Durchführung

Zunächst werden Objekte, die nur zwei verschiedene Zustände z_1, z_2 annehmen können, benötigt, wie zum Beispiel Magnetplättchen (»Rot« oder »Blau«) oder die von uns benutzten Fische (»auf dem Rücken schwimmen« oder »auf dem Bauch schwimmen«).

1. Ein Zuschauer legt die Fische mit beliebigem Zustand in einer $m \times n$ Matrix aus.
2. Ich lege eine weitere Spalte und eine weitere Zeile mit den Objekten genau so an, dass eine gerade Anzahl eines bestimmten Zustandes z_1 (z. B. gerade Anzahl »Fische auf dem Rücken«) in jeder Zeile und jeder Spalte ist.
3. Der Zuschauer dreht nun einen Fisch, ohne dass ich hinsehe.
4. Um den »Fehler« zu finden suche ich zeilenweise nach einer ungeraden Anzahl von z_1 (»Fische auf dem Rücken«). Ich merke mir diese Zeile j. Nun suche ich spaltenweise nach einer ungeraden Anzahl von z_1. Ich merke mir diese Spalte k. Das vom Zuschauer umgedrehte Objekt befindet sich in der j. Zeile und in der k Spalte.

Benötigte Materialien

Zweifarbige Magnetplättchen sind im Schulbedarf erhältlich. Der Trick kann auch mit anderen Elementen, die in einer Matrix angeordnet werden und zwei Zustande haben können, durchgeführt werden.

Zum Beispiel können Kreise und Kreuze an die Tafel gezeichnet werden. Für die Übung der Teilnehmenden haben wir Fische, die auf dem Rücken und dem Bauch schwimmend angeordnet werden konnten (siehe Vorlage unten), verwendet.

Wie funktioniert das Ganze?

Mit dem Anfügen eines Paritätsbits für jede Zeile und Spalte einer binären Matrix wird redundante Information angefügt. Dadurch wird ein zweidimensionaler Fehlerkorrekturcode erschaffen, sodass man einen Fehler nicht nur finden, sondern auch den ursprünglichen Zustand der Bits wiederherstellen kann. Bei 2 oder 3 Fehlern können diese zwar erkannt, aber nicht mehr korrigiert werden. Sind 4 Fehler in der Matrix, so kann es passieren, dass diese nicht mehr erkannt werden.

Paritätsbits werden bei Datenübertragung und Datenspeicherung benutzt, um Fehler zu erkennen. Ein praktisches Beispiel ist die zerkratzte CD, bei der Kompression und Redundanz gleichzeitig angewendet werden und zu einer Fehlerkorrektur führen.

Literatur

Tim Bell, Mike Fellows und Ian H. Witten. *Computer Science Unplugged*. 2006. URL: `https://csunplugged.org/de/` (besucht am 24. 03. 2019)

