Belegarbeit zur Quellenkodierungsmethode **"Run-Length Encoding"** im Modul "statistische Nachrichtentheorie"

Von: Monique Golnik (563075)
Dozent: Prof. Dr. Christoph Lange

17. Januar 2022

Inhaltsverzeichnis

Kapitel 1

Einleitung

In der heutigen Zeit ist es wichtiger den je sich mit der Komprimierung von Daten auseinander zu setzen. Auch wenn die Speicherkapazität einem das Gefühl der Grenzenlosigkeit vermittelt, entspricht das nicht der Realität - Kompressionsverfahren sind nicht mehr wegzudenken und gewinnen stetig an Bedeutung.

Gegenstand dieser Belegarbeit ist die nähere Betrachtung einer Quellcodierung, wobei **Run-Length Encoding** im Fokus steht.

Ziel ist es, dass ein Verständnis der Funktionsweise dieses Verfahrens und dessen Grenzen vermittelt werden.

Zunächst werden in Kapitel 2 grundlegende Begrifflichkeiten näher erläutert und es wird in die Thematik der Quellenkodierung und Datenkompression eingeführt. In Kapitel 3 wird das Verfahren Run-Length Encoding definiert, Vor- und Nachteile benannt sowie Anwendungsbeispiele dargestellt, um den Einsatz zu verdeutlichen

Kapitel 2

Quellenkodierung und Datenkompression

Bei der Quellkodierung ist es das Ziel, dass die Symbolentropie erhöht und so die Redundanz reduziert wird. Sie ist eine eineindeutige Darstellung der Quelleninformation in einer realisierbaren und möglichst redundanzfreien beziehungsweise -armen Form.

Die Aufgabe der Quellenkodierung ist die Kodierung eines Datensatzes, der von einer Informationsquelle abgegeben wurde, in einen Binärcode mit möglichst kleiner Stellenzahl. [Lan21, Seite 47 ff.]

„Bei der Datenkompression werden Dateien in eine alternative Darstellung überführt, die effizienter ist als die ursprüngliche. Ziel dieser Codierung ist es, sowohl den benötigten Speicherplatz als auch die Übertragungszeit zu verringern.“ [SE21]

Durch zwei unterschiedliche Ansätze, lässt sich so ein Codiergewinn erreichen:

- **Redundanz-Kompression**: Auf der Grundlage einer Redundanzreduktion lassen sich Daten nach der Kompression verlustfrei wieder dekomprimieren - eine solche Kompression ist nur möglich, wenn ein Datensatz sich wiederholende Zeichen beinhaltet
- **Irrelevanz-Kompression**: Bei diesem Ansatz werden irrelevante Informationen entfernt, um einen Datensatz zu komprimieren. Jedoch lässt sich dadurch der Datensatz nicht Bit-genau wiederhergestellten und ist somit verlustbehaftet

Im Folgenden wird der Blick auf die verlustfreie Kompressionsmethode **Run-Length Encoding** gerichtet und näher erläutert.

Kapitel 3

Run-Length Encoding

3.1 Redundanz-Kompression durch Run-Length Encoding

Die Lauflängencodierung [1] beschreibt eine verlustfreie Kompression, die sinnvoll angewandt werden kann, wenn der Datensatz Sequenzen mit sich mehrfach wiederholenden Zeichen enthält.

Die RLE ist den sogenannten **Phrasencodern** zugeordnet, wie in Abbildung 3.1 dargestellt.

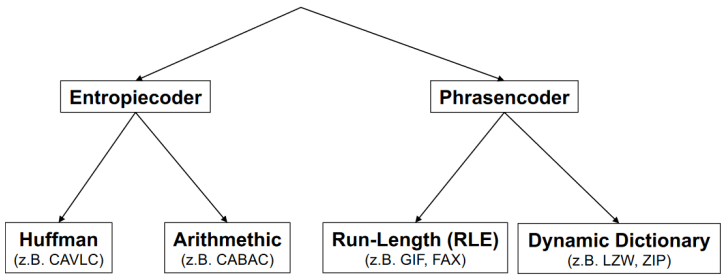

Abbildung 3.1: Klassifikation von verlustlosen Kompressionsmethoden [Bro21, Seite 8]

Bei dieser Art der Codierung wird eine Folge von gleichen Zeichen, Buchstaben oder Zahlen durch ein Symbol und die Angabe über die Häufigkeit der gleichen Symbole substituiert. Die Abfolge von identischen Zeichen und der Länge der jeweiligen Sequenz, die in einem sogenannten Run Counter gespeichert wird, wird als **Run** bezeichnet. Aufeinanderfolgende Vorkommen eines Datensatze $d = Run$. Anzahl der Wiederholungen n von $d = RunLength$

[1] (Run-Length Encoding (RLE), Run-Length Coding (RLC), Lauflängencodierung = Synonyme

(Lauflänge) Diese Form der Kompression eignet sich sehr gut um Redundanzen in Grafiken und Bilddateien mit wenigen Farben zu beseitigen.

Ist die zu komprimierende Grafik sehr detailreich, besitzt also sehr viele Farben oder Kontraste, ist die Lauflängencodierung nicht mehr geeignet, da so nur sehr selten gleiche Zeichen in einer Sequenz aufeinander folgen.[Gmb12] Auch für Texte eignet sich diese Methode eher nicht, da hier zwar Farben eine untergeordnete Rolle spielen, jedoch die aufeinander folgenden Buchstaben sehr selten gleich sind und so einzelne Zeichen nicht effizient zusammengefasst werden können.[Lan21, Seite 61]

Bei der Kompression werden identische Zeichen so lange eingelesen, bis sich einer ändert - Zeichen und Anzahl die gleich sind, werden festgehalten. Bei der Dekompression wird der festgehaltene Wert ausgelesen und die dem entsprechende Anzahl an Bits ausgegeben.[Gmb12]

Sendet die Quellen neben Buchstaben auch Ziffern, so muss das Verfahren um Sonderzeichen erweitert werden, so dass die resultierende Angabe weiterhin eindeutig als Lauflänge gekennzeichnet ist. [Lan21, Seite 62] Ein Beispiel zur Verdeutlichung findet sich in Abbildung 3.2.

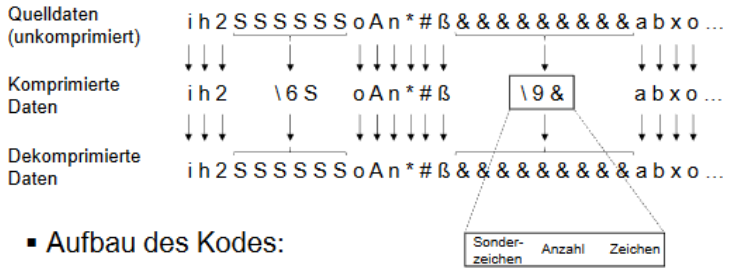

Abbildung 3.2: Beispiel Verfahrenserweiterung
[Lan21, Seite 62]

Der Kompressionsgrad des Run-Length Encodings hängt stark von der Charakteristik der Quelldaten ab. [Lan21, Seite 62]

3.2 Vor- und Nachteile gegenüber anderen Verfahren

Vorteile: Das RLE-Verfahren zeichnet sich durch seine Geschwindigkeit und Einfachheit aus. Zudem zeichnet es sich durch eine signifikante Datenkompression aus, wie im Kapitel **Einsatz & Anwendungsbeispiele** verdeutlicht wird.
Nachteile: Auch wenn RLE signifikante Vorteile bietet, ist es aber zur Kompression komplexer Farbbilder ungeeignet- auch wenn das Verfahren weiterentwickelt wurde und dadurch die Zunahme des Datenvolumens bei Reihen

ohne Wiederholungen kein Problem mehr darstellt, so sind die Kompressionsraten bei solchen Bildern doch denkbar schlecht. Der sinnvolle Einsatz von RLE beschränkt sich daher auf bestimmte Grafiktypen.[Koe06] Bei sehr kleinen Zeichenfolgen kann es passierten, dass sich die Daten sogar vergrößern, wie beispielsweise bei einem Schabrettmuster. Daher muss vorher eine Mindestanzahl von Werten die zusammengefügt werden gesetzt werden.

3.3 Einsatz & Anwendungsbeispiele

Die Lauflängencodierung findet Anwendung bei verschiedenen Grafikdateiformaten, wie beispielsweise dem TIFF[2], dem TGA- und dem Bitmap- Dateiformat.In diesem Kapitel werden Beispiele aus unterschiedlichen Bereichen dargestellt, um zu verdeutlichen, wo sich die Lauflängencodierung eignet und wie sie angewandt wird.

1. Einsatz in der Computergrafik
In diesem Bereich wird das RLE- Verfahren bei speicherintensiven Rastergrafiken angewendet und ist besonders effizient, wenn es sich um Grafiken mit wenigen aber großflächigen Farben handelt. Das Verfahren kann dabei sogar noch optimiert werden, in dem eine Wegoptimierung durch die Wahl des effektivsten Laufweges anhand eines Algorithmus vorgenommen wird. Dieser Laufweg könnte beispielsweise zeilen-sequenziell, im Zick-Zack oder mäanderförmig[3] siehe Abbildung 3.3 sein.

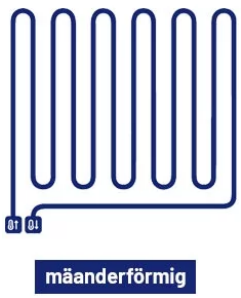

Abbildung 3.3: Wegoptimierung: Mäanderförmig
[hei]

Grafik mit 48 Bytes: Die Lauflängencodierung lässt sich auf sogenannte Rastergrafiken[4] übertragen, in dem die RGB-Werte genutzt werden.

- **zwei Beispielabfolgen**: siehe unter RLE-Code (rechts) in der Grafik 3.4

[2]TIFF = Tagged Image File Format
[3]Mäander = Bezeichnung einer Flussschlinge in einer Abfolge weiterer Flussschlingen
[4]Rastergrafik = Pixelgrafik

- **RGB-Farbwerte:** Diese stellen die jeweiligen Farben dar

  Sind viele gleichfarbige Pixel (hier als Kästchen dargestellt) aufeinander folgend, können diese gut zusammengefasst werden, in dem die Anzahl vor dem Farbwert übertragen wird.

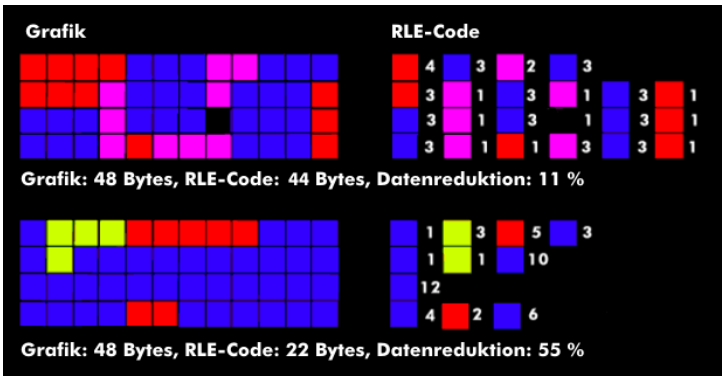

Abbildung 3.4: Beispiel mit geringer und höherer Effizienz [Gmb12]

In der Grafik 3.4 wird auch schnell deutlich, dass ein sehr detailreiches Bild ungeeignet für die RLE-Codierung ist, da kaum gleichfarbige Pixel aufeinander folgen. Die Datenreduktion beträgt im oberen Grafikbeispiel gerade einmal 11%. In der unteren, in der sich sehr viele nebeneinander liegende blaue Pixel befinden, beträgt die Datenreduktion effiziente 55%.

2.Abfolge bestimmter Nukleinbasen (U,G,C,A) [Juc21]
Diese Art der Kompression lässt sich nicht nur auf Grafiken anwenden sondern auch auf Zeichenketten, wie beispielsweise die Abfolge bestimmter Nukleinbasen. Hierbei wird anstatt der RGB-Farbwert die ASCII- Zeichen genutzt.

- **Beispielabfolge**: **A G G U A** | C G ..
- **ASCII-Code**: 01000001 01000111 01000111 01010101 01000001

  Es wäre nun möglich eine binäre Codierung vorzunehmen, da in diesem Beispiel nur vier Zeichen (Buchstaben) auftauchen, würden 2 Bit für eine eindeutige Zuordnung ausreichen:

Zahl	Binär	Zeichen
0	00	A
1	01	C
2	10	G
3	11	U

Tabelle 3.1: Binärcodierungstabelle für Beispiel-Zeichen

Dadurch würde sich der betrachtete Code folgendermaßen kürzen: 00 10 10 11 00. Die Kompressionsrate ist bereits hier merklich gesunken, wurden zuvor noch 8 Bit benötigt und jetzt nur 2, was bedeutet, dass der benötigte Speicherplatz auf 25% komprimiert werden konnte. Jedoch darf hierbei nicht vergessen werden, dass die Zuordnungstabelle bisher nur dem Sender bekannt ist, diese muss für eine Dekompression jedoch auch dem Empfänger bekannt sein - Daher muss die Tabelle beispielsweise im Dateikopf mitübertragen werden. Das entspricht daher einen tatsächlichen Bedarf von $4 * (2 + 8)$ Bit $= 40$ Bit allein für die Tabelle - bis hier würde sich bei dieser Übertragung wider erwartend eher ein Nachteil ergeben (Bei der Kürze der hier genutzen Zeichen).

- **RLE-Methode**: **1**A **2**G **1**U **1**A

  Es wird deutlich, dass sich der zu übertragende Code stark verkürzt hat, in dem die jeweilige Anzahl der aufeinanderfolgenden Buchstaben angegeben wird. Bei der Übertragung wird zuerst die Anzahl und dann das dazugehörige Zeichen gesendet. Auf den ersten Blick wüsste man jedoch nicht, ob es sich gerade um eine Zahl oder einen Buchstaben handelt. **Merke:** Ob beispielsweise eine 01 als C oder als 2 interpretiert wird, wird durch die Position der Zeichenkette festgelegt.

Code	Zeichen	Code	Zahl
00	A	00	1
01	C	01	2
10	G	10	3
11	U	11	4

Tabelle 3.2: RLE-Codiertabelle für Beispiel-Zeichen

Der Speicherbedarf reduziert sich auf 32 Bit + Dateikopf

- **RLE-Methode (verbessert)**:**1**A **2**G U A

  Die Einsen vor den Buchstaben wurden entfernt, da diese auch keine weitere Ausdruckskraft hätten. Des Weiteren bietet sich die Idee an,

eine gemeinsame Codiertabelle mit 3 Bit zu erstellen, um Verwechslungen gänzlich zu vermeiden:

Binär	Zeichen
000	A
001	C
010	G
011	U
100	2
101	3
110	4

Tabelle 3.3: gemeinsame Codiertabelle

Dadurch verringert sich die Größe des Dateikopfes, da die Tabelle nun kleiner (zusammengefasst) ist: $6 * 3$ Bit $= 18$ Bit + Dateikopf.

Kapitel 4

Fazit

An den aufgeführten Anwendungsbeispielen wird deutlich, dass dieses Codierverfahren nicht perfekt ist, jedoch bereits einen ursprünglich hohen Speicherplatzbedarf deutlich komprimieren kann.

Derzeit ist noch kein perfektes Codierverfahren bekannt - aber vielleicht kommt dieses noch!

Kapitel 5

Eigenständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Belegarbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel verfasst habe. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt.

Berlin, den 17. Januar 2022

Monique Golnik

Abbildungsverzeichnis

Tabellenverzeichnis

Onlinequellen

[Bro21] Benjamin Bross. *Bildkompression*. deutsch. Frauenhofer Heinrich-Hertz Institute. Mai 2021. URL: https://moodle.htw-berlin.de/pluginfile.php/1216458/mod_resource/content/2/MMK_SoSe2021_3_Bildkompression.pdf.

[Gmb12] DATACOM Buchverlag GmbH. „Lauflängencodierung“. In: (Okt. 2012). URL: https://www.itwissen.info/RLE-run-length-encoding-Lauflaengencodierung.html.

[Juc21] Frank Juchim. *Kompression und Lauflängencodierung - Codierung und Datenübertragung*. Apr. 2021. URL: https://www.youtube.com/watch?v=DRH1eAd1Psk.

[Koe06] Mirjam Koeck. *RLE - Runlength-Encoding*. Deutsch. Digital Media for Artists. Apr. 2006. URL: http://www.dma.ufg.ac.at/app/link/Grundlagen:2D-Grafik/module/12946?step=all.

[Lan21] Prof. Dr. Christoph Lange. *Einführung in die Informations- und-Kodierungstheorie I*. deutsch. HTW. Nov. 2021. URL: https://moodle.htw-berlin.de/pluginfile.php/1361027/mod_resource/content/1/SNT_HTW_Lange_WS_2021_22_04_InfKodTheorie_1.pdf.

[SE21] IONOS SE. „Datenreduktion durch Deduplikation und Kompression“. In: *Digital Guide IONOS by 11* (Jan. 2021). URL: https://www.ionos.de/digitalguide/server/knowhow/datenreduktion-durch-deduplikation-und-kompression/.

Bildquellen

[hei] kocer heiztech. „Funktionsweise einer Fußbodenheizung“. In: (). URL: https://www.kocer-heiztech.at/blog/fussbodenheizungen/.