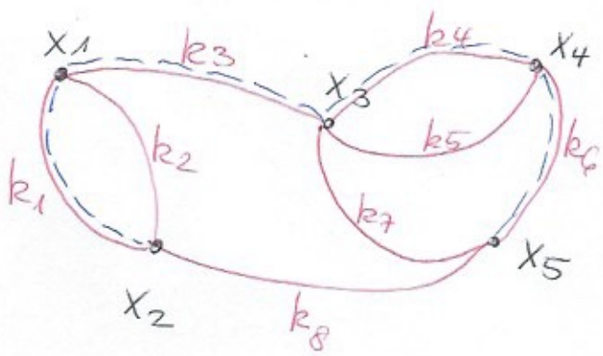
Das historische Beispiel, das Königsberger Brückenproblem, ist ein erstes Beispiel für eine sog. "Graphen zu durchlaufen".
Daher ist es wichtig, einige Begriffe zu definieren:

Def.: (Weg, Kantenfolge)
Gegeben sei ein Graph G. Ein Teilgraph G^* mit der Eigenschaft $v(k_i) = \{x_{i-1}, x_i\}$ $i = 1, \dots, m$ heißt Weg oder Kantenfolge, wenn die Knoten paarweise verschieden sind. Häufig wird die Kantenfolge durch die Folge seiner Knoten angegeben. Ist $x_0 = x_m$, also der Anfangsknoten auch der Endknoten des Weges, so heißt die Kantenfolge geschlossen, im anderen Fall spricht man von einer offenen Kantenfolge. Eine geschlossene Kantenfolge heißt auch Zyklus oder Kreis.
Die Länge L einer Kantenfolge wird definiert durch die Anzahl der zugehörigen Kanten.
Bei gerichteten Graphen müssen die aufeinanderfolgenden Knoten durch eine in diese Richtung weisende gerichtete Kante verbunden sein.

Bp:



Kantenfolge: $k_1\ k_3\ k_4\ k_6$ — Beschreibung mittels der Kanten

Weg $x_2\ x_1\ x_3\ x_4\ x_5$ — Beschreibung mittels der Knoten

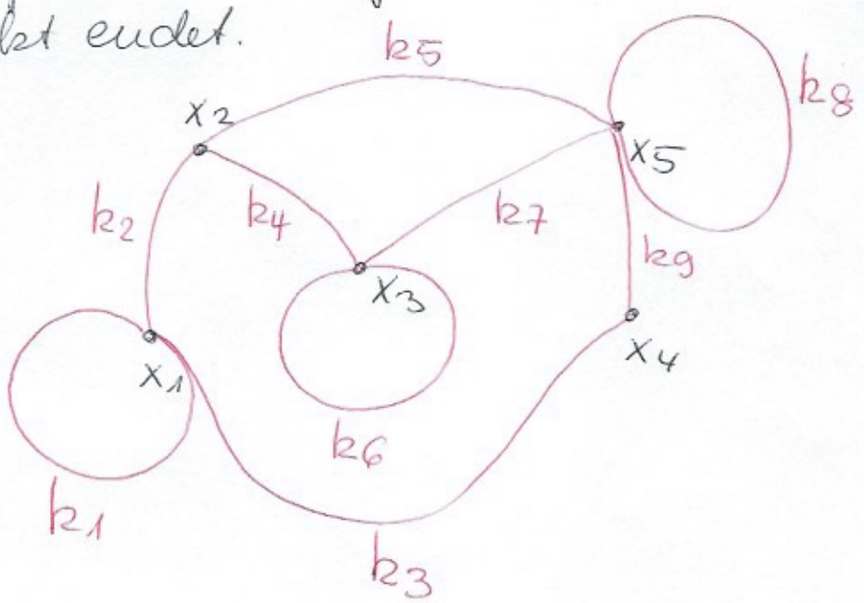
Def.: (Kantenzug, Kantenweg, Kantenzyklus)

Kommt in einer Kantenfolge keine Kante zweimal vor, spricht man von einem (offenen bzw. geschlossenen) Kantenzug. Ein Kantenzug, in dem kein Knoten zweimal vorkommt, heißt Kantenweg.

Def.: Eulerzug

Ein Kantenzug, der jede Kante eines Graphen genau einmal enthält, heißt Eulerzug oder Eulersche Linie.
Dieser kann geschlossen sein, d.h. er endet da, wo gestartet wurde, er kann aber auch alle Kanten eines Graphen durchlaufen, ohne, dass man am Startpunkt endet.

Bp:



Eulersche Linie

Start bei x_2: $k_2\ k_1\ k_3\ k_9\ k_7\ k_6\ k_4\ k_5\ k_8$ Ende bei x_5

Der folgende Satz von L. Euler (1736) sagt etwas über die Existenz von Eulerschen Linien in einem Graphen aus.

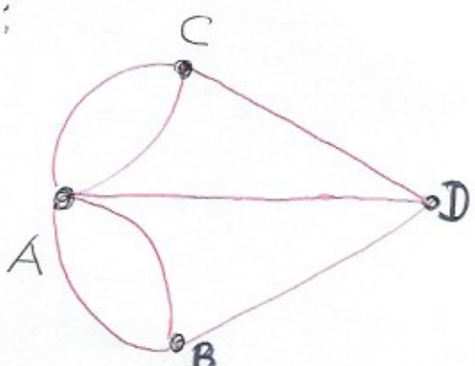
Satz: (Euler 1736)

Ein endlicher, ungerichteter, nicht notwendig schlichter Graph $G = (M, K, v)$ besitzt genau dann eine Eulersche Linie, wenn G bis auf isolierte Knoten zusammenhängend und die Zahl z der Knoten mit ungeradem Grad höchstens 2 ist.

Denn, ist ein Knotengrad ungerade, so muss die Euler'sche Linie in diesem Knoten beginnen oder enden. Sind zwei Knotengrade ungerade, so beginnt die Euler'sche Linie in einem dieser Knoten und endet in dem anderen dieser Knoten.

Mit anderen Worten: Ein Knoten muss über eine Kante erreicht und über eine andere Kante wieder verlassen werden (jede Kante nur einmal verwendet). Bis auf Anfangs- und Endknoten muss der Knotengrad immer gerade sein.

Betrachten wir noch einmal das Königsberger Brücken-problem:



hier die graphentheoretische Interpretation

Frage: Gibt es nun einen Euler'sche Linie (jede Kante nur einmal (d.h. für Euler jede Brücke nur einmal) in diesem Graphen?

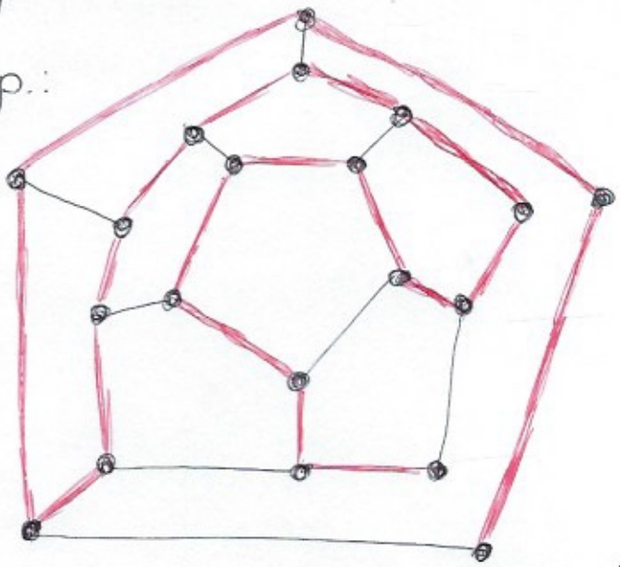
Antwort: Es gibt keine Eulersche Linie, da jeder Knotengrad ungerade ist.

Bem.: Der Begriff Weg, Kantenzug, Euler'sche Linie, Eulerzug wird in der Literatur nicht einheitlich verwendet.

Def.: (Hamiltonsche Linie)

Ein Weg, der jeden Knoten eines Graphen genau einmal enthält, heißt Hamilton'sche Linie. Wenn Anfangs- und Endknoten zusammenfallen, spricht man von einem Hamilton'schen Kreis.

Bp.:



20 Knoten
Kanten schwarz
Die rot markierten Kanten markieren einen Weg, der jeden Knoten nur einmal besucht

Spezielles Problem:
Problem des Handlungsreisenden

Im Gegensatz zum Königsberger Brückenproblem gelöst das Aufsuchen einer Hamilton'schen Linie (Kreis), zu den math. Problemen, die noch nicht gelöst werden konnten.

Im Folgenden werden mit weiteren Definitionen die Grundlagen gelegt, um das Thema: Durchlaufen von Graphen in seinen verschiedenen Herangehensweisen zu besprechen.

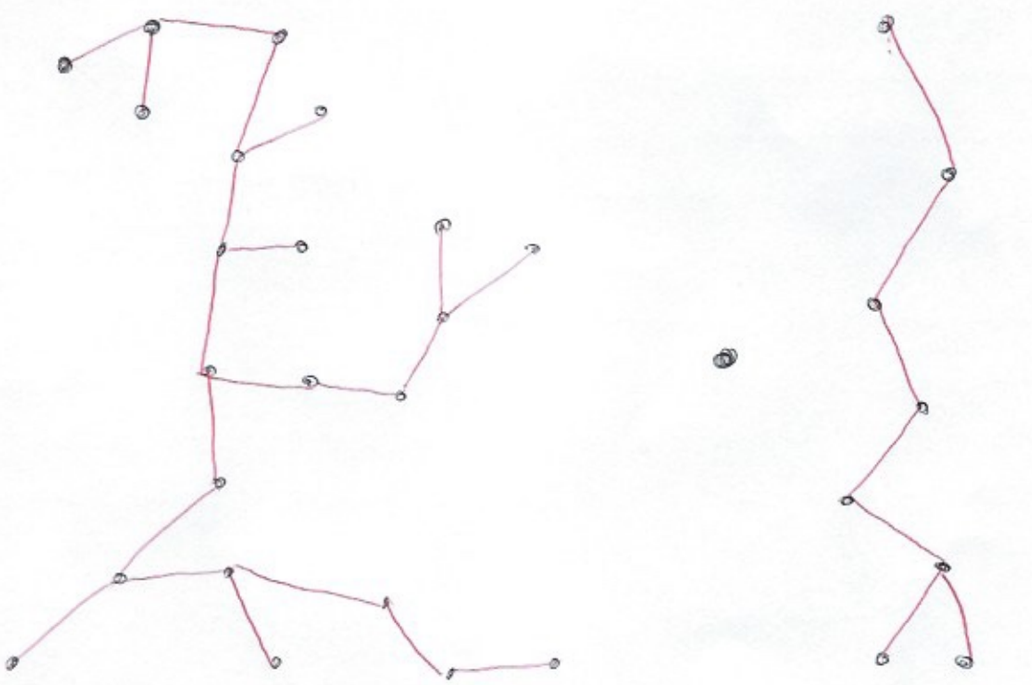
Bäume und Wälder

Ein Graph, der keine geschlossene Kantenfolge (Zyklus) enthält, wird als Wald bezeichnet.

Def. (Baum)

Ein zusammenhängender Graph ohne geschlossene Kantenfolge, in dem je zwei Knoten durch genau einen Weg verbunden sind, heißt Baum.

Bem: In jedem Baum ist jeder Knoten x_i mit $d(x_i) > 1$ ein trennender Knoten (Artikulationspunkt) und jede Kante eine Brücke.
Die Knoten eines Baumes mit $d(x_i) = 1$, heißen Randknoten oder Blätter.



Baum Baum

Wald

Satz: Ein Baum mit n Knoten hat genau n-1 Kanten.

Beweis: vollständige Induktion nach der Anzahl der Knoten des Baumes

Induktionsanfang: $n = 2$

2 Knoten → 1 Kante

Induktionsvoraussetzung:

Beh. gilt für $n = k$ Knoten, also k-1 Kanten

Induktionsschritt: $k \to k+1$

Ein Baum mit k+1 Knoten besteht aus einem Baum mit k Knoten und k-1 Kanten, kommt ein Knoten hinzu, so wird dieser durch eine Kante mit einem vorhandenen Knoten verbunden, es sind dann $k-1+1=k$ Kanten, d.h. $k+1-1=k$

Induktionsschluss: Die Beh. gilt für alle $n \in \mathbb{N}$.

Bem: Entfernt man von einem Baum ein Blatt, so ist der Rest immer noch ein Baum.

Satz: Für einen Graphen G sind die folgenden Aussagen äquivalent

(1) G ist ein Baum

(2) Zwischen je zwei Knoten enthält G genau einen Weg

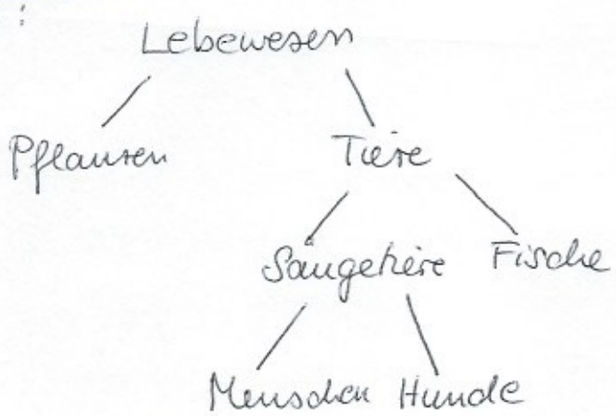
(3) G ist minimalzusammenhängend, d.h. G ist zusammenhängend, wird eine Kante k "aus der Mitte" entfernt, zerfällt der Graph in zwei Zusammenhangskomponenten.

(4) G ist maximal kreislos, d.h. G ist zykelfrei, aber für je zwei nicht benachbarte Knoten x_i und x_j enthält $G + \{x_i, x_j\}$ einen Zyklus.

Auf den Beweis dieses Satzes soll an dieser Stelle verzichtet werden, er ist nicht schwer und eine gute Übung, nochmals die Definitionen zu überprüfen.

Wo findet man Bäume?

In der Informatik werden Datenstrukturen durch Bäume dargestellt.
In der Biologie werden Stammbäume durch Bäume dargestellt:

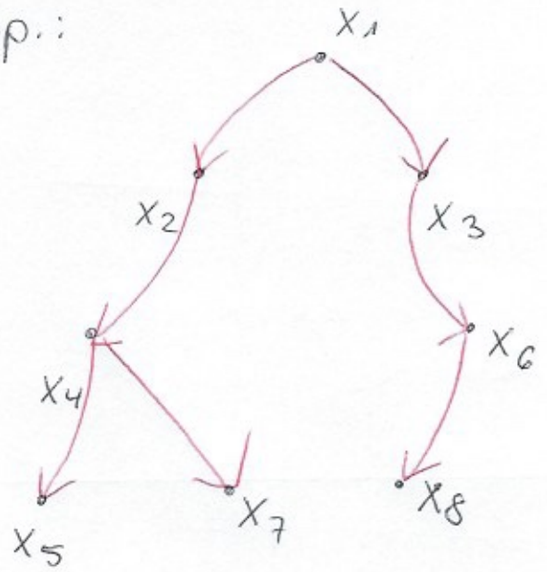


Von besonderer Bedeutung unter den Bäumen sind die Wurzelbäume

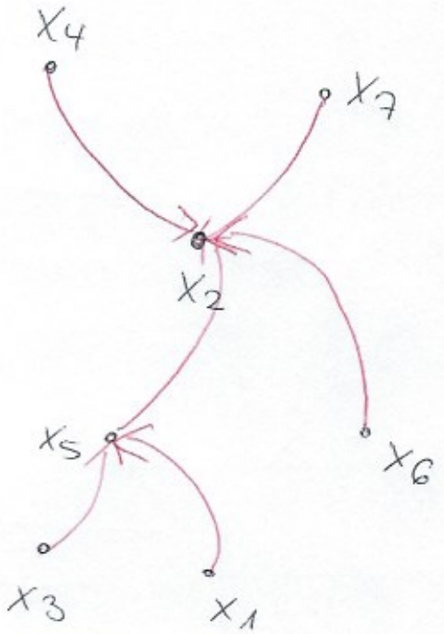
Def: (Wurzelbaum)

Ein Wurzelbaum ist ein gerichteter Graph, der ein Baum ist und der einen ausgezeichneten Knoten (die Wurzel) hat, von dem aus alle anderen Knoten erreichbar sind oder die von allen anderen Knoten erreicht werden kann.
Im Falle eines ungerichteten Graphen könnte jeder Knoten die Wurzel sein, daher macht es nur Sinn, gerichtete Graphen zu betrachten.
Hier kann man zwischen sog. Out-Trees und In-Trees unterscheiden.

Bp.:



Out-Tree
mit x_1 als Wurzel

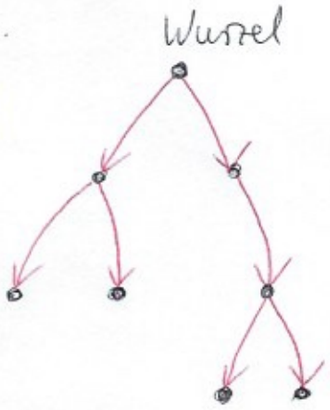


In-Tree
mit x_2 als Wurzel

Def: (Höhe eines Wurzelbaumes)

Die maximal mögliche Länge eines Weges von der Wurzel zum Blatt heißt die Höhe des Wurzelbaumes. Je nachdem, ob man die Anzahl der durchlaufenen Kanten zählt, oder die Anzahl der Knoten nimmt, kann sich diese Zahl unterscheiden.

Bp.:



Anzahl der durchlaufenen Kanten: 3

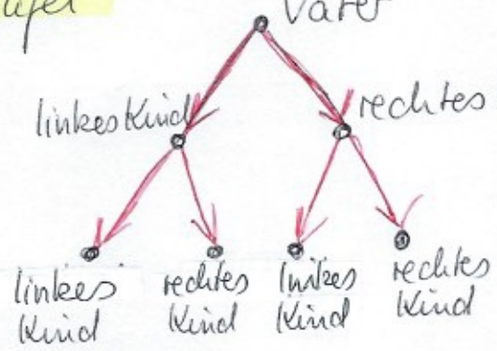
Anzahl der durchlaufenen Knoten: 4

Von ganz besonderer Bedeutung unter den Wurzelbäumen sind die Binärbäume.

Def. (Binärbaum)

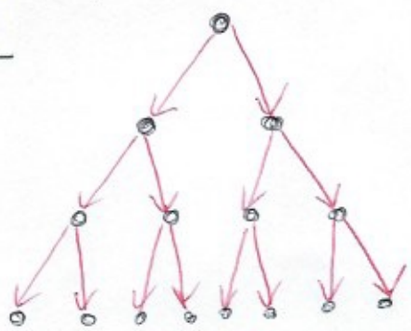
Ein Wurzelbaum, bei dem jeder Knoten höchstens zwei Nachfolger hat (Kindknoten, linkes und rechtes Kind, häufig auch: linker und rechter Sohn)
Hat jeder Knoten entweder 0 oder 2 Nachfolger, so heißt der Baum **regulärer Binärbaum**.

Bp: Ahnentafel



Üblicherweise sind Binärbäume Out-Trees.
Die Wurzel hat den Eingangsgrad 0, die anderen Knoten haben den Eingangsgrad 1
Beim Binärbaum ist der **Ausgangsgrad** der Knoten 0, 1 oder 2
Knoten mit Ausgangsgrad 0 sind Blätter.
Knoten mit **Ausgangsgrad ≥ 1** sind **innere Knoten**.
Ein vollständiger Binärbaum besteht aus Knoten, die immer 2 Nachfolger haben.

Bp.:



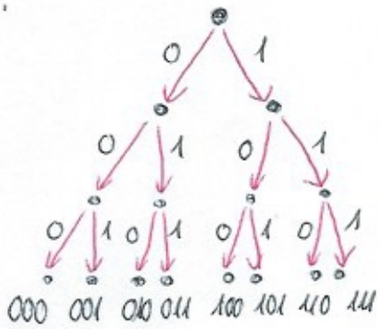
Höhe H: Anzahl der Knoten im längsten Weg
hier $H = 4$

Anzahl der Knoten im Baum:
$2^H - 1$ hier: $2^4 - 1 = 15$

Bp. (aus der Aussagenlogik):
Geg.: n logische Ausdrücke $A_0, A_1, \ldots A_{n-1}$ mit den jeweiligen Wahrheitswerten 0 oder 1.
Ausgehend von einer Wurzel (Niveau 0) wird jede Entscheidung A_i durch eine Stufe (Niveau)

dargestellt ($i = 0, 1, \ldots, n-1$)

Bp hier $n = 3$:



Die 8 möglichen Belegungen mit Wahrheitswerten von 3 logischen Ausdrücken finden sich an den Blättern.

Wo findet man Binärbäume? Wichtige Anwendungen

— Informatik: **Binärer Suchbaum** (BST: Binary Search Tree)

Wie der Name "Suchbaum" ausdrückt, dient ein Binärer Suchbaum dazu, aus großen Datenbeständen Einträge schnell wiederzufinden.
Zum schnellen Wiederfinden von Daten müssen die Daten vorher in eine passende Anordnung gebracht werden.
Das heißt die Daten müssen einen sortierbaren Schlüssel besitzen. In einem binären Suchbaum tragen die Knoten die "Schlüssel". Die Schlüssel des linken Teilbaums sind "kleiner oder gleich" dem Schlüssel des Knotens, die Schlüssel des rechten Teilbaums sind "größer oder gleich" als der Schlüssel des Knotens.
Dabei sind "kleiner gleich" und "größer gleich" vom Anwender festzulegen.
Allgemein kann man für einen binären Suchbaum folgende Eigenschaften definieren:

- Jeder Knoteninhalt besitzt als Komponente einen Schlüssel, für diesen Schlüssel muss eine Ordnung definiert sein.
- Für jeden Knoten K, der kein Blatt ist (innere Knoten) gilt: alle Schlüssel in den Knoten des linken Teilbaums sind kleiner, alle Schlüssel in den Knoten des rechten Teilbaums sind größer als der Schlüssel in K.

Die Suche geht wie folgt:

Sei S der Schlüssel

Ist S gleich dem Schlüssel, so ist die Suche beendet.

Falls $S < X$: Suche begeint im linken Teilbaum, beim "linken" Kind (Sohn)

Falls $S > X$: Suche begeint im rechten Teilbaum, beim "rechten" Kind (Sohn)

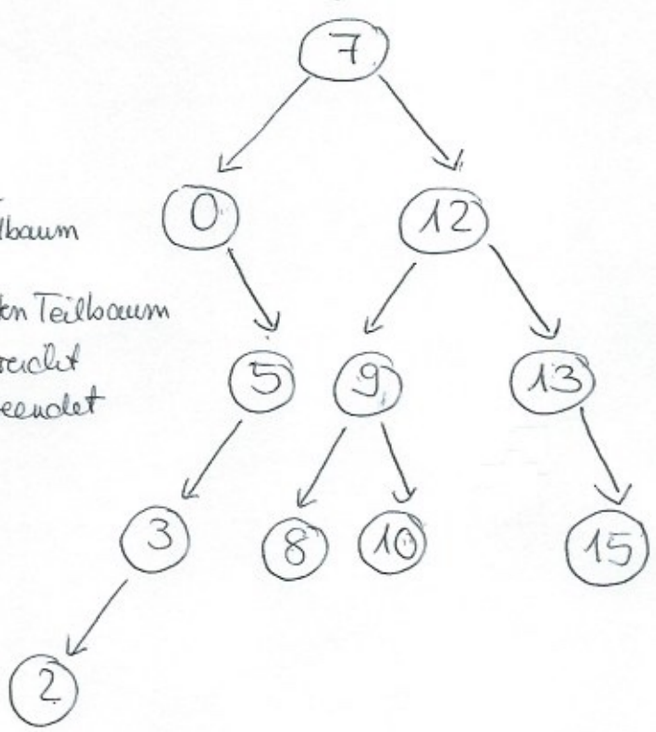
Die Suche wird solange durchgeführt, bis das Ziel erreicht ist.

Bp: Der folgende Baum ist ein binärer Suchbaum mit ganzen Zahlen als Schlüssel:

Gegeben ist folgende Datenmenge:

[7, 12, 0, 5, 9, 3, 8, 2, 13, 10, 15, 1]

Darstellung dieser Zahlen in einem binären Suchbaum (Zahlen sind Schlüssel und < (links) > (rechts) ist die Teilordnung)



Suche: (13)

13 > 7 ⇒ Suche im rechten Teilbaum

13 > 12 ⇒ Suche im rechten Teilbaum

(13) = (13) Ziel erreicht Suche beendet

Suche: (3)

Man vergleicht 3 mit dem Wurzelknoten:

3 < 7

⇒ Suche im linken Teilbaum

3 > 0 ⇒ Suche im rechten T.

3 < 5 ⇒ Suche im linken T.

3 = (3) Ziel erreicht

Suche beendet

Umfang der Suche:

In einem Baum der Höhe H benötigt man maximal
V = H Vergleiche.

Zur Erinnerung: H = Anzahl der Knoten im längsten Weg

Knotenzahl: maximal $N = 2^{H-1}$

Frage: Wieviele Einträge sind z.B. bei 10 Vergleichen speicherbar?

Antwort: Die Zahl der Einträge entspricht der Anzahl der Knoten maximal, also mit H = 10:

$$N = 2^{10} - 1 = 1024 - 1 = 1023$$

— Huffman-Code (1952) (David A. Huffman 1925–1999)

Als eine weitere wichtige Anwendung der Binärbäume kann man die Huffman-Codierung ansehen.
Die primäre Idee bei der Huffman-Codierung ist die, häufig vorkommende Zeichen (z.B. das "e") durch möglichst kurze Codewörter zu codieren. Wenn man jedoch unterschiedlich lange Codes hat, besteht das Problem, dass bei der Rückcodierung der Code nicht mehr eindeutig ist, so könnte es Anfang einer längeren Bitfolge sein. Dies muss ausgeschlossen werden und man muss an einen solchen Code die Forderung stellen, dass kein Codewort Präfix eines anderen Codeworts ist, man fordert einen präfix-freien Code.

Bp.: 0, 10, 11 präfixfrei

0, 01, 10 nicht präfixfrei, da 0 in 01 enthalten

Anschaulich kann man sich die Problematik bei der Telefonnummernvergabe vorstellen:
Jeder Anschluss muss durch seine Telefonnummer eindeutig identifizierbar sein, d.h. beim Wählen darf es zwischendurch nicht schon bei einem anderen Teilnehmer klingeln. So beginnt in Deutschland keine Telefonnummer mit 112, denn dann würde man nach Eingabe der drei Ziffern beim Notruf landen.
Beim Huffman-Code werden in Abhängigkeit von der Häufigkeit des Auftretens der Buchstaben im Text, die Buchstaben mit unterschiedlich langen binären Ziffernfolgen codiert. Das führt zu einem sparsameren Speicherplatzverbrauch.

Frage: Was hat das alles mit einem Binärbaum zu tun?

Zur Darstellung des Codes wird ein Binärbaum verwendet, die Blätter sind die zu codierenden Zeichen, der Pfad von der Wurzel zum Blatt liefert mit seinen Kantenmarkierungen die Bitfolge, die aus 0 und 1 besteht.

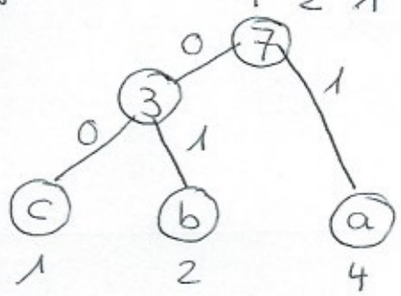
Aufbau des Baumes

Zunächst wird für jeden Buchstaben und jedes Zeichen (auch Leerzeichen) in der zu codierenden Sequenz die (relative) Häufigkeit ermittelt.
Jeder Buchstabe und jedes Zeichen wird zu einem Blatt des Binärbaumes. Man schreibe die Häufigkeit dazu.
Danach werden die beiden Blätter mit der geringsten Häufigkeit zusammengefasst. Der neue innere Knoten enthält die Summe der beiden "Blatthäufigkeiten".
In einem weiteren Schritt werden nun wieder die beiden Knoten mit der nun geringsten Häufigkeit zusammengefasst.

Dabei kann auch ein bereits erzeugter "Innenknoten" in Frage kommen.
Dieser Schritt wird so lange wiederholt, bis alle Buchstaben und Zeichen "abgearbeitet" sind. Der letzte Knoten, der im Binärbaum gezeichnet wird, enthält im Falle von markierten relativen Häufigkeiten die 1, im Falle der markierten absoluten Häufigkeiten, die Anzahl der vorkommenden verschiedenen Buchstaben und Zeichen.

Kleines Beispiel: abacaba soll codiert werden

Häufigkeiten: a b c
4 2 1



c=00 b=01 a=1

Dekodierung 01 11 00 01 01 11 00
b a a c b b a a c

Tipps für das systematische Vorgehen bei der Codierung

Bp: MISSISSIPPI␣IST␣MISSISSIPPI

Wir zählen die Buchstaben:

M:2 I:9 S:9 P:4 T:1 ␣:2

Sortieren der Häufigkeit nach:

T:1
␣:2
M:2
P:4
I:9
S:9
] ③

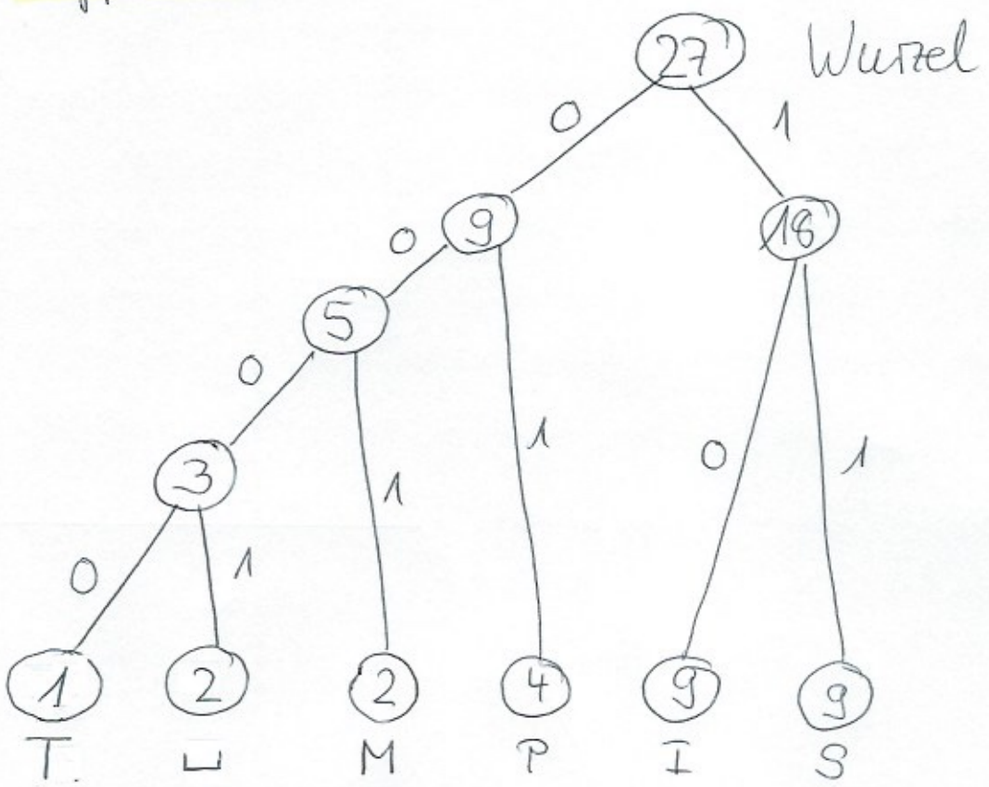
M:2
T␣:3
P:4
I:9
S:9
] ⑤

P:4
MT␣:5
I:9
S:9
] 9

] ⑱

] ㉗

Huffman-Baum:



Codes: 0000 0001 001 01 10 11

Dekodieren Sie: 00110111101101110010110

M I S S I S S I P P I

Anwendung der Huffman-Codierung:

Kompression von Texten in der Fax-Übertragung

Bilddatenkompression für JPEG

Man spricht bei der Huffman-Codierung auch von einer Entropiecodierung, das bedeutet eine Datenkompression ohne Verluste.

Weitere Beispiele zu Binärbäumen:

Darstellung arithmetischer Ausdrücke durch Binärbäume