Technische Hochschule Rosenheim

# Grundlagen der Informatik

Prof. Dr. J. Schmidt
Fakultät für Informatik

GDI – WS 2020/21

Information und Quellencodierung
Arithmetische Codierung

# Verfahren zur Datenkompression (1)



- Mittlerweile:
  Vielzahl von Verfahren zur Datenkomprimierung

- Aufteilung in zwei Gruppen
  - Verlustfreie Datenkompression
  - Verlustbehaftete Datenkompression

# Verlustfreie Datenkompression

- Ziel der Codierung
  - Redundanz-Reduktion möglichst auf Null (Übertragungszeit und Speicherplatz ↓)
- Wesentliche Forderung
  - Die in den Daten enthaltene Information bleibt ohne Änderung erhalten
  - D.h. decodierte Daten unterscheiden sich nicht von den Originaldaten
- Zum Beispiel
  - Codierung von Texten und Tabellen

# Verfahren zur Datenkompression (3)



## Verlustbehaftete Datenkompression

- Ziel der Codierung
  - Über verlustfreie Datenkompression hinausgehende Verringerung der Datenmenge
- Information bleibt im Wesentlichen erhalten, aber gewisser Informationsverlust akzeptiert
  - D.h. Teil der Information geht verloren
  - somit unterscheiden sich die decodierten Daten von den Originaldaten
- (wesentlich) höhere Kompressionsraten lassen sich erzielen
- Zum Beispiel
  - Standbilder, Audio- oder Videodateien (wahrnehmungspsychologische Eigenschaften der Augen/Ohren werden berücksichtigt)

# Bereits bekanntes Kompressionsverfahren

- Huffman-Codierung
  - Durch den Huffman-Algorithmus wird ein Code-Baum iterativ von unten nach oben aufgebaut
  - Ergebnis: Code mit variabler Wortlänge
- Datenkompression im Vergleich zu Block-Codes aufgrund der Redundanzminimierung
  - Maß für die Datenkompression: Vergleich der mittleren Wortlänge des Huffman-Codes mit der konstanten Wortlänge des Block-Codes
  - berücksichtigt nur einzelne Zeichen, aber z.B. keine Wiederholungen innerhalb eines Wortes

## Alternative Verfahren

- Arithmetische Codierung
- Lauflängen-Codierung
- LZW-Algorithmus

# Arithmetische Codierung (1)



- Prinzip
  - Dem gesamten Quelltext wird eine Gleitpunktzahl x im Intervall $0 \leq x < 1$ zugeordnet
- Informationsgehalt
  - Einzelzeichen können implizit auch einen nicht-ganzzahligen Informationsgehalt tragen
    - Bei Huffman-Code erhält jedes Zeichen des Quelltextes ein Code-Wort mit ganzzahliger Länge
  - Arithmetische Codierung kann Redundanz meist noch etwas weiter verringern

# Arithmetische Codierung (2)



- Prinzip
  - Dem gesamten Quelltext wird eine Gleitpunktzahl x im Intervall $0 \leq x < 1$ zugeordnet
- Beispiel

| Quelltext | | Codierung |
|---|---|---|
| ESSEN | ➔ | 0.24704 |

| Codierung | | Decodierung |
|---|---|---|
| 0.24704 | ➔ | ESSEN |

## Vorgehen

- Vor der eigentlichen Codierung eines Quelltexts mit n Zeichen wird erst die Häufigkeitsverteilung der n Zeichen ermittelt
- Ausgehend vom Intervall [0,1[ wird dieses in n aneinander anschließende Intervalle aufgeteilt
- Jedem Intervall wird ein Zeichen zugeordnet
- Die Länge der Intervalle entspricht den Auftrittswahrscheinlichkeiten der Zeichen

# Beispiel

- Quelltext ESSEN ist arithmetisch zu codieren

- Notwendige Vorbereitungsschritte
  - Ermittlung der Auftrittswahrscheinlichkeiten $p_i$ der einzelnen Zeichen
  - Zuordnung eines Intervalls $[u, o[$ zu jedem Zeichen, wobei die Länge zu den jeweiligen Auftrittswahrscheinlichkeiten proportional ist

| Zeichen $c$ | Auftrittswahrsch. $p_i$ | Intervall $[u(c), o(c)[$ |
|---|---|---|
| E | $^2/_5$ | $[0.0, 0.4[$ |
| S | $^2/_5$ | $[0.4, 0.8[$ |
| N | $^1/_5$ | $[0.8, 1.0[$ |

## Kompressions-Algorithmus

- Initialisiere untere und obere Grenze

$$O := 1$$
$$U := 0$$

- Lies nächstes Eingabezeichen c und berechne

$$l := O - U$$
$$O := U + l \cdot o(c)$$
$$U := U + l \cdot u(c)$$

… aktuelle Länge des Intervalls
… neue Obergrenze, $o(c)$ aus Tabelle
… neue Untergrenze, $u(c)$ aus Tabelle

bis Textende erreicht ist

- Ergebnis $x$ (codierte Eingabedaten)

$$x := \frac{U+O}{2}$$

… (oder auch $x = U$)

# Arithmetische Codierung (6)

## Beispiel

- Kompression des Textes ESSEN

| Zeichen $c$ | Auftrittswahrsch. $p_i$ | Intervall $[u(c), o(c)[$ |
|---|---|---|
| E | $^2/_5$ | [0.0, 0.4[ |
| S | $^2/_5$ | [0.4, 0.8[ |
| N | $^1/_5$ | [0.8, 1.0[ |

| $c$ | $l$ | $O$ | $U$ |
|---|---|---|---|
| | - | 1.0 | 0.0 |
| E | 1.0 | 0.4 | 0.0 |
| S | 0.4 | 0.32 | 0.16 |
| S | 0.16 | 0.288 | 0.224 |
| E | 0.064 | 0.2496 | 0.224 |
| N | 0.0256 | 0.2496 | 0.24448 |

… Initialisierung

$$l := O - U$$
$$O := U + l \cdot o(c)$$
$$U := U + l \cdot u(c)$$

- Das Ergebnis ist $x$ = 0.24704

$$x := \frac{U + O}{2}$$

## Dekompressions-Algorithmus

- Lies Code $x$
- Solange $x$ > 0 (bzw. nicht alle Zeichen sind dekodiert)

Suche Zeichen $c$, in dessen Intervall $x$ liegt
Gib $c$ aus

$$l = o(c) - u(c)$$ … Länge des Intervalls

$$x := \frac{x - u(c)}{l}$$ … Neuer Code

# Beispiel

- Aus dem codierten Text (Gleitpunktzahl x = 0.24704) kann schrittweise der Ursprungstext wieder gewonnen werden

| Zeichen $c$ | Auftrittswahrsch. $p_i$ | Intervall $[u(c), o(c)[$ |
|---|---|---|
| E | $^2/_5$ | [0.0, 0.4[ |
| S | $^2/_5$ | [0.4, 0.8[ |
| N | $^1/_5$ | [0.8, 1.0[ |

| $x$ | $c$ (Ausgabe) | $O$ | $U$ | $l$ |
|---|---|---|---|---|
| 0.24704 | E | 0.4 | 0.0 | 0.4 |
| 0.6176 | S | 0.8 | 0.4 | 0.4 |
| 0.544 | S | 0.8 | 0.4 | 0.4 |
| 0.36 | E | 0.4 | 0.0 | 0.4 |
| 0.9 | N | 1.0 | 0.8 | 0.2 |

Suche Zeichen $c$, in dessen Intervall $x$ liegt. Gib $c$ aus

$l = o(c) - u(c)$

$x := \frac{x - u(c)}{l}$

# Aufgabe

- Codieren Sie das Wort IBIS arithmetisch!
- Decodieren Sie das Ergebnis!

## Codierung

$$l := O - U$$
$$O := U + l \cdot o(c)$$
$$U := U + l \cdot u(c)$$

$$x := \frac{U + O}{2}$$

## Decodierung

Suche Zeichen $c$, in dessen Intervall $x$ liegt. Gib $c$ aus

$$l = o(c) - u(c)$$
$$x := \frac{x - u(c)}{l}$$

# Arithmetische Codierung (10)



## Probleme

- Immer kleiner werdende Teilintervalle mit jedem neu zu codierenden Zeichen
  - Abhängig von Wortbreite haben Rechner aber nur eine begrenzte Genauigkeit für Gleitpunktzahlen
  - Ab einer Grenze ist „Codezahl“ nicht mehr darstellbar
- Auftrittswahrscheinlichkeit der Zeichen muss vor der Codierung bekannt sein
  - Verwendung der immer gleichen Auftrittswahrscheinlichkeiten
  - Verwendung semi-adaptives/ adaptives Verfahren
- Wesentlich rechenintensiver als Huffman

# Arithmetische Kodierung – Verwendung



- H.264/MPEG 4 AVC
  - (verlustbehaftete) Videocodierung
  - z.B. Blu-ray oder DVB-S2
  - arithmetische Codierung optional an Stelle von Huffman für Entropiecodierung verwendbar
- HEVC
  - auch: H.265/MPEG-H Teil 2
  - Nachfolgeformat von H.264
  - z.B. UHD-Bluy-ray (4k), DVB-T2, Streaming
  - arithmetische Codierung obligatorisch, kein Huffman