RoboLab

Assignments

01 - Hamming Codes

Florian Lubitz
Fakultät Informatik
5106912
florian.lubitz@mailbox.tu-dresden.de

Task 1

The constructed generator matrix $G'_{4,7}$ and the corresponding parity-check matrix $H'_{3,7}$ without extension look like this:

$$G'_{4,7} := \begin{array}{c} \begin{matrix} p_1 & p_2 & d_1 & p_3 & d_2 & d_3 & d_4 \end{matrix} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

$$\Rightarrow A^T = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \rightarrow A = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix}$$

$$\Rightarrow H'_{3,7} := \begin{array}{c} \begin{matrix} p_1 & p_2 & d_1 & p_3 & d_2 & d_3 & d_4 \end{matrix} \\ \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

Adding the additional parity bit p_4 for the extended Hamming Code for the generator matrix $G_{4,8}$ looks like this:

$$G'_{4,8} := \begin{array}{c} \begin{matrix} p_1 & p_2 & d_1 & p_3 & d_2 & d_3 & d_4 & p_4 \end{matrix} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

And the corresponding parity-check matrix $H_{4,8}$ looks like this:

$$\Rightarrow H'_{4,8} := \begin{array}{c} \begin{matrix} p_1 & p_2 & d_1 & p_3 & d_2 & d_3 & d_4 & p_4 \end{matrix} \\ \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

Task 2

$$
\begin{array}{c}
\begin{array}{cccccccc} p_1 & p_2 & d_1 & p_3 & d_2 & d_3 & d_4 & p_4 \end{array} \\
G'_{4,8} := \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix}
\end{array}
$$

To convert this non-systematic generator matrix into a systematic one, we have to apply the following steps:

Step 1:

$$
\begin{array}{r}
\\ r_2 + r_1 \\ \\ r_4 + r_1
\end{array}
\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}
$$

Step 2:

$$
\begin{array}{r}
r_1 + r_2 \\ \\ r_3 + r_2 \\ \\
\end{array}
\begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}
$$

Step 3:

$$
\begin{array}{r}
\\ r_2 + r_3 \\ \\ r_4 + r_3
\end{array}
\begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}
$$

Step 4:

$$
\begin{array}{r}
r_1 + r_4 \\ r_2 + r_4 \\ \\ \\
\end{array}
\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}
$$

$$
\begin{array}{c}
\begin{array}{cccccccc} d_1 & d_2 & d_3 & d_4 & p_1 & p_2 & p_3 & p_4 \end{array} \\
\Rightarrow G_{4,8} \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}
\end{array}
$$

As G is defined as $G := (I_k|-A^T)$, we can read the parity-check matrix H directly from the matrix A: $H := (-A|I_{n-k})$:

$$H_{4,8} := \begin{array}{c} \begin{matrix} d_1 & d_2 & d_3 & d_4 & p_1 & p_2 & p_3 & p_4 \end{matrix} \\ \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

Task 3

$$\begin{aligned}
\vec{x_1} &= (\vec{a_1} \cdot G') \mod 2 = (0100) \cdot G' \mod 2 = (10011001) \mod 2 = (10011001)\, p_4 = 1 \\
\vec{x_2} &= (\vec{a_2} \cdot G') \mod 2 = (1001) \cdot G' \mod 2 == (22110011) \mod 2 = (00110011)\, p_4 = 01 \\
\vec{x_3} &= (\vec{a_3} \cdot G') \mod 2 = (0011) \cdot G' \mod 2 == (10000111)\, p_4 = 1 \\
\vec{x_4} &= (\vec{a_4} \cdot G') \mod 2 = (1101) \cdot G' \mod 2 == (10101010)\, p_4 = 0
\end{aligned}$$

Task 4

$$\begin{aligned}
\vec{x_1} &= (11001101) \\
\vec{x_2} &= (10011001) \\
\vec{x_3} &= (11011011) \\
\vec{x_4} &= (11010101)
\end{aligned}$$

Parity bit p_4

$$\begin{aligned}
\vec{x_1} &= (11001101) \rightarrow p_4 = 1 \rightarrow \text{error} \\
\vec{x_2} &= (10011001) \rightarrow p_4 = 1 \rightarrow \text{correct} \\
\vec{x_3} &= (11011011) \rightarrow p_4 = 1 \rightarrow \text{correct} \\
\vec{x_4} &= (11010101) \rightarrow p_4 = 1 \rightarrow \text{error}
\end{aligned}$$

Syndrome vector

The syndrome vector $\vec{z}$ is calculated by multiplying the received vector $\vec{x}$ with the parity-check matrix H, where the last bit is the parity bit p_4 and the first three bits are the syndrome vector $\vec{z}$:

$$\vec{z} = (\vec{x} \cdot H) \mod 2$$

This results in the following syndrome vectors:

$$\begin{aligned}
\vec{z_1} &= (\vec{x_1} \cdot H) \mod 2 = ((11001101) \cdot H) \mod 2 \\
&= \begin{pmatrix} 2 & 2 & 2 & 5 \end{pmatrix} \mod 2 = \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix} \\
&\Rightarrow \text{error in parity bit} \\
\vec{z_2} &= (\vec{x_2} \cdot H) \mod 2 = ((10011001) \cdot H) \mod 2 \\
&= \begin{pmatrix} 2 & 2 & 2 & 4 \end{pmatrix} \mod 2 = \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix} \\
&\Rightarrow \text{no error} \\
\vec{z_3} &= (\vec{x_3} \cdot H) \mod 2 = ((11011011) \cdot H) \mod 2 \\
&= \begin{pmatrix} 3 & 2 & 4 & 6 \end{pmatrix} \mod 2 = \begin{pmatrix} 1 & 0 & 0 & 1 \end{pmatrix} \\
&\Rightarrow \text{error on position 1, try correction} \\
\vec{z_4} &= (\vec{x_4} \cdot H) \mod 2 = ((11010101) \cdot H) \mod 2 \\
&= \begin{pmatrix} 2 & 3 & 3 & 5 \end{pmatrix} \mod 2 = \begin{pmatrix} 0 & 1 & 1 & 1 \end{pmatrix} \\
&\Rightarrow \text{error on position 6, try correction}
\end{aligned}$$

Error correction

Correct $\vec{x_3}$

For error correction, we have to flip the bit in the received vector $\vec{x_3}$ at the position given by the syndrome vector $\vec{z_3}$:

$$\begin{aligned}
\vec{z_3} = (1001) \to \text{remove parity bit} \to \vec{z_3} = (100) \to \text{convert to decimal (LSB)} \to 1 \\
\vec{x_3'} = (11011011) \to \text{flip bit at position 1} \to \vec{x_4} = (01011011)
\end{aligned}$$

To validate the correction, we calculate the syndrome vector again:

$$\begin{aligned}\vec{z}_3' &= (\vec{x}_3' \cdot H) \mod 2 = ((01011011) \cdot H) \mod 2 \\ &= \begin{pmatrix}1 & 1 & 1 & 0\end{pmatrix} \Rightarrow \text{multiple errors}\end{aligned}$$

Correct $\vec{x}_4$

For error correction, we have to flip the bit in the received vector $\vec{x}_4$ at the position given by the syndrome vector $\vec{z}_4$:

$$\begin{aligned}\vec{z}_4 = (0111) \rightarrow \text{remove parity bit} \rightarrow \vec{z}_4 = (011) \rightarrow \text{convert to decimal (LSB)} \rightarrow 6 \\ \vec{x}_4 = (11010101) \rightarrow \text{flip bit at position 6} \rightarrow \vec{x}_4' = (11010001)\end{aligned}$$

To validate the correction, we calculate the syndrome vector again:

$$\begin{aligned}\vec{z}_4' &= (\vec{x}_4' \cdot H) \mod 2 = ((11010001) \cdot H) \mod 2 \\ &= \begin{pmatrix}0 & 0 & 1 & 1\end{pmatrix} \Rightarrow \text{error on position 4, repeat} \\ \vec{x}_4' &= (11010001) \rightarrow \text{flip bit at position 4} \rightarrow \vec{x}_4'' = (11000001) \\ \vec{z}_4'' &= (\vec{x}_4'' \cdot H) \mod 2 = \begin{pmatrix}1 & 1 & 0 & 1\end{pmatrix} \Rightarrow \text{error on position 3, repeat} \\ \vec{x}_4'' &= (11000001) \rightarrow \text{flip bit at position 3} \rightarrow \vec{x}_4''' = (11100001) \\ \vec{z}_4'' &= (\vec{x}_4''' \cdot H) \mod 2 = \begin{pmatrix}0 & 0 & 0 & 0\end{pmatrix} \Rightarrow \text{no error}\end{aligned}$$

Appendix

Python code for encoding and decoding

```
# %%
from typing import Tuple, Dict

Matrix = Tuple[Tuple[int, ...], ...]


def transpose(matrix: Matrix) -> Matrix:
    """Return the transpose of a matrix."""
    return tuple(zip(*matrix))
```

```

# %%
G_non_sys: Matrix = (
    (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, 0, 1, 0, 0, 1, 0),
)
H_non_sys: Matrix = (
    (1, 0, 1, 0, 1, 0, 1, 0),
    (0, 1, 1, 0, 0, 1, 1, 0),
    (0, 0, 0, 1, 1, 1, 1, 0),
    (1, 1, 1, 1, 1, 1, 1, 1),
)
G_sys: Matrix = (
    (1, 0, 0, 0, 0, 1, 1, 1),
    (0, 1, 0, 0, 1, 0, 1, 1),
    (0, 0, 1, 0, 1, 1, 0, 1),
    (0, 0, 0, 1, 1, 1, 1, 0),
)
H_sys: Matrix = (
    (0, 1, 1, 1, 1, 0, 0, 0),
    (1, 0, 1, 1, 0, 1, 0, 0),
    (1, 1, 0, 1, 0, 0, 1, 0),
    (1, 1, 1, 0, 0, 0, 0, 1),
)

# %%
Vector = Tuple[int, ...]


def encode(a: Vector, G: Matrix) -> Vector:
    """Encode a message vector using a generator matrix."""
    return tuple(sum(a * b for a, b in zip(a, col)) % 2 for col in transpose(G))

# %%
words: Dict[str, Vector] = {
    "1": (0, 1, 0, 0),
    "2": (1, 0, 0, 1),
    "3": (0, 0, 1, 1),
    "4": (1, 1, 0, 1),
```

```
}

print("Task 3, encode with non-systematic matrix")
for word, vector in words.items():
    encoded = encode(vector, G_non_sys)
    print(
        f"{word}: {encoded}, Parity: {encoded[-1]} -> {sum(encoded) % 2 == 0}"
    )

# %%


# %%
words_2: Dict[str, Vector] = {
    "1": (1, 1, 0, 0, 1, 1, 0, 1),
    "2": (1, 0, 0, 1, 1, 0, 0, 1),
    "3": (1, 1, 0, 1, 1, 0, 1, 1),
    "4": (1, 1, 0, 1, 0, 1, 0, 1),
}

# %%
def get_syndrome(x: Vector, H: Matrix) -> Vector:
    """Check if a vector has the correct parity."""
    return tuple(sum(a * b for a, b in zip(x, col)) % 2 for col in H)


print("Task 4, check with non-systematic matrix")
for word, vector in words_2.items():
    *syndrome, overall_parity = get_syndrome(vector, H_sys)
    syndrome_number = sum(2**i * bit for i, bit in enumerate(syndrome))
    print(
        f"Word {word}: Syndrome: {syndrome} ({syndrome_number}), overall_parity: {overall_parity}"
    )
    if syndrome_number == 0 and overall_parity == 0:
        print("\tno error")
    elif syndrome_number == 0 and overall_parity == 1:
        print("\tError in p4")
    elif syndrome_number >= 1 and overall_parity == 1:
        while syndrome_number >= 1 and overall_parity == 1:
```

```
            print("\tError on position " + str(syndrome_number) + ". Try to
                correct it")
            vector = tuple(
                bit ^ (1 if i == syndrome_number - 1 else 0)
                for i, bit in enumerate(vector)
            )
            print(f"\tCorrected vector: {vector}")
            print(f"\tCorrected check: {get_syndrome(vector, H_sys)}")
            *syndrome, overall_parity = get_syndrome(vector, H_sys)
            syndrome_number = sum(2**i * bit for i, bit in enumerate(syndrome))
            if(syndrome_number == 0 and overall_parity == 0):
                print("\tCorrected!")
                break
            elif(syndrome_number >= 1 and overall_parity == 0):
                print("\tMultiple errors")
                break
    else:
        print("\tMultiple errors")
    print()
```