Goethe-Universität Frankfurt am Main

Fachbereich 12 / Informatik

Aktuelle Themen der Angewandten Informatik:

Datenkompression

Lempel-Ziv-Algorithmus

Betreuung:

Dr.-Ing. The Anh Vuong

Autoren:

Boui Bilal (6985624)

EL Masiani Mimoun (6859378)

Melissopoulos Loukas (7020025)

Abgabedatum:

28.02.2021

Inhaltsverzeichnis

1 Theoretische Grundlagen der Datenkompression 3
2 Biographien 4
2.1 Abraham Lempel 4
2.2 Jacob Ziv 4
2.3 Terry Welch 5
3 Verfahrensbeschreibung 6
3.1 Lempel-Ziv-77 6
3.2 Lempel-Ziv-78 6
3.3 Präsentation/Demonstration Kit 6

4 Lempel-Ziv-Welch-Algorithmus 8
4.1 Funktionsweise 8
4.2 Präsentation/Demonstration Kit 8

5 Qualitätsbewertung 9
6 Erklärung: Selbstständige Verfassung der Seminararbeit 10
7 Referenzen 11

1 Theoretische Grundlagen der Datenkompression

Im Allgemeinen beschreibt die Datenkompression (auch Datenkomprimierung genannt) die Verdichtung oder auch Reduzierung von einer Anzahl von Daten.

Bei der Übertragung oder auch Speicherung von Informationen soll der Aufwand der Technik möglichst gering sein. Hierbei sollen die Anzahl der Bits, die pro Zeiteinheit übertragen werden, von Informationen gering codiert sein.

Die Datenkompression unterscheidet zwischen zwei Codierungsformen:

1. Verlustbehaftete Codierung

Im Fall der verlustbehaften Codierung werden Nachrichten beziehungsweise Daten nicht zu 100 Prozent verlässlich übertragen. Folglich enthält der Empfänger unter Umständen nicht Informationen, die jedoch von Wichtigkeit sind. Diese Art von Informationen, Signalen etc. sind nicht mehr verfolgbar. Dementsprechend ist eine Dekonstruktion nicht mehr möglich.

In diesem Zusammenhang illustrieren wir ein kleines Beispiel:

Der Satz „Der Ball ist gr“ stellt sinnbildlich die verlustbehaftete Codierung dar. Der Empfänger ist nicht in der Lage zu entziffern, welche Information ihm mittgeteilt werden soll. Unterdies gibt es verschiedene Attribute, die man dem Ball zuschreiben könnte. Dazu gehören: grün, groß, grau, etc.

2. Verlustfreie Codierung

Im Fall der verlustfreien Codierung werden Nachrichten beziehungsweise Daten komplett komprimiert übertragen, ohne das Informationen verloren gehen. Redundante Signale beinhalten ohne Informationsverlust wegzulassende Informationen. Folglich erhalten wir eine Vielzahl von Informationen, die oft vorkommt und durch ein bestimmtes Zeichen ergänzt wird. Zu verlustfreie Codierungsalgorithmen gehören:

1. Lempel-Ziv-Verfahren (wo auch unser Fokus liegt)
2. Lempel-Ziv-Welch-Verfahren
3. Huffmann-Codierung

2 Biographien

In diesem Abschnitt erhalten Sie Informationen zu den Lempel-Ziv-Algorithmus-Entwicklern.

2.1 Abraham Lempel

Abraham Lempel ist am 10. Februar 1936 in Lemberg (Polen) geboren. Er ist ein polnisch-israelischer Informatiker.

Lempel studierte an der technischen Universität in Haifa Informatik. Im Jahr 1963 machte er seinen Bachelor und 1965 sein Master. Darüber hinaus erhielt er im Jahr 1965 seinen Doktorgrad. Später ging er in die Vereinigten Staaten von Amerika, um dort als wissenschaftlicher Mitarbeiter tätig zu sein. Ab 1971 arbeitete er als Professor an der technischen Universität in Haifa.

Im Mai 1977 wurde in der wissenschaftlichen Zeitschrift „IEEE Transactions on Information Theory“ der Codierungsalgorithmus „Lempel-Ziv-77-Algorithmus“ (LZ77) präsentiert. In den Jahren darauf wurden immer mehr optimierte Algorithmen des Lempel-Ziv-Verfahrens veröffentlicht. Eine preisliche Anerkennung erhielt er 1997 mit dem „Paris-Kanellakis-Preis“.

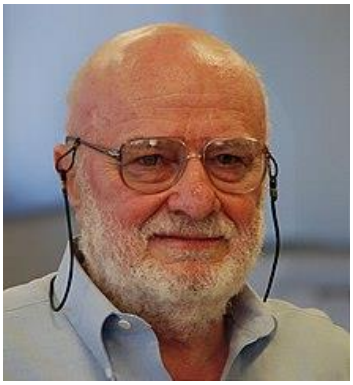


Bild von Abraham Lempel

Quelle des Bildes:

https://de.wikipedia.org/wiki/Abraham_Lempel#/media/Datei:Abraham_Lempel.JPG

2.2 Jacob Ziv

Jacob Ziv ist am 27. Januar 1931 in Tiberias (Palästina) geboren. Er ist ein israelischer Elektrotechniker.

Ziv besuchte wie Lempel an der technischen Hochschule in Haifa. Dort studierte er Elektrotechnik. Im Jahr 1965 promovierte er an der „Massachusetts Institute of Technology“. Im Jahr 1970 kehrte er an die technische Hochschule in Haifa zurück, wo er als Professor tätig war. Im Mai 1977 wurde in der wissenschaftlichen Zeitschrift „IEEE Transactions on Information Theory“ der Codierungsalgorithmus „Lempel-Ziv-77-Algorithmus (LZ77) präsentiert. Er gewann zahlreiche Auszeichnungen. Dazu gehören.

- Paris-Kanellakis-Preis (1997)
- Claude E. Shannon Award (1997)
- BBVA Foundation Frontiers of Knowledge Award (2008)

In diesem Jahr (2021) erhielt er die „IEEE Medal of Honor“.

Bild von Jacob Ziv

Quelle des Bildes:

https://upload.wikimedia.org/wikipedia/commons/thumb/d/d0/Jacob_Ziv.jpeg/330px-Jacob_Ziv.jpeg

2.3 Terry Welch

Terry Welch ist im Jahr 1939 geboren und am 22. November 1988 in Travis County (USA) verstorben. Welch war ein US-amerikanischer Informatiker.

Am „Massachusetts Institute of Technology“ studierte er Elektrotechnik und promovierte dort. Nach dem Studium lehrte er an der „University of Texas at Austin“ beziehungsweise ab 1976 war er Manager im Forschungszentrum der „Sperry Corporation“ tätig.

Welch entwickelte (basierend auf Arbeiten von Jacob Ziv und Abraham Lempel) den weltweit benutzten LZW-Algorithmus. (Lempel-Ziv-Welch-Algorithmus) Im Jahr 1983 meldete er für seinen Arbeitgeber US-Patente beziehungsweise im Jahr 1984 internationale Patente. Danach veröffentlichte er seinen Algorithmus.

Im Jahr 1988 starb Welch an einem Hirntumor.

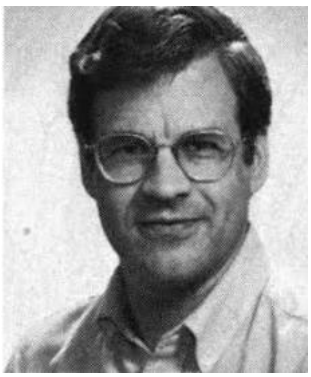


Bild von Terry Welch

Quelle des Bildes:

https://openpreservation.org/blogs/compression-at-your-discretion/

3 Verfahrensbeschreibung

3.1 Lempel-Ziv-77

Der Lempel-Ziv-77-Algorithmus ist das Komprimierungsverfahren, das Abraham Lempel und Jacob Ziv. Dieses Verfahren, wie eingangs in beiden Biographien erwähnt, ist im Artikel „A Universal Algorithm for Sequential Data Compression“ in der „IEE Transactions of Information Theory“ im Jahr 1977 veröffentlicht worden.

Dieses Verfahren hat einen sehr hohen Stellenwert, da dies das erste Kompressionsverfahren ist, welches sich mit einer Tabelle steuern lässt. Hierbei existiert ein Zeiger, welches den Eingabetext entlangläuft und neue Elemente in das Wörterbuch hinzufügt.

In diesem Zusammenhang entscheiden wir zwischen drei Codierungsformen:

1. Wir besitzen eine Übereinstimmung mit dem Wörterbuch.
2. Wir besitzen keine Übereinstimmung mit dem Wörterbuch.
3. Wir besitzen eine Übereinstimmung mit dem Wörterbuch. Jedoch geht es über die Länge des Wörterbuches hinaus.

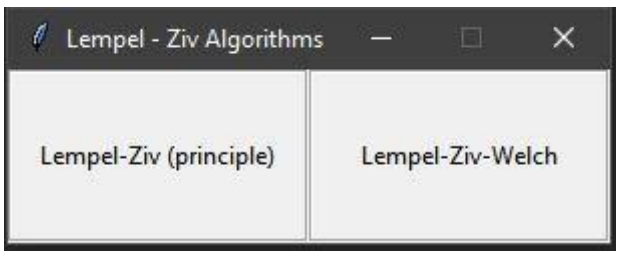
3.2 Lempel-Ziv-78

Der Lempel-Ziv-78-Algorithmus ist eine modifizierte Version des Lempel-Ziv-77-Algorithmus, welches ebenfalls tabellengesteuert ist. Dieser ist im Artikel „Compression of Individual Sequences via Variable-Rate Coding“ in „IEEE Transactions on Information Theory“ vorgestellt worden. Der Unterschied hierbei ist nun, dass die Tabelle des Eingabetextes anfangs leer ist und dann Teile hinzugefügt werden.

In diesem speziellen Wörterbuch haben wir bis zu 4096 Einträgen beziehungsweise eine festgelegte Codelänge von 12 Bit. Erwähnenswert ist die Tatsache, dass die Übertagung von Einzelzeichen verhindert wird, da diese möglicherweise bereits vorher in das Wörterbuch hinzugefügt wurde.

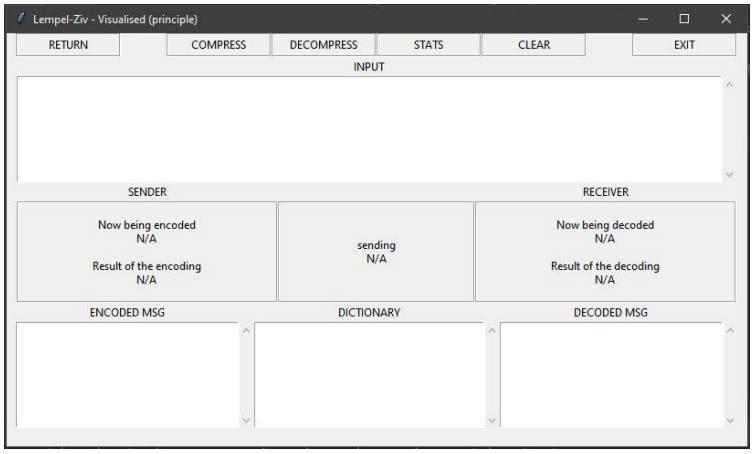
3.3 Präsentation/Demonstration Kit

Die folgenden Bildschirmfotographien verdeutlichen den Frontend des Lempel-Ziv-Algorithmus. (Grundprinzip)



Auf dieser Bildschirmfotographie sieht man das Auswahlmenü unseres Programmes.

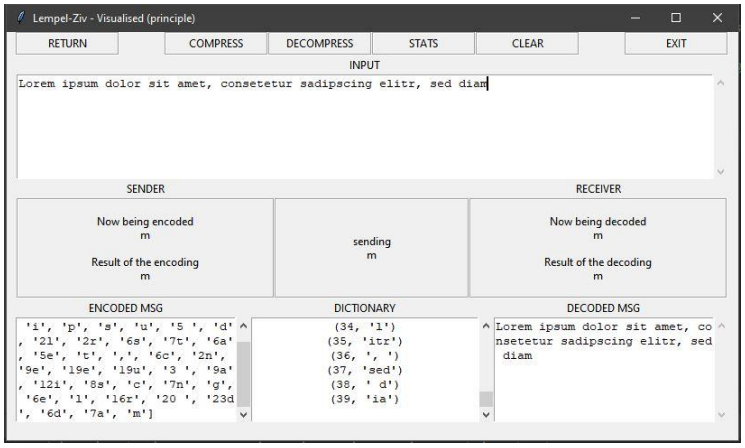
Es ist erkennbar, dass das Prinzip des LZ und das LZW implementiert wurde.



Auf dieser Bildschirmfotographie sieht man die „Arbeitsfläche“ des Lempel-Ziv-Algorithmus.

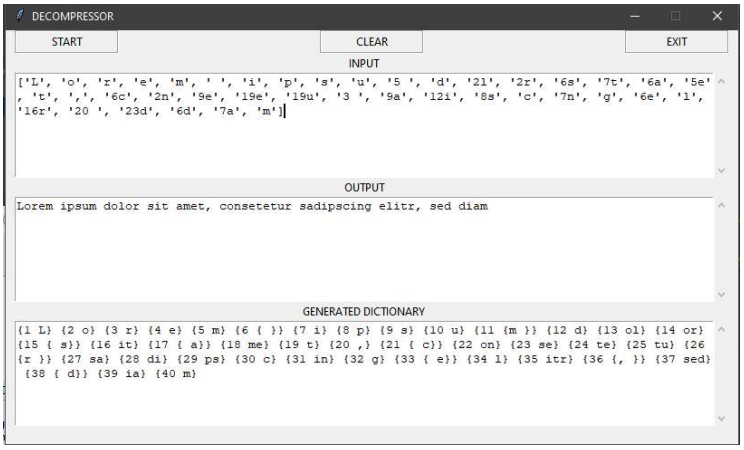
Im Feld „Input“ ist der Nutzer in der Lage, eine individuellen Eingabetext zu kreieren.

Anschließend wir der Eingabetext encodiert und decodiert. Zwischenzeitlich entsteht auch ein individuelles Wörterbuch.



Diese Bildschirmfotographie illustriert die Verwendungsweise des der eingangs erwähnten „Arbeitsfläche“.

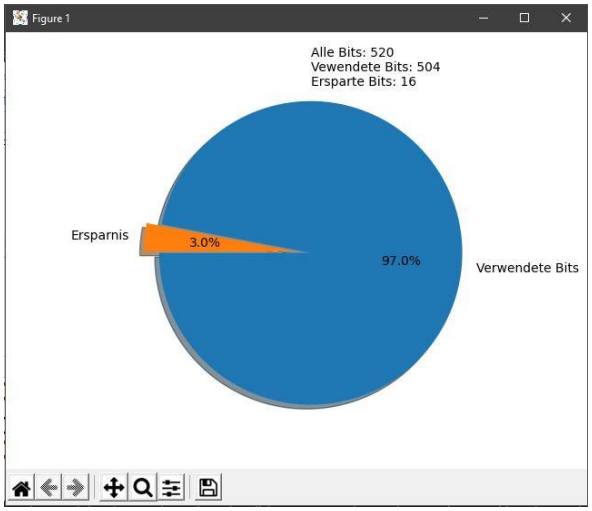
Der Nutzer gibt in im Feld „Input“ eine individuelle Zeichenkette ein und wird in den unteren Feldern bearbeitet.



Diese Bildschirmfotographie illustriert die Dekompression im Grundprinzip des Lempel-Ziv-Algorithmus.

Im Feld „Input“ wird der komprimierte String eingegeben und erhalten im Feld „Output“ unseren Ausgangstext zurück.

Im Feld „Generated Dictionary“ wird für unser Eingabetext ein individuelles Wörterbuch erstellt.



Diese Bildschirmfotographie illustriert den tatsächlichen Nutzen des Lempel-Ziv-Algorithmus.

Man erkennt, wie viele Bits der Eingabetext einnimmt beziehungsweise, wie viele Bits nach der Kompression verwendet worden ist.

In einem Kreisdiagramm sieht man einmal mehr die Ersparnis in Prozent.

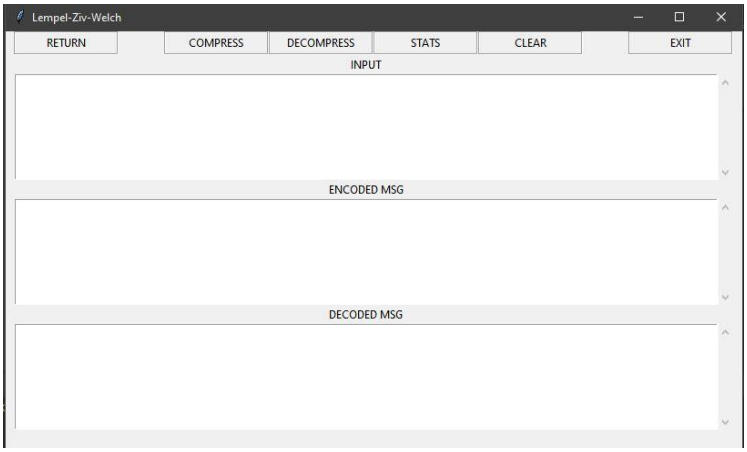
4 Lempel-Ziv-Welch-Algorithmus

4.1 Funktionsweise

Der Lempel-Ziv-Welch-Algorithmus (LZW) ist eine modifizierte Version des LZ78. Der Algorithmus lässt sich visuell ebenfalls als Wörterbuch beschreiben. Hierbei haben wir ein Wörterbuch von bis zu 4096 Einträgen beziehungsweise eine feste Codelänge von 12 Bit. Der Lempel-Ziv-Welch-Algorithmus hat wie der Lempel-Ziv Algorithmus einen Kompressionsvorgang und einen Dekompressionsvorgang. Erwähnenswert ist hierbei, dass die ersten 256 Einträge mit allen möglichen Zeichen belegt sind.

Grund hierbei ist die Tatsache, dass die Übertragung von Einzelzeichen verhindert wird, da diese möglicherweise bereits vorher in das Wörterbuch hinzugefügt wurde. Dadurch haben wir eine optimierte Laufzeit, da aus dem Input Muster erkannt werden, wodurch nicht das komplette Muster übertragen wird, sondern nur dessen Index.

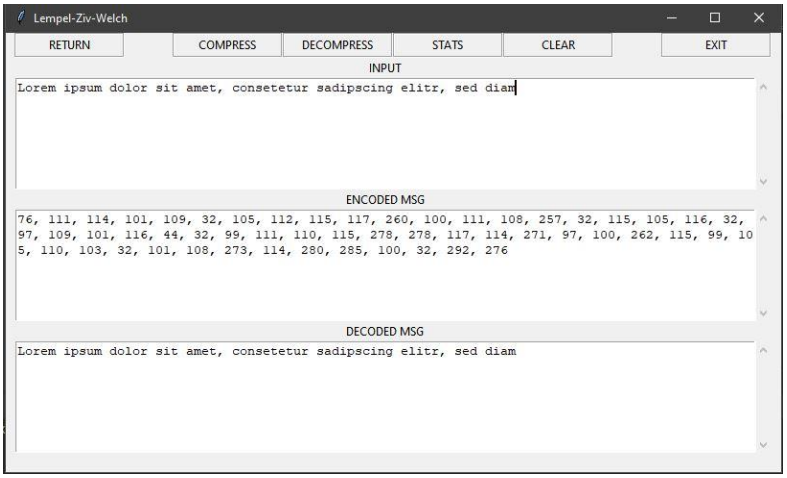
4.2 Präsentation/Demonstration Kit



Auf dieser Bildschirmfotographie sieht man die „Arbeitsfläche“ des Lempel-Ziv-Welch-Algorithmus.

Im Feld „Input“ ist der Nutzer in der Lage, eine individuellen Eingabetext zu kreieren.

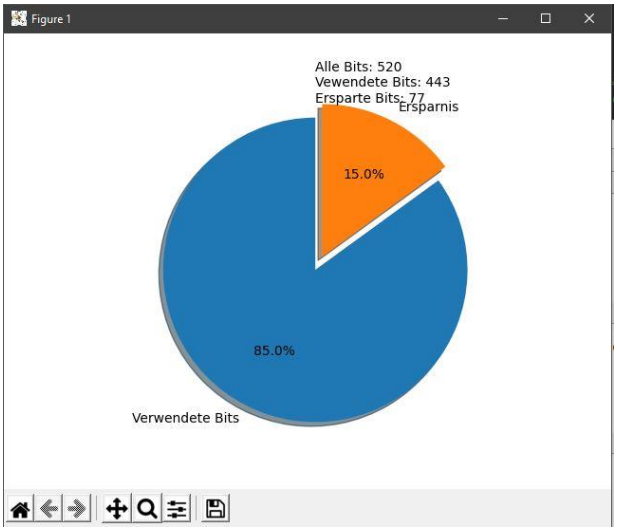
Anschließend wir der Eingabetext encodiert und decodiert.



Diese Bildschirmfotographie illustriert das Verfahren des Lempel-Ziv-Welch-Algorithmus.

Im Feld „Input“ gibt der Nutzer seinen individuellen Eingabetext ein.

Anschließend wird der Eingabetext encodiert und dekodiert. Man erkennt, dass die encodierte Nachricht der Korrektheit entspricht, da die dekodierte Nachricht mit der des Eingabetextes übereinstimmt.



Diese Bildschirmfotographie illustriert den tatsächlichen Nutzen des Lempel-Ziv-Welch-Algorithmus.

Man erkennt, wie viele Bits der Eingabetext einnimmt beziehungsweise, wie viele Bits nach der Kompression verwendet worden ist.

In einem Kreisdiagramm sieht man einmal mehr die Ersparnis in Prozent.

5 Qualitätsbewertung

In der Informatik ist der Lempel-Ziv-(Welch)-Algorithmus weitestgehend interessant, wenn man sich die Laufzeit näher betrachtet. Im Allgemeinen ist der Lempel-Ziv-(Welch)-Algorithmus nachweislich sehr effizient, da ihre Laufzeit asymptotisch optimal sind.

Ihre relative Redundanz ist bei $\lim_{n\to\infty} = 0$. (n = Länge des Strings)

Die relative Redundanz beschreibt die die Wiederholung eines Elementes im Eingabetexte/String. Das bedeutet: Wenn ein Eingabetext gegen unendlich viele Zeichen geht, so ist bei der Kompression die Anzahl der Wiederholung eines bestimmten Zeichens gleich null.

Alles in Allem lässt sich schlussfolgern, dass es sich bei den Algorithmen um gute Verfahren handelt, die ihre Aufgabe als Datenkompressoren gewissenhaft erfüllen. Dabei bieten der Lempel-Ziv-77 und der Lempel-Ziv-78 eine gute Basis der Datenkompression. Jedoch sind diese nicht als optimaler Datenkompressionsalgorithmus zu betrachten. An den obigen Kreisdiagrammen ist die prozentuale Ersparnis bei der Kompression mithilfe des Lempel-Ziv-Welch-Algorithmus deutlich höher. (im Vergleich zum Grundprinzip des Lempel-Ziv-Algorithmus, worauf sich LZ77 und LZ78 basieren) Selbstverständlich hat das Grundprinzip des Lempel-Ziv-Algorithmus den Vorteil, dass ein Eingabetext ohne Vorkenntnisse komprimiert werden kann. Jedoch findet bei Eingabetexte mit wenig Länge wenig bis keine Kompression statt. Folglich erhalten wir bezüglich der Größe an Datenmenge entweder keine Reduzierung oder eine Zunahme.

In diesem Zusammenhang gewinnt der Lempel-Ziv-Welch-Algorithmus immer mehr an Popularität. Heutzutage basieren vielerlei Formate auf den LZW. Dazu gehören Formate wie ARC, GIF, ZIP. In den modifizierten Algorithmen ist stets eine Verbesserung zu sehen, da sie jede Art von Zeichen akzeptieren und diese komprimieren können.

6 Erklärung: Selbstständige Verfassung der Seminararbeit

Hiermit erklären wir, dass wir die vorliegende Hausarbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt haben.

Die Stellen der Hausarbeit, die anderen Quellen im Wortlaut oder dem Sinn nach entnommen wurden, sind durch Angaben der Herkunft kenntlich gemacht. Dies gilt auch für Zeichnungen, Skizzen, bildliche Darstellungen sowie für Quellen aus dem Internet.

Boui Bilal	Melissopoulus Loukas	EL Masiani Mimoun

7 Referenzen

- https://de.wikipedia.org/wiki/LZ77

- https://www.tu-chemnitz.de/informatik/ThIS/downloads/courses/ws02/datkom/LZ-Codierung.pdf

- https://www.youtube.com/watch?v=RV5aUr8sZD0

- https://de.wikipedia.org/wiki/Abraham_Lempelhttps://upload.wikimedia.org/wikipedia/commons/thumb/d/d0/Jacob_Ziv.jpeg/330px-Jacob_Ziv.jpeg

- http://www.gm.fh-koeln.de/~hk/lehre/ala/ws0506/Praktikum/Projekt/D_rot/Ausarbeitung.pdf

- https://de.wikipedia.org/wiki/Terry_Welch

- https://rosettacode.org/wiki/LZW_compression