

***Einführung in die Informatik II***  
***Aus der Informationstheorie:***  
***Datenkompression***

Prof. Bernd Brügge, Ph.D  
Institut für Informatik  
Technische Universität München

Sommersemester 2001

23. Juli 2001

# *Überblick über die Vorlesung*

## ❖ Informationstheorie

- ◆ Entropie
- ◆ Verlustfreie vs. verlustbehaftete Kompression
- ◆ Huffman-Code

# *Codierung*

- ❖ (Wiederholung aus Info I):
  - ◆ **Nachricht:** Eine Mitteilung, bei der wir von dem Übertragungsmedium und der Darstellung durch Signale oder Inschriften abstrahieren.
  - ◆ **Repräsentation:** Die äußere Form einer Nachricht.
  - ◆ **Information:** Der abstrakte Gehalt bzw. die *Bedeutung (Semantik)* einer Nachricht.
- ❖ Information muss für die Zwecke der maschinellen Speicherung und Verarbeitung stets durch exakt festgelegte Formen der Repräsentation dargestellt werden.
- ❖ Die Suche nach einfachen und ökonomischen Repräsentationen führt zu Fragen der Codierung von Informationen und der Auswahl von Codes.
  - ◆ Ein Code erlaubt den Übergang von einer Repräsentation zu einer anderen Repräsentation derselben Information.

# *Auswahl von Codes*

- ❖ Bei der Auswahl von Codes stehen zwei Ziele im Vordergrund:
- ❖ **Hohes Maß an Fehlersicherheit**
  - ◆ Beim Auftreten von Übertragungsfehlern oder Verarbeitungsfehlern wollen wir geringfügig veränderte Wörter (codierte Nachrichten) zumindest als gestört erkennen oder vielleicht sogar trotz der Störung richtig decodieren.
- ❖ **Ökonomie der Darstellung und Verarbeitung**
  - ◆ Aus Effizienzgründen sind wir daran interessiert, die Wörter möglichst klein zu machen.
  - ◆ Außerdem soll die Verarbeitung in der gewählten Repräsentation einfach sein.

# *Für welche Aufgaben werden Codes eingesetzt?*

## ❖ **Erkennung/Beseitigung von Übertragungsfehlern**

- ◆ Hamming-Abstand

- ◆ Parität, Prüfsumme

→ Vorlesung (HS): *Rechnernetze (Hegering)*

## ❖ **Verschlüsselung** (siehe Info I - Vorlesung 14)

→ Vorlesung (HS): *Kryptologie (Gerold)*

## ❖ **Kompression von Daten**

- ◆ Informationsgehalt, Entropie, Redundanz

- ◆ verlustbehaftete vs. verlustfreie Kompression

→ Vorlesung (HS): *Effiziente Algorithmen und Datenstrukturen (Mayr)*

→ Vorlesung (HS): *Data Compression (Khuri, WS 99/00)*

# *Stochastische Nachrichtenquelle*

- ❖ Gegeben sei eine Zeichenmenge  $A$ , wobei wir für jedes Zeichen  $a$  in  $A$  dessen relative Häufigkeit  $p_a$  kennen.
- ❖ **Definition:** Eine Quelle für Zeichen bzw. Zeichenfolgen, bei der zu jedem Zeitpunkt die Wahrscheinlichkeit, dass ein bestimmtes Zeichen gesendet wird, der relativen Häufigkeit des Zeichens in der Zeichenmenge entspricht, heißt **stochastische** oder **Shannonsche Nachrichtenquelle**.

# *Entscheidungsgehalt und Entropie*

- ❖ **Definition:** Der **Informationsgehalt** eines Zeichens  $a \in A$  ist der Kehrwert der Wahrscheinlichkeit  $p_a$  seines Auftretens:  $1/p_a$
- ❖ **Definition:** Der **Entscheidungsgehalt** eines Zeichens  $a \in A$  ist der binäre Logarithmus seines Informationsgehaltes:  $\text{ld}(1/p_a)$
- ❖ **Definition:** Der mittlere Entscheidungsgehalt  $H$  eines Zeichenvorrates  $A$  ist definiert als  $H = \sum (p_a \cdot \text{ld}(1/p_a))$
- ❖  $H$  bezeichnet man auch als **Entropie der Nachrichtenquelle**. Die Entropie gibt ein Maß für die Schwankung der Wahrscheinlichkeiten der von einer Nachrichtenquelle ausgesendeten Zeichen.
  - ◆ Ist die Entropie klein, dann sind die Wahrscheinlichkeiten der Zeichen sehr unterschiedlich. Ist die Entropie groß, dann sind die Wahrscheinlichkeiten nicht sehr unterschiedlich.
  - ◆ Die Entropie ist am größten, wenn alle Zeichen gleichwahrscheinlich sind.

## *Beispiel: Berechnung der Entropie*

- ❖ Gegeben sei das Alphabet  $A$  mit zwei Zeichen  $a$  und  $b$  und deren Wahrscheinlichkeiten  $p_a = 0.75$  und  $p_b = 0.25$ .
- ❖ Die Entropie der Nachrichtenquelle über  $A$  ist  $H = \sum (p_a \cdot \text{ld}(1/p_a))$
- ❖ Also:
  - ◆  $H = 0.75 \cdot \text{ld}(1/0.75) + 0.25 \cdot \text{ld}(1/0.25)$
  - ◆  $H = 0.75 \cdot \text{ld}(4/3) + 0.25 \cdot \text{ld}(4)$
  - ◆  $H \approx 0.3 + 0.5$
  - ◆  $H \approx 0.8$
- ❖ Zum Vergleich: mit  $p_a = 0.5$  und  $p_b = 0.5$  ergibt sich
  - ◆  $H = 0.5 \cdot \text{ld}(2) + 0.5 \cdot \text{ld}(2) = 1.0$

# *Binärcodierung*

- ❖ Sei  $A$  ein Alphabet von Zeichen, wobei  $p_a$  die Wahrscheinlichkeit des Zeichens  $a$  bedeutet.
- ❖ Eine Abbildung  $c: A \rightarrow \mathbb{B}^*$  bezeichnen wir als **Binärcodierung**.
- ❖ Mit  $|c(a)|$  bezeichnen wir die Wortlänge des Binärwortes  $c(a) \in \mathbb{B}^*$ .
- ❖ Für eine gegebene Codierung  $c$  erhalten wir dann die mittlere Wortlänge durch
  - ◆  $L = \sum (p_a \cdot |c(a)|)$
- ❖ Aus Gründen der Ökonomie sind wir daran interessiert, mit Binärcodierungen zu arbeiten, bei denen die mittlere Wortlänge der Codierung möglichst klein ist.
- ❖ Allerdings treten wir schnell an die Grenzen des Möglichen.
  - ◆ Bei einer zwei-elementigen Menge kann  $L$  nie kleiner als 1 sein.

## *Binäre Wortcodierung*

- ❖ Bessere Werte für die mittlere Wortlänge erhalten wir, wenn wir nicht Einzelzeichen, sondern Wörter über  $A$  codieren.
- ❖ Ein Wort über  $A$  wird dann codiert, indem wir seine Elemente zu Gruppen zusammenfassen.
- ❖ Eine Abbildung  $c: A^m \rightarrow \mathbb{B}^*$  definieren wir als eine **(binäre) Wortcodierung** über einem Zeichenvorrat  $A$ , wobei  $m$ -elementige Wörter  $w \in A^m$  auf Binärwörter der Länge  $N_w$  abgebildet werden.
- ❖ Die Wahrscheinlichkeit  $q_w$  des Wortes  $w$  ergibt sich dabei aus dem Produkt der Wahrscheinlichkeiten der Einzelzeichen.
- ❖ Die mittlere Wortlänge einer binären Wortcodierung ergibt sich aus
  - ◆  $L = 1/m \cdot \sum (q_w \cdot N_w)$ , wobei  $w \in A^m$
- ❖ Für  $L$  gibt es eine untere Grenze, die durch die Entropie bestimmt ist.

# *Shannonsches Codierungstheorem*

- ❖ Das **Shannonsche Codierungstheorem** sagt
  - (1) Für beliebige binäre Wortcodierungen gilt:  $H \leq L$ , das heißt, die mittlere Wortlänge ist gleich oder größer als die Entropie
  - (2) Jede stochastische Nachrichtenquelle kann durch binäre Wortcodierungen so codiert werden, dass der nicht-negative Wert  $L - H$  beliebig klein wird.
- ❖ Den Wert  $L - H$  bezeichnet man auch als **Coderedundanz**.
- ❖ Die Entropie stellt also die untere Grenze für die mittlere Wortlänge einer binären Wordcodierung dar.
  - ◆ Die mittlere Wortlänge wird klein, wenn es uns gelingt, die Codierung so zu wählen, dass für jedes neue Bit die Wahrscheinlichkeit für L und O gleich groß ist.
  - ◆ Dann hat zu jedem Zeitpunkt jedes Bit die Wahrscheinlichkeit 0.5 und damit den gleichen Entscheidungsgehalt.

# *Codebaum*

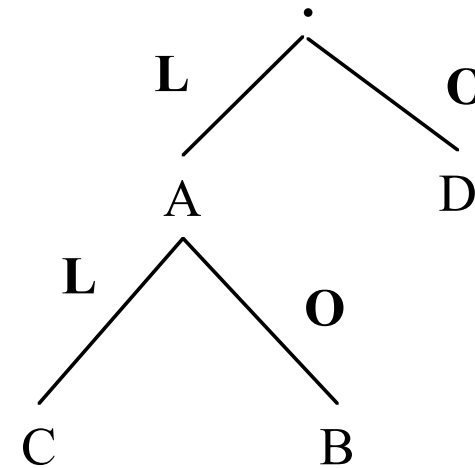
- ❖ Lesen wir die Bits von einer Shannonschen Nachrichtenquelle, dann enthält jedes Bit neue Informationen über das Zeichen, das gerade durch ein Codewort übertragen wird.
- ❖ Jedes Bit gestattet also, die Menge der in Frage kommenden Zeichen aus dem Alphabet  $A$  einzuschränken.
- ❖ Es entsteht so ein Entscheidungsbaum für die schrittweise Codierung, den wir als **Codebaum** bezeichnen.
- ❖ Binärcodierungen und binäre Wortcodierungen lassen sich durch einen binären Codebaum repräsentieren.
  - ◆ Auch die Codierung und Decodierung von Binärwörtern kann mit Hilfe eines binären Codebaums vorgenommen werden.

# Beispiel eines Codebaums

## Binärcodierung

Zeichen	Code
A	L
B	LO
C	LL
D	O

## Zugehöriger Codebaum



Zeichenkette: LLL00LL

Mögliche Worte: AAADDAA, CADDCC, ...

❖ Bei Codes mit variierender Wortlänge ist die Decodierung nicht eindeutig

### ❖ Fano-Bedingung:

Kein Codewort darf der Anfang eines anderen Codewortes sein.

❖ Ein Code der die Fano-Bedingung erfüllt, heißt **Präfix-Code**

# *Datenkompression*

- ❖ **Ziel:** Möglichst viele Informationen möglichst kompakt in einer Nachricht codieren.
- ❖ **Verlustfreie Kompression:** Alle Informationen können aus der Nachricht wieder decodiert werden.
- ❖ **Verlustbehaftete Kompression:** Ein Teil der Informationen geht bei der Codierung der Nachricht verloren.
- ❖ Einsatzbereiche für Datenkompression:
  - ◆ Photos im WWW (JPEG)
  - ◆ Audio-Übertragung über das WWW (MP3)
  - ◆ DVD (MPEG2)
  - ◆ Mobil-Kommunikation (WAP, UMTS)
- ❖ Im folgenden betrachten wir ein verlustfreies Kompressionsverfahren  
⇒ Huffman-Codierung

verlustbehaftete  
Kompressions-  
verfahren

# *Beispiel für verlustfreie Codierung: Huffman-Code*

## ❖ **Problembeschreibung:**

- ◆ Wir wollen eine lange Nachricht komprimieren (z.B. Buchtext, Bild).
- ◆ die Nachricht ist aus vielen einzelnen Elementen aufgebaut (z.B. Buchstaben, farbige Pixel).
- ◆ Die Elementmenge (Alphabet, Farbpalette) ist festgelegt.
- ◆ Die Häufigkeitsverteilung für die Elementmenge ist bekannt, d.h. für jedes Element aus der Elementmenge wissen wir, wie oft es in der Nachricht vorkommt.

- ❖ **Grundidee:** Wir versuchen, die einzelnen Elemente möglichst effizient zu codieren, d.h. häufig vorkommende Elemente sollen einem kürzeren Code haben als selten auftretende Elemente.

## *Anwendungsbeispiel aus der Biologie: DNS-Sequenzen*

- ❖ (fiktive) DNS-Sequenz: **CAAGAATGCATAATAGATAG**
- ❖ Länge bei Codierung nach ISO 8859-1(8 bit ASCII): **160 bit** (20 Byte)
- ❖ Für 4 Zeichen reichen  $\log_2(4) = 2$  bit pro Zeichen, z.B.:
  - A: 00    C: 01    G: 10    T: 11**    (*2-bit-Code*)
  - Codierung: **0100001000001110010011000011001000110001**
  - Länge der codierten Nachricht: **40 bit**
- ❖ **Frage:** Gibt es einen noch kompakteren Code für diesen Text?
  - A: 0      C: 110    G: 111    T: 10**    (*Huffman-Code*)
  - Codierung: **110001110010111110010001001110100111**
  - Länge der codierten Nachricht: **36 bit**
- ❖ **Frage:** Wie kommt man zu diesem Code?

# *Aufbau eines Codebaums für Huffman-Codes*

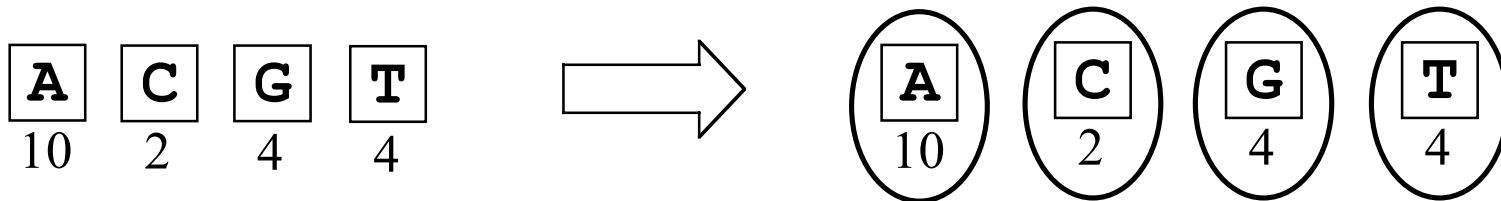
## ❖ **Algorithmus:**

- ◆ Erzeuge für jedes Element einen (1-elementigen) Baum mit dem Element als Inhalt und dessen Häufigkeit als zusätzlichem Attribut
- ◆ solange mehr als ein Baum existiert:  
Erstelle einen neuen Binärbaum mit leerer Wurzel:
  - ◆ linker Unterbaum: der Baum mit der niedrigsten Häufigkeit
  - rechter Unterbaum: der Baum mit der zweitniedrigsten Häufigkeit
  - ◆ die linke Kante wird mit **0** beschriftet, die rechte Kante mit **1**.
  - ◆ die Häufigkeit des neuen Binärbaums ist die Summe der Häufigkeiten seiner Unterbäume.

# *Aufbau eines Huffman-Codebaums: Beispiel (1)*

## ❖ Algorithmus:

- ◆ Erzeuge für jedes Element einen (1-elementigen) Baum mit dem Element als Inhalt und dessen Häufigkeit als zusätzlichem Attribut



## Aufbau eines Huffman-Codebaums: Beispiel (2)

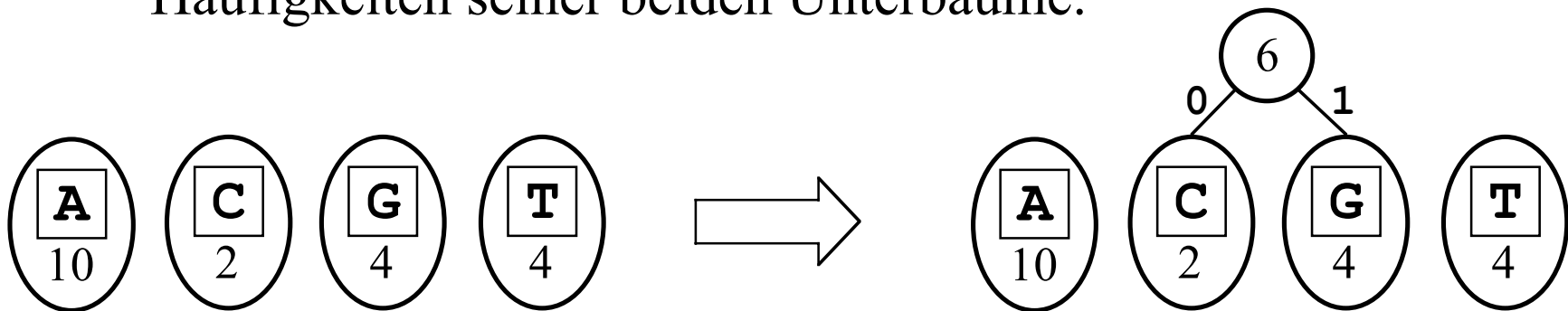
### ❖ Algorithmus:

◆ ...

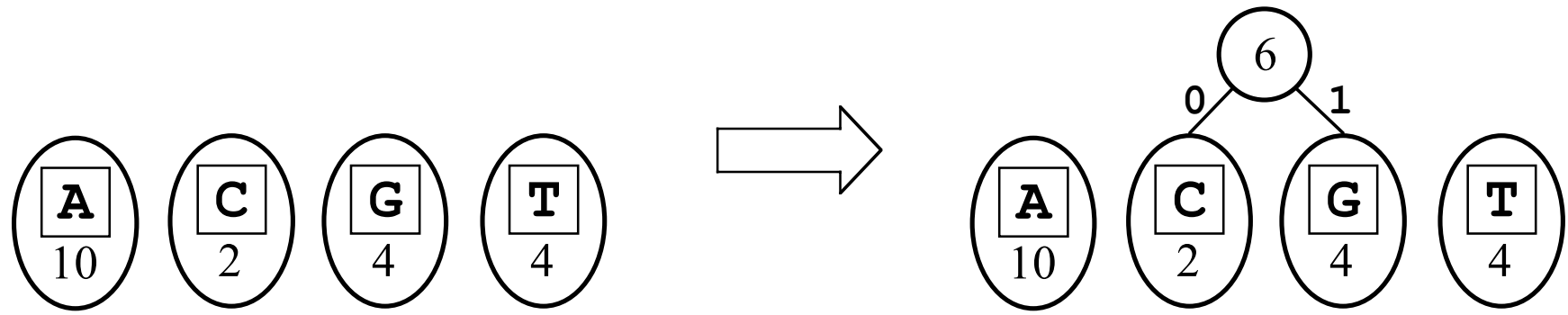
◆ solange mehr als ein Baum existiert:

Erstelle einen neuen Binärbaum mit leerer Wurzel:

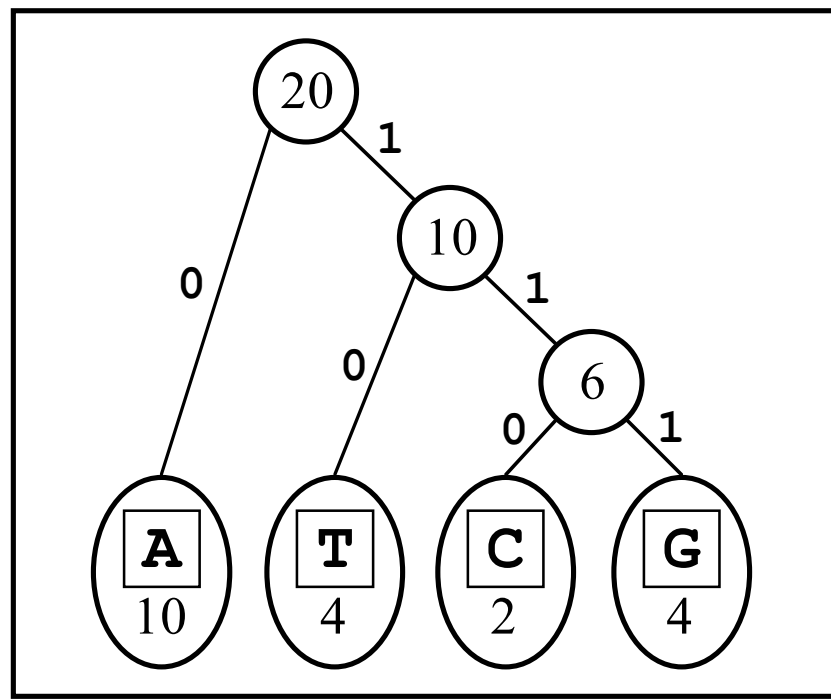
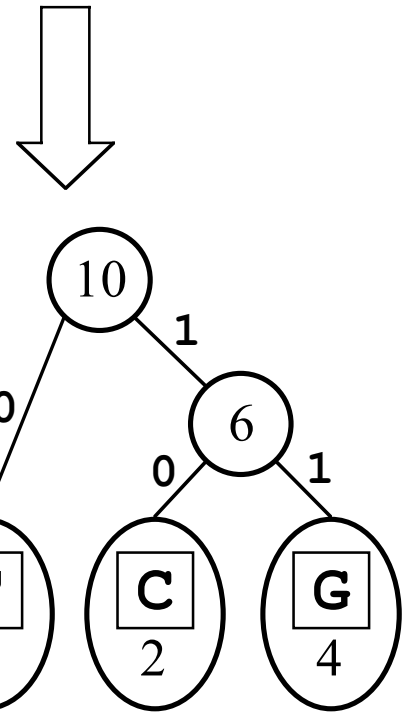
- ◆ linker Unterbaum: der Baum mit der niedrigsten Häufigkeit
- rechter Unterbaum: der Baum mit der zweitniedrigsten Häufigkeit
- ◆ die linke Kante wird mit **0** beschriftet, die rechte Kante mit **1**.
- ◆ die Häufigkeit des neuen Binärbaums ist die Summe der Häufigkeiten seiner beiden Unterbäume.



# Aufbau eines Huffman-Codebaums: Beispiel (3)

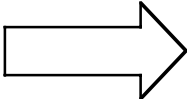


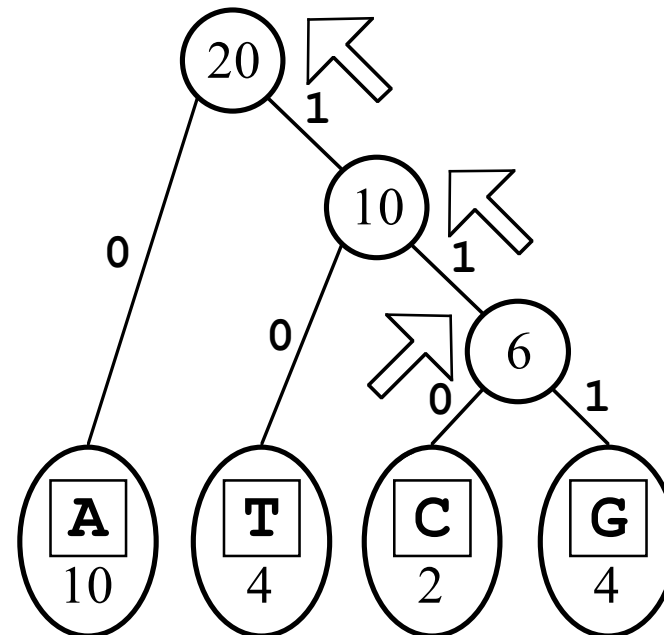
Häufigkeit von **T** ist kleiner als Häufigkeit von **C** und **G** ⇒ **T** linker Unterbaum



# Codierung mit Huffman-Codebaum

- ❖ Der Code für ein Element ergibt sich, wenn man den Codebaum von dem Elementknoten aus in Richtung Wurzel durchläuft und dabei die Markierung der zuletzt durchlaufenen Kante *links* an den bereits vorhandenen Code anfügt.
- ❖ Beispiel: Codierung von **C**

**C**            **1 1 0**

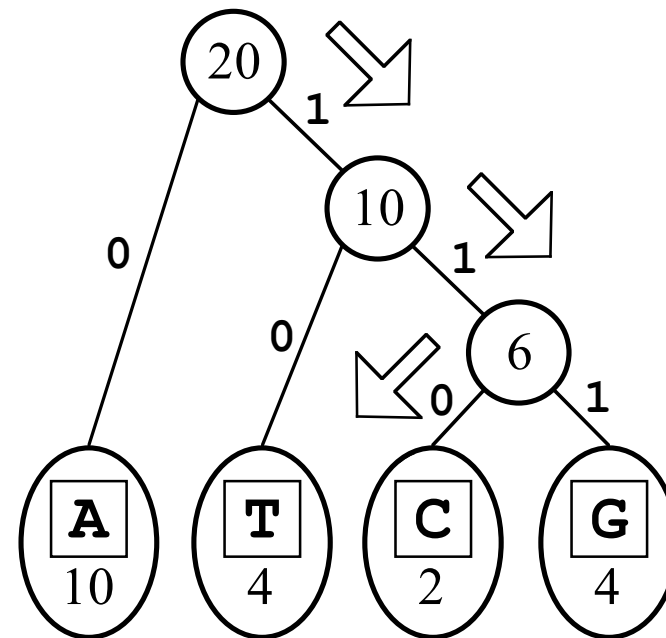


# Decodierung mit Codebaum

- ❖ Um aus einem Code ein Element zu decodieren, durchläuft man den Codebaum von der Wurzel des Codebaums aus:
  - ◆ falls der aktuelle Knoten Unterknoten hat, wird die Kante durchlaufen, deren Markierung dem zuletzt gelesenen Codezeichen entspricht.
  - ◆ wenn der aktuelle Knoten ein Blattknoten ist, ist sein Inhalt das gesuchte Element.

❖ Beispiel: Decodierung von **110**

1 1 0       $\longrightarrow$       **C**



## *Huffman-Codes sind Präfix-Codes*

- ❖ Der Huffman-Code erfüllt die Fano-Bedingung, d.h. der Huffman-Code ist ein sog. *Präfix-Code*.
- ❖ *Präfix-Code*:  
Für jedes codierbare Element  $e$  gilt: es gibt kein anderes Element  $e' \neq e$ , dessen Codierung die Codierung von  $e$  als *Präfix* enthält.
  - ◆ Anschauliche Interpretation dieser Aussage anhand des Codebaums:  
*Nur die Blattknoten des Codebaums enthalten codierbare Elemente!*
- ❖ Die Präfix-Code-Eigenschaft ist vor allem wichtig für die Decodierung:
  - ◆ Ein Code mit variabler Länge, der kein Präfix-Code ist, ist u.U. ohne zusätzliche Informationen nicht eindeutig decodierbar!
  - ◆ Beispiel: **A: 0**      **C: 00**      **G: 1**      **T: 11**  

```
graph TD; 1100 --> TC; 1100 --> TAA; 1100 --> GGC; 1100 --> GGAA;
```
- ❖ Codes mit fester Länge (z.B. ASCII) müssen keine Präfix-Codes sein, da die Elementgrenzen durch die Codelänge eindeutig festgelegt sind.

## *Wann ist Kompression mit Huffman-Codes sinnvoll?*

- ❖ Huffman-Codes erlauben eine verlustfreie Kompression.
- ❖ Die Verwendung eines Huffman-Codes bietet sich vor allem an bei
  - ◆ umfangreichen Nachrichten mit wenig verschiedenen Elementen
  - ◆ deutlich unterschiedlichen Häufigkeiten der einzelnen Elemente (bei gleichverteilten Häufigkeiten einfacher Code mit fester Länge)
- ❖ Treten ganze Element-Sequenzen gehäuft auf (z.B. Silben in Texten, Muster in Bildern), sind andere Kompressionsverfahren vorzuziehen (z.B. LZW).
- ❖ Verlustbehaftete Kompressionsverfahren (z.B. JPEG) ermöglichen eine erheblich höhere Kompressionsrate.
  - ◆ Sie sind jedoch mit einem Informationsverlust verbunden und sollten daher nur eingesetzt werden, wenn sichergestellt ist, dass keine wichtigen Informationen entfernt werden!

# *Zusammenfassung*

- ❖ **Stochastische Nachrichtenquelle:** Eine Quelle für Zeichenfolgen mit festgelegten Wahrscheinlichkeiten für die erzeugbaren Zeichen.
- ❖ **Shannonsches Codierungstheorem:** Die mittlere Wortlänge einer Codierung ist mindestens so groß wie die Entropie der Nachrichtenquelle.
- ❖ Das Ziel der *Datenkompression* ist, möglichst viele Informationen möglichst kompakt in einer Nachricht codieren
- ❖ *Verlustbehaftete vs Verlustfreie Kompressionen*
  - ◆ Verlustbehaftete Kompressionsverfahren (z.B. JPEG, MP3) ermöglichen eine erheblich höhere Kompressionsrate (bei gleichzeitigem Informationsverlust).
- ❖ **Huffman-Codes** erlauben verlustfreie Kompression.