Kanalkodierung

Leistungsnachweis

Aufgabe 1

Vergleich der Leistungsfähigkeit serieller und paralleler Verkettung am HAMMING-Kode

Sommersemester 2012

Bearbeiter:
Linda Leuschner(s8796190@mail.zih.tu-dresden.de)
Rene Kowalewski(s9248090@mail.zih.tu-dresden.de)

Betreuer:
Dr.-Ing. Dagmar Schönfeld

Inhaltsverzeichnis

1 Grundlagen

In diesem Kapitel gehen wir auf theoretische Grundlagen ein. Zunächst erklären wir die Prinzipien der parallelen und seriellen Kodeverkettung und stellen die Hamming-Matrizen auf, welche diese Prinzipien mathematisch beschreiben. Anschließend werden wir das bit-flipping-Dekodierungsverfahren untersuchen. Dabei beschreiben wir bit-flipping sowohl für hard-decision als auch unter Verwendung von soft-Zuverlässigkeiten. Zuletzt wenden wir uns dem Dekodieren mittels Kontrollgleichungen zu, betrachten dabei allerdings nur parallele Kodeverkettung.

1.1 Kodeverkettung

Wir untersuchten Kodeverkettung am Beispiel des (7,4,3)-Hammingkodes. Dabei wird ein Quellkodewort nicht einzeln kodiert, sondern vier Quellkodewörter als ein Quellkodeblock der Größe 4×4 aufgefasst. Dieser Block wird dann sowohl horizontal als auch vertikal dekodiert. Wir bezeichnen ein Quellkodewort des (7,4,3)-Hammingkodes als Quellkodewort, eine Zusammenfassung von vier Quellkodewörtern als Quellkodeblock.

1.1.1 Parallele Kodeverkettung

Bei der parallelen Kodeverkettung werden die Spalten und Zeilen des Kodeblocks unabhängig voneinander kodiert. Im Falle einer parallelen Hammingkodeverkettung besteht ein Quellkodeblock aus vier Quellkodewörten bzw. aus 16 Quellkodezeichen. Diese vier Wörter werden zunächst jeweils mittels Hammingkode kodiert. Anschließend werden die jeweils 1. Zeichen der Wörter, die jeweils 2. Zeichen der Wörter usw. als Quellkodewörter aufgefasst und ebenfalls mittels Hammingkode kodiert. Die entstehenden 24 redundanten Stellen werden zusammen mit den 16 Quellkodezeichen übertragen. Abbildung 1 veranschaulicht die parallele Kodeverkettung.
Dieser Kodierungsvorgang lässt sich auch mittels einer Kontrollmatrix $H \in \{0,1\}^{40,16}$ darstellen. Diese Matrix ist in Abbildung 2 abgebildet. Die ersten 12 Zeilen sorgen für die horizontale Kodierung. Jeweils drei Zeilen kodieren ein Quellkodewort, relevant sind dabei jeweils nur die Stellen, welche zugleich Positionen des Quellkodewortes sind. Die restlichen Zellen der jeweiligen Spalte sind mit Nullen besetzt, mit Ausnahme der Position, an der die berechnete redundante Stelle platziert wird. Die restlichen 12 Zeilen werden für die vertikale Kodierung benötigt.

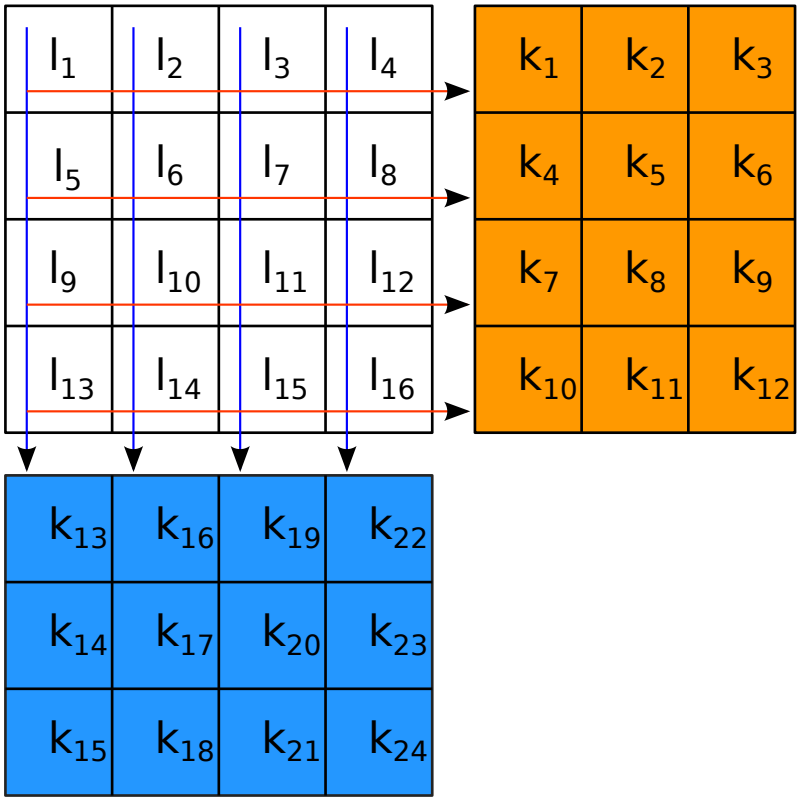

Abbildung 1: Schematische Darstellung der parallelen Kodeverkettung

Auch hier werden alle nicht relevanten Stellen mit Nullen besetzt, die Positionen aus denen sich im Quellkodeblock ein vertikales Quellkodewort zusammensetzt, werden entsprechend der Hammingkodierung belegt.

1.1.2 Serielle Kodeverkettung

Bei der seriellen Kodeverkettung werden bei der vertikalen Kodierung neben den Informationsstellen auch die redundanten Stellen der horizontalen Dekodierung berücksichtigt (Abbildung 3). Das heißt, die jeweils 1., 2. usw. Kontrollstellen der horizontalen Kodierung bilden ein neues Wort, welches mittels Hammingkode kodiert wird. Es entsteht ein (49,16)-Kode. Auch dieser Kodierungsvorgang lässt sich mittels einer Kontrollmatrix beschreiben (Abbildung 4).

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

Abbildung 2: Kontrollmatrix für die parallele Kodeverkettung.

1.2 Bit-Flipping

Bit-flipping ist ein iteratives Dekodierungsverfahren. In jeder Iteration werden aus allen Positionen der empfangenen Kanalkodefolge eine oder mehrere Positionen ermittelt, welche (heuristisch) am meisten zu einer Korrektur des Wortes beitragen und anschließend „geflippt“. Bei hard-decision Dekodierung heißt das, dass bei einer Kanalkodefolge, welche nur aus Nullen und Einsen besteht (bzw. genauer aus -1 und 1), eine zu flippende Eins in eine Null und eine zu flippende Null in eine Eins ungewandelt wird.
Bei soft-decision Dekodierung liegt eine Kanalkodefolge $b \in \mathbb{R}^n$ vor, d.h. die Kanalzeichen können auch Werte zwischen -1 und 1 annehmen. Weighted bit-flipping ist ein Algorithmus, welcher eine hart entschiedene Kanalkodefolge dekodiert, dabei aber weiche Zuverlässigkeiten einbezieht.

1.2.1 Bit-Flipping mit Hard Decision

Für eine fehlerbehaftete Kanalkodefolge b und die Kontrollmatrix H (siehe Abbildung 2 bei paralleler Kodeverkettung und Abbildung 4 bei serieller Kodeverkettung) kann mittels $e = (H \cdot b^T)^T \cdot H$ ein Gewichtsvektor e berechnet werden. Durch $(H \cdot b^T)$ wird dabei das Syndrom

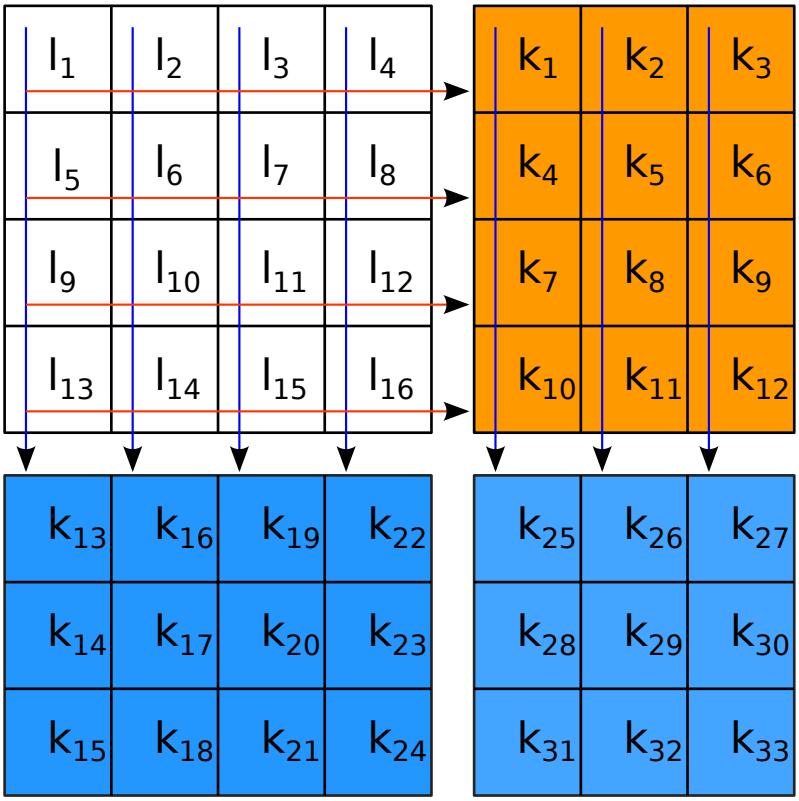

Abbildung 3: Schematische Darstellung der seriellen Kodeverkettung

s berechnet, ist dieses ungleich dem Nullvektor, so liegen Übertragungsfehler vor. Der Vektor $s^T \cdot H$ zählt für jede Spalte der Matrix - und somit für jede Position in dem Kanalkodewort - wie viele Einsen in der Spalte und im Syndrom übereinstimmen. Je mehr Einsen übereinstimmen, desto mehr bewirkt (heuristisch) ein Flip an dieser Position eine Veränderung des Syndroms in Richtung des Nullvektors.
Allerdings kann man dadurch im Allgemeinen nicht die exakte oder die exakten Positionen bestimmen, durch die die Empfangsfolge auf Anhieb zu b_{korr} korrigiert wird, sodass das Syndrom $s = H \cdot (b_{korr})^T$ dem Nullvektor entspricht. In vielen Fällen werden sogar zunächst die falschen Stellen geflippt. Durch die iterative Anwendung der oben genannten Berechnung können solche Fehlentscheidungen allerdings bei ausreichend kleiner Fehleranzahl wieder rückgängig gemacht werden.
Für das übertragene Kanalkodewort $b = b^0$ wird das Syndrom s^0 berechnet, und, falls dieses nicht dem Nullvektor entspricht, der Gewichtsvektor e^0. Für alle Positionen in e^0, die einen vorgegebenen Schwellwert T überschreiten, wird die entsprechende Position in b geflippt. Für

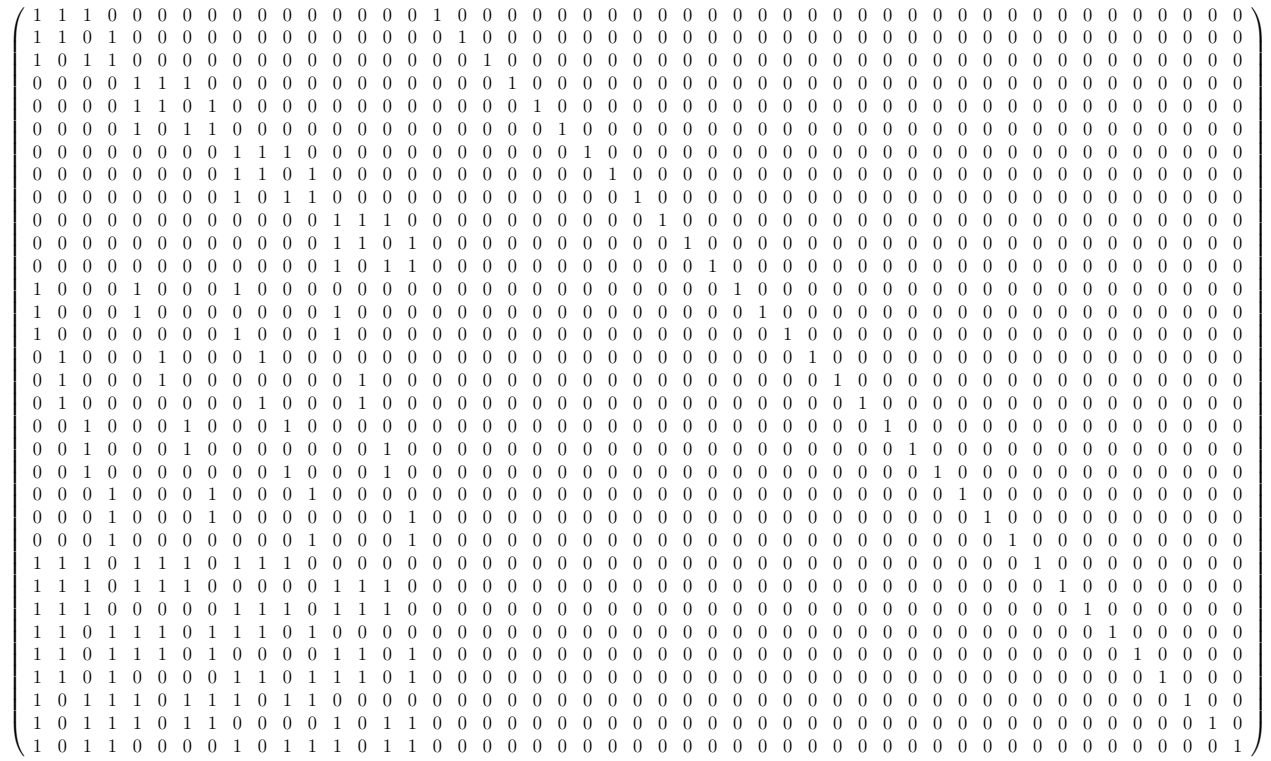

Abbildung 4: Kontrollmatrix für die serielle Kodeverkettung.

den entstehenden Vektor b^1 wird erneut das Syndrom s^1 berechnet, und, falls dieses wiederum nicht den Nullvektor entspricht, der Gewichtsvektor e^1 usw. Ist in der i-ten Iteration das Syndrom $s^i = \vec{0}$, so ist die Dekodierung abgeschlossen.

1.2.2 Weighted Bit-Flipping

Auch bei weighted bit-flipping wird der Algorithmus abgebrochen, wenn das Syndrom 0 ist. Da das Empfangsfolge $y = (y_1, \ldots, y_n)$ ($n = 40$ für parallele Verkettung, $n = 49$ für serielle Verkettung), welches dem Algorithmus als Eingabe dient, allerdings weiche Werte beinhaltet, müssen diese für die Berechnung des Syndroms zunächst hart entschieden werden.
Falls die Abbruchbedingung nicht erfüllt ist, wird zunächst ein Vektor $w \in \mathbb{R}^{k=24}$ bzw. $w \in \mathbb{R}^{k=33}$ für die serielle Verkettung berechnet. Dieser Vektor enthält für jede Kontrollgleichung das betragsmäßig kleinste Gewicht der Kanalkodefolge, welches in dieser Gleichung relevant

ist, ist also

$$w_i = \min_{\substack{j \in \{1,\ldots,n\} \\ H_{i,j}=1}} |y_j|$$

für jedes $i \in \{1, \ldots, k\}$. Diese Information dient dazu, nach der Auswertung über alle k, die Stelle zu flippen, welche in ihrem hard-decision-Wert am wenigsten sicher ist.
Der Gewichtsvektor $e = (e_1, \ldots, e_n)$ berechnet sich nun durch

$$e_j = \sum_{\substack{i=1 \in \{1,\ldots,k \\ H_{i,j}=1\}}} (2s_i - 1) \cdot w_i$$

für alle $j \in \{1, \ldots, n\}$. An den Positionen des Gewichtsvektors, an denen das Gewicht maximal ist, wird geflippt.

1.3 Dekodieren mittels Kontrollgleichungen

Beim Dekodieren mittels Kontrollgleichungen kann wiederum mit soft-Zuverlässigkeiten oder hart entschiedenen Werten gearbeitet werden. Wir untersuchten Dekodierung mittels hard-decision bei paralleler Kodeverkettung.
Durch die Kodeverkettung muss auch hier ein iteratives Verfahren angewandt werden, allerdings entscheidet sich bereits nach wenigen Iterationen, ob erfolgreich dekodiert werden kann, oder ob Dekodierungsversagen vorliegt.
Bei der parallelen Verkettung werden zunächst alle Spalten unabhängig voneinander mithilfe des (7,4,3)-Hammingkodes dekodiert. Die Änderungen werden in die Empfangsfolge eingetragen, anschließend werden die Zeilen unabhängig voneinander dekodiert. Ist das Syndrom nicht 0, wird der Vorgang mit der veränderten Efpmangsfolge wiederholt, die Änderungen wiederum eingetragen usw., bis entweder kein Fehler mehr vorliegt, oder eine bestimmte Anzahl von Iterationen erreicht ist. Im letzteren Fall liegt Dekodierungsversagen vor.

2 Programm

2.1 Überblick

Um die Leistungsfähigkeit der parallelen und seriellen Verkettung eines (7,4,3) HAMMING-Kodes zu vergleichen ist ein Java Programm entstanden, welches ein Eingabewort kodiert, dieses fehlerhaft überträgt und anschließend, durch ein iteratives LDPC-Dekodierungsverfahren (bit-flipping oder weighted bit-flipping) dekodiert. Dabei können alle drei Dekodierungsergebnisse (richtige Dekodierung, Falsche Dekodierung und Dekodierungsversagen) auftreten. Zur Bewertung der Leistungsfähigkeit wurden Messungen mit dem Programm vorgenommen.

2.2 Oberfläche

Die Oberfläche des Programms (Abbildung 5) besteht aus drei Teilen: dem Kodierer (1,2,3), dem Übertragungskanal (4,5,6) und dem Dekodierer (7,8,9). Elemente zur Anzeige (10) und zum neu starten des Programms (11) sind vorhanden.

2.3 Bedienung

Kodierer

In den Kodierer ist ein 16 Bit Eingabewort einzugeben, welches nur aus 0en und 1en besteht (1). Als nächstes ist die Wahl zwischen paralleler oder serieller Verkettung, zu treffen (2). Mit dem Button „Kodieren“ (3) wird das Eingabewort kodiert und im Ausgabewort (4) ausgegeben.

Übertragungskanal

Nach betätigen des Buttons „Übertragen“ (5) wird das Ausgabewort (4) nun mit einem Signal-Rauschabstand (SNR) oder einer bestimmten Anzahl von Fehlern übertragen und ausgegeben (6).

Dekodierer

Zunächst ist hier der iterative Dekodierungsalgorithmus festzulegen (7), dabei kann zwischen bit-flippling oder weighted bit-flipping gewählt werden. Danach wird festgelegt wie viele Iter-

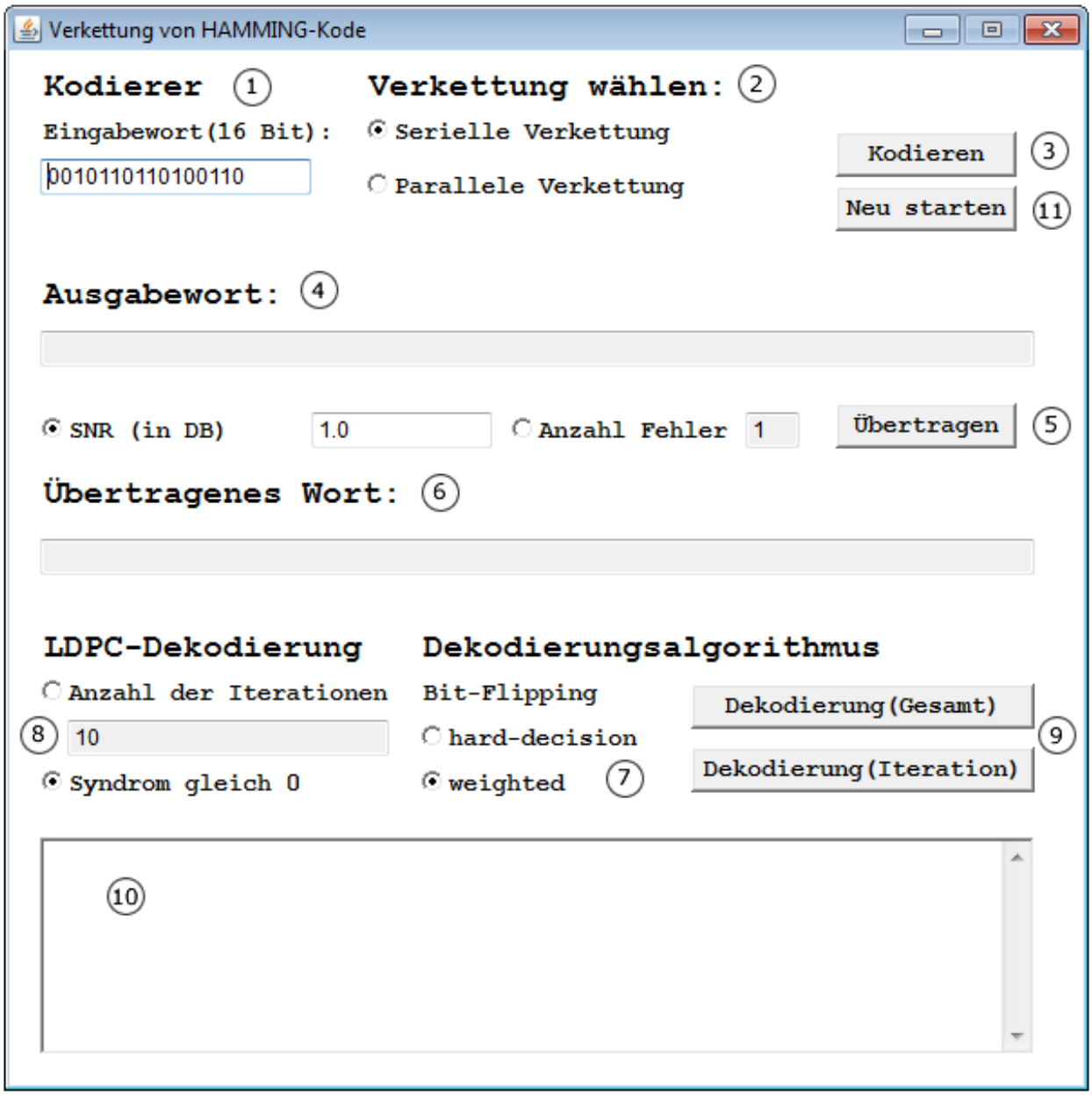

Abbildung 5: Programmoberfläche

ationen dekodiert (8) werden sollen, dabei stehen zur Auswahl die genaue Anzahl an Iterationen oder solange bis das Syndrom null ist. Mit dem Button „Dekodierung (Gesamt)“ wird die Dekodierung gestartet, solange bis das Syndrom gleich 0 oder die Anzahl der Iterationen erreicht ist. Da es passieren kann, dass das Syndrom nicht null wird und somit unendlich viele Iterationen zu durchlaufen wären, wird nach 500 Iterationen abgebrochen. Der Button „Dekodierung (Iteration)“ dekodiert immer nur eine Runde bis das gewählte Ereignis erreicht ist.

Ausgabe und Neu starten

Nach jeder Iteration wird das aktuelle Ausgabewort b, das Syndrom und der Gewichtsvektor e zur Anzeige (10) gebracht. Sobald das Syndrom gleich null ist wird die Dekodierung gestoppt und die Hintergrundfarbe ändert sich zu grün oder rot. Grün bedeutet, dass das übertragene Wort richtig dekodiert wurde und rot, dass das übertragene Wort falsch dekodiert wurde. Mit dem Button „Neu starten“ kann das Programm zurückgesetzt werden.

3 Auswertung

3.1 Messungen

Um die Leistungsfähigkeit paralleler und serieller Verkettung festzustellen wurden im Programm Messungen vorgenommen. Dazu wurde ein 16 Bit Eingabewort gewählt, welches dann kodiert wurde und anschließend 10000 mal übertragen und dekodiert wurde. Dabei wurden falsche Dekodierung und Dekodierungsversagen als „false“ und die richtige Dekodierung als „true“ gezählt. Bei der Übertragung mit SNR wird das kodierte Eingabewort mit einer normal verteilte Zufallszahl ($z_j = ZZ_{NV[0,1]} * \sigma$) addiert, dabei ist σ die Standartabweichung und wird folgendermaßen berechnet

$$\sigma = 1/\sqrt{2 * R * 10^{\frac{E_b}{N_0}/10}}$$

Es wurden folgende 7 Messungen durchgeführt:

- serielle Verkettung mit SNR (0.1-10.0) und bit-flipping
- parallele Verkettung mit SNR (0.1-10.0) und bit-flipping
- serielle Verkettung mit SNR (0.1-10.0) und weighted bit-flipping
- parallele Verkettung mit SNR (0.1-10.0) und weighted bit-flipping
- serielle Verkettung mit Anzahl Fehler (0-10) und bit-flipping
- parallele Verkettung mit Anzahl Fehler (0-10) und bit-flipping
- parallele Verkettung mit Anzahl Fehler (0-10) und Kontrollgleichungen

3.2 Messergebnisse

Vergleich zwischen paralleler und serieller Verkettung

In der ersten Gegenüberstellung wird als iteratives Dekodierungsverfahren bit-flipping verwendet und der Signal-Rausch Abstand (SNR) liegt im Bereich von 0.1 bis 10.0 . Dabei ist in Abbildung 6 zu erkennen, dass die serielle Verkettung gegenüber der parallelen Verkettung ein wenig besser ist.

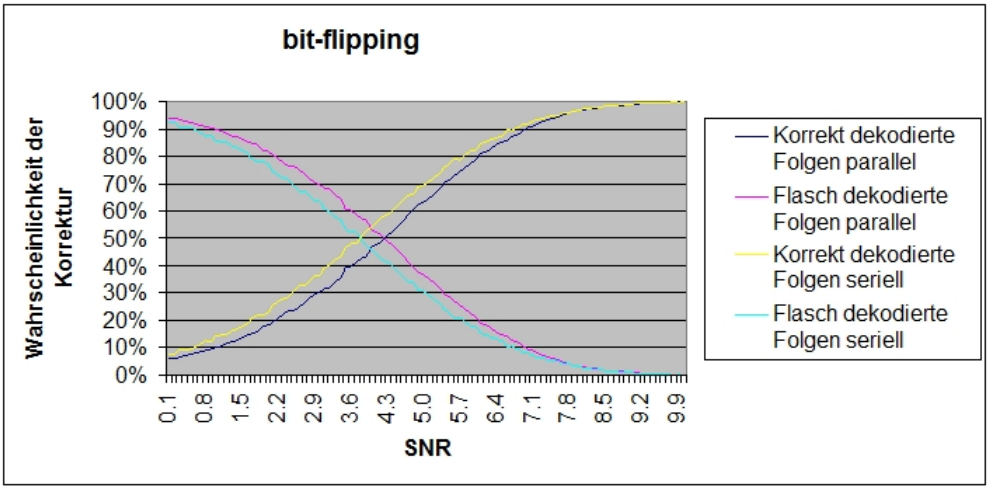

Abbildung 6: Vergleich zwischen paralleler und serieller Verkettung mit bit-flipping

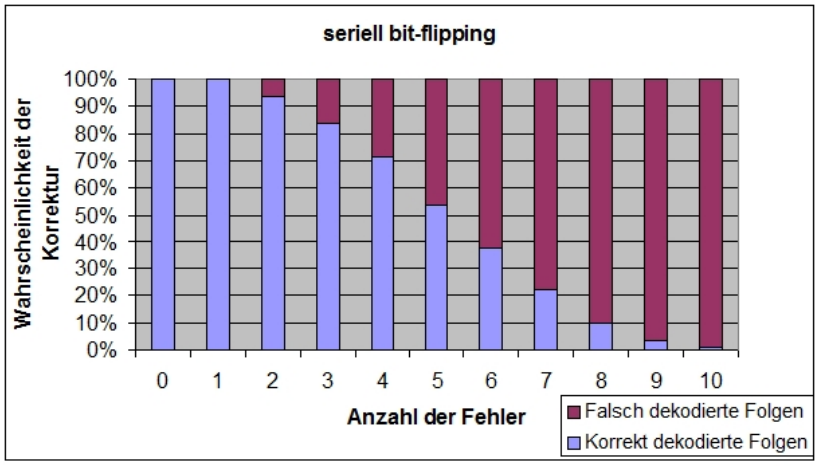

Abbildung 7: serielle Verkettung mit bit-flipping

Im zweiten Vergleich wird ebenfalls bit-flipping verwendet, aber die Anzahl der Fehler liegt jetzt im Bereich von 0 bis 10. Daraus lässt sich zwischen Abbildung 7 und Abbildung 8 noch eindeutiger erkennen, dass die serielle Verkettung wesentlich besser ist als die parallele Verkettung.

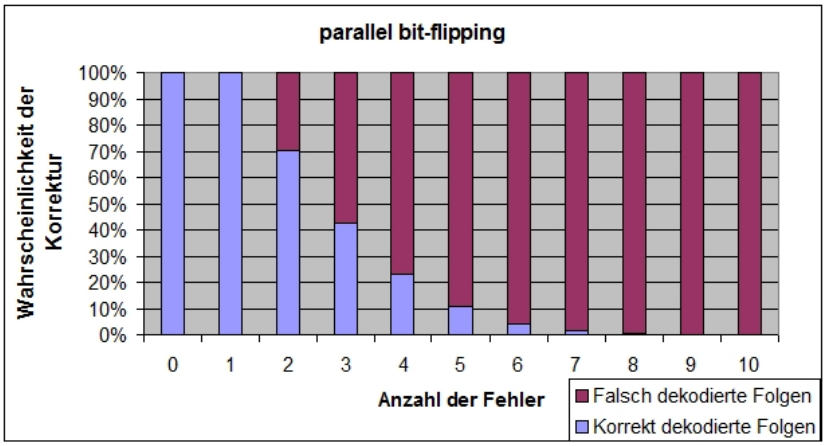

Abbildung 8: parallele Verkettung mit bit-flipping

Wird bei der parallelen Verkettung die iterative Dekodierung auf Basis von Kontrollgleichungen durchgeführt (Abbildung 9), so zeigt sich eine wesentliche Verbesserung der Ergebnisse gegenüber der parallelen und seriellen Verkettung mit bit-flipping. Im letzten Vergleich wird

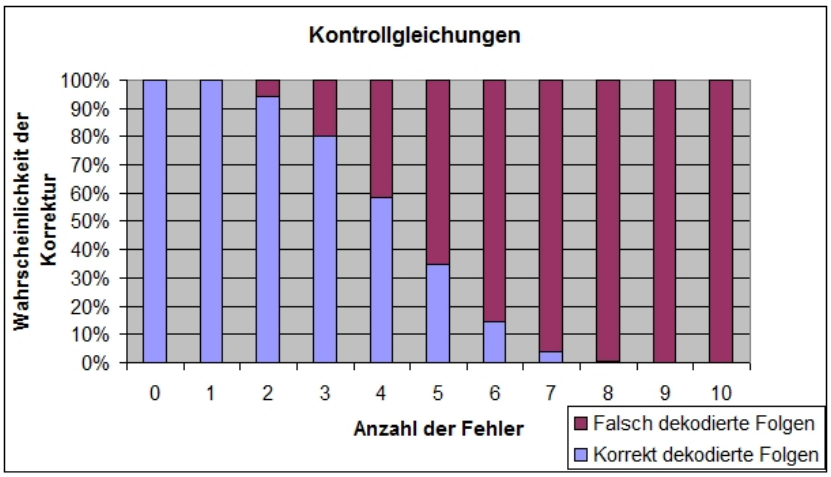

Abbildung 9: parallele Verkettung mit Kontrollgleichungen

als iteratives Dekodierungsverfahren weighted bit-flipping verwendet und der Signal-Rausch

Abstand (SNR) liegt im Bereich von 0.1 bis 10.0 . Aus der Abbildung 10 ist nun deutlich zu erkennen, dass die parallele Verkettung leistungsfähiger ist als die seriellele Verkettung.

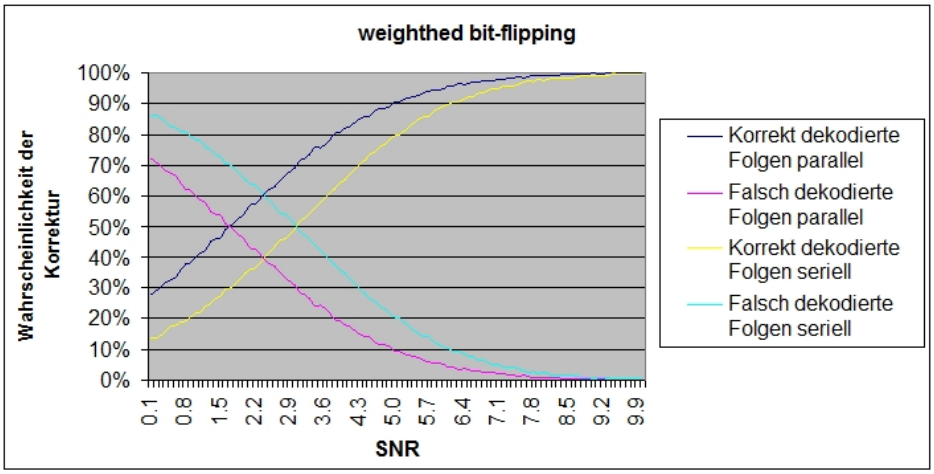

Abbildung 10: Vergleich zwischen paralleler und serieller Verkettung mit weighted bit-flipping

In Abbildung 11 wird nochmals deutlich, dass weighted bit-flipping in serieller und paralleler Verkettung eine höhere Leistungsfähigkeit besitzt als bit-flipping.

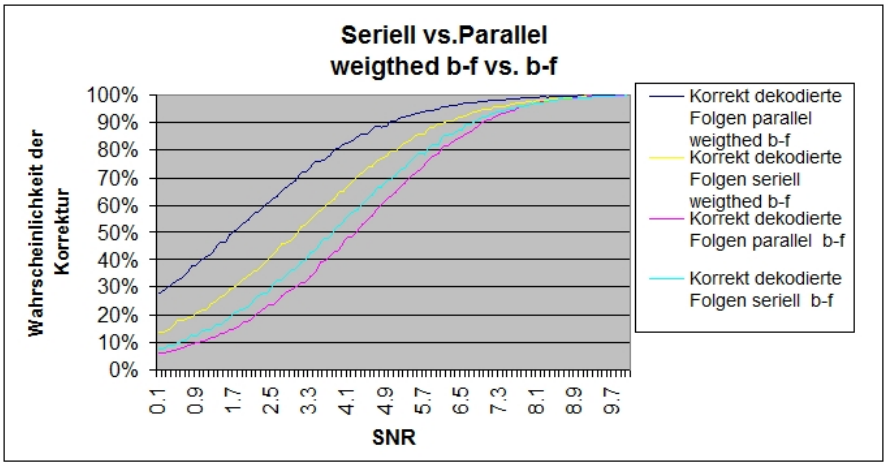

Abbildung 11: Vergleich zwischen serieller und parallelerVerkettung mit bit-flipping und weighted bit-flipping