

# Aufgabe 4: Zara Zackigs Zurückkehr

Teilnahme-ID: 62454

Bearbeiter dieser Aufgabe:  
Philip Gilde

19. April 2022

## Inhaltsverzeichnis

<b>1</b>	<b>Lösungsidee</b>	<b>1</b>
<b>2</b>	<b>Umsetzung</b>	<b>2</b>
<b>3</b>	<b>Laufzeit</b>	<b>3</b>
<b>4</b>	<b>Beispiele</b>	<b>3</b>
<b>5</b>	<b>Quellcode</b>	<b>3</b>

## 1 Lösungsidee

Von den gegebenen  $n$  Karten mit jeweils  $m$  Bits werden  $p$  Karten gesucht, so dass das exklusive Oder (im Folgenden als XOR abgekürzt) von  $p - 1$  Karten gleich der  $p$ . Karte ist.

$$\begin{array}{lcl} & k_1 \oplus k_2 \oplus \dots \oplus k_{p-1} = k_p & | \oplus k_p \\ \Leftrightarrow & k_1 \oplus k_2 \oplus \dots \oplus k_{p-1} \oplus k_p = 0 & \end{array}$$

Diese Gleichung lässt sich zu jeder der  $p$  Karten umstellen. Es werden also  $p$  Karten gesucht, deren XOR gleich einer Karte mit  $m$  Nullen ist. Dieses Problem lässt sich umformulieren zu einem linearen Gleichungssystem im Galois-Feld  $GF(2)$ . Dieses besteht nur aus den beiden Elementen 0 und 1. Die Addition im Feld entspricht dem XOR, die Multiplikation einem UND. Die gemischten Karten entsprechen der Matrix  $K \in GF(2)^{n \times m}$ .  $K_n$  ist dabei die  $n$ -te Karte und  $K_{n,m}$  das  $m$ -te Bit der  $n$ -ten Karte. Gesucht wird der Vektor  $v \in GF(2)^n$ , so dass dieses lineare Gleichungssystem gilt:

$$\begin{array}{l} K_{1,1}v_1 + K_{2,1}v_2 + \dots + K_{n,1}v_n = 0 \\ K_{1,2}v_1 + K_{2,2}v_2 + \dots + K_{n,2}v_n = 0 \\ \dots \\ K_{1,m}v_1 + K_{2,m}v_2 + \dots + K_{n,m}v_n = 0 \end{array}$$

In Matrixform:

$$K^T v = 0$$

Dabei bestimmt  $v_n$ , ob die  $n$ -te Karte zu den gesuchten Karten gehört.

Die Menge von Vektoren, die sich oben für  $v$  einsetzen lassen, wird als Nullraum oder Kern der Matrix  $K^T$  bezeichnet. In  $GF(2)$  kann ein Vektor nicht skaliert werden, weil er nur mit entweder 0 oder 1 multipliziert werden kann. Somit besteht der Nullraum aus allen möglichen Kombinationen von Summen

der Basisvektoren. Wenn  $r$  der Rang von  $K^T$  ist, dann ist  $q = n - r$  die Anzahl der Basisvektoren des Nullraums. Der Rang von  $K^T$  ist die Anzahl linear unabhängiger Zeilen beziehungsweise Spalten (diese beiden Werte sind gleich). Wenn, wie in der ursprünglichen Aufgabe,  $n < m$ , dann ist der Rang in der Regel  $n - 1$ , denn nur eine Karte, die Wiederherstellungskarte, ist linear abhängig von den anderen. Die Wahrscheinlichkeit, dass  $h$  Karten mit jeweils  $b$  Bits voneinander linear unabhängig sind, ist, wenn diese zufällig und erzeugt sind und Nullen und Einsen gleich wahrscheinlich sind, wovon der Einfachheit halber ausgegangen wird, nach [1] gegeben durch

$$p_h = \prod_{i=1}^h (1 - 2^{i-1-b})$$

Diese ist für  $h = n - 1 = 110$  und  $b = m = 128$  hoch genug, um den anderen Fall zunächst außen vor zu lassen. Somit sind alle bis auf eine der 111 Karten linear unabhängig voneinander. Der Nullraum besteht damit aus nur  $q = n - (n - 1) = n - n + 1 = 1$  Vektor. Dieser muss an den Stellen der 11 echten Karten 1 und sonst überall 0 sein. Somit haben wir die echten Karten gefunden.

Um die Basisvektoren zu finden, wird das in [2] beschriebene Verfahren verwendet. Dabei wird zuerst  $K^T$  mit der Identitätsmatrix zu  $\begin{bmatrix} K^T \\ I \end{bmatrix}$  erweitert.  $\begin{bmatrix} K^T \\ I \end{bmatrix}^T$  wird dann mithilfe des Gauß-Jordan-Algorithmus in Stufenform gebracht, was dann in der Matrix  $\begin{bmatrix} B \\ C \end{bmatrix}^T$  resultiert. Die Basis des Nullraums bilden die Spalten von  $C$ , deren entsprechende Spalten in  $B$  Null sind. Das lässt sich damit begründen, dass die elementaren Reihentransformationen der Transponierten beziehungsweise elementaren Spaltentransformationen der Multiplikation mit einer Matrix  $P$  entsprechen, also  $\begin{bmatrix} K^T \\ I \end{bmatrix} P = \begin{bmatrix} B \\ C \end{bmatrix}$ . Daraus folgt  $IP = P = C$  und  $K^T P = K^T C = B$ .

$$\begin{array}{lll} & K^T C = B & | \cdot C^{-1} \\ \Leftrightarrow & K^T = BC^{-1} & | \cdot v \\ \Leftrightarrow & K^T v = BC^{-1}v & | \text{Es wird } C^{-1}v = w \text{ gesetzt} \\ & = Bw & | \text{Damit } v \text{ zum Nullraum gehört, wird verlangt:} \\ & = 0 & \end{array}$$

Weil alle Spalten in  $B$ , die nicht Null sind, linear unabhängig voneinander sind (Das bedeutet nämlich Spaltenstufenform), gilt  $Bw = 0$  nur wenn die Einträge von  $w$ , die nicht Null sind, den Nullspalten von  $B$  entsprechen. Die Basis der Vektoren  $w$ , für die  $Bw = 0$  gilt, sind also die verschiedenen Vektoren, die eine Eins bei einer Nullspalte von  $B$  und sonst überall Nullen haben. Da  $C^{-1}v = w \Leftrightarrow v = Cw$  definiert wurde, sind die Spalten von  $C$ , die den Nullspalten von  $B$  entsprechen, die Basis des Nullraums.

Wenn  $q > 1$  ist, kann es sein, dass keiner der Basisvektoren des Nullraums 11 Einsen beinhaltet, sondern eine Linearkombination dieser. In diesem Fall wird jede der  $2^q$  Kombinationen der Basisvektoren durchprobiert, bis eine davon 11 Einsen enthält.

Die richtige Karte für das  $s$ -te Haus kann gefunden werden, in dem man die Karten aufsteigend sortiert und zuerst die  $s$ -te und dann die  $s + 1$ -te Karte ausprobiert. Wenn die Sicherungskarte kleiner als die Öffnungskarte des  $s$ -ten Hauses ist, liegt sie davor im Stapel und die Öffnungskarte an der Stelle  $s + 1$ . Wenn sie größer ist, dann liegt sie dahinter im Stapel und die Öffnungskarte an Stelle  $s$ .

Das Verfahren stößt an seine Grenzen, wenn  $n > m$  ist. Der Rang von  $K^T$  ist dann nämlich höchstens  $m$ , wodurch der Nullraum  $q = n - m$  Basisvektoren hat. Diese Basisvektoren müssen nun nicht zwangsweise einen mit 11 Einsen beinhalten, dieser könnte auch eine Linearkombination der anderen Basisvektoren sein. Wenn das der Fall ist, müsste man alle  $2^q$  Kombinationen von Basisvektoren durchprobieren, bis eine davon 11 Einsen beinhaltet, was für die Beispieleingabe auf der BwInf-Webseite mit 161 Karten zu je 128 Bit  $2^q = 2^{n-m} = 2^{161-128} = 2^{33} = 8.589.934.592$  Kombinationen sind. Diese lassen sich nicht wirklich in überschaubarer Zeit durchprobieren.

## 2 Umsetzung

Die Lösung wurde in Python umgesetzt. Dabei wurde die Bibliothek `NumPy` verwendet, um die Koeffizientenmatrizen darzustellen.

### 3 Laufzeit

Ein Durchlauf des Gauß-Jordan-Algorithmus hat eine Laufzeit von  $\mathcal{O}(n^3)$ . Der Algorithmus wird einmal ursprünglich und dann für jeden der  $q$  Basisvektoren des Nullraums durchgeführt, somit ist die Laufzeit  $\mathcal{O}((q+1)n^3)$

### 4 Beispiele

### 5 Quellcode

### Literatur

- [1] A. F. (<https://math.stackexchange.com/users/54227/alfonso-fernandez>), “Probability that a random binary matrix will have full column rank?” Mathematics Stack Exchange. URL:<https://math.stackexchange.com/q/564699> (Version: 2013-11-12).
- [2] Wikipedia contributors, “Kernel (linear algebra) — Wikipedia, the free encyclopedia.” [https://en.wikipedia.org/w/index.php?title=Kernel\\_\(linear\\_algebra\)&oldid=1070674634](https://en.wikipedia.org/w/index.php?title=Kernel_(linear_algebra)&oldid=1070674634), 2022. [Online; Abgerufen 18. April 2022].