation of 30.

Eve can tell who scored more points between Alice and Bob.

☐ TRUE ■ FALSE

Solution: Since $s_1 \oplus s_2 = s_2 \oplus s_1$, Eve cannot tell whether $2^{s_1} > 2^{s_2}$ or vice versa given $s_1 \oplus s_2$.

Frank can help Eve infer if Alice got a grade of 5.75 or more.

■ TRUE ☐ FALSE

Solution: Frank knows that s_2 was either 11100, 11101 or 11110. Eve and Frank can thus infer that Alice scored 24 points only or more if the first two bits of $s_1 \oplus s_2$ are 0.

Question 9

[2 Points] If a finite commutative group contains an element that is its own inverse but is *not* the identity element, then the group must have an even number of elements.

■ TRUE ☐ FALSE

Solution: An element (not the identity) that is its own inverse has order 2. By the Lagrange theorem, we know that the order of every element must divide the number of elements in the group. Therefore, in this case, 2 is a divisor of the number of elements in the group.

Question 10
[4 Points] How many elements does the multiplicative group $(\mathbb{Z}/2535\mathbb{Z}^*, \cdot)$ have?

- ☐ 845
- ☐ 0
- ■ 1248
- ☐ 1252

Solution: The number of elements in the multiplicative group $(\mathbb{Z}/m\mathbb{Z}^*, \cdot)$ for a natural number m is given by $\phi(m)$ where $\phi(.)$ is Euler's totient function.
To find $\phi(m)$, the key is the prime factorization of $m = 2535$, which is slightly cumbersome. But clearly, 2535 is divisible by 5, so we find $2535 = 5 \cdot 507$. What about 507? The trick is always the same: Try some small divisors. 2 does not work, so try 3. You quickly realize that indeed, $507 = 3 \cdot 169$. Now, you probably recall that $169 = 13^2$. So, we have found the prime factorization $2535 = 5 \cdot 3 \cdot 13^2$. From the Homework and the formula collection, we recall that for coprime m and n, $\phi(mn) = \phi(m)\phi(n)$. Using this here, $\phi(2535) = \phi(5 \cdot 3)\phi(13^2)$. The first expression is our old friend from RSA (since both factors are prime), thus $\phi(5 \cdot 3) = 4 \cdot 2 = 8$. The second is the prime-power formula in the formula collection (we proved it together in class), namely, $\phi(13^2) = 13^2 - 13 = 156$. Hence, $\phi(2535) = 8 \cdot 156 = 1248$.
Alternatively, perhaps you recalled the general formula that we derived in the homework. For $m = \prod_{p_i|m} p_i^{q_i}$, where the p_i's are prime, the value of the totient function is given as

$$\phi(m) = m \prod_{p_i|m} \left(1 - \frac{1}{p_i}\right).$$

For $m = 2535 = 3 \cdot 5 \cdot 13^2$, this equals 1248.

Question 11
[4 Points] Which of the following is **not** a valid RSA encoding exponent for the modulus $m = 3403 = 83 \times 41$?

- ☐ 999
- ☐ 7
- ☐ 949
- ■ 943

Solution: We know from the homeworks that the encoding exponent needs to be invertible with respect to both $p - 1$ and $q - 1$ (and therefore coprime to both of them), where p and q are the prime factors of the modulus. Thus, e needs to be coprime to both $82 = 41 \times 2$ and $40 = 5 \times 2^3$. Since none of the options are even, we check for divisibility by 41. The quickest way to do this is to find multiples of 41 between 900 and 1000. We find that the multiples are 902, 943, and 984.

Question 12
[3 Points] Find $4^{14553} \mod 13$.

- ■ 12
- ☐ 9
- ☐ 3
- ☐ 2

Solution: Notice that $4^3 \equiv -1 \mod 13$, and $14553 = 3 \times 4651$. Therefore, $4^{14553} \equiv (-1)^{4651} \times 4 \equiv -1 \equiv 12 \mod 13$.

Question 13

[4 Points] Let $(m_1 = p_1q_1, e)$ be the public parameters of an RSA cryptosystem, with (secret) decrypting exponent d_1. Moreover, let $(m_2 = p_2q_2, e)$ be the public parameters of another RSA cryptosystem, with (secret) decrypting exponent d_2. We assume that p_1, q_1, p_2 and q_2 are four distinct prime numbers.
Alex has encrypted the plaintext t with the RSA cryptosystem with public parameters $(m = p_1q_2, e)$ to form the cryptogram c. Which of the following is a correct decryption?
(Note: Here, we use $(a)_k^{-1}$ to denote the multiplicative inverse of a in the multiplicative group $(\mathbb{Z}/k\mathbb{Z}^*, \cdot)$ if it exists. If it does not exist, we set $(a)_k^{-1} = 0$.)

- [x] $\left[a \cdot \left(c^{d_1} \mod p_1\right) + b \cdot \left(c^{d_2} \mod q_2\right)\right] \mod m$,
  where $a = q_2v$ and $b = p_1u$ for some integers u and v satisfying $p_1u + q_2v = 1$.
- [ ] $c^d \mod m$,
  where $d = d_1(q_2 - 1)(q_2 - 1)_{p_1-1}^{-1} + d_2(p_1 - 1)(p_1 - 1)_{q_2-1}^{-1}$.
- [ ] None of the given formulas work.
- [ ] $\left(c^{d_1} \mod m_1 + c^{d_2} \mod m_2\right) \mod m$

Solution: The correct formula is $\left[a \cdot \left(c^{d_1} \mod p_1\right) + b \cdot \left(c^{d_2} \mod q_2\right)\right] \mod m$, where $a = q_2v$ and $b = p_1u$ for some integers u and v satisfying $p_1u + q_2v = 1$.
The idea is to pass through the Chinese remainder theorem, exactly like we did in our proof that RSA can indeed be decrypted in the claimed fashion. The main part of the proof is that we are constructing

$$[t]_{p_1} \text{ and } [t]_{q_2}.$$

Since p_1 and q_2 are coprime, then by the Chinese remainder theorem, we know that these two uniquely characterize $[t]_{p_1q_2}$. And that we can find the latter simply with the inversion formula for the Chinese remainders that we derived in class. Which is exactly the proposed formula.
Now, how can we be sure that $c^{d_1} \mod p_1$ is indeed equal to $t \mod p_1$? Write

$$c^{d_1} \mod p_1 = (t^e \mod p_1q_2)^{d_1} \mod p_1 = ((t^{ed_1}) \mod p_1q_2) \mod p_1$$

So far, so easy. The only tricky part is what to do with the two successive modulos. For this, we need a simple and pretty obvious property of successive modulo: For every integer x, we have that $(x \mod 2m) \mod m = x \mod m$, and this holds not just for $2m$, but for any multiple of m. Therefore, we can conclude

$$((t^{ed_1}) \mod p_1q_2) \mod p_1 = (t^{ed_1}) \mod p_1$$

But of course, since we know that d_1 is a valid decoding exponent for the RSA system with parameters $(m_1 = p_1q_1, e)$, we know that $ed_1 \mod (p_1 - 1) = 1$, which is precisely what we used in our analysis of RSA. Thus,

$$c^{d_1} \mod p_1 = (t^{ed_1}) \mod p_1 = t \mod p_1.$$

The same goes through for $c^{d_2} \mod q_2$, and thus the claimed formula works.

Question 14

[2 Points] A finite cyclic group with an even number of elements has exactly one element of order 2.

- [x] TRUE
- [ ] FALSE

Solution: Let g be a generator of the group $(G, \star)$.. Then, because the group is cyclic, every element can be written as $a = g^i$, where i ranges over $1, 2, \ldots, |G|$. Now, a is an element of order two if it is *not* the identity element e, but $a^2 = e$. Hence, we are looking at $a^2 = (g^i)^2 = g^{2i}$. Clearly, since $|G|$ is even by assumption, we may select $i = |G|/2$, in which case the element $a^2 = g^{2i} = e$, but $a = g^i \neq e$. So, there always exists an element of order 2. Could there be another element of order 2? Consider again $a = g^i$ with $a^2 = g^{2i}$. But $g^{2i} = e$ if and only if $2i$ is a multiple of $|G|$, thus i a multiple of $|G|/2$. Hence, the element of order 2 that we have constructed is unique.

Third part: Channel Coding

Question 15

[4 points] Suppose we want to send a k-bit message, where we assume that $k = rc$ for two positive integers r and c. We can shape the $k = rc$ bits into a rectangular array with r rows and c columns. For example, if $k = 12$, then the array could be of size 1×12, 2×6, 3×4, 4×3, 6×2, or 12×1. The following table shows an example for the case of 3×4.

x_{11}	x_{12}	x_{13}	x_{14}	q_1
x_{21}	x_{22}	x_{23}	x_{24}	q_2
x_{31}	x_{32}	x_{33}	x_{34}	q_3
p_1	p_2	p_3	p_4	

Define q_i to be the parity bit of the message bits in row i of the array, i.e.,

$$q_i = x_{i1} + \cdots + x_{ic} \bmod 2$$

and p_j to be the parity bit of the message bits in column j of the array, i.e.,

$$p_j = x_{1j} + \cdots + x_{rj} \bmod 2$$

The codeword w is the concatenated sequence of the message bits $x_{11}, \ldots, x_{rc}$, the row parity bits, and the column parity bits:

$$w = [x_{11}, x_{12}, \ldots, x_{rc}, q_1, \ldots, q_r, p_1, \ldots, p_c]$$

w is then transmitted over the (binary) error channel. Which of the following statements is correct?

- ☐ The minimum distance of the rectangular parity code depends on the choice of the positive integers k and r.
- ☐ Suppose we use the 1×12 rectangular parity code. Suppose the error channel output has a single position j for which $x_{1j} + p_j = 1$, while all other column parity bits are consistent, and the row parity bit is consistent, too. Then, minimum distance decoding amounts to changing bit x_{1j} and nothing further.
- ☐ For $k \geq 2$, no error pattern of weight two or more can be corrected.
- ■ For any $k \geq 1$, the rectangular parity code can correct any error pattern of weight 1.

Solution: The first key observation is that this code is linear. (In fact, it is straightforward to give a systematic generator matrix for the code.)
Hence, the minimum distance is equal to the minimum weight of the code. For this, observe the following:

- If we put a single 1 in the information bits (any position whatsoever), the weight of the codeword will be three (since exactly two of the parity bits will also be one). So, clearly, the minimum distance cannot be larger than 3.
- Could it be two? That is, could we find a codeword of weight 2? It is clear that this is not possible: We have to at least place *two* ones into the information bits. But this will create *at least* one parity bit that is one, hence, again a weight of three. So, the minimum distance is three. Hence, it can correct any error of weight 1.

By this argument, the minimum weight (and thus, the minimum distance) of the code is always 3 no matter k, r, and c. This means that it is guaranteed that we can correct an error of weight one.

Question 16

[4 Points] Suppose a Reed-Solomon code over $\mathbb{F}_{11}$ of length $n = 7$ and dimension $k = 3$ was used with $a_1 = 0, a_2 = 1, a_3 = 2, a_4 = 3, a_5 = 4, a_6 = 5$, and $a_7 = 6$. We intercept the following output of the erasure channel:

$$\vec{y} = (2, 3, 6, ?, ?, ?, ?)$$

Which one of the following sequences could be the codeword that generated this channel output?

- [x] $\vec{c} = (2, 3, 6, 0, 7, 5, 5)$
- [ ] $\vec{c} = (2, 3, 6, 1, 3, 5, 5)$
- [ ] None of the listed sequences.
- [ ] $\vec{c} = (2, 3, 6, 0, 7, 5, 6)$

Solution: The key is to recall (or read in the Formula Collection) that an RS codeword is determined by the evaluations of a fixed polynomial $p(x)$ at the given locations $x = a_i$. All we need to find is: Which polynomial? We are given 3 valuations of the unknown polynomial in the problem statement. To *uniquely* fit a polynomial to 3 valuations, we proceed by Lagrange interpolation: Find a polynomial with the smallest possible degree passing through the 3 given points. We know that this is a polynomial of degree no more than two, given by

$$\begin{aligned} p(x) &= 2\,(2)^{-1}\,(x-1)(x-2) + 3\,(-1)^{-1}\,x(x-2) + 6\,(2)^{-1}\,x(x-1) \\ &= (x-1)(x-2) + (-3)(-1)\,(-1)^{-1}\,x(x-2) + 3\cdot 2\,(2)^{-1}\,x(x-1) \\ &= (x-1)(x-2) - 3x(x-2) + 3x(x-1) \\ &= x^2 - 3x + 2 - 3x^2 + 6x + 3x^2 - 3x \\ &= x^2 + 2. \end{aligned}$$

Plugging in $x = a_4 = 3$, we find $3^2 + 2 = 11 \equiv 0$. Plugging in $x = a_5 = 4$, we find $4^2 + 2 = 18 \equiv 7$. Plugging in $x = a_6 = 5$, we find $5^2 + 2 = 27 \equiv 5$. Plugging in $x = a_7 = 6$, we find $6^2 + 2 = 36 + 2 = 38 \equiv 5$.

Question 17:

[8 points] Consider a binary linear (n, k) block code $\mathcal{C}_1$ with (full-rank) generator matrix G and (full-rank) parity check matrix H. The binary linear block code $\mathcal{C}_2$ has generator matrix H. Answer the following True/False questions.

G is a parity check matrix of $\mathcal{C}_2$.

■ TRUE ☐ FALSE

Solution: Denote the rows of G by $\vec{g}_i$, for $i = 1, 2, \ldots, k$. Denote the rows of H by $\vec{h}_i$, for $i = 1, 2, \ldots, n-k$. Then, every codeword of $\mathcal{C}_2$ can be expressed as $\vec{c} = \sum_{i=1}^{n-k} u_i \vec{h}_i$. We need to show that every such codeword satisfies $\vec{c}G^T = \vec{0}$. This is the same as saying $\sum_{j=1}^{n} c_j g_{i,j} = 0$ for every $i = 1, 2, \ldots, k$. But $c_j = \sum_{i=1}^{n-k} u_i h_{i,j}$. Plugging in, $\sum_{j=1}^{n} c_j g_{i,j} = \sum_{j=1}^{n} \sum_{i=1}^{n-k} u_i h_{i,j} g_{i,j} = \sum_{i=1}^{n-k} u_i \sum_{j=1}^{n} h_{i,j} g_{i,j}$, and the last sums are zero for every i since H is a parity check matrix for the the code with generator matrix G.

$\mathcal{C}_1$ and $\mathcal{C}_2$ have no codewords in common.

☐ TRUE ■ FALSE

Solution: Evidently, they have the all-zero codeword in common.

The union of $\mathcal{C}_1$ and $\mathcal{C}_2$ is $\mathbb{F}_2^n$, that is, $\mathcal{C}_1 \cup \mathcal{C}_2 = \mathbb{F}_2^n$.

☐ TRUE ■ FALSE

Solution: Just by counting: $\mathcal{C}_1$ has 2^k codewords, $\mathcal{C}_2$ has 2^{n-k} codewords. So, the union has no more than $2^k + 2^{n-k} - 1$ codewords. But the total number of sequences in $\mathbb{F}_2^n$ is 2^n, much larger in general.
Or via an example: Let $\mathcal{C}_1$ be the code containing all sequences with an even number of ones (which we have seen in class). As you may recall, the parity check matrix of this code is a single vector, namely, the all-ones vector. This code has only two codewords. Clearly, the sequence starting with three ones, followed by all zeros, is neither in $\mathcal{C}_1$ nor in $\mathcal{C}_2$.

$\mathcal{C}_2$ is a binary linear $(n, n-k)$ block code.

■ TRUE ☐ FALSE

Solution: This is true by definition, and since we are assuming H to be full rank.

Question 18

[4 points] Consider a binary linear code spanned by the following vectors:

$$\{v_1 = (110010), v_2 = (011100), v_3 = (111001), v_4 = (100101)\}$$

Which one of the following statements is correct?

- [x] If we receive from the Binary Symmetric Channel (with crossover probability less than 0.5) a word (101111), to minimize the error probability, we will deduce the transmitted codeword is (101110)
- [ ] Every three columns in its parity-check matrix H in systematic form are linearly independent.
- [ ] The cardinality of the code is 16.
- [ ] If we receive from the erasure channel a word (?10?11), the transmitted codeword is (010011)

Solution: The main point here is that the 4 given vectors are linearly dependent. For example, by inspection, one quickly realizes that $v_3 = v_2 + v_4 \mod 2$. (More systematically, one can do Gaussian elimination.) Perhaps a nice quick step is to write the generator matrix:

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

Taking the current row 3 and adding to it the first and the second rows, leads to

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

Finally, taking the middle row and adding to it the last row gives

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

which is systematic, thus

$$H = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix},$$

- *If we receive from the Binary Symmetric Channel (with crossover probability less than* 0.5*) a word* (101111)*, to minimize the error probability, we will deduce the transmitted codeword is* (101110) This is correct: To minimize error probability on this channel, we need to perform minimum distance decoding. The proposed decoding has a Hamming distance of only 1, and the proposed decoded sequence is indeed a codeword (can be see for example via the parity check matrix).
- *If we receive from the erasure channel a word* (?10?11)*, the transmitted codeword is* (010011) This is incorrect: It is not a codeword. Rather, the transmitted codeword is (010111)
- *Every three columns in its parity-check matrix H in systematic form are linearly independent.* This is incorrect: We have the parity check matrix above, and can verify that, *e.g.,* the first, fourth and sixth columns are linearly dependent.
- *The cardinality of the code is* 16. This is incorrect: From the generator matrix, we can see that the dimension is $k = 3$, so there are 8 codewords.

Question 19

[4 points] Consider a binary code $\mathcal{C}$ of length $n = 5$. The code is made up of all sequences that contain exactly three ones. Which of the following statements is true about this code?

- [x] The minimum distance of $\mathcal{C}$ is 2.
- [ ] Overall there are 8 codewords in $\mathcal{C}$.
- [ ] It can correct every error with weight no more than 1.
- [ ] It is a linear code.

Solution: Perhaps the easiest to rule out is 8 codewords: the codewords are all sequences of length $n = 5$ with 3 ones, that is $\binom{5}{3} = 10$ codewords. Which also directly rules out linearity (this is also ruled out since the all-zero sequence is not in the code). Can it correct every error with weight no more than 1? A quick example reveals that this is not possible: Starting from codeword 11100, an error of weight 1 is 01100. Clearly, there are three minimum-distance codewords: Not just 11100, but also 01110 and 01101. So, by exclusion, the minimum distance apparently is 2. Which can of course also be derived directly, but is a little harder to argue in a clean fashion.

Fourth part: open questions

Answer in the empty space below. Your answer should be carefully justified, and all the steps of your argument should be discussed in details. If you have the correct answer, but do not show the full work, you will not receive full credit. Leave the check-boxes empty, they are used for the grading.

Question 20: *This question is worth* $3 \times 8 = 24$ *points.*

In this problem you are an eavesdropper and codebreaker! You eavesdrop on a communication and manage to overhear the following (noisy) channel output sequence:

$$\vec{y} = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Your task in this problem is to find the original message that was transmitted.
Here is what you know about the system used for transmission:

- Source Coding: You know the original message was either '*foo*' or '*bar*'. You know that the original message was compressed letter-by-letter using a Huffman code, but you don't know which Huffman code. You only know that the Huffman code was designed for the symbol probabilities $p(X = r) = 0.5, p(X = o) = 0.25, p(X = a) = 0.125, p(X = f) = 0.0625, p(X = b) = 0.0625$.

- Encryption: The binary string coming from the Huffman code is now encrypted using RSA. This is a little tricky since RSA does not directly take in binary strings. Here is how they proceeded: The binary string is first cut into chunks of 4 bits each. Each chunk is converted into an integer following the usual binary expansion rule, and then adding '1' (so that we do not feed a '0' to RSA encryption), as in the following table:

chunks of 4 bits of source coding output	input to RSA	RSA output	towards input to channel code
0000	1	1	00001
0001	2	2	00010
0010	3	3	00011
⋮	⋮	⋮	⋮
1111	16	20	10100

  The integer "input to RSA" is encrypted with RSA with public parameters $(m = 21, e = 5)$. The output of the RSA encryption is an integer between 1 and 20 (since we never use the input '0'). We convert this integer into 5 bits as in the above table (using standard binary representation). In this way, note that if the original string (source coding output) was $2 \times 4 = 8$ bits long, then the output of the RSA encryption will be $2 \times 5 = 10$ bits long.

- Channel Coding: From the RSA stage, we get 10 bits. They simply append a '0' at the end so as to make it 11 bits. Then, what was used is the *systematic* $(15, 11)$ Hamming code, specifically with the following parity check matrix:

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

$$\vec{y} = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

  where we have included, purely for your convenience, again the noisy channel output sequence $\vec{y}$.

(a) [8 Points] *Reprinted for your convenience*
Channel Coding: From the RSA stage, we get 10 bits. They simply append a '0' at the end so as to make it 11 bits. Then, what was used is the *systematic* $(15, 11)$ Hamming code, specifically with the following parity check matrix:

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

$$\vec{y} = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

where we have included, purely for your convenience, again the noisy channel output sequence $\vec{y}$.

Solution:

- First, we check if the received noisy channel output happens to be a codeword. For this, it suffices to compute the syndrome:

  $$\vec{s} = \vec{y}H^T = (1, 1, 1, 0).$$

  Clearly, our $\vec{y}$ is not a codeword.

- We are using a Hamming code. Hamming codes have minimum distance 3. So, they can correct a single error. This means that we are looking to correct a single error. Remember that a special feature of Hamming codes is that each column of the parity check matrix is unique. All we have to do is to find the column corresponding to the syndrome $\vec{s} = \vec{y}H^T = (1, 1, 1, 0)$. This is the fourth column. Therefore, the error happened in the fourth column, and the minimum distance decoder will output the codeword

  $$\vec{c} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

- Finding the 11 information bits that actually correspond to this length-15 codeword is, of course, a bit of hard work in general. But luckily, here, we know that a systematic code was used. Therefore, the information bits are simply given by the first 11 bits of the length-15 codeword that we found, that is

  $$0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0 \tag{1}$$

  The last '0' can be removed as it was only appended to get to 11 bits, so as to fit the Hamming code. Therefore, the two 5-bit sequences are 00100 and 01101.

(b) [8 Points] *Reprinted for your convenience*
Encryption: The binary string coming from the Huffman code is now encrypted using RSA. This is a little tricky since RSA does not directly take in binary strings. Here is how they proceeded: The binary string is first cut into chunks of 4 bits each. Each chunk is converted into an integer following the usual binary expansion rule, and then adding '1' (so that we do not feed a '0' to RSA encryption), as in the following table:

chunks of 4 bits of source coding output	input to RSA
0000	1
0001	2
0010	3
⋮	⋮
1111	16

RSA output	towards input to channel code
1	00001
2	00010
3	00011
⋮	⋮
20	10100

The integer "input to RSA" is encrypted with RSA with public parameters $(m = 21, e = 5)$. The output of the RSA encryption is an integer between 1 and 20 (since we never use the input '0'). We convert this integer into 5 bits as in the above table (using standard binary representation). In this way, note that if the original string (source coding output) was $2 \times 4 = 8$ bits long, then the output of the RSA encryption will be $2 \times 5 = 10$ bits long.
Backup plan for Part *(b):* [Maximum 7 points] If you could not solve Part *(a)*, you can do Part *(b)* assuming that the answer to Part *(a)* was $0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0$ (which are the 11 original transmitted bits making up the input of channel coding). Recall that the last '0' was only appended to get to 11 bits and can thus be removed.
Solution:

- In Part *(a)*, we found the 10 information bits:

$$0, 0, 1, 0, 0, 0, 1, 1, 0, 1 \tag{2}$$

  Therefore, the two 5-bit sequences are 00100 and 01101. Using the mapping table given in the problem statement, these correspond to the integers 4 and 13, respectively.

- Next, we need to perform RSA decryption. For this, we need to find the decoding exponent. We have $21 = 3 \times 7$. Therefore, $k = \text{lcm}(2, 6) = 6$. We need $ed + \ell k = 1$. Or $5d + 6\ell = 1$. You can do this with Extended Euclid, but the numbers are so small that we can just try out (or guess the solution): Plug in $d = 5$, then $5d = 25$, but of course $4 \times 6 = 24$, so selecting $\ell = -4$ does the trick. That is, we have found our decoding exponent to be $d = 5$.

- It is a simple matter to find $[4^5]_{21} = [16]_{21}$ and $[13^5]_{21} = [13]_{21}$.

- So, we continue with 16 and 13. Using now the other conversion table given in the problem statement, we find the corresponding bit strings to be 1111 and 1100.

Solution for the First Backup Plan: For the first backup plan, we start with

$$0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0 \tag{3}$$

The last '0' can be removed as it was only appended to get to 11 bits, so as to fit the Hamming code. Therefore, the two 5-bit sequences are 01111 and 00010. Using the mapping table given in the problem statement, these correspond to the integers 15 and 2, respectively. See above in the main solution: the decoding exponent for RSA is $d = 5$. Thus, we find $[15^5]_{21} = [15]_{21}$ and $[2^5]_{21} = [11]_{21}$.
So, we continue with 15 and 11. Using now the other conversion table given in the problem statement, we find the corresponding bit strings to be 1110 and 1010.

(c) [8 Points] *Reprinted for your convenience*
You know the original message was either '*foo*' or '*bar*'. You know that the original message was compressed letter-by-letter using a Huffman code, but you don't know which Huffman code. You only know that the Huffman code was designed for the symbol probabilities $p(X = r) = 0.5, p(X = o) = 0.25, p(X = a) = 0.125, p(X = f) = 0.0625, p(X = b) = 0.0625$.
Backup plan for Part *(c):* [Maximum 7 points] If you could not solve Part *(b)*, you can do Part *(c)* assuming that in Part *(b)*, RSA decrpytion gave you the integers 2 and 6 (in this order).
(*Note:* If you solved Part *(b)* with the "Backup plan" and continue with your solution from Part *(b)*, then you can get a score of up to 8 Points for Part *(c)*.)
Solution:

- In Part *(b)*, we found 16 and 13. Using now the other conversion table given in the problem statement, we find the corresponding bit strings to be 1111 and 1100.
- We append the two to obtain 11111100. We know that this came from a Huffman code designed for the given probabilities.
- Therefore, let us discuss Huffman codes for the given source probabilities:
  (a) The key observation here is to realize that the shape of the tree is unique.
  (b) The only question is, for each branching point, which side to put the zero and which side to put the one.
  (c) We know that the original sequence starts with either 'f' or 'b', which are at the very bottom of the tree, meaning that each must be four bits. But our sequence starts with 1111. So, we conclude that the Huffman codewords for 'f' or 'b' must be 1110 and 1111, but we cannot know which is which.
  (d) Let us write this insight into our Huffman tree. Now, looking at the tree, we know that we *must* have that the codeword for *r* is '0', for 'o' is '10' and for 'a' is '110'. The remaining four bits of our sequence, which are 1100, must be decoded in this fashion. You can directly see that this must be 'ar', so we can conclude that 'bar' was transmitted.

Solution for the First Backup Plan, continued: To continue the first backup plan, we found 15 and 11. Using now the other conversion table given in the problem statement, we find the corresponding bit strings to be 1110 and 1010.
We append the two to obtain 11101010. This came from a Huffman code. So, let us discuss Huffman codes for the given source probabilities. The key observation here is to realize that the shape of the tree is unique. The only question is, for each branching point, which side to put the zero and which side to put the one. We know that the original sequence starts with either 'f' or 'b', which are at the very bottom of the tree, meaning that each must be four bits. But our sequence starts with 1110. So, we conclude that the Huffman codewords for 'f' or 'b' must be 1110 and 1111, but we cannot know which is which. Let us write this insight into our Huffman tree. Now, looking at the tree, we know that we *must* have that the codeword for *r* is '0', for 'o' is '10' and for 'a' is '110'. The remaining four bits of our sequence, which are 1010, must be decoded in this fashion. You can directly see that this must be 'oo', so we can conclude that 'foo' was transmitted.
Solution for the Second Backup Plan: For the second backup plan, we directly start with the two original integers 2 and 6. Using now the conversion table given in the problem statement, we find the corresponding bit strings to be 0001 and 0101. We append the two to obtain 00010101. This came from a Huffman code. So, let us discuss Huffman codes for the given source probabilities. The key observation here is to realize that the shape of the tree is unique. The only question is, for each branching point, which side to put the zero and which side to put the one. We know that the original sequence starts with either 'f' or 'b', which are at the very bottom of the tree, meaning that each must be four bits. But our sequence starts with 0001. So, we conclude that the Huffman codewords for 'f' or 'b' must be 0001 and 0000, but we cannot

know which is which. Let us write this insight into our Huffman tree. Now, looking at the tree, we know that we *must* have that the codeword for r is '1', for 'o' is '01' and for 'a' is '001'. The remaining four bits of our sequence, which are 0101, must be decoded in this fashion. You can directly see that this must be 'oo', so we can conclude that 'foo' was transmitted.