Selbstkorrigierende Kodierungen: Korrektur von zwei und drei Fehlern mit dem erweiterten Kartentrick

Fachdidaktik Informatik I HS 22

Manuela Fischer & Manuel Wettstein

19. Februar 2023

Bei der Informationsübertragung und Datenspeicherung passieren oftmals Fehler, welche undetektiert gravierende Konsequenzen haben können, von einfachem Datenverlust bis zu dem Verlust eines Menschenlebens. Fehlererkennende oder gar -korrigierende Kodierungen können das so entstehende Risiko stark reduzieren und sind deshalb in der modernen Technologie allgegenwärtig.

Aus Kapitel 4 des ABZ-Lehrbuchs "INFORMATIK: Data Science und Sicherheit" kennen wir den Kartentrick, der uns erlaubt, einen Fehler zu korrigieren und zwei Fehler zu detektieren. In der Praxis ist natürlich eine grössere Anzahl Fehler möglich und leider auch üblich. Wir untersuchen deshalb, wie der Kartentrick verallgemeinert werden kann für die Korrektur von zwei und drei Fehlern. Für diesen Zweck wird sich die Berechnung und Übertragung von Prüfbits entlang anderer Richtungen als nur horizontal und vertikal als hilfreich herausstellen.

Wir erarbeiten das Thema mit Hilfe von Aufgaben. Diese sind in drei Kategorien unterteilt und können anhand des Symbols, mit dem sie gekennzeichnet sind, unterschieden werden.

- Durch Bearbeiten dieser *Knobelaufgaben* wird die Schwierigkeit des zu lösenden Problems erkannt, ein Interesse für die Lösung geweckt und es werden wichtige Erkenntnisse für die Lösung erworben.
- Durch diese *Lern- und Projektaufgaben* wird entweder eine Thematik weiter vertieft oder selbständig ein Schritt in Richtung Lösung des Problems gegangen.
- Durch Lösen dieser *Routineaufgaben* wird das erworbene Wissen gefestigt und Verständnislücken können aufgedeckt werden.

Zudem gibt es für die selbständige Auseinandersetzung mit dem Material die folgenden Hilfestellungen.

- Am Ende jedes Abschnitts werden die wichtigsten Konzepte kurz und prägnant zusammengefasst.
- Ausführliche Lösungen zu den Aufgaben befinden sich im Anhang.

Inhaltsverzeichnis

1 Der Kartentrick: 1-Fehlerkorrektur

Schauen wir doch kurz auf den Kartentrick zurück, um uns die wichtigsten Begriffe und Ideen ins Gedächtnis zu rufen. Beim *Kartentrick* schreiben wir eine als Bitsequenz gegebene Nachricht in Form eines Rechtecks mit den Seitenlängen a und b auf. Der Einfachheit halber nehmen wir im Folgenden immer an, dass $a = b$ gilt und das Rechteck also ein Quadrat ist. Dies ist keine Einschränkung, da wir jede Nachricht durch Anhängen von Nullen auf die gewünschte Länge $a^2 = b^2$ bringen können.

Bei jeder *Zeile* und *Spalte* des entstandenen Quadrats fügen wir jetzt ein *Prüfbit* an, so dass die Summe (modulo 2) aller Bits entlang jeder solchen Zeile oder Spalte gleich 0 ist. Wir vervollständigen das Quadrat mit einem zusätzlichen Prüfbit oben rechts, so dass sich auch die Prüfbits selber zeilen- und spaltenweise jeweils zu 0 aufsummieren. Dies stellt eine Symmetrie her zwischen den Prüfbits und den eigentlichen Bits in der Nachricht: Es gibt also keinen Grund mehr, explizit zwischen diesen beiden Arten von Bits zu unterscheiden. Beide sind ganz einfach Teil eines Quadrats, in dem sich jede Zeile und jede Spalte zu 0 aufsummiert (modulo 2). Das könnte zum Beispiel für $a = b = 5$ wie folgt aussehen.

0	1	1	1	0
1	0	0	0	1
0	1	0	1	0
1	0	1	0	0
0	0	1	0	0

Nachricht ohne Prüfbits

0	0	1	0	1	0
0	1	1	1	0	1
1	0	0	0	1	0
0	1	0	1	0	0
1	0	1	0	0	0
0	0	1	0	0	1

Codewort mit Prüfbits

0	0	1	0	1	0
0	1	1	1	0	1
1	0	0	0	1	0
0	1	1	1	0	0
1	0	1	0	0	0
0	0	1	0	0	1

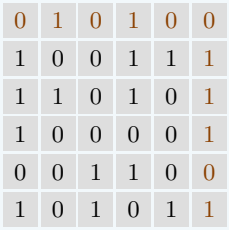
Fehler auf der Kreuzung

Falls beim Übertragen des so generierten *Codeworts* nun ein einziges Bit geflippt wird, summieren sich die Zeile und die Spalte, die dieses Bit enthalten, nicht mehr zu 0, sondern zu 1 auf. Wir erkennen also genau, wo der Fehler passiert ist, nämlich auf der *Kreuzung* ebendieser Zeile und Spalte. In diesem Sinne wird jedes Bit im Codewort von genau zwei Prüfbits überwacht.

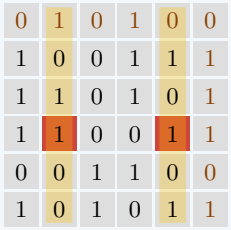
Weil ein Bit nur die Werte 0 und 1 annehmen kann, wissen wir nicht nur, wo der Fehler passiert ist, sondern können ihn auch direkt korrigieren: Die originale Nachricht wird durch Zurückflippen des fehlerhaften Bits wiederhergestellt. Die Kartentrick-Kodierung ist also 1-*fehlerkorrigierend*. Sie muss folglich *Abstand* mindestens 3 haben und somit mindestens 2-*fehlererkennend* sein. Doch welchen Abstand hat sie tatsächlich?

Aufgabe 1.1

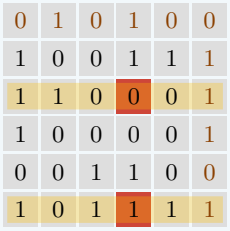
Betrachten Sie das folgende Codewort und die drei Varianten (a), (b) und (c), bei denen sich je zwei fehlerhafte Bits eingeschlichen haben. Die Fehler sind jeweils rot markiert; die Zeilen und Spalten, die sich nicht zu 0 aufsummieren, sind orange hinterlegt. Wie viele zusätzliche Bits müssen Sie bei (a), (b) und (c) jeweils mindestens flippen, um ein anderes gültiges Codewort zu erhalten? Versuchen Sie aus Ihren Überlegungen allgemein zu schliessen, dass die Kartentrick-Kodierung den Abstand 4 hat.



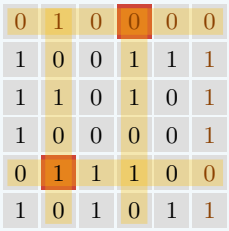
Codewort



(a)



(b)



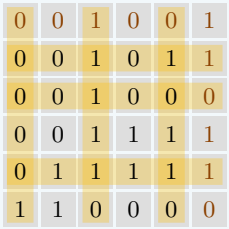
(c)

Lösung auf Seite 25

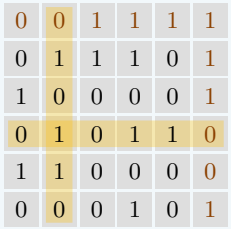
Die Kartentrick-Kodierung hat also tatsächlich Abstand 4. Das bedeutet, dass zwei unterschiedliche Codewörter sich immer durch mindestens vier Flips unterscheiden. Daraus folgt, dass diese Kodierung sogar 3-fehlererkennend ist. Bei bis zu drei Fehlern können wir also immer erkennen, ob die Übermittlung fehlerhaft war oder nicht. Heisst das aber auch, dass wir im fehlerhaften Fall genau angeben können, wie viele Fehler (1, 2 oder 3) passiert sind?

Aufgabe 1.2

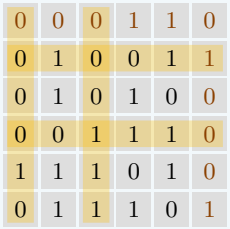
Bei den folgenden Codewörtern haben sich jeweils 0, 1, 2 oder 3 Fehler eingeschlichen. Um Ihnen Rechenarbeit zu ersparen, sind die Zeilen und Spalten, die sich nicht zu 0 aufsummieren, bereits eingezeichnet, jedoch nicht die Positionen der eigentlichen Fehler. Versuchen Sie für jeden Fall zu entscheiden, wie viele Fehler tatsächlich passiert sind. Ist die Antwort immer eindeutig?



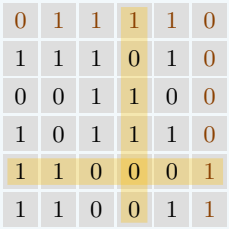
(a)



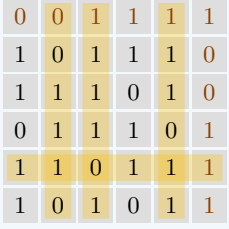
(b)



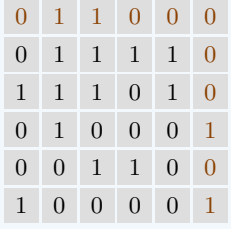
(c)



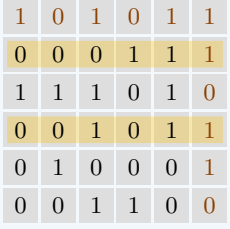
(d)



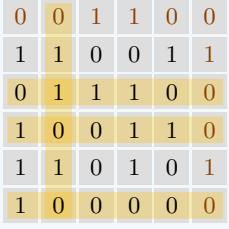
(e)



(f)



(g)



(h)

Lösung auf Seite 26

Wenn wir nur daran interessiert sind herauszufinden, wie viele Fehler passiert sind, dann gibt es bei der Kartentrick-Kodierung also nur eine sehr spezielle Anordnung von drei Fehlern, die sich nicht von dem Fall mit einem einzigen Fehler unterscheiden lässt. In allen anderen Fällen können wir zuverlässig entscheiden, ob 0, 1, 2 oder 3 Fehler passiert sind.

Zusammenfassung

Die Kartentrick-Kodierung hat Abstand 4 und ist somit bereits 3-fehlererkennend. Wir nähern uns demzufolge unserem Ziel einer 3-fehlerkorrigierenden Kodierung. In den meisten Fällen können wir sogar bereits die genaue Anzahl der Fehler (0, 1, 2 oder 3) bestimmen. Diese Unterscheidung wird sich bei der 3-Fehlerkorrektur als besonders wichtig herausstellen.

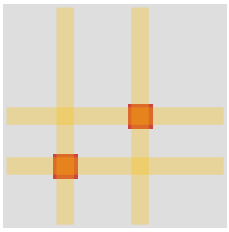
2 Erweiterter Kartentrick: 2-Fehlerkorrektur

Das Ziel dieses Abschnitts ist die Erweiterung des Kartentricks durch eine geschickte Wahl zusätzlicher Prüfbits, welche die Korrektur von zwei Fehlern erlauben. Als ersten Schritt möchten wir genauer verstehen, weshalb die Kartentrick-Kodierung nicht bereits 2-fehlerkorrigierend ist. Dazu betrachten wir nochmals die zwei generischen Fälle aus Aufgabe 4.30 b des Lehrbuchs für die Positionierung von zwei Fehlern.

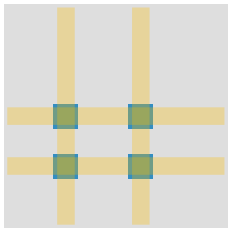
2.1 Fall 2A: Zwei Fehler auf unterschiedlichen Zeilen und Spalten

Zuerst widmen wir uns dem Fall mit zwei Fehlern, die auf unterschiedlichen Zeilen und Spalten liegen. Es gibt also genau zwei Zeilen und zwei Spalten, die einen Fehler anzeigen. Wir verwenden den Begriff *Linie* für eine Zeile oder eine Spalte und nennen eine Linie *fehlerhaft*, falls sie einen Fehler anzeigt, also falls die Summe (modulo 2) aller Bits auf dieser Linie nicht gleich 0 ist.

Die Situation könnte also wie folgt aussehen. Wie Ihnen schon aus dem ersten Abschnitt bekannt sein sollte, markieren wir die Fehler jeweils rot und hinterlegen die fehlerhaften Linien mit oranger Farbe.



2 Fehler



4 Kreuzungen

Die vier fehlerhaften Linien bilden vier *Kreuzungen*, zwei davon mit Fehlern, zwei ohne Fehler. Durch Flippen der zwei fehlerhaften Bits wird das originale Codewort wiederhergestellt. Jedoch können die Fehler auch durch Flippen der Bits auf den zwei anderen Kreuzungen behoben werden, was zu einem alternativen Codewort führt. In anderen Worten ausgedrückt hätte genauso gut das alternative Codewort mit zwei Fehlern an diesen anderen zwei Kreuzungen übertragen werden können. Es gibt also keinen guten Grund, das originale Codewort dem alternativen vorzuziehen. Somit ist die Fehlerkorrektur ohne weitere Information nicht möglich.

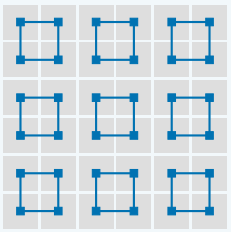
Doch welche Information haben wir denn eigentlich? Wir wissen, dass die Fehler an zwei der vier Kreuzungen liegen müssen, können diese aber noch nicht genau bezeichnen. Die Hoffnung ist, mit zusätzlichen Prüfbits die Fehler auf diesen vier verbleibenden möglichen Positionen zu lokalisieren.

Aufgabe 2.1

Frida hat einen tollen Vorschlag. Sie berechnet für jeden 2×2-Block des Codeworts (also des Quadrats bestehend aus Nachrichtenbits und den Zeilen- und Spaltenprüfbits wie bisher) je ein zusätzliches Prüfbit. Sie addiert dazu alle Bits im gleichen 2×2-Block (markiert durch zusammenhängende blaue Linien im Bild unten) auf und wählt das Prüfbit, so dass die 2×2-Block-Summe plus das Prüfbit gleich 0 ist (modulo 2).

0	1	1	1	0	1
0	1	0	0	1	0
1	0	1	1	1	0
0	1	1	0	0	0
1	1	0	1	0	1
0	0	1	1	0	0

Codewort



2×2-Blöcke

0	0	0
0	1	1
0	1	1

Zusätzliche Prüfbits

Geben Sie für die folgenden vier Fälle an, ob Frida mit Hilfe der Information gegeben durch ihre zusätzlichen Prüfbits die zwei Fehler eindeutig korrigieren kann. Die Fehler sind in der Abbildung zwar bereits markiert, Sie sollen aber natürlich davon ausgehen, dass Frida selber die Fehler noch nicht kennt.

0	1	1	0	0	0
0	1	1	0	1	0
1	1	1	0	0	0
0	1	0	0	0	1
0	0	1	1	0	0
1	0	0	0	0	1

(a)

1	0	1	0	0	0
1	1	0	1	0	1
1	0	1	0	1	0
1	1	0	1	1	1
1	1	0	1	1	0
1	1	1	0	1	0

(b)

1	0	1	1	0	0
0	0	1	0	0	1
0	1	1	1	1	1
1	1	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	0

(c)

0	1	0	0	0	1
1	0	0	0	0	1
0	1	1	0	1	0
0	1	1	1	1	1
1	0	0	0	0	1
0	0	0	1	1	0

(d)

Lösung auf Seite 27

Fridas Blockmethode gibt uns leider auch nicht genügend Information, um in allen Fällen die zwei Fehler korrigieren zu können. Ihr Trick funktioniert nur dann, wenn die Fehler so liegen, dass die vier Kreuzungen der fehlerhaften Linien sich in vier unterschiedlichen 2×2-Blöcken befinden.

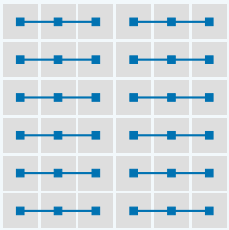
Fridas grundsätzliche Idee, für weitere Bereiche zusätzlich zu Zeilen und Spalten ein Prüfbit zu berechnen, ist aber durchaus erfolgversprechend. Wir wollen diese Idee weiterverfolgen, indem wir versuchen, die Bereiche noch ein bisschen geschickter zu wählen.

Aufgabe 2.2

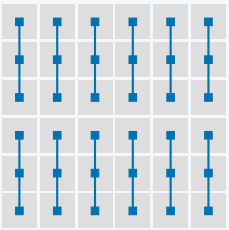
Betrachten Sie die folgende Situation eines übertragenen Codeworts mit zwei Fehlern.

1	1	1	1	0	1
1	1	0	0	1	1
0	1	0	1	0	1
0	0	1	0	0	1
0	0	1	0	1	0
0	1	1	1	1	0

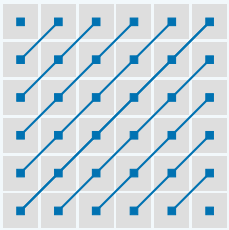
Bestimmen Sie für die vier folgenden Fälle, ob zusätzliche Prüfbits für die dargestellten Bereiche genügen, um die Fehler zu korrigieren. Sie würden also für jeden Bereich von zusammenhängenden blauen Linien je ein Prüfbit berechnen: für die zwölf Zeilenhälften, für die zwölf Spaltenhälften, für die elf Vorwärtsdiagonalen oder für die elf Rückwärtsdiagonalen.



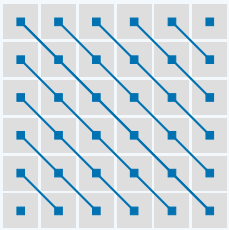
Zeilenhälften



Spaltenhälften



Vorwärtsdiagonalen



Rückwärtsdiagonalen

Lösung auf Seite 27

Die Fehler im spezifischen Codewort von Aufgabe 2.2 lassen sich mit Bereichen definiert sowohl durch *Vorwärtsdiagonalen* als auch durch *Rückwärtsdiagonalen* korrigieren. Tatsächlich genügen beide Arten von *Diagonalen* separat für eine 2-fehlerkorrigierende Kodierung. Wir entscheiden uns willkürlich für die Vorwärtsdiagonalen und werden uns im Folgenden vergewissern, dass diese nicht nur im oben betrachteten Fall, sondern auch im Allgemeinen die nötige Information für eine zuverlässige 2-Fehlerkorrektur enthalten. Die Analyse für die Rückwärtsdiagonalen wäre identisch.

Bei dem in Aufgabe 2.2 betrachteten Fall hatten wir die hilfreiche Eigenschaft, dass keine der vier Kreuzungen auf der gleichen Vorwärtsdiagonalen liegt. Dies lässt uns von einer fehlerhaften Vorwärtsdiagonalen direkt auf den Fehler auf der entsprechenden Kreuzung schliessen. Das gleiche Argument funktioniert auch dann, wenn die zwei Kreuzungen ohne Fehler auf einer gemeinsamen Vorwärtsdiagonalen liegen, solange die beiden Kreuzungen mit Fehler je eine eigene Vorwärtsdiagonale haben. Doch was wäre, wenn die zwei Fehler tatsächlich auf einer gemeinsamen Vorwärtsdiagonalen liegen würden?

Aufgabe 2.3

Flippen Sie im folgenden Codewort zwei Bits, so dass kein Vorwärtsdiagonalen-Prüfbit einen Fehler anzeigen würde. Wieso wäre es dann mit der gegebenen Information trotzdem möglich, die beiden Fehler zu korrigieren?

1	1	0	1	0	1
0	1	1	0	0	0
0	0	0	1	0	1
1	0	1	1	0	1
1	0	0	1	0	0
1	0	0	0	0	1

Lösung auf Seite 28

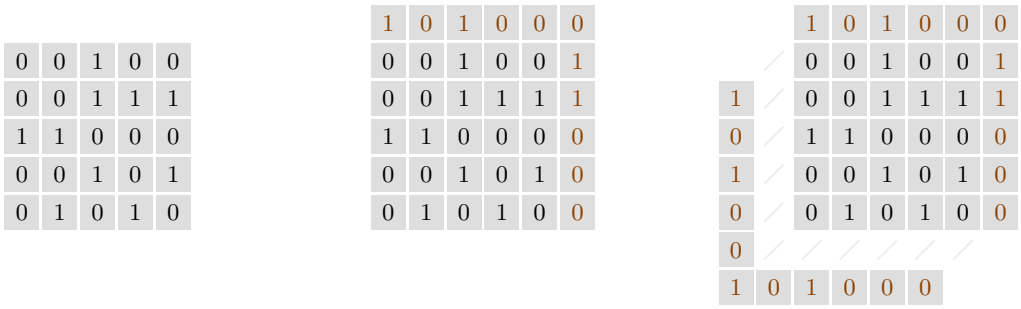
Wenn die zwei Fehler auf der gleichen Vorwärtsdiagonalen liegen, zeigt keine der Vorwärtsdiagonalen einen Fehler an. Dennoch können wir die Position der fehlerhaften Bits herausfinden. Da die zwei Vorwärtsdiagonalen durch die Kreuzung oben links und die Kreuzung unten rechts keinen Fehler anzeigen, können die Fehler nicht dort positioniert sein. Die fehlerhaften Bits müssen also beim Kreuzungspaar sein, das auf der gemeinsamen Vorwärtsdiagonalen liegt.

Zusammenfassung

Wenn die zwei Fehler auf unterschiedlichen Zeilen und Spalten liegen, haben wir vier fehlerhafte Linien, die vier Kreuzungen bilden, auf welchen die Fehler positioniert sein können. Die zusätzlichen Prüfbits für die Vorwärtsdiagonalen ermöglichen es uns, für jede dieser vier Kreuzungen herauszufinden, ob sie ein fehlerhaftes Bit enthält.

2.2 Erste Erweiterung der Kodierung

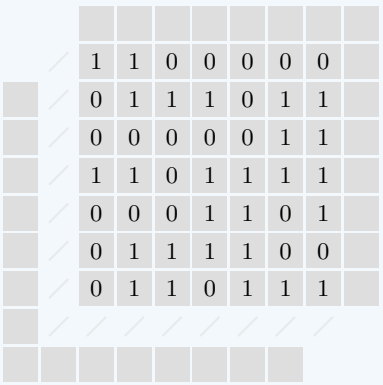
Für die erste *Erweiterung der Kartentrick-Kodierung* mit Vorwärtsdiagonalen gehen wir wie folgt vor: Wir ordnen die Nachricht wie gewohnt in einem hinreichend grossen Quadrat an, erweitern dieses mit einer Zeile und einer Spalte für die Prüfbits wie beim originalen Kartentrick und berechnen schliesslich für jede Vorwärtsdiagonale des Kartentrick-Codeworts die Summe (modulo 2), welche dann, wie unten illustriert, als zusätzliches Prüfbit in das Diagramm eingefügt wird. Wir haben mit den Vorwärtsdiagonalen jetzt also einen dritten Typ *Linien* (zusätzlich zu Zeilen und Spalten), auf denen wir Prüfbits bestimmen.



Nachricht | Kartentrick-Codewort | Erweitertes Codewort

Aufgabe 2.4

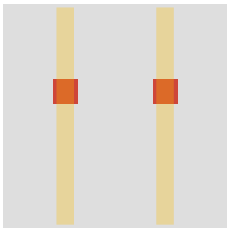
Bestimmen Sie für die folgende Nachricht alle Prüfbits und fügen Sie sie in das untenstehende Diagramm ein.



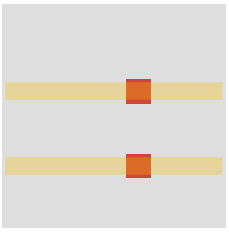
Lösung auf Seite 28

2.3 Fall 2B: Zwei Fehler auf gleicher Zeile oder gleicher Spalte

Wir haben den Fall von zwei Fehlern auf unterschiedlichen Zeilen und Spalten hinreichend verstanden. Doch wie sieht es aus, wenn die zwei Fehler auf einer gemeinsamen Zeile oder Spalte liegen?



2 Fehler auf einer Zeile

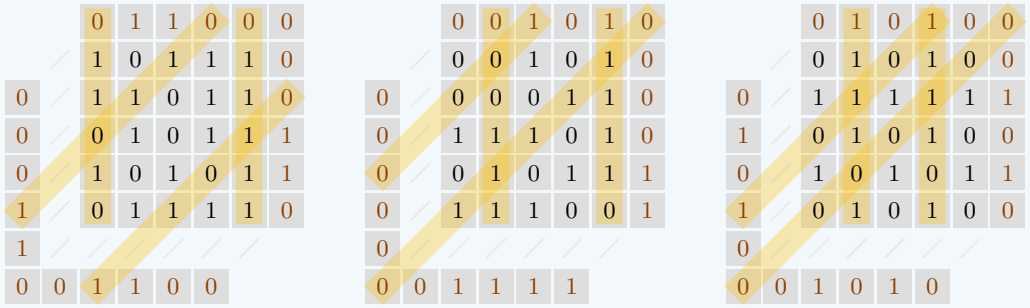


2 Fehler auf einer Spalte

Schauen wir uns zunächst den Fall von zwei Fehlern auf einer Zeile an. Wir entdecken mit dem originalen Kartentrick zwar die zwei fehlerhaften Spalten, können aber die dazugehörige Zeile nicht bestimmen, da es keine fehlerhafte Zeile gibt. Wir wissen also nicht, auf welcher der $a+1$ Zeilen das Fehlerpaar sitzt. Nun ist die Frage zu beantworten, ob uns die zusätzlichen Prüfbits der Vorwärtsdiagonalen dabei helfen, das richtige Fehlerpaar zu entdecken.

Aufgabe 2.5

In den folgenden drei Codewörtern haben wir jeweils zwei Fehler auf einer Zeile versteckt. Können Sie alle Fehler finden? Um Ihnen Rechenarbeit zu ersparen, sind die fehlerhaften Linien bereits markiert.



(a) (b) (c)

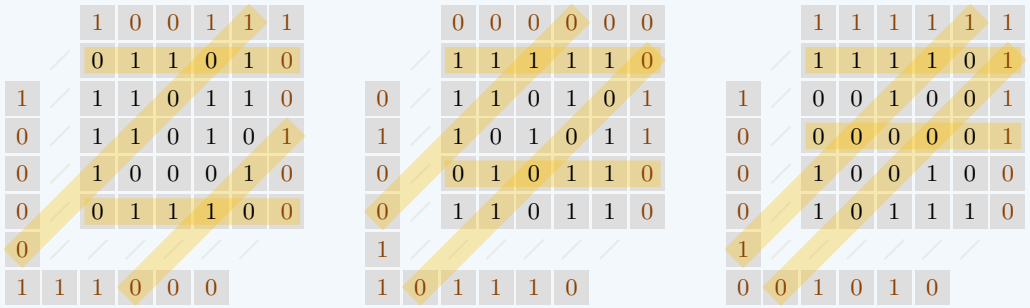
Lösung auf Seite 29

Die Vorwärtsdiagonalen helfen uns also tatsächlich dabei, zwei Fehler auf einer Zeile zu finden: Die fehlerhaften Spalten und fehlerhaften Vorwärtsdiagonalen bilden bis zu vier Kreuzungen, wovon genau zwei auf einer Zeile liegen. Dies sind die gesuchten Fehlerpositionen.

Wie sieht es mit zwei Fehlern auf einer gemeinsamen Spalte aus? Mit dem originalen Kartentrick erkennen wir zwar die fehlerhaften Zeilen, können aber nicht bestimmen, auf welcher der $a+1$ Spalten die zwei Fehler liegen. Helfen uns die zusätzlichen Prüfbits der Vorwärtsdiagonalen auch hierbei?

Aufgabe 2.6

In den folgenden drei Codewörtern gibt es jeweils zwei Fehler auf einer gemeinsamen Spalte. Können Sie alle Fehler finden?



(a) (b) (c)

Lösung auf Seite 29

Die Vorwärtsdiagonalen helfen uns also analog auch dabei, zwei Fehler auf einer gemeinsamen Spalte zu finden: Die fehlerhaften Zeilen und fehlerhaften Vorwärtsdiagonalen bilden bis zu vier Kreuzungen, wovon genau zwei auf einer Spalte liegen. Dies sind wieder die gesuchten Fehlerpositionen.

Zusammenfassung

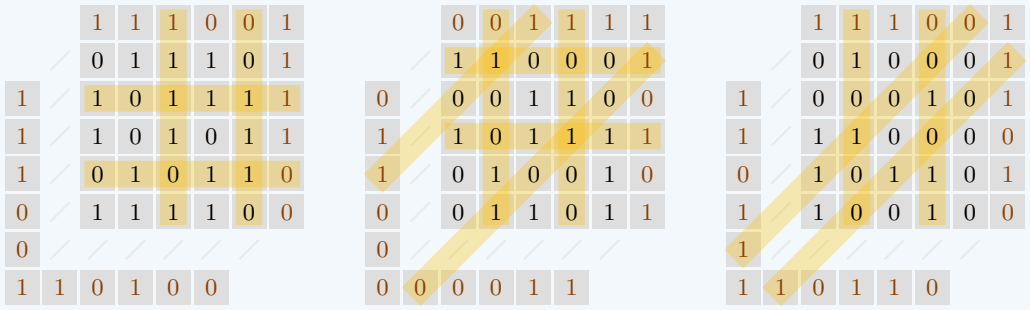
Bei zwei Fehlern auf einer Zeile (oder auf einer Spalte) bilden die fehlerhaften Vorwärtsdiagonalen und Spalten (oder Zeilen) bis zu vier Kreuzungen, wovon genau zwei auf einer Zeile (oder auf einer Spalte) liegen. Die Fehler können also mit den zusätzlichen Prüfbits lokalisiert werden.

2.4 Überblick

Wenn wir wissen, dass wir zwei Fehler auf unterschiedlichen Zeilen und Spalten (Abschnitt 2.1) oder zwei Fehler auf einer gemeinsamen Zeile oder auf einer gemeinsamen Spalte (Abschnitt 2.3) haben, können wir die Fehler korrigieren. Doch wie erkennen wir, in welcher Situation wir uns befinden? Wir haben tatsächlich bereits alle Informationen, um die Fälle unterscheiden zu können.

Aufgabe 2.7

Bei den folgenden drei Situationen gibt es jeweils zwei Fehler im Quadrat des Codeworts. Finden und korrigieren Sie die Fehler.



(a) (b) (c)

Lösung auf Seite 30

Bis jetzt haben sich die Fehler immer nur im Quadrat befunden. Doch was wäre, wenn ein Fehler bei den Prüfbits der Vorwärtsdiagonalen passiert? Und was, wenn wir nicht wissen, ob es genau 2 Fehler gibt?

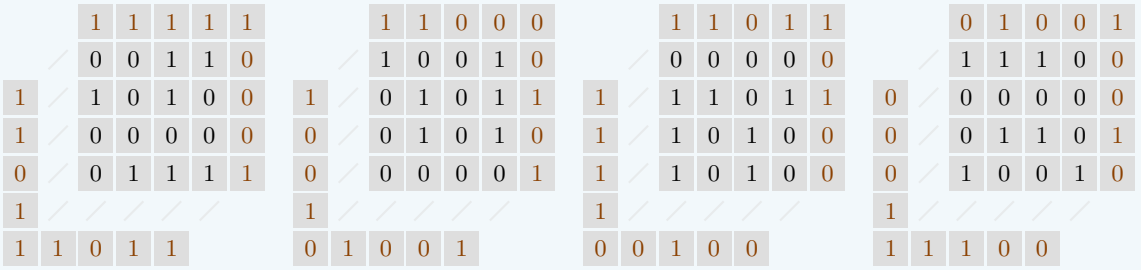
Aufgabe 2.8

Bei den folgenden vier Situationen gibt es jeweils entweder 0, 1 oder 2 Fehler.

(1) Bestimmen Sie für jede Situation die fehlerhaften Zeilen, Spalten und Vorwärtsdiagonalen.

(2) Finden Sie für jede Situation zuerst die Anzahl Fehler im Quadrat und dann die Anzahl Fehler in den Prüfbits der Vorwärtsdiagonalen. Falls Sie Hilfe benötigen, denken Sie an Aufgabe 1.2 zurück.

(3) Finden und korrigieren Sie die Fehler.



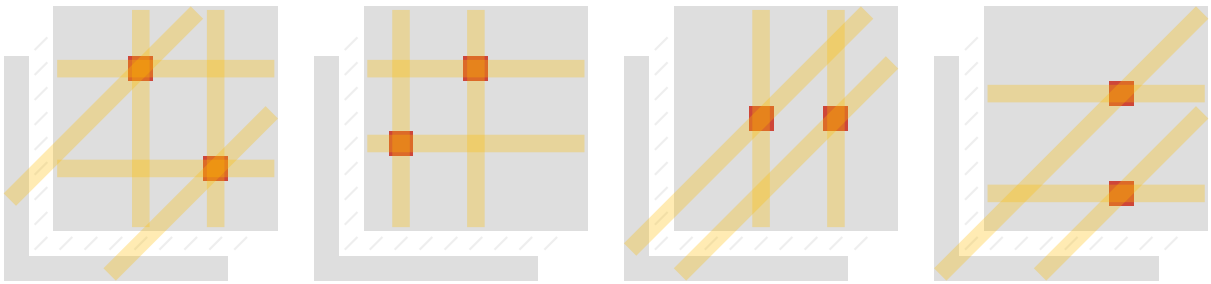
Lösung auf Seite 31

Aufgabe 2.9

Fridolin wird stutzig. Hatten wir einfach nur Glück mit den oben dargestellten Situationen? Oder funktioniert unsere Strategie tatsächlich immer? Helfen Sie Fridolin bei seiner Argumentation, weshalb das Vorgehen von Aufgabe 2.8 nicht nur in den spezifischen dargestellen Fällen, sondern auch im Allgemeinen bei höchstens zwei Fehlern funktioniert.

Lösung auf Seite 32

Wie wir in Abschnitt 1 gesehen haben, geben uns die Prüfbits des originalen Kartentricks genug Information, um zu entscheiden, ob es 0, 1 oder 2 Fehler im Quadrat gibt. Bei 0 Fehler im Quadrat kann die Nachricht trivialerweise wiederhergestellt werden; bei 1 Fehler hilft uns der originale Kartentrick (ohne die Information der Prüfbits der Vorwärtsdiagonalen) bei der Wiederherstellung. Bei 2 Fehlern im Quadrat kann es keine Fehler in den Prüfbits der Vorwärtsdiagonalen geben, also können wir die gelernten Methoden anwenden. Zusammengefasst gibt es bei zwei Fehlern im Quadrat die folgenden vier Fälle:



(i) Zwei Fehler auf unterschiedlichen Zeilen, Spalten und Vorwärtsdiagonalen: Es gibt je zwei fehlerhafte Zeilen, Spalten und Vorwärtsdiagonalen. An den Positionen der zwei Fehler kreuzen sich je drei Linien.

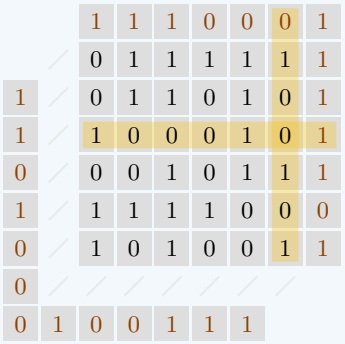
(ii) Zwei Fehler auf unterschiedlichen Zeilen und Spalten, aber auf der gleichen Vorwärtsdiagonalen: Es gibt je zwei fehlerhafte Zeilen und Spalten und keine fehlerhafte Vorwärtsdiagonale. Die zwei Kreuzungen auf einer gemeinsamen Vorwärtsdiagonalen müssen die Fehler enthalten.

(iii) Zwei Fehler auf einer Zeile: Es gibt je zwei fehlerhafte Spalten und Vorwärtsdiagonalen. Die Fehler befinden sich auf den zwei Kreuzungen, die auf einer Zeile liegen.

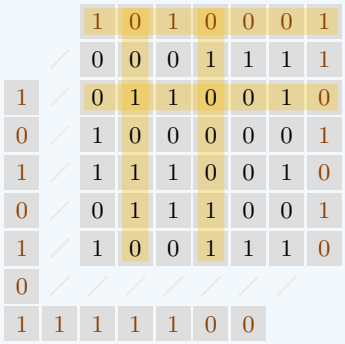
(iv) Zwei Fehler auf einer Spalte: Es gibt je zwei fehlerhafte Zeilen und Vorwärtsdiagonalen. Die Fehler befinden sich auf den zwei Kreuzungen, die auf einer Spalte liegen.

Aufgabe 2.10

Finden und korrigieren Sie die Fehler. Alle fehlerhaften Linien sind bereits markiert.



(a)



(b)

Lösung auf Seite 32

Unsere erweiterte Kartentrick-Kodierung ist also tatsächlich 2-fehlerkorrigierend und hat somit Abstand mindestens 5. Nun möchten wir noch verstehen, was der genaue Abstand ist.

Aufgabe 2.11

Finden Sie zwei Codewörter beliebiger Länge mit Abstand 6.

Lösung auf Seite 33

Aufgabe 2.12

Begründen Sie, warum es keine zwei Codewörter mit Abstand 5 geben kann.

Lösung auf Seite 34

Die erweiterte Kartentrick-Kodierung hat also Abstand 6. Sie ist somit 5-fehlererkennend und, wie bereits gesehen, 2-fehlerkorrigierend. Doch wie viele Bits brauchen wir für die Darstellung eines Codeworts?

Aufgabe 2.13

Betrachten Sie ein Quadrat der Seitenlänge x. Welche Längen können die Vorwärtsdiagonalen haben? Wie viele Vorwärtsdiagonalen jeder Länge gibt es? Wie viele Vorwärtsdiagonalen gibt es insgesamt?

Lösung auf Seite 34

Aufgabe 2.14

Wie viele Bits brauchen Sie für das Codewort einer Nachricht mit 20, 25 und 30 Bits? Wie viele Bits brauchen Sie für das Codewort einer Nachricht mit $n = x^2$ Bits?

Lösung auf Seite 35

Zusammenfassung

Wir haben gesehen, dass das Einführen von Prüfbits für weitere Linien (Vorwärtsdiagonalen zusätzlich zu Zeilen und Spalten) uns die nötige Information liefert, um zwei Fehler zu korrigieren. Die so entstandene erweiterte Kartentrick-Kodierung ist mit Abstand 6 zudem auch 5-fehlererkennend.

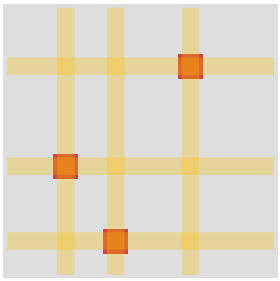
3 Zweifach erweiterter Kartentrick: 3-Fehlerkorrektur

Im vorherigen Abschnitt haben wir gesehen, dass das Berechnen von Prüfbits auf zusätzlichen Bereichen zu mehr Information führt. Diese Information kann für die Korrektur einer grösseren Anzahl Fehler verwendet werden. Wenn wir uns bei den Bereichen aber nur auf Vorwärtsdiagonalen beschränken, ist es noch nicht möglich, drei Fehler zuverlässig zu korrigieren (siehe Aufgabe 2.11). In diesem Abschnitt werden wir deshalb untersuchen, welche Bereiche für die Korrektur von drei Fehlern nützlich sind.

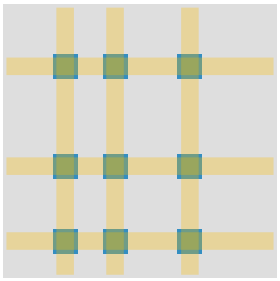
Es wird sich dabei herausstellen, dass das Finden und Korrigieren eines ersten Fehlers eine besonders wichtige Rolle spielt. Deshalb wollen wir uns vorläufig nur auf das Lokalisieren eines einzigen der drei Fehler konzentrieren.

3.1 Fall 3A: Drei Fehler auf unterschiedlichen Zeilen und Spalten

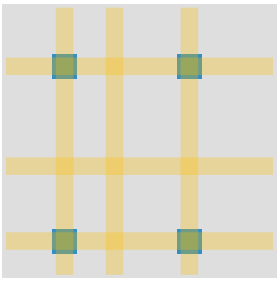
Erinnern wir uns an unser Vorgehen am Anfang von Abschnitt 2, bei dem es um die Korrektur zweier Fehler ging: Dort haben wir in Abschnitt 2.1 zuerst den Fall betrachtet, bei dem die beiden Fehler weder auf der gleichen Zeile noch auf der gleichen Spalte liegen. Was wäre die entsprechende Situation jetzt, wo wir drei Fehler haben? Das könnte einfach wieder bedeuten, dass keine zwei der drei Fehler auf der gleichen Zeile oder auf der gleichen Spalte liegen. Diese Art Fehler zu platzieren führt zu drei fehlerhaften Zeilen und drei fehlerhaften Spalten. Die Situation könnte zum Beispiel wie ganz links in der folgenden Abbildung aussehen.



3 Fehler



9 Kreuzungen



4 kritische Punkte

Angenommen wir wüssten nicht, wo die drei Fehler tatsächlich liegen, dann hätten wir zumindest die möglichen Positionen der Fehler auf die neun markierten Kreuzungen in der Mitte der Abbildung oben reduziert. Von diesen neun möglichen Positionen werden die vier Eckpunkte oben links, oben rechts, unten links sowie unten rechts eine besondere Rolle spielen. Diese vier speziellen Positionen, die wir ganz rechts in der Abbildung oben markiert haben, nennen wir deshalb *kritische Punkte*.

Aufgabe 3.1

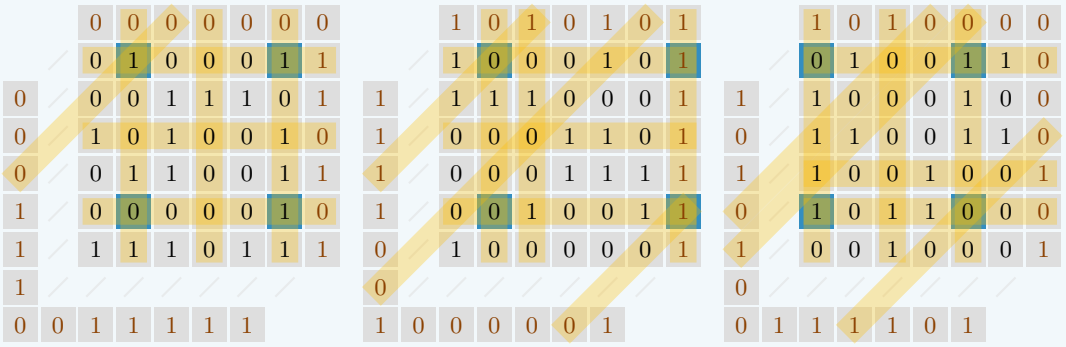
Wo können in der oben beschriebenen Situation die drei Fehler liegen? Anders gefragt, auf wie viele Arten kann man die drei Fehler auf die neun Kreuzungen verteilen? Fällt Ihnen zusätzlich etwas auf bezüglich der kritischen Punkte?

Lösung auf Seite 35

Es gibt also insgesamt sechs unterschiedliche Arten, die drei Fehler zu verteilen. Und dabei gibt es immer mindestens einen kritischen Punkt, der einen Fehler enthält. Weil unser erstes Ziel war, nur einen der drei Fehler zu finden, können wir uns also bei der Suche auf diese vier kritischen Punkte beschränken. Es wird sich zeigen, dass die Prüfbits der Vorwärtsdiagonalen bei dieser Suche in den meisten Fällen bereits ausreichen, aber leider nicht immer.

Aufgabe 3.2

Bestimmen Sie für die folgenden Fälle je mindestens einen der markierten kritischen Punkte, der einen Fehler enthält. Die fehlerhaften Zeilen, Spalten und Vorwärtsdiagonalen sind bereits eingezeichnet.

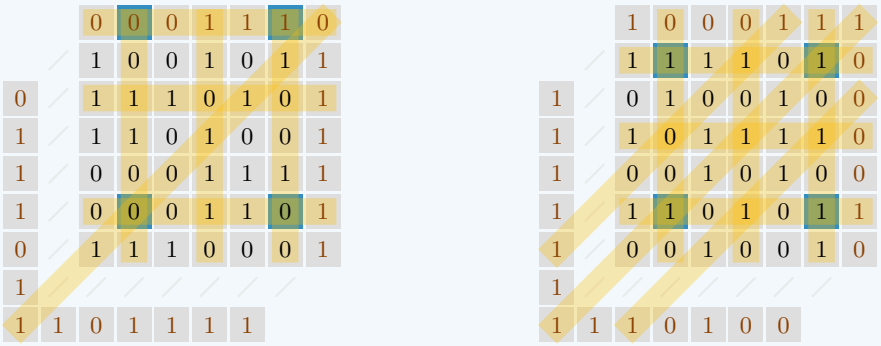


(a) (b) (c)

Lösung auf Seite 36

Aufgabe 3.3

Funktioniert Ihre Strategie aus der vorherigen Aufgabe auch hier? Begründen Sie Ihre Aussage.



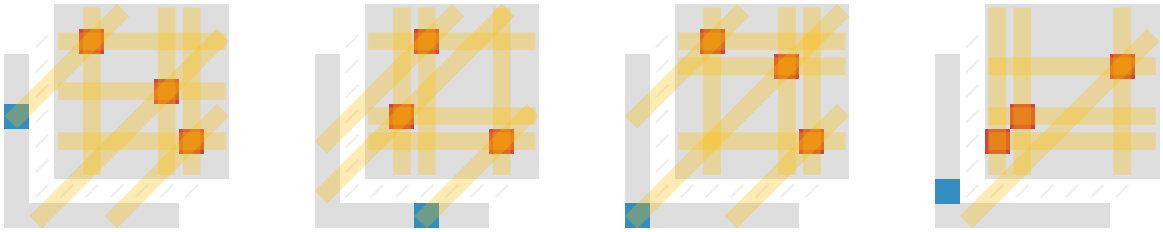
(a) (b)

Lösung auf Seite 36

Die Strategie, Fehler auf kritischen Punkten mittels fehlerhaften Vorwärtsdiagonalen zu finden, funktioniert nur dann zuverlässig, wenn die entsprechende Vorwärtsdiagonale keine weitere der neun Kreuzungen enthält. Es ist deshalb kein Zufall, dass die Strategie in Aufgabe 3.2 immer und in Aufgabe 3.3 nur manchmal erfolgreich ist. Der grundlegende Unterschied zwischen den zwei Aufgaben besteht in der Positionierung des Fehlers auf den kritischen Punkten. In Aufgabe 3.2 liegt er immer entweder oben links oder unten rechts; in Aufgabe 3.3 hingegen immer oben rechts oder unten links.

- Für die ersteren zwei kritischen Punkte (also oben links oder unten rechts) verläuft die entsprechende Vorwärtsdiagonale entweder ganz links oder ganz rechts von allen anderen Kreuzungen und somit durch kein weiteres fehlerhaftes Bit (siehe auch (i) und (ii) weiter unten). Ein Fehler bei so einem kritischen Punkt macht sich also zwingend durch die fehlerhafte Vorwärtsdiagonale bemerkbar.
- Für die letzteren zwei kritischen Punkte (also oben rechts oder unten links) kann die entsprechende Vorwärtsdiagonale im Prinzip durch bis zu zwei weitere Kreuzungen verlaufen. Es kann also sein, dass man von einer fehlerhaften Vorwärtsdiagonalen nicht eindeutig auf den Fehler auf einem spezifischen kritischen Punkt schliessen kann (siehe auch (iii) weiter unten). Es kann aber auch sein, dass die Vorwärtsdiagonale den Fehler auf einem kritischen Punkt zwar enthält, aber wegen eines weiteren Fehlers gar nicht anzeigt (siehe auch (iv) weiter unten).

In den folgenden Situationen sind jeweils die Prüfbits derjenigen Vorwärtsdiagonalen blau markiert, auf die wir uns fokussieren wollen. Bei (i) und (ii) erlaubt dieses Prüfbit die zuverlässige Identifizierung des Fehlers auf dem entsprechenden kritischen Punkt. Bei (iii) zeigt das Prüfbit einen Fehler an, obwohl sich gar keine Fehler auf den entsprechenden kritischen Punkten befinden. Bei (iv) zeigt das Prüfbit keinen Fehler an, obwohl der entsprechende kritische Punkt einen Fehler enthält.



(i) (ii) (iii) (iv)

Aufgabe 3.4

Haben Sie eine Idee, wie wir die Kodierung noch ein zweites Mal erweitern könnten, so dass sich die zwei oben beschriebenen Typen von kritischen Punkten nicht mehr grundlegend unterscheiden und wir somit in jedem Fall einen ersten Fehler auf einem kritischen Punkt finden können?

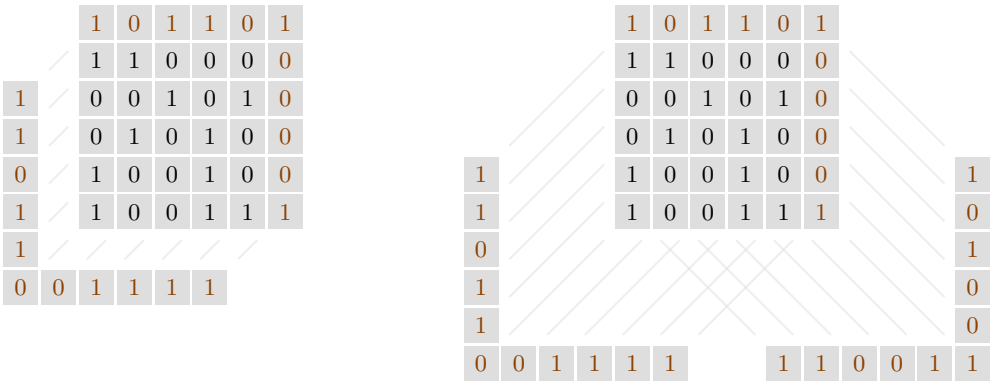
Lösung auf Seite 36

Zusammenfassung

Bei der Suche nach einem ersten Fehler in einem Codewort mit je drei fehlerhaften Zeilen und Spalten haben wir uns auf vier kritische Punkte beschränkt. Von diesen kritischen Punkten wissen wir, dass mindestens einer einen Fehler enthält. Fehler auf den kritischen Punkten oben links oder unten rechts sind zuverlässig mit den Prüfbits der Vorwärtsdiagonalen identifizierbar. Bei den kritischen Punkten oben rechts oder unten links funktioniert das aber nicht gleich gut. Deshalb haben wir uns überlegt, zusätzliche Prüfbits auch für Rückwärtsdiagonalen zu berechnen.

3.2 Zweite Erweiterung der Kodierung

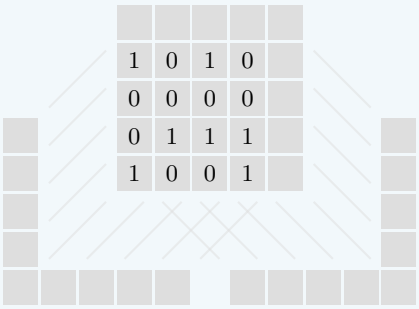
Basierend auf den Überlegungen aus Aufgabe 3.4 definieren wir die *zweifach erweiterte Kartentrick-Kodierung* wie folgt: Wir nehmen die bereits erweiterte Kartentrick-Kodierung einer gegebenen Nachricht und berechnen die Prüfbits für alle Rückwärtsdiagonalen auf analoge Weise.



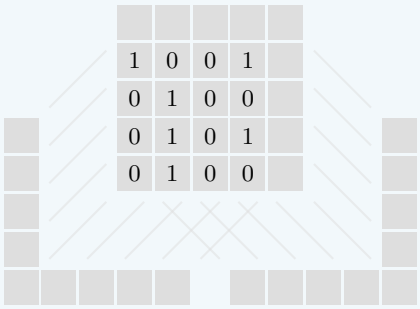
Einfache Erweiterung Zweifache Erweiterung

Aufgabe 3.5

Berechnen Sie für die folgenden Nachrichten das Codewort der zweifach erweiterten Kartentrick-Kodierung, indem Sie das entsprechende Diagramm vervollständigen.



(a)

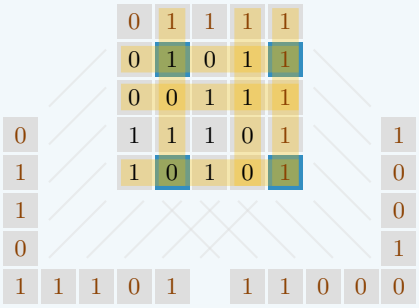


(b)

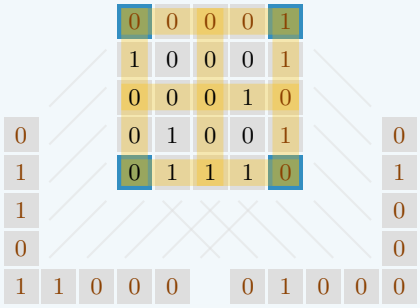
Lösung auf Seite 37

Aufgabe 3.6

Bestimmen Sie in den folgenden Situationen zuerst die fehlerhaften Vorwärts- und Rückwärtsdiagonalen. Die fehlerhaften Zeilen und Spalten sowie die kritischen Punkte sind bereits markiert. Finden Sie danach je einen Fehler auf einem der kritischen Punkte.



(a)



(b)

Lösung auf Seite 37

Für den Spezialfall, bei dem drei Fehler auf unterschiedlichen Zeilen und Spalten sind, scheint unsere Kodierung also tatsächlich mächtig genug zu sein, so dass wir einen ersten Fehler finden können. In der folgenden Aufgabe wollen wir nun endlich verstehen, wieso wir dann immer auch gleich die ganze Nachricht wiederherstellen können.

Aufgabe 3.7

Fridolin erhält von Frida ein Codewort der zweifach erweiterten Kartentrick-Kodierung mit drei Fehlern. Er ist sich sicher, dass er die Position eines der drei fehlerhaften Bits kennt. Wie muss er vorgehen, um die verbleibenden zwei Fehler auch finden und korrigieren zu können?

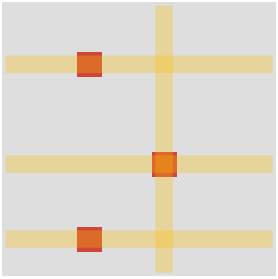
Lösung auf Seite 37

Zusammenfassung

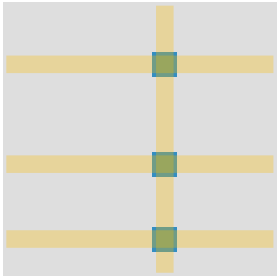
Die zweifach erweiterte Kartentrick-Kodierung besteht zusätzlich aus den Prüfbits aller Rückwärtsdiagonalen. Sobald ein einziger von drei Fehlern in einem gegebenen Codewort identifiziert und korrigiert ist, können wir die restlichen zwei Fehler mit unserem Wissen aus Abschnitt 2 auch korrigieren.

3.3 Fall 3B: Zwei von drei Fehlern auf gleicher Zeile oder gleicher Spalte

Wie sieht es denn eigentlich aus, wenn nicht alle drei Fehler auf unterschiedlichen Zeilen und Spalten liegen? Es könnten beispielsweise auch zwei der Fehler auf einer gemeinsamen Spalte sein, wie in der folgenden Abbildung links zu sehen ist.



3 Fehler



3 Kreuzungen

Die Situation zeichnet sich dadurch aus, dass nur noch eine Spalte fehlerhaft ist. Insbesondere gibt es nur drei Kreuzungen (in der Abbildung oben rechts markiert), in denen sich möglicherweise Fehler verstecken könnten. Die dargestellte Situation links zeigt deutlich, dass diese drei Kreuzungen alleine uns nicht direkt zu allen Fehlern führen werden. In der folgenden Aufgabe wird sich zeigen, dass diese drei speziellen Positionen trotzdem unser Augenmerk verdient haben.

Aufgabe 3.8

Auf welchen der drei markierten Kreuzungen können im Prinzip Fehler liegen? Können vielleicht sogar mehrere Kreuzungen gleichzeitig von Fehlern besetzt sein? Auf insgesamt wie viele Arten können die drei Fehler in der dargestellten Situation verteilt sein?

Lösung auf Seite 38

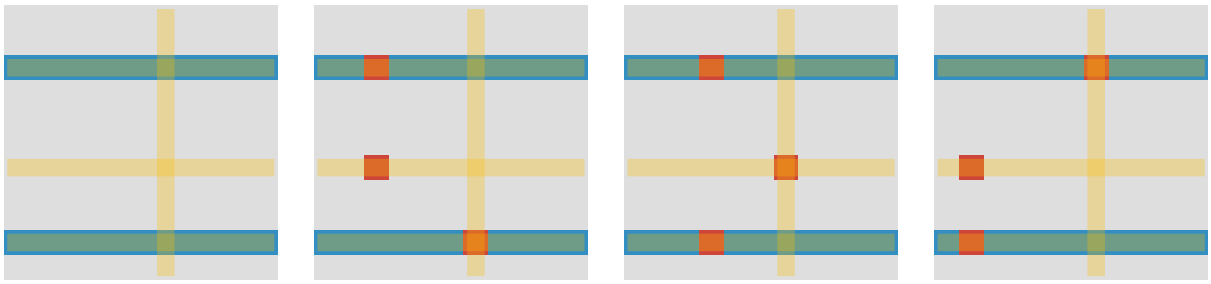
Es ist also insbesondere so, dass immer mindestens eine der drei markierten Kreuzungen einen Fehler enthält. Wie wir bereits in Aufgabe 3.7 gelernt haben, wären wir schon fertig mit der Wiederherstellung der Nachricht, falls wir auch nur diesen einzelnen Fehler identifizieren könnten.

Aufgabe 3.9

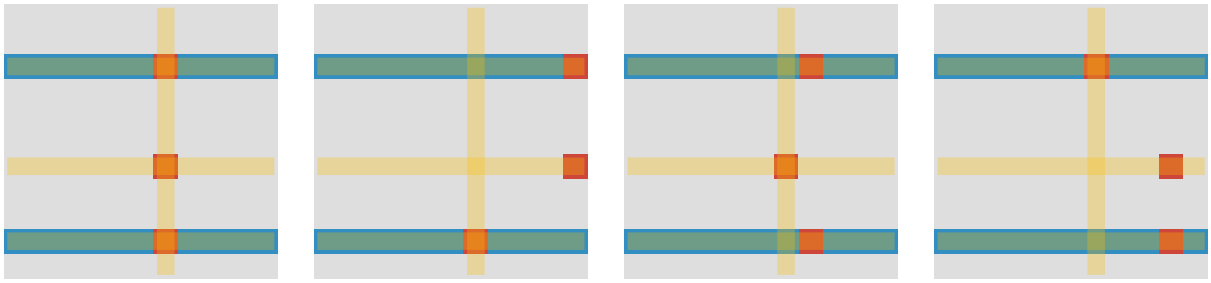
Fridolin hat genau aufgepasst. Er erinnert sich an die vier kritischen Punkte von Abschnitt 3.1. Er glaubt, dass wir hier halt einfach nur drei kritische Punkte haben, die wir aber auf gleiche Weise mittels Diagonalen-Prüfbits testen können. Stimmt das?

Lösung auf Seite 38

Die Idee, die drei Kreuzungen als kritische Punkte zu bezeichnen und uns bei der Suche nach dem ersten Fehler auf diese drei Bits zu beschränken, scheint leider nicht in jedem Fall erfolgversprechend zu sein. Trotzdem wollen wir uns auch hier wieder so eine geschickt gewählte Menge von Kandidaten überlegen: Wir definieren alle Positionen auf der oberen und der unteren fehlerhaften Zeile als *kritische Punkte*. In der folgenden Darstellung ganz oben links sind alle kritischen Punkte markiert.



Alle kritischen Punkte | Fehler oben links (ol) | Fehler links (l) | Fehler unten links (ul)



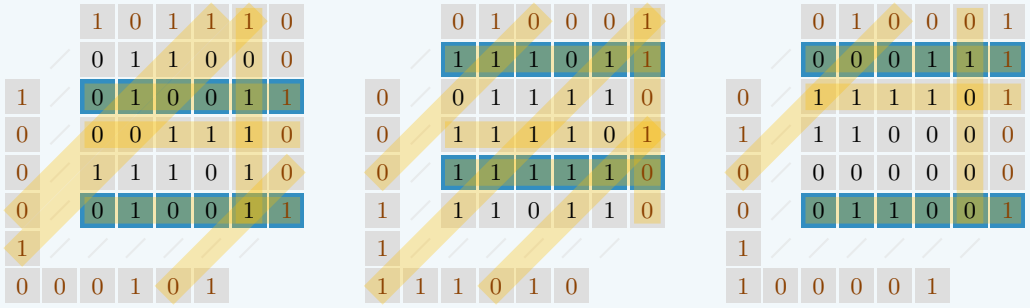
Fehler mittig (m) | Fehler oben rechts (or) | Fehler rechts (r) | Fehler unten rechts (ur)

In den anderen sieben Abbildungen oben haben wir alle grundlegend verschiedenen Fälle, die darstellen, wie die Fehler über die kritischen Punkte verteilt sein können, aufgelistet. Zum Beispiel verlangen wir beim Fall (m), dass alle drei Fehler auf der fehlerhaften Spalte liegen. Beim Fall (ol) soll der linke Teil der oberen fehlerhaften Zeile (also alle Positionen strikt links von der fehlerhaften Spalte) einen Fehler enthalten, aber gleichzeitig soll der linke Teil der unteren fehlerhaften Zeile keinen Fehler enthalten. Beim Fall (l) hingegen sollen sowohl der linke Teil der oberen fehlerhaften Zeile als auch der linke Teil der unteren fehlerhaften Zeile je einen Fehler enthalten. Die restlichen vier Fälle sind auf analoge Weise zu verstehen.

In der folgenden Serie von Aufgaben werden Sie selbständig eine Strategie basierend auf diesen sieben Fällen entwickeln. Stellen Sie bitte sicher, dass Sie die Lösung einer Aufgabe wirklich verstanden haben, bevor Sie zur nächsten weitergehen.

Aufgabe 3.10

Frida weiss bereits, dass es sich in den folgenden Situationen um Codewörter mit drei Fehlern handelt, welche entsprechend den Fällen (ol) oder (l) im Quadrat angeordnet sind. Sie weiss also insbesondere, dass auf dem linken Teil der oberen fehlerhaften Zeile ein Fehler versteckt ist. Sie hat bereits die kritischen Punkte markiert und alle fehlerhaften Linien eingezeichnet. Helfen Sie ihr, nur anhand der Prüfbits der Vorwärtsdiagonalen diesen einen Fehler auf der oberen fehlerhaften markierten Zeile zu finden.



(a) (b) (c)

Lösung auf Seite 39

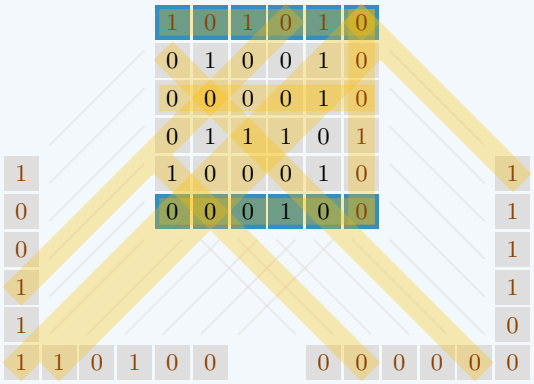
Aufgabe 3.11

Erklären Sie für jeden der sechs Fälle (ol), (l), (ul), (or), (r) und (ur) einzeln, wie Sie die Strategie aus der vorherigen Aufgabe anpassen müssen, um einen ersten Fehler auf den kritischen Punkten zu finden. Erwähnen Sie für jeden Fall, ob die Vorwärts- oder die Rückwärtsdiagonalen relevant sind.

Lösung auf Seite 39

Aufgabe 3.12

Bei folgendem Codewort mit drei Fehlern ist sich Frida nicht ganz so sicher wie vorher. Sie weiss aber zumindest, dass die drei Fehler entsprechend den Fällen (ol), (l) oder (ul) angeordnet sind. Das heisst, sie weiss, dass entweder der linke Teil der oberen fehlerhaften Zeile oder der linke Teil der unteren fehlerhaften Zeile (oder vielleicht sogar beide) einen Fehler enthält. Wie muss sie jetzt vorgehen, um diesen ersten Fehler auf den kritischen Punkten zu finden?



Lösung auf Seite 40

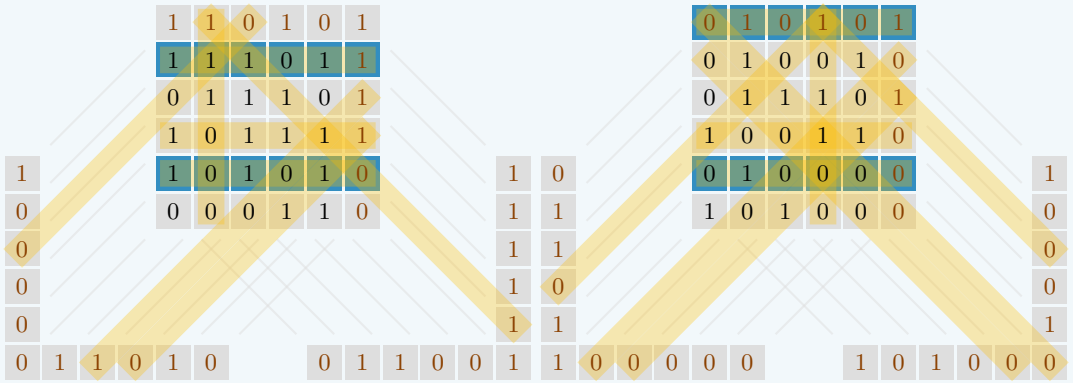
Aufgabe 3.13

Erklären Sie, wie Sie die Strategie aus der vorherigen Aufgabe anpassen müssen, wenn die drei Fehler entsprechend den Fällen (or), (r) oder (ur) angeordnet sind, aber Sie nicht genau wissen, welcher dieser drei Fälle zutrifft.

Lösung auf Seite 40

Aufgabe 3.14

Frida hat soeben zwei neue Codewörter mit drei Fehlern erhalten. Sie weiss noch nicht, um welche Fälle es sich dabei handelt. Helfen Sie ihr wieder beim Finden eines ersten Fehlers auf den kritischen Punkten.



(a) (b)

Lösung auf Seite 41

Wir haben es endlich geschafft. Wir verstehen jetzt, wie man vorgehen muss, wenn zwei der drei Fehler auf der gleichen Spalte liegen. Das einzige, was für diesen Fall noch übrigbleibt, ist die Frage, was sich genau ändert, wenn zwei der drei Fehler auf der gleichen Zeile (anstatt auf der gleichen Spalte) liegen.

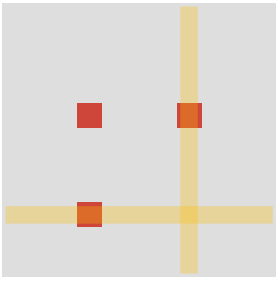
Da Sie jetzt schon geübt sind mit der Anpassung von bekannten Strategien auf neue Situationen, wollen wir Ihnen diese Frage aber ersparen. Man überlegt sich schnell, dass alles hier Gelernte auf analoge Weise angewendet werden kann. Falls Ihnen diese einfache Erklärung Schwierigkeiten bereitet, dann stellen Sie sich bitte vor, wie Sie das Codewort zuerst um 90 Grad drehen und dann die bekannten Strategien genauso wie zuvor darauf anwenden.

Zusammenfassung

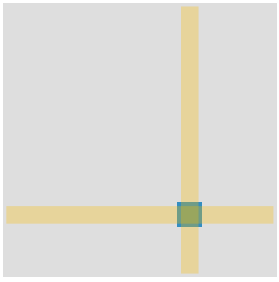
Wir haben gelernt, wie man bei zwei Fehlern auf einer gemeinsamen Zeile oder gemeinsamen Spalte einen ersten der insgesamt drei Fehler finden kann. Wie wir bereits wissen, reicht dies dann auch aus, um alle drei Fehler zuverlässig zu korrigieren. Im Gegensatz zu dem Fall mit drei Fehlern auf unterschiedlichen Zeilen und Spalten haben wir hier eine viel grössere Anzahl kritischer Punkte berücksichtigt.

3.4 Fall 3C: Drei Fehler auf insgesamt zwei Zeilen und zwei Spalten

Es bleibt nur noch eine sehr spezielle Situation übrig, die wir bis jetzt noch nicht betrachtet haben. Es können nämlich zwei der drei Fehler auf einer gemeinsamen Zeile sein und gleichzeitig zwei der drei Fehler auf einer gemeinsamen Spalte. Ein Beispiel finden Sie in der folgenden Abbildung links.



3 Fehler

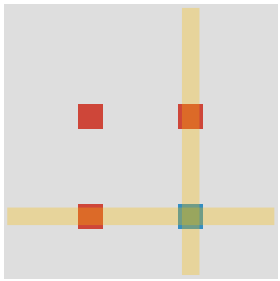


1 Kreuzung

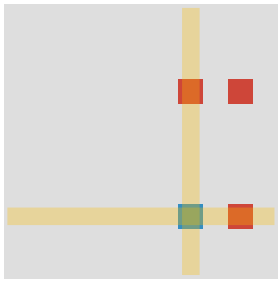
Dieser Fall zeichnet sich dadurch aus, dass jeweils nur eine Zeile und eine Spalte fehlerhaft ist. Basierend auf dieser Information ist also noch nicht einmal klar, dass überhaupt drei Fehler passiert sind (denken Sie zurück an Aufgabe 1.2). Doch zu dieser Frage kommen wir erst später. Zuerst möchten wir wie immer besser verstehen, wie die drei Fehler überhaupt angeordnet sein können.

Es muss sich je ein Fehler auf der fehlerhaften Zeile und ein Fehler auf der fehlerhaften Spalte befinden. Wo das genau ist, das ist egal, solange diese zwei Fehler nicht auf der markierten Kreuzung sind. Sobald diese zwei Fehler platziert sind, ist auch klar, wo der dritte Fehler sein muss: Er muss auf der gleichen Spalte wie der erste und auf der gleichen Zeile wie der zweite Fehler sein.

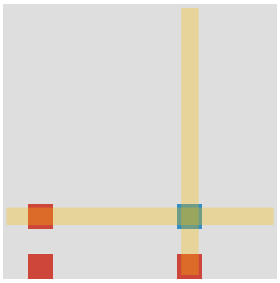
Allgemein ist weiter zu beobachten, dass die fehlerhafte Zeile und Spalte das Quadrat in vier unterschiedliche Quadranten zerteilt. Damit meinen wir die Regionen oben links, oben rechts, unten links sowie unten rechts. Bezüglich der Position des dritten Fehlers gibt es also die folgenden vier Fälle.



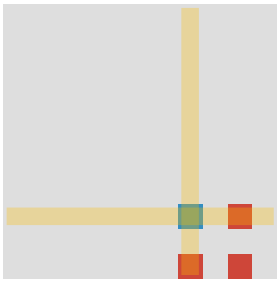
Fehler oben links (ol)



Fehler oben rechts (or)



Fehler unten links (ul)



Fehler unten rechts (ur)

Aufgabe 3.15

Auf wie viele unterschiedliche Arten können die drei Fehler in diesem Fall angeordnet sein?

Lösung auf Seite 41

Aufgabe 3.16

Nehmen Sie an, dass sich der dritte Fehler im oberen linken Quadranten befindet, also dass der Fall (ol) zutrifft. Was können Sie dann über die Prüfbits der Vorwärtsdiagonalen aussagen? Wie sieht es bei den Rückwärtsdiagonalen aus? Können Sie nur aus dieser Information herleiten, dass der dritte Fehler tatsächlich im oberen linken Quadranten sein muss?

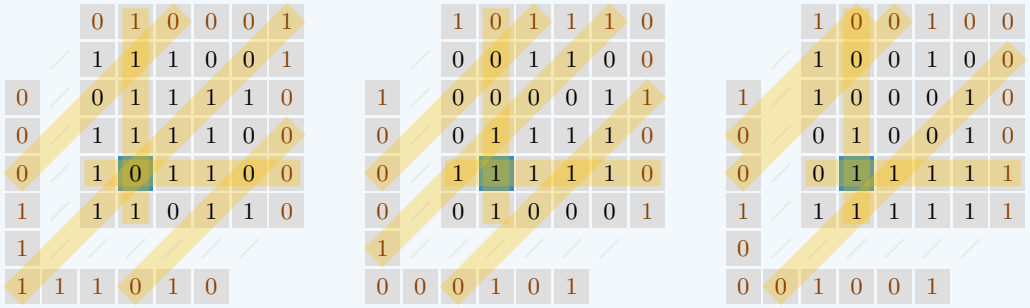
Lösung auf Seite 42

Wie soeben gelernt in Aufgabe 3.16 können wir mit Hilfe der Diagonalen-Prüfbits immer herausfinden, in welchem der vier Quadranten sich der dritte Fehler befindet. In anderen Worten können wir also die vier Fälle (ol), (or), (ul) und (ur) ganz einfach unterscheiden. Mit dieser Information ist es jetzt nicht mehr schwierig, einen ersten Fehler wirklich ausfindig zu machen.

Falls sich der dritte Fehler im oberen rechten oder unteren linken Quadranten befindet, also die Fälle (or) oder (ul), dann werden sich die Prüfbits der Vorwärtsdiagonalen als ausschlaggebend herausstellen. Falls sich hingegen der dritte Fehler im oberen linken oder unteren rechten Quadranten befindet, also die Fälle (ol) oder (ur), dann werden sich die Prüfbits der Rückwärtsdiagonalen als ausschlaggebend herausstellen. Falls also bei der Identifizierung des richtigen Quadranten zum Beispiel die Vorwärtsdiagonalen-Prüfbits relevant waren, dann werden wir bei der darauf folgenden Fehlersuche nur noch auf die Rückwärtsdiagonalen-Prüfbits zurückgreifen, und umgekehrt. Überzeugen Sie sich selbst in der folgenden Aufgabe.

Aufgabe 3.17

Bei den folgenden Codewörtern haben sich jeweils drei Fehler so eingeschlichen, dass der Fall (or) zutrifft. Verwenden Sie die Prüfbits der Vorwärtsdiagonalen, um je einen der Fehler zu finden. Die Rückwärtsdiagonalen haben wir weggelassen, weil sie für diese Aufgabe nicht mehr benötigt werden.



Lösung auf Seite 42

Zusammenfassung

Für den Fall, bei dem gleichzeitig zwei Fehler auf einer Zeile und zwei Fehler auf einer Spalte liegen, haben wir zuerst gelernt, wie man mit Hilfe der Diagonalen-Prüfbits eines bestimmten Typs (also entweder vorwärts oder rückwärts) den Quadranten mit dem dritten Fehler bestimmt. Dann haben wir weiter gesehen, wie man mit dem jeweilig anderen Typ von Diagonalen-Prüfbits einen ersten Fehler tatsächlich finden kann. Wie üblich reicht die Identifizierung dieses ersten Fehlers bereits aus, um das gesamte Codewort zu korrigieren.

3.5 Überblick

Wir sind in diesem Abschnitt bis jetzt immer davon ausgegangen, dass wir keine Fehler auf den zusätzlichen Diagonalen-Prüfbits haben. Anders ausgedrückt, bis jetzt befanden sich immer drei Fehler im Quadrat des originalen Kartentricks, also entweder in einem Nachrichtenbit oder in einem Prüfbit einer Zeile oder Spalte. Diese Annahme ist natürlich nicht ohne Weiteres berechtigt, denn bei der Datenübertragung können es auch genau diese zusätzlichen Diagonalen-Prüfbits sein, die vielleicht geflippt werden. Es wird sich aber herausstellen, dass die Erkennung eines einzigen Fehlers auf den Vorwärts- und Rückwärtsdiagonalen-Prüfbits bereits ausreicht, um dieses Problem zu beheben.

Aufgabe 3.18

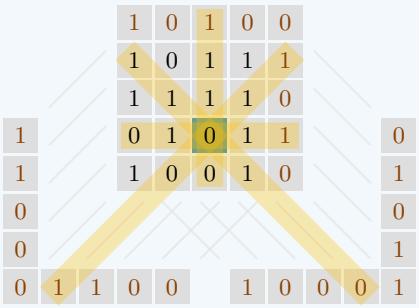
Nehmen Sie an, Sie haben ein möglicherweise fehlerhaftes Codewort der zweifach erweiterten Kartentrick-Kodierung erhalten. Nehmen Sie ferner an, dass es entweder *kein* oder *nur ein* fehlerhaftes Diagonalen-Prüfbit hat. Können Sie dann nur anhand der Diagonalen-Prüfbits entscheiden, ob tatsächlich ein fehlerhaftes Diagonalen-Prüfbit existiert? Falls ja, können Sie auch entscheiden, ob es sich dabei um ein Vorwärts- oder ein Rückwärtsdiagonalen-Prüfbit handelt? Für die Lösung dieser Aufgabe kann es hilfreich sein, sich zuerst zu überlegen, was die Summe aller Vorwärtsdiagonalen-Prüfbits darstellt.

Lösung auf Seite 43

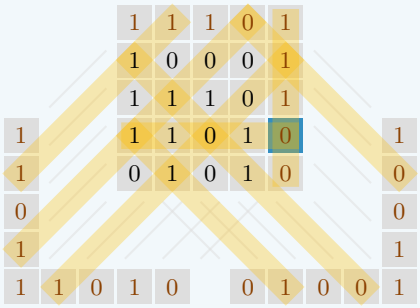
In Aufgabe 1.2 haben wir gelernt, dass man in fast allen Fällen dem Quadrat des originalen Kartentricks ansehen kann, wie viele Fehler es enthält. Der einzige Fall, den wir damals nicht unterscheiden konnten, war derjenige, in dem es je eine fehlerhafte Zeile und Spalte gibt und wir somit nicht wissen können, ob es nur einen oder drei sehr speziell angeordnete Fehler im Quadrat gibt. Mit der zweifach erweiterten Kartentrick-Kodierung ist es nun aber möglich, diese Fälle zu unterscheiden. Probieren Sie es selbst aus!

Aufgabe 3.19

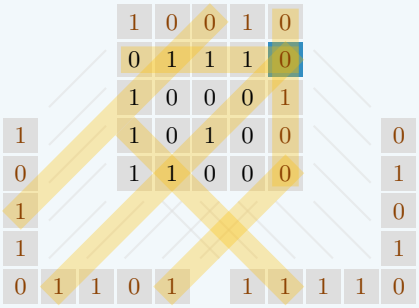
Bestimmen Sie in den folgenden vier Fällen, ob das Quadrat einen oder drei Fehler enthält. Beachten Sie, dass die Diagonalen-Prüfbits jetzt auch Fehler enthalten können. Um Ihnen den Überblick ein wenig zu erleichtern, haben wir die Kreuzung der fehlerhaften Zeile und Spalte bereits markiert.



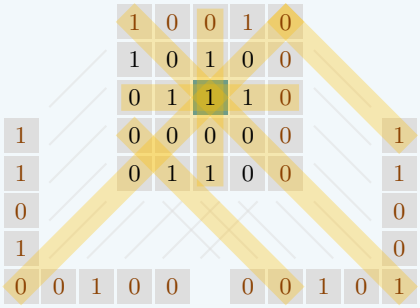
(a)



(b)



(c)



(d)

Lösung auf Seite 44

Für jedes fehlerhafte Codewort des zweifach erweiterten Kartentricks können wir somit nun erkennen, ob das Quadrat des ursprünglichen Kartentricks 0, 1, 2 oder 3 Fehler enthält. Doch wie müssen wir in den jeweiligen Fällen weiter vorgehen? Dies ist die letzte Frage; sobald Sie sie beantwortet haben, besitzen Sie alle Zutaten, um mit dem zweifach erweiterten Kartentrick bis zu drei beliebige Fehler zuverlässig zu korrigieren.

Aufgabe 3.20

Frida hat von Fridolin ein (möglicherweise fehlerhaftes) Codewort der zweifach erweiterten Kartentrick-Kodierung erhalten. Sie kennt bereits die Anzahl f der Fehler im Quadrat. Erklären Sie ihr, wie Sie weiter vorgehen muss für die Fälle $f = 0$, $f = 1$, $f = 2$ und $f = 3$.

Lösung auf Seite 45

Jetzt, wo wir die 3-Fehlerkorrektur mit dem zweifach erweiterten Kartentrick verstanden haben, möchten wir noch wissen, wie viele Bits diese Kodierung denn benötigt.

Aufgabe 3.21

Fridolin möchte Frida Nachrichten der Länge 20, 25 und 30 mit der zweifach erweiterten Kartentrick-Kodierung übermitteln. Wie viele Bits braucht er dazu jeweils? Wie viele Bits braucht er für das Codewort einer Nachricht mit $n = x^2$ Bits?

Lösung auf Seite 45

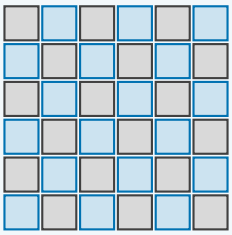
Zusammenfassung

Wir wissen jetzt, wie man mit der zweifach erweiterten Kartentrick-Kodierung in jedem beliebigen Fall drei Fehler zuverlässig erkennen und korrigieren kann. Dabei spielt es keine Rolle, ob sich diese drei Fehler im Quadrat der ursprünglichen Kartentrick-Kodierung oder auf den Prüfbits der zusätzlichen Diagonalen befinden. Ausschlaggebend ist, dass wir in einem ersten Schritt die genaue Anzahl Fehler im Quadrat bestimmen können, was uns dann erlaubt, direkt die Methoden des jeweilig zutreffenden Falles aus Abschnitt 2 und Abschnitt 3 anzuwenden.

4 Kontrollaufgaben

Aufgabe 4.1

Fridolin findet das Aufschreiben von so vielen Prüfbits anstrengend und hat eine Idee, wie man das Ziel einer 2-fehlerkorrigierenden Kodierung mit weniger Prüfbits erreichen kann. Er benutzt den originalen Kartentrick und möchte zusätzlich für den blauen und grauen Bereich in der folgenden Abbildung je ein Prüfbit berechnen. Funktioniert das? Begründen Sie Ihre Aussage.



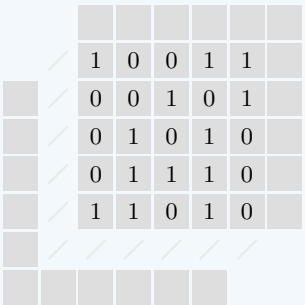
Zwei Bereiche für Prüfbits

Lösung auf Seite 45

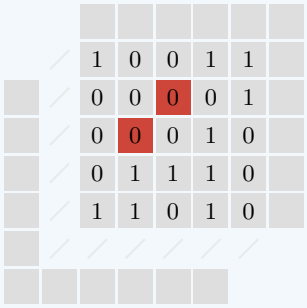
Aufgabe 4.2

Wir möchten gerne die Nachricht 10011001010101001110 mit dem einfach erweiterten Kartentrick kodieren.

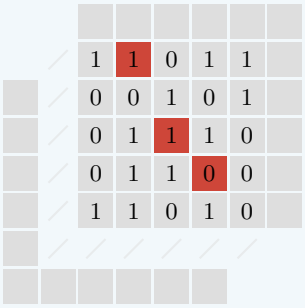
(1) Wir haben die Nachricht bereits in das untenstehende Diagramm eingetragen. Berechnen Sie die Prüfbits und vervollständigen Sie das Diagramm.



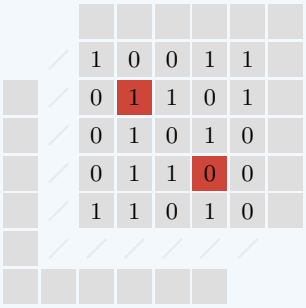
(2) Zeichnen Sie für die folgenden drei Fälle die fehlerhaften Linien ein und entscheiden Sie, ob die rot markierten Fehler lokalisiert und korrigiert werden können. Erklären Sie Ihr Vorgehen.



(a)



(b)



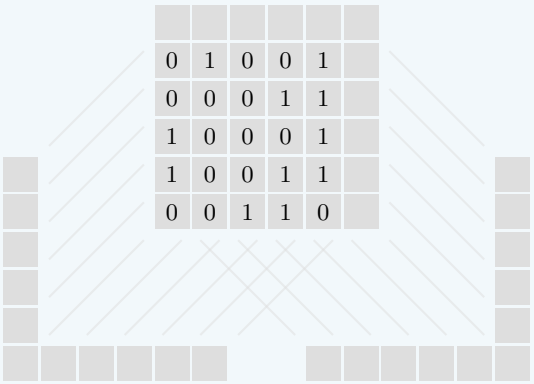
(c)

Lösung auf Seite 46

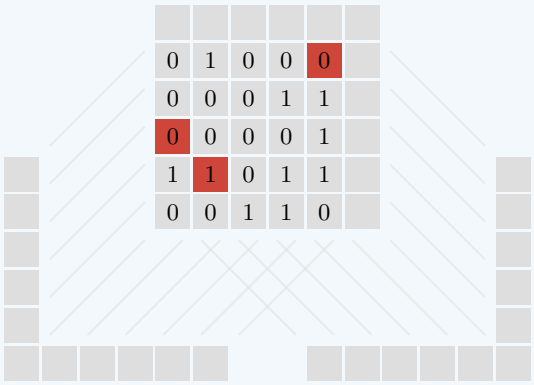
Aufgabe 4.3

Wir möchten gerne die Nachricht 0100100011100011001100110 mit dem zweifach erweiterten Kartentrick kodieren.

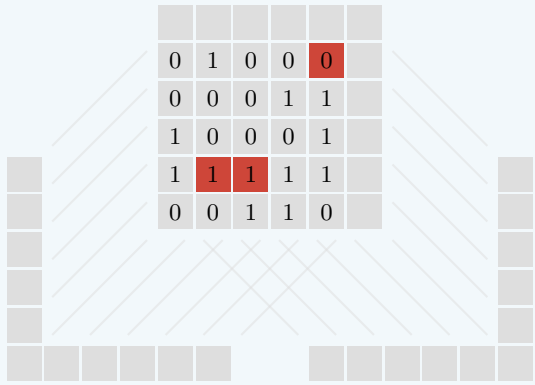
(1) Wir haben die Nachricht bereits in das untenstehende Diagramm eingetragen. Berechnen Sie die Prüfbits und vervollständigen Sie das Diagramm.



(2) Zeichnen Sie für die folgenden zwei Fälle mit drei Fehlern die fehlerhaften Linien ein und erklären Sie, wie Sie einen ersten Fehler finden können.



(a)



(b)

Lösung auf Seite 47

5 Lösungen

☑ Lösung zu Aufgabe 1.1 auf Seite 3

Konkret: Wir argumentieren zuerst separat für die drei konkreten Aufgabenstellungen.

(a) Es gibt insgesamt zwei Spalten, die sich nicht zu 0 aufsummieren. Die einzige Möglichkeit, eine solche fehlerhafte Spalte zu reparieren, ist, ein zusätzliches Bit in ebendieser Spalte zu flippen. Da es, wie schon erwähnt, zwei solcher Spalten gibt, müssen wir also mindestens zwei zusätzliche Bits flippen, um wieder ein gültiges Codewort zu erhalten. Ferner sei angemerkt, dass zwei zusätzliche Flips tatsächlich auch ausreichen, solange sie auf einer gemeinsamen Zeile passieren.

(b) Es gibt insgesamt zwei Zeilen, die sich nicht zu 0 aufsummieren. Die einzige Möglichkeit, eine solche fehlerhafte Zeile zu reparieren, ist, ein zusätzliches Bit in ebendieser Zeile zu flippen. Da es, wie schon erwähnt, zwei solcher Zeilen gibt, müssen wir also mindestens zwei zusätzliche Bits flippen, um wieder ein gültiges Codewort zu erhalten. Ferner sei angemerkt, dass zwei zusätzliche Flips tatsächlich auch ausreichen, solange sie auf einer gemeinsamen Spalte passieren.

(c) Es gibt insgesamt zwei Spalten sowie zwei Zeilen, die sich nicht zu 0 aufsummieren. Die gleiche Argumentation wie entweder bei Fall (a) oder (b) lässt uns also darauf schliessen, dass auch in diesem Fall mindestens zwei zusätzliche Bits geflippt werden müssen. Interessanterweise reichen aber auch hier zwei zusätzliche Flips aus, nämlich genau dann, wenn sie auf denjenigen zwei Kreuzungen passieren, auf denen sich noch kein Fehler befindet.

In allen betrachteten Fällen benötigen wir also mindestens zwei zusätzliche geflippte Bits, um wieder ein gültiges Codewort zu erhalten. Insgesamt müssen wir also immer mindestens vier Bits flippen. Dies ist ein Indiz dafür, dass die Kartentrick-Kodierung den Abstand 4 hat, aber noch kein Beweis.

Allgemein: Für den allgemeinen Beweis stellen wir uns jetzt ein beliebiges Codewort w vor. Wir wollen uns vergewissern, dass mindestens vier Bits in w geflippt werden müssen, um zu einem anderen gültigen Codewort zu gelangen. Da wir bereits wissen, dass die Kartentrick-Kodierung Abstand mindestens 3 hat, wissen wir auch, dass mit zwei oder weniger Flips noch kein anderes Codewort erreicht werden kann. Deshalb wählen wir zuerst einmal völlig willkürlich zwei Bits in w aus, die geflippt werden. Dadurch erhalten wir eine neue Nachricht w', die sich an genau zwei Stellen von w unterscheidet, aber selber noch kein gültiges Codewort sein kann.
Wir nehmen nun eine Fallunterscheidung vor über die mögliche Anordnung der zwei bereits geflippten Bits. Diese zwei Bits befinden sich entweder

(a) auf einer gemeinsamen Zeile oder

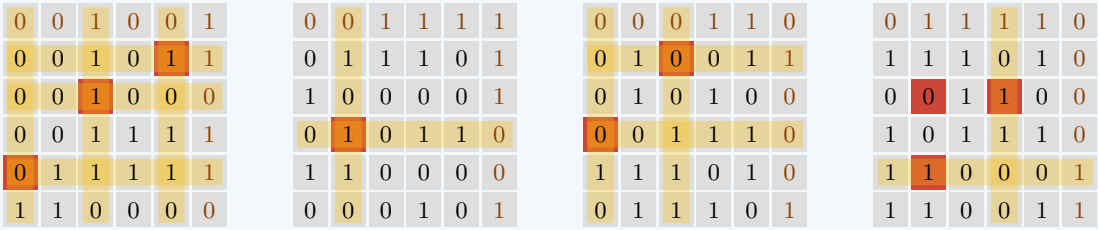
(b) auf einer gemeinsamen Spalte oder

(c) weder auf einer gemeinsamen Zeile noch auf einer gemeinsamen Spalte.

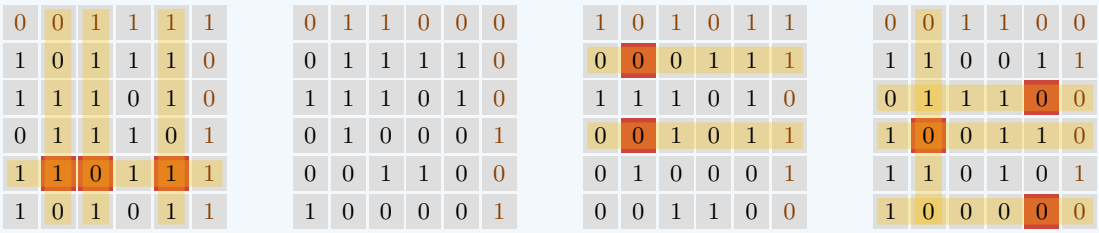
Diese drei allgemeinen Fälle entsprechen exakt den drei vorangegangenen konkreten Aufgabenstellungen. Insbesondere lässt sich die vorangegangene Argumentation direkt auf die allgemeinen Fälle hier übertragen. Zum Beispiel sieht man bei Fall (a), dass zwei geflippte Bits auf einer gemeinsamen Zeile in w immer zu zwei fehlerhaften Spalten in w' führen. Dies lässt uns nun allgemein darauf schliessen, dass mindestens zwei zusätzliche Bits in w' (also vier Bits insgesamt in w) geflippt werden müssen, um wieder ein gültiges Codewort zu erhalten. Bei diesem Argument spielt es auch gar keine Rolle, ob es sich bei den zwei bereits geflippten Bits um Nachrichten- oder Prüfbits handelt. Die Fälle (b) und (c) lassen sich auf die gleiche einfache Weise behandeln.

☑ Lösung zu Aufgabe 1.2 auf Seite 3

Zuerst einmal möchten wir die acht Codewörter mit aufgedeckten Fehlern zeigen. Beachten Sie bitte, dass es sich jeweils nur um eine mögliche Anordnung handelt und dass man in allen Fällen ausser bei (f) die Fehler auch anders verteilen könnte.



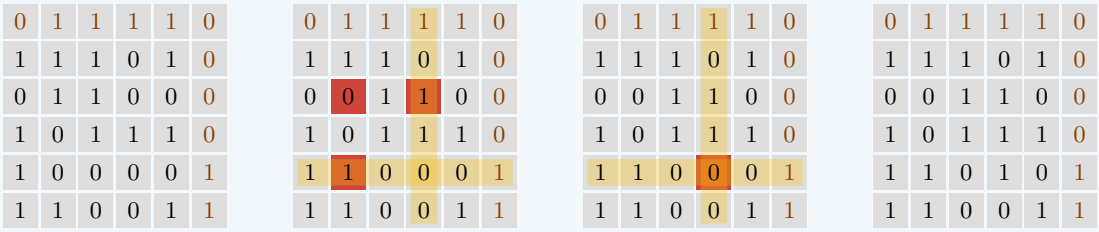
(a) (b) (c) (d)



(e) (f) (g) (h)

Auch wenn die genaue Anordnung der Fehler nicht eindeutig ist, ist es trotzdem in den meisten Fällen möglich, die Anzahl der Fehler zu bestimmen. Wir zeigen nun, wie man die acht gegebenen Codewörter systematisch den folgenden vier Kategorien zuordnen kann.

- **Drei Fehler.** Bei (a), (e) und (h) haben wir entweder drei unterschiedliche Zeilen oder drei unterschiedliche Spalten (oder beides), die sich nicht zu 0 aufsummieren. Das kann natürlich nur sein, wenn wir auch drei Fehler haben.
- **Zwei Fehler.** Bei (c) und (g) haben wir entweder zwei unterschiedliche Zeilen oder zwei unterschiedliche Spalten (oder beides), die sich nicht zu 0 aufsummieren. Das kann natürlich nur mit *mindestens zwei* Fehlern passieren. Man kann aber weiter beobachten, dass das nur mit *genau zwei* Fehlern möglich ist: Bei drei Fehlern wäre man nämlich gezwungen, den dritten Fehler wieder auf eine der bereits fehlerhaften Zeilen (oder Spalten) zu legen, welche sich dann wieder korrekt zu 0 aufsummieren würde.
- **Null Fehler.** Bei (f) summieren sich alle Zeilen und Spalten zu 0 auf, es handelt sich also um ein gültiges Codewort. Da der Abstand der Kodierung 4 ist, können wir 1, 2 oder 3 Fehler ausschliessen und wissen somit, dass wir in diesem Fall 0 Fehler haben müssen.
- **Unklar.** Es bleiben (b) und (d) oder im Allgemeinen Fälle, bei denen sich genau eine Zeile und genau eine Spalte nicht zu 0 aufsummieren. Interessanterweise können wir hier nicht darauf schliessen, wie viele Fehler es gibt. Sowohl ein Fehler als auch drei Fehler können auf unterschiedlichen Codewörtern so angeordnet werden, dass genau dieses Muster von fehlerhaften Zeilen und Spalten entsteht. In der folgenden Illustration sehen wir, wie diese Uneindeutigkeit zum Beispiel für den Fall (d) zustande kommen könnte.



Codewort w | w mit 3 Fehlern | w' mit 1 Fehler | Codewort w'

☑ Lösung zu Aufgabe 2.1 auf Seite 5

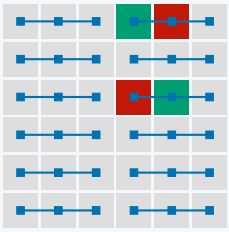
Frida kann die Fehler korrigieren, wenn jede der vier Kreuzungen in einem anderen 2×2-Block liegt.

- In den Fällen (a) und (c) liegen alle vier Kreuzungen in unterschiedlichen 2×2-Blöcken. Es gibt genau zwei fehlerhafte 2×2-Blöcke. Frida kann also entscheiden, welche zwei Kreuzungen einen Fehler enthalten.
- Im Fall (b) gibt es einen fehlerhaften 2×2-Block, der alle vier Kreuzungen enthält. Frida hat nicht genügend Information, um zu entscheiden, welche der vier Bits fehlerhaft sind.
- Im Fall (d) gibt es zwei fehlerhafte 2×2-Blöcke, die jeweils zwei Kreuzungen enthalten. Frida kann nicht entscheiden, welches Kreuzungspaar für die Fehler verantwortlich ist.

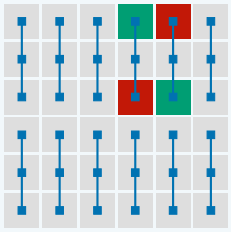
☑ Lösung zu Aufgabe 2.2 auf Seite 5

Von den Zeilen- und Spalten-Prüfbits wissen wir bereits, dass nur die zwei Paare von gegenüberliegenden Kreuzungen als mögliche fehlerhafte Bits in Frage kommen. In den folgenden Darstellungen markieren wir die tatsächlichen Fehler rot und das andere mögliche Fehlerpaar grün. Wenn alle vier Kreuzungen in unterschiedlichen Bereichen liegen, können wir die Fehler lokalisieren und korrigieren, weil dann nur genau die zwei Bereiche mit fehlerhaftem Bit einen Fehler anzeigen würden.

- Es liegen je ein Fehler und eine Kreuzung ohne Fehler auf der gleichen Zeilenhälfte. Diese zwei Zeilenhälften zeigen somit einen Fehler an, aber wir können nicht entscheiden, welches der zwei Paare dafür verantwortlich ist. Das Gleiche gilt für die Spaltenhälften.

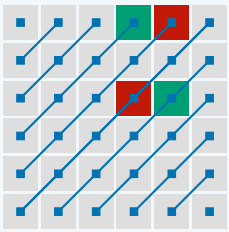


Zeilenhälften

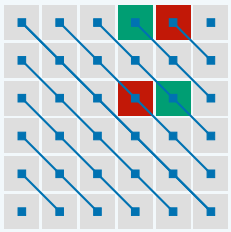


Spaltenhälften

- Es gibt vier Vorwärtsdiagonalen, die je durch genau eine der Kreuzungen gehen. Wir können also entscheiden, welche zwei der vier Kreuzungen einen Fehler enthalten. Analog gilt dies auch für die Rückwärtsdiagonalen.



Vorwärtsdiagonalen



Rückwärtsdiagonalen

☑ Lösung zu Aufgabe 2.3 auf Seite 6

Wir könnten zum Beispiel die in der folgenden Darstellung rot markierten Bits flippen. Da die zwei geflippten Bits sich auf der gleichen Vorwärtsdiagonalen befinden, gleichen sich die Fehler wieder aus.

1	1	0	1	0	1
0	1	1	0	1	0
0	0	0	1	0	1
1	0	1	1	0	1
1	1	0	1	0	0
1	0	0	0	0	1

Aufgrund der Zeilen- und Spalten-Prüfbits wissen wir wie üblich, welche vier Kreuzungen für die zwei Fehler in Frage kommen. Die Tatsache, dass kein Vorwärtsdiagonalen-Prüfbit einen Fehler anzeigt, lässt uns darauf schliessen, dass weder die Kreuzung oben links noch die Kreuzung unten rechts einen Fehler enthalten kann. Also müssen die Fehler tatsächlich bei der Kreuzung oben rechts und der Kreuzung unten links sein.

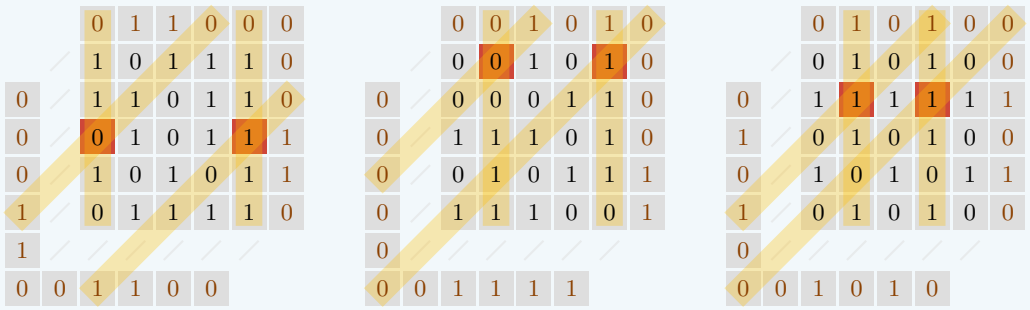
☑ Lösung zu Aufgabe 2.4 auf Seite 7

Hier ist das vervollständigte Diagramm.

		0	1	1	0	0	0	1	1
		1	1	0	0	0	0	0	0
0		0	1	1	1	0	1	1	1
0		0	0	0	0	0	1	1	0
0		1	1	0	1	1	1	1	0
1		0	0	0	1	1	0	1	1
0		0	1	1	1	1	0	0	0
0		0	1	1	0	1	1	1	1
1									
0	0	0	0	0	0	1	1		

Lösung zu Aufgabe 2.5 auf Seite 8

Die Fehler haben wir wie folgt versteckt.

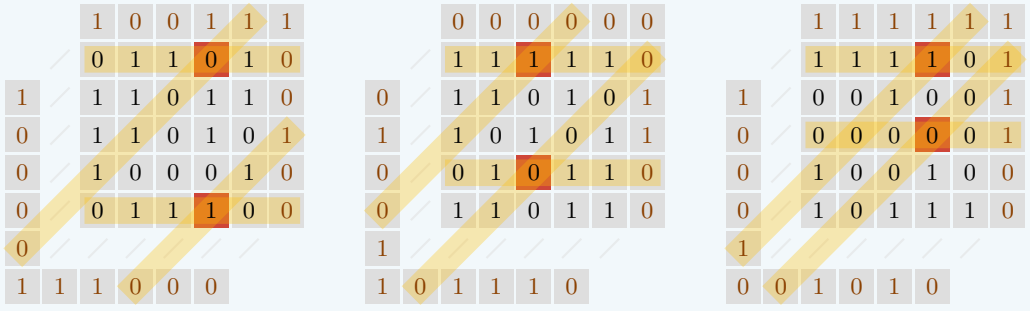


(a) (b) (c)

Doch wie können wir sie finden? Wir beobachten, dass die zwei Fehler immer auf zwei unterschiedlichen Spalten und zwei unterschiedlichen Vorwärtsdiagonalen liegen. Diese vier Linien zeigen also je einen Fehler an. Wir wissen, dass die fehlerhaften Bits auf den Kreuzungen liegen müssen; wir wissen jedoch nicht, auf welchen genau sie liegen, denn die vier Linien können bis zu vier Kreuzungen bilden. Aber nur genau zwei dieser Kreuzungen liegen auf einer gemeinsamen Zeile. Dies sind unsere fehlerhaften Bits.
In (a) haben wir nur zwei Kreuzungen; in (b) drei Kreuzungen, zwei auf einer Zeile, eine unterhalb; und in (c) haben wir vier Kreuzungen, zwei auf einer Zeile, eine oberhalb und eine unterhalb.

Lösung zu Aufgabe 2.6 auf Seite 8

Die Fehler haben wir wie folgt versteckt.



(a) (b) (c)

Wieder bilden die vier fehlerhaften Linien (zwei Zeilen und zwei Vorwärtsdiagonalen) zwischen zwei bis vier Kreuzungen, wovon genau zwei auf einer Spalte liegen. Dies sind die gesuchten Fehlerpositionen.

☑ Lösung zu Aufgabe 2.7 auf Seite 9

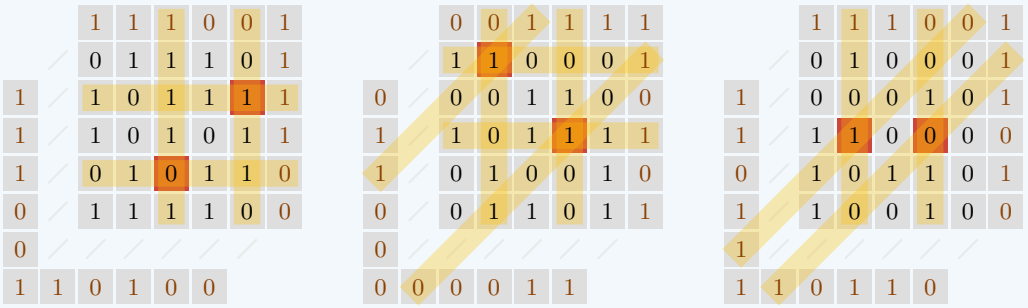
Wir können die drei möglichen Fälle von zwei Fehlern im Quadrat unterscheiden anhand der Anzahl fehlerhafter Zeilen, Spalten und Vorwärtsdiagonalen.

(i) **Zwei Fehler auf einer Zeile.** In diesem Fall sehen wir zwei fehlerhafte Spalten und zwei fehlerhafte Vorwärtsdiagonalen und keine fehlerhaften Zeilen.

(ii) **Zwei Fehler auf einer Spalte.** Dieser Fall führt zu je zwei fehlerhaften Zeilen und zwei fehlerhaften Vorwärtsdiagonalen und zu keiner fehlerhaften Spalte.

(iii) **Zwei Fehler auf unterschiedlichen Zeilen und Spalten.** In diesem Fall werden uns je zwei fehlerhafte Zeilen und zwei fehlerhafte Spalten angezeigt und entweder null oder zwei fehlerhafte Vorwärtsdiagonalen.

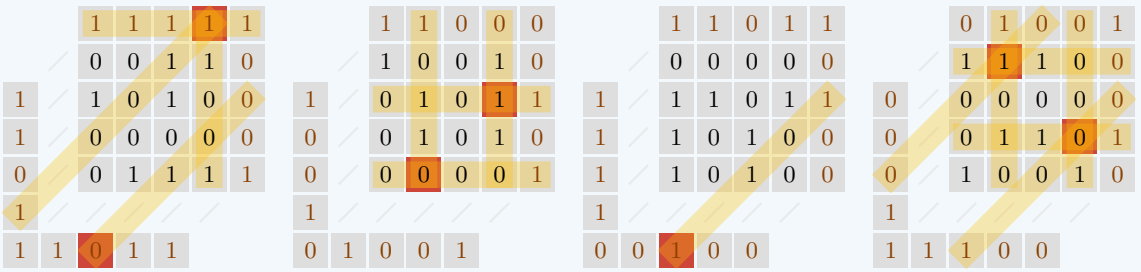
In (a) haben wir zwei fehlerhafte Zeilen und zwei fehlerhafte Spalten: Wir sind im Fall (iii). Wir müssen uns also entscheiden, welches der zwei Paare von gegenüberliegenden Kreuzungen die fehlerhaften Bits enthält. Da keine Vorwärtsdiagonalen einen Fehler anzeigen, müssen die zwei Fehler auf der gleichen Vorwärtsdiagonalen liegen. Nur eines der zwei Paare liegt auf einer gemeinsamen Vorwärtsdiagonalen. Wir haben die Fehler gefunden.
In (b) haben wir wiederum zwei fehlerhafte Zeilen und zwei fehlerhafte Spalten, also wiederum Fall (iii), aber dieses Mal mit zwei fehlerhaften Vorwärtsdiagonalen. Die Fehler befinden sich also auf dem Kreuzungspaar, das nicht auf einer gemeinsamen Vorwärtsdiagonalen liegt.
In (c) haben wir zwei fehlerhafte Spalten und zwei fehlerhafte Vorwärtsdiagonalen und keine fehlerhaften Zeilen: Wir sind im Fall (i). Die Spalten und Vorwärtsdiagonalen führen zu vier Kreuzungen, wovon genau zwei auf einer Zeile liegen. Dies sind die gesuchten Fehlerpositionen.



(a) (b) (c)

Lösung zu Aufgabe 2.8 auf Seite 9

Die Lösung sieht folgendermassen aus.



(a) (b) (c) (d)

Nun folgt die Begründung.

(1) In (a) haben wir zwei fehlerhafte Vorwärtsdiagonalen und je eine fehlerhafte Spalte und Zeile.

In (b) gibt es je zwei fehlerhafte Zeilen und Spalten.

In (c) ist nur eine Vorwärtsdiagonale fehlerhaft.

In (d) haben wir je zwei Vorwärtsdiagonalen, Zeilen und Spalten, die fehlerhaft sind.

(2) Wir haben in Aufgabe 1.2 gesehen, dass zwei fehlerhafte Zeilen oder Spalten nur dann auftreten können, wenn wir auch tatsächlich zwei Fehler im Quadrat haben. Demnach haben wir in den Fällen (b) und (d) zwei Fehler im Quadrat und null Fehler in den Vorwärtsdiagonalen-Prüfbits.

Zudem haben wir gesehen, dass je eine fehlerhafte Zeile und Spalte entweder auf einen Fehler oder auf einen speziellen Fall mit drei Fehlern hinweist. Da wir höchstens zwei Fehler haben, handelt es sich bei (a) um einen Fehler im Quadrat, den wir an der Kreuzung der Zeile und der Spalte (und der einen Vorwärtsdiagonalen) finden. Die andere fehlerhafte Vorwärtsdiagonale muss auf ein fehlerhaftes Vorwärtsdiagonalen-Prüfbit zurückgehen.

Wenn weder Zeilen noch Spalten einen Fehler anzeigen, so kann es keinen Fehler im Quadrat geben. Die fehlerhafte Vorwärtsdiagonale in (c) muss somit auf ein fehlerhaftes Vorwärtsdiagonalen-Prüfbit zurückzuführen sein.

(3) Bei (a) finden wir den Fehler im Quadrat an der Kreuzung der fehlerhaften Zeile und Spalte (und der Vorwärtsdiagonalen). Zudem ist die andere Vorwärtsdiagonale fehlerhaft. Das ist für uns aber nicht wichtig: Sobald wir alle Fehler im Quadrat und somit in der Nachricht korrigiert haben, können wir die Prüfbits der Vorwärtsdiagonalen (falls nötig) neu berechnen.

Bei (b) zeigt keine Vorwärtsdiagonale einen Fehler an. Die zwei Fehler müssen also an den zwei Kreuzungen liegen, die auf einer gemeinsamen Vorwärtsdiagonalen liegen.

Bei (c) liegt der einzige Fehler bei dem Prüfbit der fehlerhaften Vorwärtsdiagonalen.

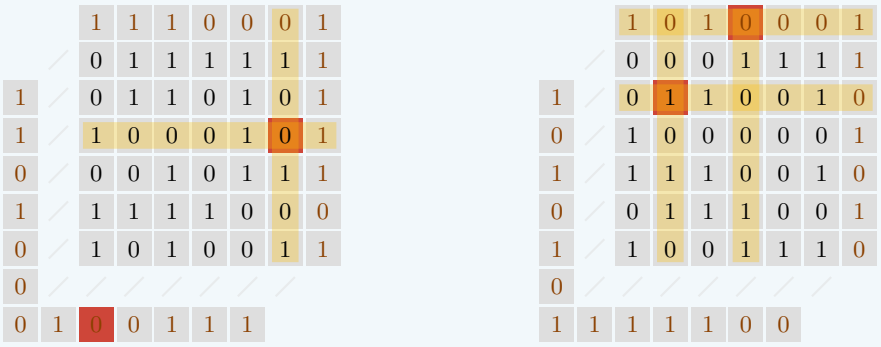
Bei (d) haben wir zwei fehlerhafte Vorwärtsdiagonalen. Die zwei Fehler müssen sich also an den entsprechenden Kreuzungen der jeweiligen Zeile, Spalte und Vorwärtsdiagonalen befinden (also auf dem Kreuzungspaar, das nicht auf einer gemeinsamen Vorwärtsdiagonalen liegt).

Lösung zu Aufgabe 2.9 auf Seite 10

Von Aufgabe 1.2 wissen wir, wie wir nur anhand des Quadrats 0, 1 und 2 Fehler im Quadrat unterscheiden können: Gibt es zwei fehlerhafte Zeilen oder Spalten, so sind wir im 2-Fehler-Fall; sind alle Zeilen und Spalten fehlerfrei, so gibt es keinen Fehler im Quadrat; und in allen anderen Fällen haben wir genau einen Fehler im Quadrat.
Wir wissen vom originalen Kartentrick, wie wir 0 und 1 Fehler korrigieren können. Dazu brauchen wir die Prüfbits der Vorwärtsdiagonalen überhaupt nicht. Falls wir potentielle Fehler darin auch korrigieren möchten, können wir dies tun, indem wir nach der Rekonstruktion der originalen Nachricht alle Prüfbits erneut berechnen.
Interessant ist also nur der Fall mit 2 Fehlern im Quadrat. Da wir höchstens zwei Fehler haben, sind die Prüfbits der Vorwärtsdiagonalen alle korrekt. Wir können die Fehler finden und korrigieren wie oben diskutiert: Wenn nur zwei Zeilen (und keine Spalten) einen Fehler anzeigen, so sind die Fehler bei den zwei Kreuzungen der fehlerhaften Spalten mit den fehlerhaften Vorwärtsdiagonalen, die auf einer Spalte liegen. Analog bei nur zwei fehlerhaften Spalten (und keinen fehlerhaften Zeilen) sind die Fehler bei den Kreuzungen, die auf einer Zeile liegen. Wenn zwei Zeilen und zwei Spalten Fehler anzeigen, dann gibt es zwei Fälle: Wenn keine Vorwärtsdiagonale fehlerhaft ist, so befinden sich die zwei Fehler auf den zwei Kreuzungen, die auf einer gemeinsamen Vorwärtsdiagonalen liegen; sonst befinden sie sich auf den anderen zwei Kreuzungen.

Lösung zu Aufgabe 2.10 auf Seite 11

Die Situationen sehen so aus.

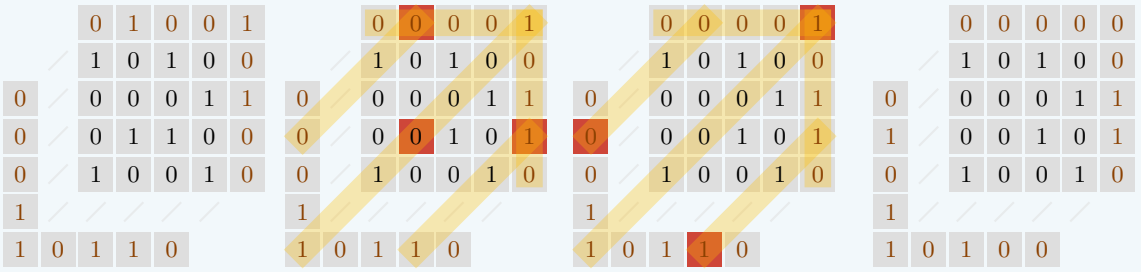


(a) (b)

Interessanterweise haben wir in beiden Situationen den Fall, dass sich zwei Fehler auf der gleichen Vorwärtsdiagonalen befinden (bei (a) ein Fehler im Quadrat und einer in den Prüfbits der Vorwärtsdiagonalen; bei (b) zwei Fehler im Quadrat) und sich somit auslöschen. Dennoch können wir sie mit unserer Strategie finden und korrigieren.

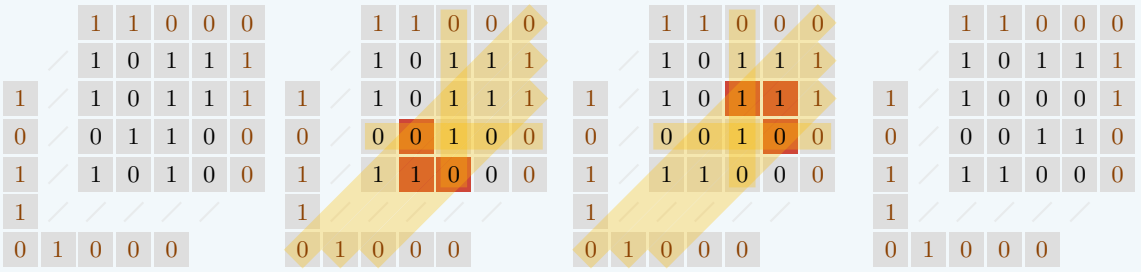
Lösung zu Aufgabe 2.11 auf Seite 11

Wenn wir zwei Codewörter mit Abstand 6 haben, können wir jeweils drei Fehler so legen, dass beide Codewörter genau Abstand 3 haben zum fehlerhaften Codewort. Wir können dann nicht entscheiden, welches Codewort das originale war. In andere Worte gefasst bedeutet dies, dass eine Kodierung mindestens Abstand 7 haben muss, um 3-fehlerkorrigierend zu sein.
Die folgenden Codewörter w und w' haben Abstand 6 zueinander und jeweils Abstand 3 zum fehlerhaften Codewort in der Mitte.



Codewort w | w mit drei Fehlern | w' mit drei Fehlern | Codewort w'

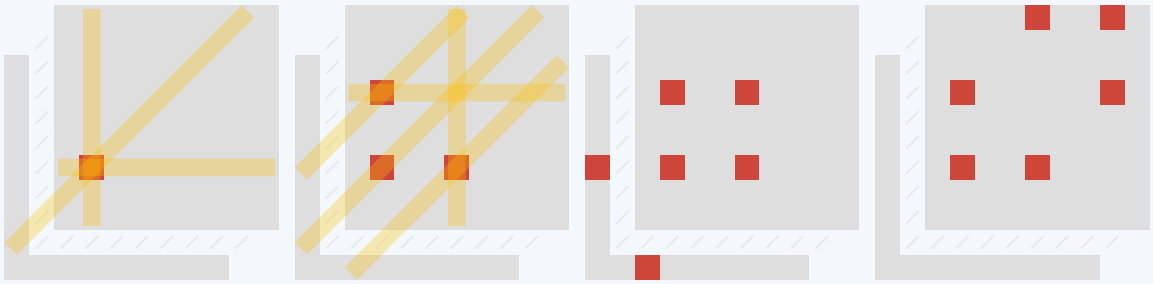
Alternativ könnten wir die zwei Codewörter w und w' auch so wählen, dass keines der Prüfbits verändert wird.



Codewort w | w mit drei Fehlern | w' mit drei Fehlern | Codewort w'

☑ Lösung zu Aufgabe 2.12 auf Seite 11

Wir betrachten zwei beliebige gültige Codewörter und zeigen, dass sie sich in mindestens 6 Bits unterscheiden müssen. Daraus folgt, dass der Abstand der Kodierung mindestens 6 ist. Wenn wir dies mit der Beobachtung aus Aufgabe 2.11 kombinieren, können wir schliessen, dass der Abstand der erweiterten Kartentrick-Kodierung genau 6 ist.
Zunächst beobachten wir, dass sich die zwei Codewörter an mindestens einer Position im Quadrat unterscheiden müssen. Ansonsten würde es sich entweder um die gleichen Codewörter handeln oder eines der Codewörter wäre nicht gültig.
Wir betrachten die unterste linke Position, an der sich die zwei Codewörter unterscheiden, also so, dass es weder links auf der gleichen Zeile noch unterhalb auf der gleichen Spalte eine Position gibt, an der sich die Bits der zwei Codewörter unterscheiden.
Wenn sich das zweite Codewort nur an dieser Position vom ersten Codewort unterscheiden würde, wären jeweils eine Zeile, eine Spalte und eine Vorwärtsdiagonale fehlerhaft (siehe (a)): Das zweite Codewort wäre nicht gültig.



(a) (b) (c) (d)

Um den Fehler auf der Zeile zu vermeiden, muss es also mindestens eine weitere Position auf der gleichen Zeile geben, an der sich die zwei Codewörter unterscheiden. Analog muss es mindestens eine weitere Position auf der gleichen Spalte geben, an der sich die zwei Codewörter unterscheiden, um den Fehler auf der Spalte zu vermeiden. Da die erste Position die unterste linke ist, befinden sich alle drei Positionen auf unterschiedlichen Vorwärtsdiagonalen, was zu drei fehlerhaften Vorwärtsdiagonalen führt (siehe (b)).
Diese drei Fehler können nur vermieden werden, wenn es mindestens drei weitere Positionen gibt (eine pro Vorwärtsdiagonale), an denen sich die Codewörter unterscheiden. Zwei Möglichkeiten, dies zu tun, sind in (c) und (d) dargestellt.
Insgesamt gibt es also immer mindestens 6 Positionen, an denen sich die zwei Codewörter unterscheiden müssen. Da dies für alle Codewörter der Fall ist, muss der Abstand der Kodierung mindestens 6 sein.

☑ Lösung zu Aufgabe 2.13 auf Seite 11

Vorwärtsdiagonalen können die Längen $1, 2, \ldots, x-1, x$ haben. Es gibt genau eine Diagonale der Länge x und jeweils zwei Diagonalen aller anderer Längen. Somit gibt es insgesamt $2(x-1)+1 = 2x-1$ Vorwärtsdiagonalen.

☑ Lösung zu Aufgabe 2.14 auf Seite 11

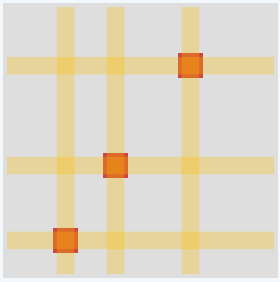
Wir runden zuerst die Nachrichtenlänge auf eine Quadratzahl auf: 20 und 25 werden zu 25; 30 wird zu 36. Für die ersten zwei Nachrichten hat das Nachrichten-Quadrat also Seitenlänge 5. Die Prüfbits führen zu einem Quadrat der Seitenlänge 6. Zu diesen 36 Bits kommen laut Aufgabe 2.13 noch 11 Vorwärtsdiagonalen-Prüfbits, also insgesamt $36 + 11 = 47$ Bits.
Analog haben wir für die letzte Nachricht eine Seitenlänge von 6 ohne und 7 mit Prüfbits. Zu diesen 49 Bits kommen 13 Vorwärtsdiagonalen-Prüfbits, insgesamt also $49 + 13 = 62$ Bits.
Im Allgemeinen brauchen wir für eine Nachricht der Länge $n = x^2$ ein Quadrat mit Seitenlänge $x + 1$ und je ein Bit für jede der $2(x + 1) - 1$ Vorwärtsdiagonalen (siehe Aufgabe 2.13). Insgesamt sind es also

$$(x + 1)^2 + 2(x + 1) - 1 = x^2 + 4x + 2 = n + 4\sqrt{n} + 2$$

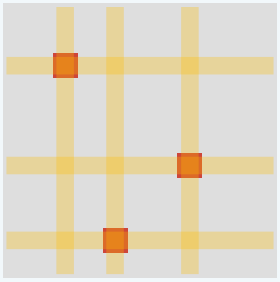
Bits, wovon n Bits für die Nachricht und $4\sqrt{n} + 2$ Bits für die Kontrollbits verwendet werden. Die Anzahl der Kontrollbits verhält sich zu der Anzahl Nachrichtenbits also ungefähr wie der Umfang eines Quadrats zu seiner Fläche.

☑ Lösung zu Aufgabe 3.1 auf Seite 12

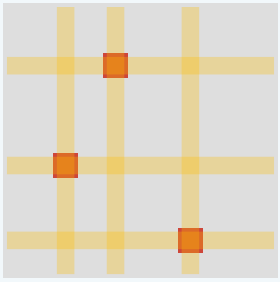
Es gibt die folgenden sechs Möglichkeiten, wie drei Fehler über die neun Kreuzungen verteilt sein können.



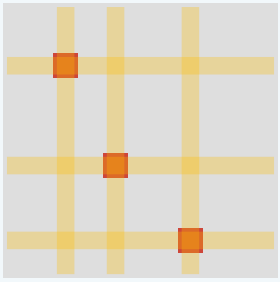
Oben rechts und unten links



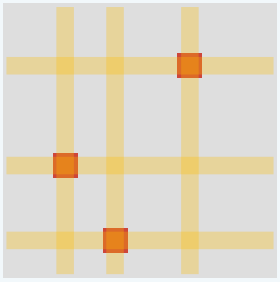
Oben links



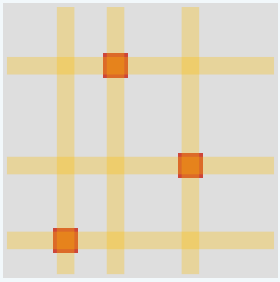
Unten rechts



Oben links und unten rechts



Oben rechts

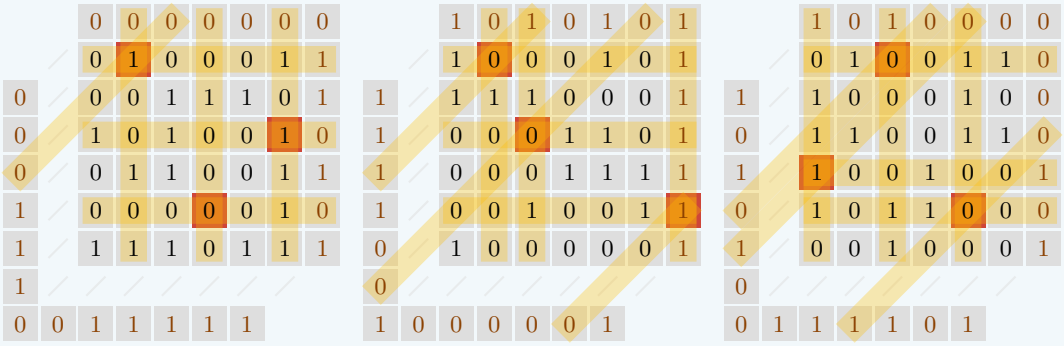


Unten links

Insbesondere fällt auf, dass bei all den sechs Möglichkeiten mindestens einer der kritischen Punkte (also die Kreuzungen ganz oben links, ganz oben rechts, ganz unten links und ganz unten rechts) einen Fehler enthält. Die jeweiligen kritischen Punkte mit Fehler sind in den einzelnen Fällen in den Beschriftungen der Abbildung oben bereits erwähnt. Diese einfache Beobachtung, dass mindestens ein kritischer Punkt einen Fehler enthält, wird beim weiteren Vorgehen noch von Nutzen sein.

Lösung zu Aufgabe 3.2 auf Seite 13

Die drei Fehler sind jeweils wie folgt positioniert.

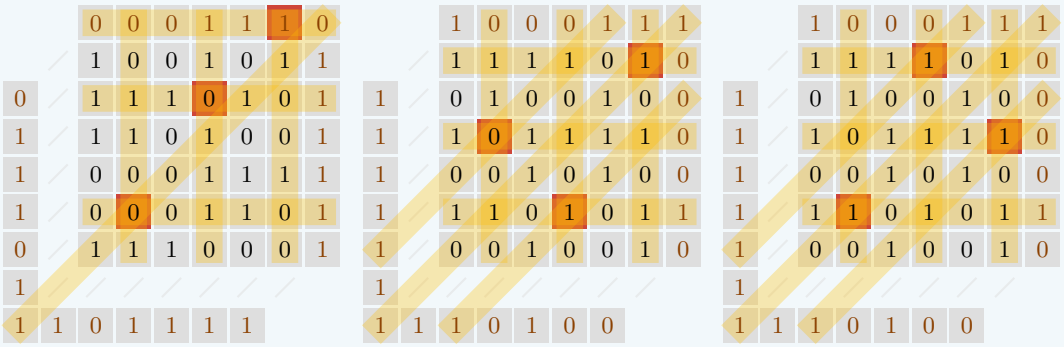


(a) (b) (c)

In allen drei Fällen gibt es mindestens eine fehlerhafte Vorwärtsdiagonale, die einen kritischen Punkt enthält, aber durch keine weitere Kreuzung verläuft. Also muss dieser kritische Punkt jeweils einen Fehler enthalten. Bei (a) ist es der kritische Punkt oben links, bei (c) der kritische Punkt unten rechts und bei (b) sind es sogar beide.

Lösung zu Aufgabe 3.3 auf Seite 13

Die Strategie funktioniert nur bei (a), aber nicht bei (b). Bei (a) gibt es wiederum eine fehlerhafte Vorwärtsdiagonale, die einen kritischen Punkt enthält, aber durch keine weitere Kreuzung führt. Bei (b) hingegen enthält jede fehlerhafte Vorwärtsdiagonale gleich mehrere Kreuzungen.
Konkreter ausgedrückt können wir also nicht entscheiden, ob der Fehler auf dem kritischen Punkt oben rechts oder unten links liegt. Tatsächlich sind beide Varianten auch möglich.



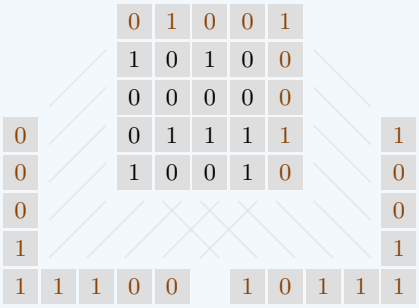
(a) Erste Variante für (b) Zweite Variante für (b)

Lösung zu Aufgabe 3.4 auf Seite 14

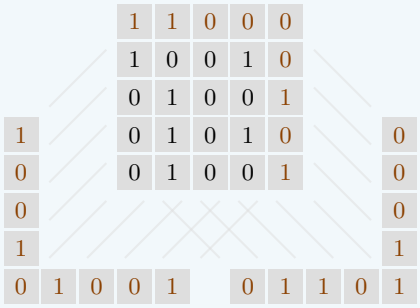
Die kritischen Punkte oben links und unten rechts liegen so, dass sie bezüglich der Richtung der Vorwärtsdiagonalen extrem sind: Eine Vorwärtsdiagonale durch einen dieser Punkte wird keine anderen Kreuzungen enthalten.
Die kritischen Punkte oben rechts und unten links haben die gleiche Eigenschaft nicht für Vorwärtsdiagonalen, aber für Rückwärtsdiagonalen (denken Sie an Aufgabe 2.2 zurück). Eine solche fehlerhafte Rückwärtsdiagonale würde dann also auch zuverlässig einen Fehler auf dem entsprechenden kritischen Punkt oben rechts oder unten links identifizieren.
Eine mögliche erfolgversprechende Idee wäre also, zusätzlich auch noch Prüfbits für alle Rückwärtsdiagonalen zu berechnen. Es wird sich zeigen, dass diese Idee auch für alle anderen Fälle gut genug ist, um drei Fehler zu korrigieren.

☑ Lösung zu Aufgabe 3.5 auf Seite 15

Hier sind alle Prüfbits für die zweifach erweiterte Kartentrick-Kodierung.



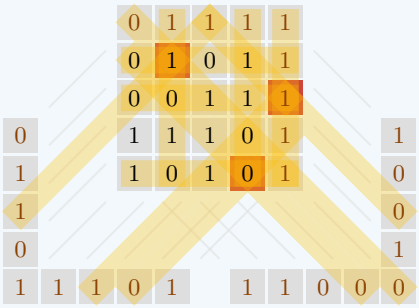
(a)



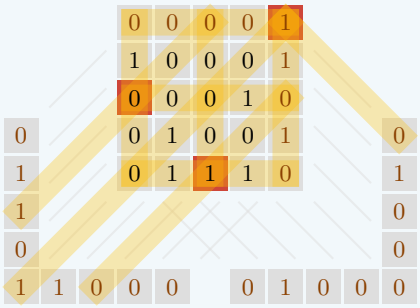
(b)

☑ Lösung zu Aufgabe 3.6 auf Seite 15

Untenstehend sehen Sie die beiden dargestellten Situationen inklusive der fehlerhaften Diagonalen sowie der markierten Fehler. Hoffentlich ist Ihnen aufgefallen, dass für das zweite Codewort gezwungenermassen die Prüfbits der Rückwärtsdiagonalen verwendet werden müssen.



(a)



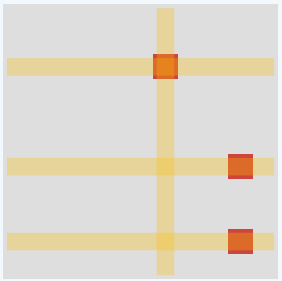
(b)

☑ Lösung zu Aufgabe 3.7 auf Seite 15

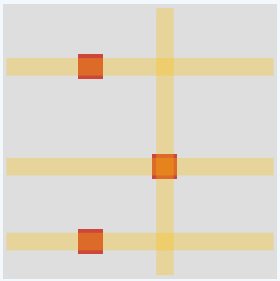
Fridolin kann ganz einfach den ersten ihm schon bekannten Fehler korrigieren, also das entsprechende Bit flippen. Wenn er alle anderen Nachrichten- und Prüfbits gleich belässt, dann erhält er auf diese Weise ein Codewort mit nur noch zwei Fehlern. Fridolin weiss bereits aus Abschnitt 2, wie zwei Fehler erkannt und korrigiert werden können. Er kann also einfach wie gewohnt die Information der (einfach) erweiterten Kartentrick-Kodierung verwenden, um die originale Nachricht wiederherzustellen. Die Prüfbits der Rückwärtsdiagonalen braucht er für diesen zweiten Schritt gar nicht mehr.

Lösung zu Aufgabe 3.8 auf Seite 16

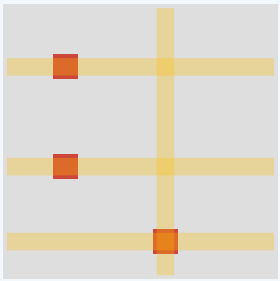
Es gibt im Wesentlichen die folgenden drei Arten, wie man drei Fehler verteilen kann.



Fehler auf oberer Kreuzung



Fehler auf mittlerer Kreuzung

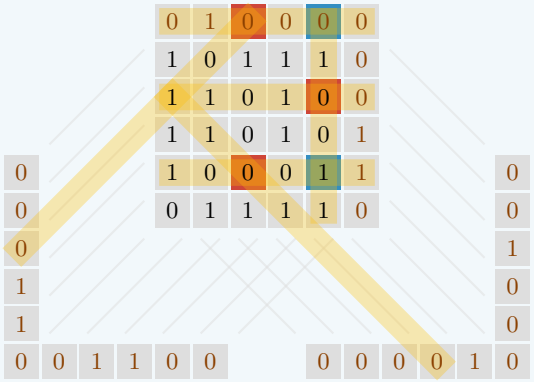


Fehler auf unterer Kreuzung

Es fällt auf, dass immer mindestens eine der drei Kreuzungen einen Fehler enthält. Es ist jedoch egal, ob es sich dabei um die obere, mittlere oder untere Kreuzung handelt.
Die anderen zwei Fehler müssen auf den übrigbleibenden fehlerhaften Zeilen liegen, aber sie können sich die gemeinsame Spalte völlig frei auswählen. Insbesondere könnte man dafür auch wieder die einzige fehlerhafte Spalte wählen, so dass alle drei Fehler auf einer Spalte liegen werden.
Wir erinnern uns, dass das Quadrat mit der Nachricht ohne die Prüfbits eine Seitenlänge von a hat. Wenn die obere Kreuzung einen Fehler enthält, dann kann man die beiden anderen Fehler also auf insgesamt $a+1$ unterschiedlichen Spalten verteilen. Das Gleiche gilt, wenn die mittlere Kreuzung oder die untere Kreuzung einen Fehler enthält. Die Anzahl Arten, wie man die drei Fehler verteilen kann, ist aber nicht, wie man denken könnte, $3 \cdot (a+1) = 3a+3$, sondern $3a+1$, weil der Fall mit drei Fehlern auf einer gemeinsamen Spalte sonst dreifach gezählt würde.

Lösung zu Aufgabe 3.9 auf Seite 16

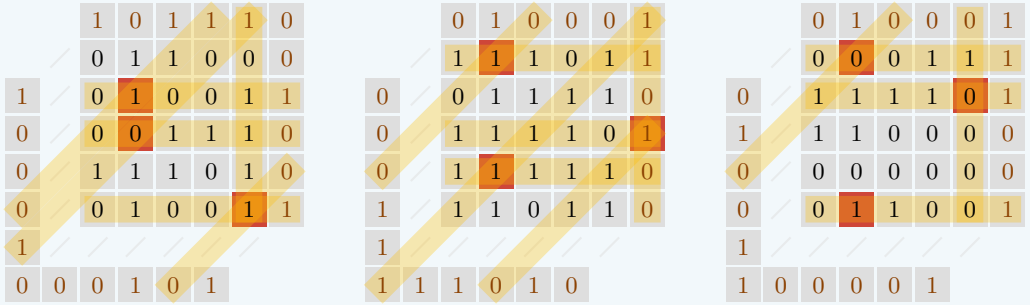
Fridolins Idee ist zwar gut und sie führt auch in vielen Situationen zum Erfolg. Aber leider klappt es nicht in allen Fällen. Schauen Sie sich zum Beispiel folgendes Codewort mit drei Fehlern an.



Es befindet sich also ein Fehler auf der mittleren Kreuzung (die beiden anderen Kreuzungen ohne Fehler sind wie gewohnt markiert). Gleichzeitig gibt es aber keine einzige fehlerhafte Diagonale (weder vorwärts noch rückwärts) durch alle drei Kreuzungen. Fridolin kann also basierend allein auf dieser Information nicht entscheiden, wo jetzt genau der Fehler liegen soll.

☑ Lösung zu Aufgabe 3.10 auf Seite 17

Die allgemeine Strategie, um den oberen linken Fehler zu finden, ist einfach: Wir nehmen die erste fehlerhafte Vorwärtsdiagonale von links ausgehend. Der gesuchte Fehler ist dann in dieser Diagonalen enthalten, während die beiden anderen Fehler auf der rechten Seite der Diagonalen positioniert sind. Überzeugen Sie sich selbst bei den Codewörtern jetzt mit aufgedeckten Fehlern.



(a) (b) (c)

☑ Lösung zu Aufgabe 3.11 auf Seite 18

Angenommen wir wüssten, in welchem spezifischen der sechs Fälle wir sind. Dann ist der gesuchte Fehler jeweils auf der folgenden Kreuzung einer Diagonalen und einer Zeile zu finden.

(ol) Die Kreuzung zwischen der *oberen* fehlerhaften Zeile und der ersten fehlerhaften *Vorwärtsdiagonalen* von *links* ausgehend.

(l) Entweder die Kreuzung von Fall (ol) oder von Fall (ul).

(ul) Die Kreuzung zwischen der *unteren* fehlerhaften Zeile und der ersten fehlerhaften *Rückwärtsdiagonalen* von *links* ausgehend.

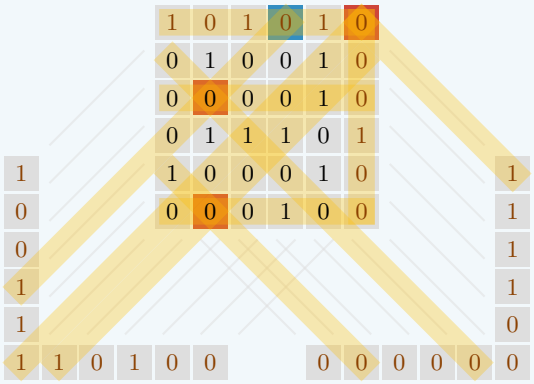
(or) Die Kreuzung zwischen der *oberen* fehlerhaften Zeile und der ersten fehlerhaften *Rückwärtsdiagonalen* von *rechts* ausgehend.

(r) Entweder die Kreuzung von Fall (or) oder von Fall (ur).

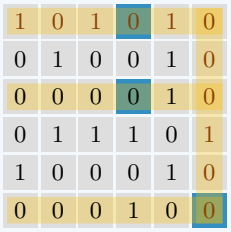
(ur) Die Kreuzung zwischen der *unteren* fehlerhaften Zeile und der ersten fehlerhaften *Vorwärtsdiagonalen* von *rechts* ausgehend.

☑ Lösung zu Aufgabe 3.12 auf Seite 18

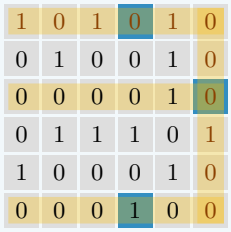
Sogar wenn man sich nicht sicher ist, welcher Fall zutrifft, kann man die beiden Strategien für die Fälle (ol) und (ul) separat ausführen. Dies ergibt zwei mögliche Positionen für den gesuchten Fehler, doch welche ist die richtige? Es muss diejenige sein, die weiter links ist.



Mit aufgedeckten Fehlern ist klar, wieso das so ist: Falls der Fehler tatsächlich auf der weiter rechten Position liegen würde (in der Abbildung oben ist diese spezifische Position markiert), dann wäre es für die fehlerhafte Diagonale durch die sich weiter links befindende Position (in der Abbildung oben ist das die erste fehlerhafte Rückwärtsdiagonale ausgehend von links) gar nicht mehr möglich, einen Fehler zu enthalten. Die zwei Möglichkeiten, wie die anderen Fehler überhaupt noch verteilt sein könnten, falls tatsächlich der gesuchte Fehler bei der Markierung oben wäre, sind im Folgenden aufgezeichnet.



Erste Möglichkeit



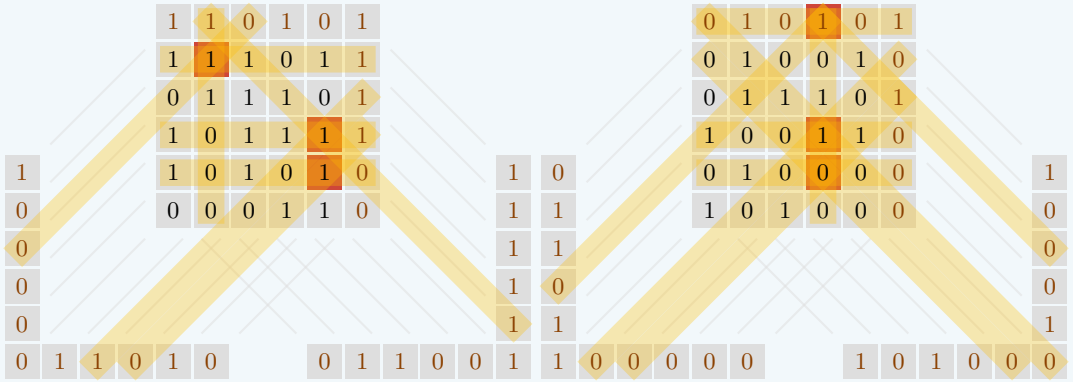
Zweite Möglichkeit

☑ Lösung zu Aufgabe 3.13 auf Seite 18

Wir führen zuerst separat die Strategien für die Fälle (or) und (ur) aus, wodurch wir zwei mögliche Positionen für einen ersten Fehler auf den kritischen Punkten erhalten. Von diesen zwei Positionen wählen wir dann diejenige aus, die weiter rechts ist.

Lösung zu Aufgabe 3.14 auf Seite 19

Die richtige Idee ist hier wieder die gleiche wie vorher, nämlich einfach die beiden Strategien von Aufgabe 3.12 und Aufgabe 3.13 separat auszuführen.
Die erste Strategie führt bei (a) nicht weit, weil keine einzige relevante fehlerhafte Diagonale die kritischen Punkte links von der fehlerhaften Spalte überhaupt enthält. Diese Beobachtung wird man immer machen, wenn die falsche Strategie für ein gegebenes Codewort ausgeführt wird. Die zweite Strategie hingegen führt bei (a) wie gewünscht zu einem ersten Fehler auf der unteren fehlerhaften Zeile.
Bei (b) führen aus demselben Grund beide Strategien zu nichts. Das heisst, das gegebene Codewort entspricht weder den Fällen der ersten Strategie (also (ol), (l) oder (ul)) noch den Fällen der zweiten Strategie (also (or), (r) oder (ur)). Das bedeutet dann aber, dass nur noch der spezielle Fall (m) übrigbleibt und deshalb alle drei Fehler auf der fehlerhaften Spalte liegen müssen.



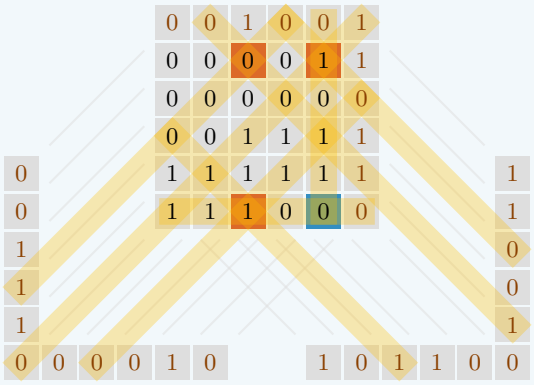
(a) (b)

Lösung zu Aufgabe 3.15 auf Seite 20

Wir müssen uns einfach fragen, wie viele Positionen es für den dritten Fehler gibt. Denn sobald der dritte Fehler gewählt ist (also derjenige Fehler, der nicht auf der fehlerhaften Zeile oder Spalte liegt), ist sofort klar, wo die beiden anderen Fehler sein müssen.
Dazu nehmen wir zuerst einfach die Fläche des gesamten Quadrats, also $(a+1)^2 = a^2 + 2a + 1$. Dann subtrahieren wir davon die Anzahl Positionen, die von der fehlerhaften Zeile oder Spalte blockiert sind, also $2a+1$. Wir erhalten somit genau a^2 unterschiedliche Arten, den dritten Fehler zu platzieren und deshalb auch genau a^2 unterschiedliche Arten, die drei Fehler anzuordnen.

☑ Lösung zu Aufgabe 3.16 auf Seite 20

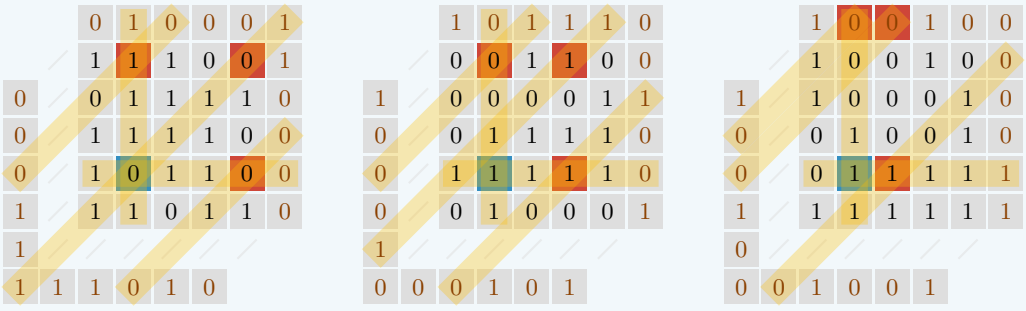
Wir betrachten dazu das folgende konkrete Codewort mit dem dritten Fehler im oberen linken Quadranten. Es sei aber angemerkt, dass alle erwähnten Eigenschaften auch im Allgemeinen gelten.



Bei den Vorwärtsdiagonalen sehen wir (von links nach rechts gehend), dass zuerst alle fehlerhaften Vorwärtsdiagonalen erscheinen und erst dann die spezielle Vorwärtsdiagonale, die durch die markierte Kreuzung verläuft. Bei den Rückwärtsdiagonalen hingegen ist es durchmischt: Es erscheint zuerst eine fehlerhafte Rückwärtsdiagonale, erst dann die spezielle Rückwärtsdiagonale durch die markierte Kreuzung und dann nochmals mindestens eine fehlerhafte Rückwärtsdiagonale.
Die beobachtete Abfolge bei den Vorwärtsdiagonalen weist also eindeutig darauf hin, dass der dritte Fehler sich im oberen linken Quadranten befindet, also dass der Fall (ol) zutrifft. Auf gleiche Weise würde die umgekehrte Abfolge (also zuerst die Vorwärtsdiagonale durch die markierte Kreuzung und erst dann alle fehlerhaften Vorwärtsdiagonalen) auf einen Fehler im unteren rechten Quadranten hinweisen, also darauf, dass der Fall (ur) zutrifft. Die Fälle für Fehler in den Quadranten oben rechts oder unten links, also die Fälle (or) und (ul), könnte man auf gleiche Weise bei den Rückwärtsdiagonalen ablesen.

☑ Lösung zu Aufgabe 3.17 auf Seite 21

Den Fehler auf der fehlerhaften Zeile kann man zum Beispiel immer auf der Kreuzung mit der letzten fehlerhaften Vorwärtsdiagonalen finden. Den Fehler auf der fehlerhaften Spalte findet man hingegen auf der Kreuzung mit der ersten fehlerhaften Vorwärtsdiagonalen. Der Vollständigkeit halber sind hier nochmals die gleichen Codewörter aufgelistet, aber jetzt mit aufgedeckten Fehlern.

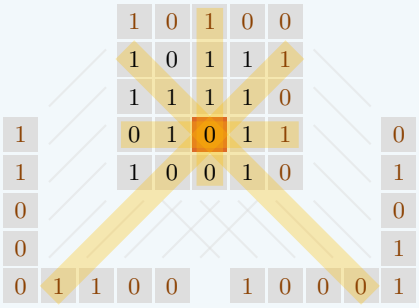


☑ Lösung zu Aufgabe 3.18 auf Seite 21

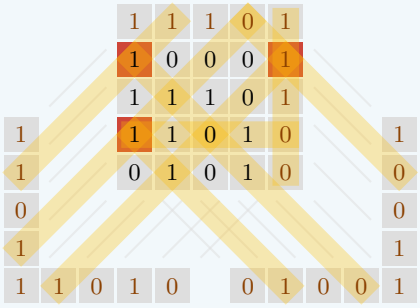
In den folgenden Überlegungen konzentrieren wir uns vorerst auf das uns unbekannte und fehlerlose Codewort. Es sei dabei $V = v_1 + \cdots + v_m$ (modulo 2) die Summe aller Vorwärtsdiagonalen-Prüfbits $v_1, \ldots, v_m$. Ferner sei $Z = z_1 + \cdots + z_m$ (modulo 2) die Summe aller Zeilensummen $z_1, \ldots, z_m$ im originalen Kartentrick-Quadrat. Man beachte, dass wir mit der Zahl z_i nicht ein Zeilen-Prüfbit meinen, sondern tatsächlich die Summe aller Nachrichten- und Prüfbits in einer gegebenen Zeile.
Wenn wir an die Abbildung mit den Vorwärtsdiagonalen in Aufgabe 2.2 zurückdenken, dann sehen wir, dass jedes Bit des originalen Kartentrick-Quadrats in die Berechnung von genau einem Vorwärtsdiagonalen-Prüfbit v_i einfliesst. Ferner sollte klar sein, dass jedes derartige Bit auch in genau einer Zeilensumme z_i vorkommt. Es folgt daraus, dass die beiden Summen V und Z die gleichen Terme aufaddieren, so dass also $V = Z$ gelten muss.
Die Summe (modulo 2) aller Bits auf einer Zeile des originalen Kartentrick-Quadrats ist immer gleich 0, es gilt also immer $z_i = 0$. Daraus folgt nun aber, dass $V = Z = 0$ gilt. Auf analoge Weise lässt sich zeigen, dass die Summe R aller Rückwärtsdiagonalen-Prübits auch gleich 0 ist (modulo 2).
Wenn wir also nun die Werte V und R für das gegebene fehlerhafte Codewort berechnen, dann erhalten wir $V = R = 0$ genau dann, wenn keines der Diagonalen-Prüfbits fehlerhaft ist. Falls es ein fehlerhaftes Vorwärtsdiagonalen-Prüfbit gibt, so erhalten wir $V = 1$. Falls es ein fehlerhaftes Rückwärtsdiagonalen-Prüfbit gibt, so erhalten wir $R = 1$. Anhand dieser Information lässt sich also einfach entscheiden, ob und wo ein fehlerhaftes Diagonalen-Prüfbit existiert.

Lösung zu Aufgabe 3.19 auf Seite 22

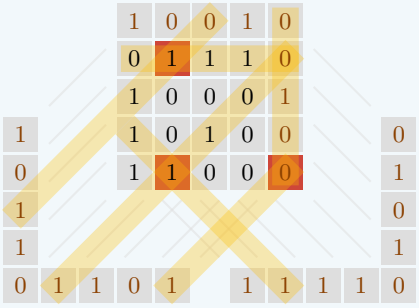
Die Anzahl Fehler im jeweiligen Quadrat können Sie den folgenden Codewörtern mit aufgedeckten Fehlern entnehmen.



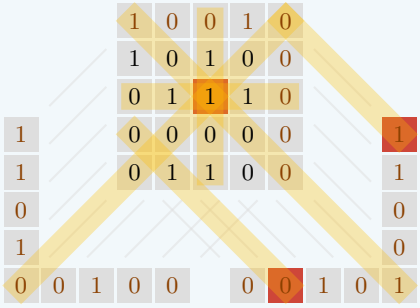
(a)



(b)



(c)



(d)

Allgemein ist zu bemerken, dass drei Fehler im Quadrat immer mit drei fehlerhaften Vorwärtsdiagonalen oder mit drei fehlerhaften Rückwärtsdiagonalen einhergehen. Dies erlaubt uns sofort, Fall (a) als 1-Fehler-Fall zu erkennen.
Fall (d) zeigt jedoch, dass die Unterscheidung nicht immer ganz einfach ist: Vom 1-Fehler-Fall ausgehend kann man mit zwei zusätzlichen Fehlern auf den Diagonalen-Prüfbits ebenso eine Konstellation mit zum Beispiel drei fehlerhaften Rückwärtsdiagonalen erzeugen. Trotzdem kann man diese Situation relativ einfach vom 3-Fehler-Fall in (c) unterscheiden, weil die fehlerhafte Diagonale in der anderen Richtung immer die Kreuzung der fehlerhaften Zeile und Spalte enthält.

☑ Lösung zu Aufgabe 3.20 auf Seite 22

Wir besprechen die einzelnen Fälle der Reihe nach.

- Falls $f = 0$ gilt, dann hat Frida nichts weiter zu tun. Sie weiss bereits, dass insbesondere die Nachrichtenbits alle korrekt sind.
- Falls $f = 1$ gilt, dann kann Frida getrost alle Diagonalen-Prüfbits ignorieren. Sie weiss, dass das Quadrat selber nur einen Fehler enthält und kann somit den einzelnen Fehler mit dem ursprünglichen Kartentrick finden und korrigieren.
- Falls $f = 2$ gilt, dann weiss Frida auch, dass höchstens eines der Diagonalen-Prüfbits falsch ist. Insbesondere bedeutet das, dass entweder alle Vorwärtsdiagonalen-Prüfbits oder alle Rückwärtsdiagonalen-Prüfbits (oder beides) korrekt sind. Falls erstere korrekt sind, kann Frida direkt die Methoden für die 2-Fehlerkorrektur aus Abschnitt 2 anwenden. Falls letztere korrekt sind, kann Frida sich überlegen, dass sie in Abschnitt 2 anstatt mit Vorwärtsdiagonalen genauso gut mit Rückwärtsdiagonalen hätte arbeiten können. Es spielt also gar keine Rolle, welche Art von Diagonalen sie verwendet, solange die entsprechenden Prüfbits alle korrekt sind. Mit Hilfe von Aufgabe 3.18 kann Frida einfach entscheiden, ob die Vorwärts- oder Rückwärtsdiagonalen-Prüfbits (oder beide) fehlerfrei sind.
- Falls $f = 3$ gilt, dann weiss Frida auch, dass alle Diagonalen-Prüfbits korrekt sind. Sie kann also die Methoden für die 3-Fehlerkorrektur, die in diesem Abschnitt unter der Annahme der Korrektheit der Diagonalen-Prüfbits entstanden sind, einfach anwenden.

☑ Lösung zu Aufgabe 3.21 auf Seite 22

Wie in Aufgabe 2.14 brauchen wir 25, 25 und 36 Bits für die Nachrichtenbits. Dann brauchen wir 36, 36 und 49 Bits für das ganze Quadrat der Kartentrick-Kodierung. Jetzt müssen wir aber, weil es sowohl Vorwärts- als auch Rückwärtsdiagonalen gibt, je zweimal die Anzahl 11, 11 und 13 von Diagonalenbits hinzuaddieren. Insgesamt erhalten wir schliesslich je 58, 58 und 75 für die Anzahl benötigter Bits für unsere drei Nachrichten.
Für eine Nachricht der Länge $n = x^2$ benötigen wir ein Quadrat der Seitenlänge $x + 1$ und je ein Bit für jede der $2(x + 1) - 1$ Vorwärtsdiagonalen und jede der $2(x + 1) - 1$ Rückwärtsdiagonalen (siehe Aufgabe 2.13). Insgesamt benötigt Fridolin also

$$(x + 1)^2 + 2(x + 1) - 1 + 2(x + 1) - 1 = x^2 + 6x + 3 = n + 6\sqrt{n} + 3$$

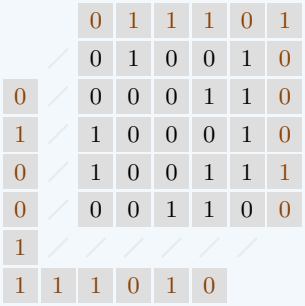
Bits, wovon n Bits für die Nachricht und $6\sqrt{n} + 3$ Bits für die Kontrollbits verwendet werden.

☑ Lösung zu Aufgabe 4.1 auf Seite 23

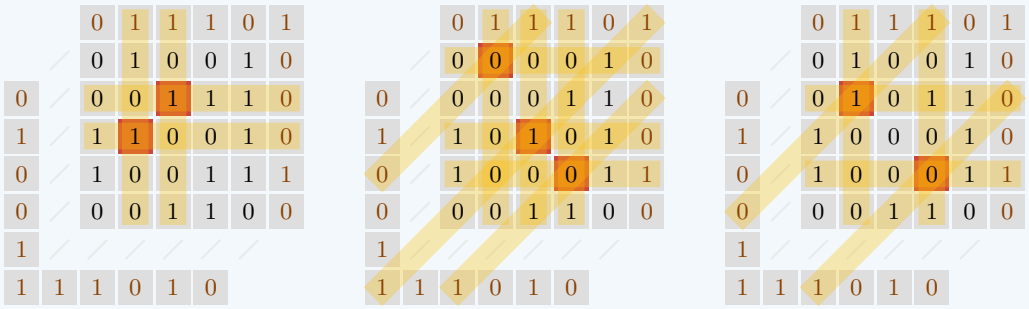
Leider funktioniert Fridolins Idee nicht. Falls die Fehler auf zwei unterschiedlichen Zeilen und Spalten platziert werden, befinden sie sich entweder im gleichen Bereich (in welchem Fall keiner der zwei Bereiche einen Fehler anzeigt) oder in zwei unterschiedlichen Bereichen mit je einer zusätzlichen Kreuzung ohne Fehler im gleichen Bereich (in welchem Fall beide Bereiche einen Fehler anzeigen, aber der Fehler nicht auf eine Kreuzung zurückgeführt werden kann). Bei zwei Fehlern auf einer Spalte oder einer Zeile ist die Sache noch hoffnungsloser. Falls beide Fehler im gleichen Bereich sind, zeigt kein Bereich einen Fehler an. Falls sie in unterschiedlichen Bereichen liegen, so gibt es zahlreiche andere Positionen, an denen die Fehler auch liegen könnten.

Lösung zu Aufgabe 4.2 auf Seite 23

(1) Wir erhalten das folgende Codewort.



(2) Die fehlerhaften Linien sehen wie folgt aus.



Tatsächlich können wir in allen drei Fällen die Fehler korrigieren.

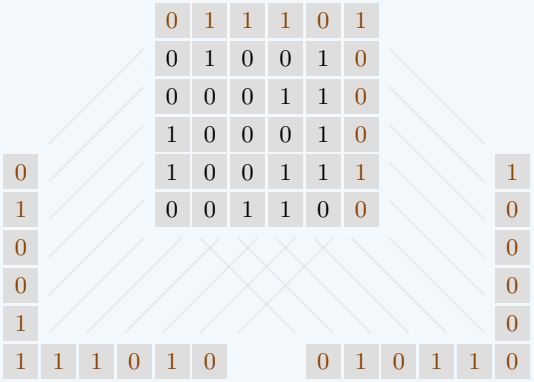
(a) Es gibt keine fehlerhafte Vorwärtsdiagonalen: Die zwei Fehler müssen also auf den zwei Kreuzungen, die auf einer Vorwärtsdiagonale sind, liegen.

(b) Eine fehlerhafte Vorwärtsdiagonale verläuft durch den kritischen Punkt oben links; eine durch den kritischen Punkt unten rechts. Damit haben wir zwei Fehler gefunden und können sie korrigieren. Der dritte Fehler muss auf der Kreuzung der nach wie vor fehlerhaften Linien liegen.

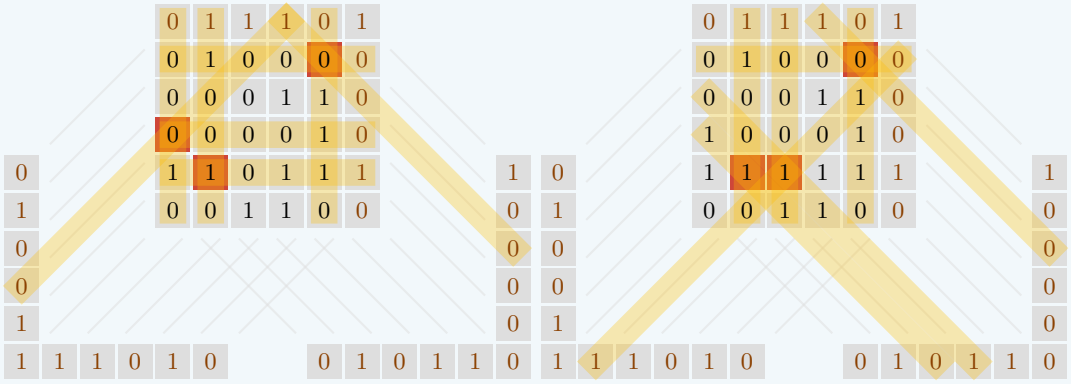
(c) Die Vorwärtsdiagonalen durch die Kreuzung oben links und durch die Kreuzung unten rechts sind fehlerhaft. Die Fehler müssen dort liegen.

Lösung zu Aufgabe 4.3 auf Seite 24

(1) Wir erhalten das folgende Codewort.



(2) Die fehlerhaften Linien sehen wie folgt aus.



Wir gehen wie folgt vor, um einen ersten Fehler zu finden.

(a) Da wir je drei fehlerhafte Zeilen und Spalten haben, muss es sich also um Fall 3A mit den vier kritischen Punkten handeln. Bei den kritischen Punkten oben links und unten rechts sind die entsprechenden Vorwärtsdiagonalen nicht fehlerhaft, also kann hier kein Fehler vorliegen. Beim kritischen Punkt oben rechts ist jedoch die entsprechende Rückwärtsdiagonale fehlerhaft, also muss an diesem Punkt ein Fehler sein.

(b) Wir haben drei fehlerhafte Spalten und nur eine fehlerhafte Zeile. Das bedeutet, dass wir im Fall 3B mit zwei Fehlern auf einer gemeinsamen Zeile sein müssen. Wir bestimmen eine mögliche Position für einen Fehler an der Kreuzung der linken fehlerhaften Spalte und der ersten Rückwärtsdiagonalen von links. Diese müssen wir vergleichen mit einer zweiten möglichen Position an der Kreuzung der rechten fehlerhaften Spalte und der ersten (in diesem Fall einzigen) Vorwärtsdiagonalen von rechts. Da die erste Position weiter unten ist als die zweite, muss dort ein Fehler liegen.