TGI Zusammenfassung

- Reguläre Sprachen α $L=\{a\}$ mit $a\in\Sigma$ $L=\{\varepsilon\}$ $L=\emptyset$
  $L=L_1\cdot L_2$ $L=L_1\cup L_2$ $L=L_1^*$

  $L(\alpha)=\{\dots\}$

  $* > \cdot > \cup$

Endlicher Automat

Zustände Eingabezeichen Übergangsfunktion Startzustand Endzustände

DEA $A=(Q, \Sigma, \delta: Q\times\Sigma\to Q, s\in Q, F\subseteq Q)$

NEA $(Q, \Sigma, \delta: Q\times(\Sigma\cup\{\varepsilon\})\to 2^Q, s\in Q, F\subseteq Q)$

ε-Abschluss $E(q)=\{p\in Q \mid p$ ist von q durch ε-Übergänge erreichbar$\}$

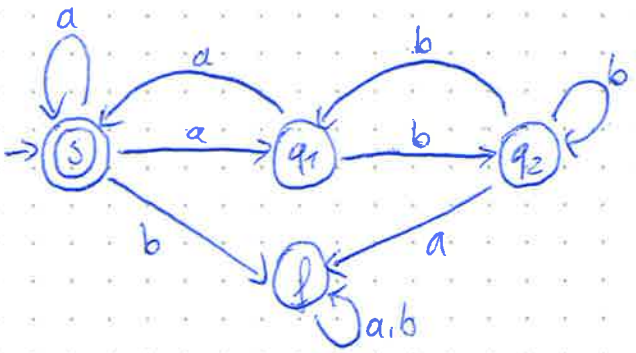
- Satz von Kleene

reguläre Sprache L $\Leftrightarrow$ regulärer Ausdruck α mit $L=L(\alpha)$ $\Leftrightarrow$ NEA A der $L(\alpha)$ erkennt $\Leftrightarrow$ DEA $\bar{A}$, der dieselbe Sprache wie A erkennt

Die von endlichen Automaten ~~erkannten~~ akzeptierten Sprachen sind genau die regulären Sprachen.

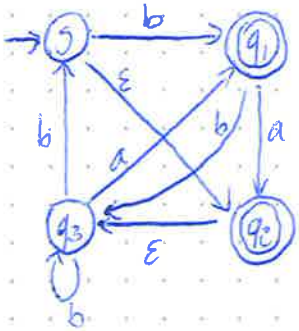
Potenzmengenkonstruktion: NEA → DEA

Zustand	Übergang a	b
$\{s\}$	$\{s,q_1\}$	$\{f\}$
$\{s,q_1\}$	$\{s,q_1\}$	$\{f,q_2\}$
$\{f\}$	$\{f\}$	$\{f\}$
$\{f,q_2\}$	$\{f,s\}$	$\{f,q_1,q_2\}$
$\{f,q_1,q_2\}$	$\{f,s,q_1\}$	$\{f,q_1,q_2\}$
$\{f,s\}$	$\{f,s,q_1\}$	$\{f,q_2\}$
$\{f,s,q_1\}$	$\{f,s,q_1\}$	$\{f,q_2\}$



Endzustände: Alle Zustände, die Endzustände aus NEA enthalten

Entfernen von ε-Übergängen



Zustand	Übergang a	b
s	q_1	s, q_1, q_2, q_3
q_1	q_2, q_3	q_3
q_2	q_1	s, q_2, q_3
q_3	q_1	s, q_2, q_3

Pumping-Lemma: $L = \{a^{2^i} \in \{a,b\}^* \mid i \in \mathbb{N}_0\}$

Erfüllt:
- "$\exists^n$": Wähle $n = 2$
- "$\forall^w$": Betrachte beliebiges $w \in L$ mit $|w| > 2$
- "$\exists^{uvx}$": Wähle Zerlegung $w = uvx$ mit $u = \varepsilon$, $v = aa$, $x = a^{2(j-1)}$
- "$\forall^i$": Für alle $i \in \mathbb{N}_0$: $uv^ix = a^{2i}a^{2(j-1)} = a^{2(i+j-1)} \in L$

Widerlegen:
- "$\forall^n$": $L = \{a^{i^2} \in \{a,b\}^* \mid i \in \mathbb{N}_0\}$
- "$\exists^w$": Betrachte beliebiges $n \in \mathbb{N}$. Sei $m = n+1$
- "$\forall^{uvx}$": Wähle das Wort $w = a^{m^2} \in L$. Es gilt $|w| \geq m > n$
- Betrachte beliebige Zerlegung $w = uvx \in L$ mit $|uv| \leq n$ und $v \neq \varepsilon$
- Beschreibe alle so möglichen Zerlegungen als $a^p a^q a^r$ (u v x) mit $p+q+r = m^2$, $p+q \leq n$ und $1 \leq q \leq n$
- "$\exists^i$": Wähle $i = 2$. Zeige $uv^2x \notin L$

$|a^{m^2}| < |uv^2x| = a^p a^{2q} a^r = |a^{m^2} a^q| \leq |a^{m^2} a^n| < |a^{m^2} a^{2m} a| = |a^{(m+1)^2}|$

Es gibt kein $j \in \mathbb{N}$, sodass $|uv^2x| = a^{j^2}$.

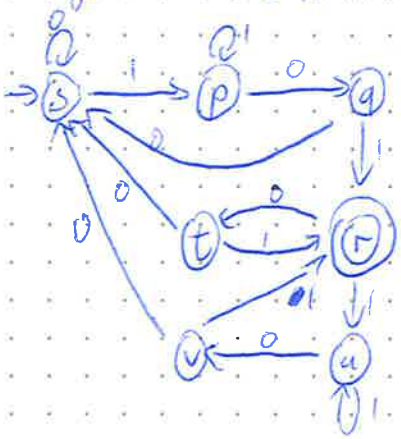
Pumping-Lemma zeigt nicht regularität, kann es aber widerlegen!

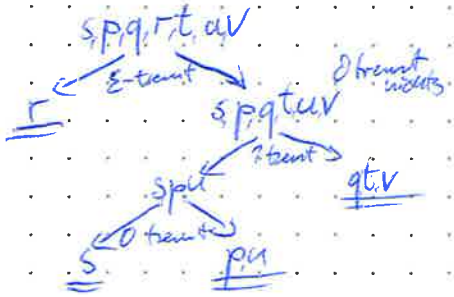
Minimierung von Automaten

Zustände äquivalent $p \equiv q$, wenn $\forall w \in \Sigma^*$ gilt $\delta(p,w) \in F \Leftrightarrow \delta(q,w) \in F$
Mit $[p]$ bezeichnen wir die Äquivalenzklasse der zu p äquivalenten Zustände.

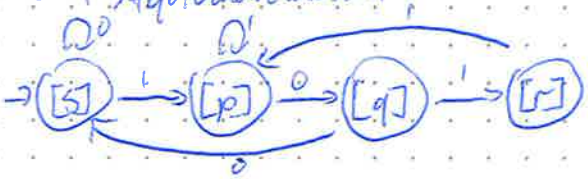
Äquivalenzklassenautomat $A^{\equiv} = (Q^{\equiv} = \{[q] \mid q \in Q\}, \Sigma^{\equiv} = \Sigma, \delta^{\equiv}([q],a) = [\delta(q,a)], s^{\equiv} = [s], F^{\equiv} = \{[f] \mid f \in F\})$

Zeuge w : $\delta(p,w) \in F$ aber $\delta(q,w) \notin F$





$\Rightarrow$ 4 Äquivalenzklassen



~~Nerode Relation~~

Nerode-Relation für alle $x,y \in \Sigma^*$

Rechtsinvariante Äquivalenzrelation R: falls xRy gilt auch $xzRyz$ für alle $z \in \Sigma^*$

Index $ind(R)$ ist Anzahl der Äquivalenzklassen von Σ^* bezüglich R.

Nerode-Relation: Für $x,y \in \Sigma^*$ gilt: $x R_L y \iff (\forall z \in \Sigma^* : xz \in L \Leftrightarrow yz \in L)$

Äquivalenzklasse $[x] = \{y \in \Sigma^* \mid x R_L y\}$

Gültige Suffixe $S_x = \{z \in \Sigma^* \mid xz \in L\}$

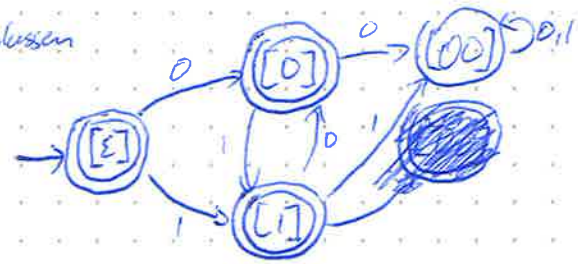
$x R_L y \iff S_x = S_y$

Bsp. $\Sigma = \{0,1\}$, $L = (0 \cup \varepsilon) \cdot (10)^* \cdot (1 \cup \varepsilon)$

$S_\varepsilon = L$
$S_0 = (10)^*(1 \cup \varepsilon)$
$S_1 = (01)^*(0 \cup \varepsilon)$
$S_{00} = S_{11} = \emptyset$
$S_{01} = (01)^*(0 \cup \varepsilon) = S_1$
$S_{10} = (10)^*(1 \cup \varepsilon) = S_0$

$[\varepsilon] = \{\varepsilon, \ldots\} = \{\varepsilon\}$
$[0] = \{0, 10, \ldots\} = 0(10)^* \cup (10)^+$
$[1] = \{1, 01, \ldots\} = 1(01)^* \cup (01)^+$
$[00] = \{00, 11, \ldots\} = (0 \cup 1)^* (00 \cup 11)(0 \cup 1)^*$

$\Rightarrow$ 4 Äquivalenzklassen



Satz von Nerode:

L regulär $\iff ind(R_L) < \infty$

Automat der Nerode-Relation ist minimal

Turing-Maschinen

$M = (Q: \text{Zustände}, \Sigma: \text{Eingabealphabet}, \sqcup \notin \Sigma: \text{Blanksymbol}, \Gamma \supseteq \Sigma \cup \{\sqcup\}: \text{Bandalphabet}, s \in Q: \text{Startzustand}, \delta: Q \times \Gamma \to Q \times \Gamma \times \{L,R,N\}: \text{Übergangsfunktion}, F \subseteq Q: \text{Endzustände})$

Zustand in F $\rightsquigarrow$ M akzeptiert w
Übergang $\delta(q,a) = (q,a,N)$ $\rightsquigarrow$ M bleibt w ab
für ewig unendlich lange $\rightsquigarrow$ M stoppt nicht

Semi-Entscheidbar: TM akzeptiert Eingabe $w \iff w \in L$
Entscheidbar: Semi-Entscheidbar und TM stoppt auf allen Eingaben

L entscheidbar $\iff$ L und L^c sind semi-entscheidbar
$\iff$ L^c ist entscheidbar
L_1 und L_2 entscheidbar $\Rightarrow$ $L_1 \cap L_2$ ist entscheidbar

Diagonalsprache $L_d = \{w \in \{0,1\}^* \mid T_w \text{ akzeptiert } w \text{ nicht}\}$ — L_d nicht entscheidbar, L_d ist semi-entscheidbar
Gödelnummer: injektive Abb aller TM nach $\{0,1\}^*$; $M \to \langle M \rangle \in \{0,1\}^*$, $w \in \{0,1\}^* \to T_w$
Halteproblem $H = \{w \# v \mid w,v \in \{0,1\}^*, T_w \text{ hält bei Eingabe } v\}$ — H nicht Entscheidbar, H nicht semi-Entscheidbar
Universelle TM: bei Eingabe (w,v) wird T_w mit Eingabe v simuliert. H ist semi-entscheidbar
Universelle Sprache $L_u = \{w \# v \mid w,v \in \{0,1\}^*, v \in L(T_w)\}$ — L_u nicht entscheidbar, L_u ist semi-entscheidbar
(T_w hält und akzeptiert bei Eingabe v)

Äquivalenzproblem $L_{\ddot{a}q} = \{w \# v \mid w,v \in \{0,1\}^*, L(T_w) = L(T_v)\}$

Satz von Rice: Sei S eine nicht-triviale Teilmenge von berechenbaren Funktionen.
Dann ist die Sprache $L(S) := \{\langle M\rangle \mid M \text{ berechnet eine Funktion aus } S\}$ nicht entscheidbar.

Komplexitätstheorie

Entscheidungsproblem: Ja/Nein
Optimalwertproblem: Optimalen Wert
Optimierungsproblem: Optimale Lösung

Ein Problem Π und Kodierungsschema $s: D_\Pi \to \Sigma^*$ zerlegen Σ^* in drei Klassen:
Wörter aus Σ^* ... die nicht Kodierung eines Bsp. aus D_Π sind
... die Kodierungen einer Nein-Instanz $I \in N_\Pi$ sind
... die Kodierungen einer Ja-Instanz $I \in J_\Pi$ sind.

Sprache $L[\Pi, s] := \{x \in \Sigma^* \mid \Sigma \text{ ist Alphabet zu } s;\ x \text{ ist Kodierung einer Ja-Instanz } I \text{ von } \Pi \text{ unter } s\ (I \in J_\Pi)\}$

Komplexitätsklassen

P: von DTM in polynomieller Zeit lösbar
NP: von NTM in polynomieller Zeit lösbar

NTM $M = (Q, \Sigma, \sqcup, s, \delta: Q \times (\Gamma \cup \{\varepsilon\}) \to 2^{Q \times \Gamma \times \{L,N,R\}}, F)$
ε-Übergänge, Wahlmöglichkeiten

Orakel-TM: 1. Es wird nichtdeterministisch auf das Band geschrieben. Schreibt Lösungsvorschlag auf das Band.
2. Deterministischer Teil erhält das gesamte Band als Input, überprüft Lösungsvorschlag.

Alle Sprachen in NP sind entscheidbar.

Polynomielle Transformation von Sprache $L_1 \subseteq \Sigma_1^*$ in $L_2 \subseteq \Sigma_2^*$ ist eine Funktion $f: \Sigma_1^* \to \Sigma_2^*$ mit Eigenschaften
- Es existiert eine DTM, die f in Polynomialzeit berechnet
- Für alle $x \in \Sigma_1^*$ gilt: $x \in L_1 \Leftrightarrow f(x) \in L_2$

Wir schreiben dann $L_1 \propto L_2$ (L_1 ist polynomial transformierbar in L_2) NP-Schwer

Ein Sprache L heißt NP-vollständig, falls gilt $L \in NP$ und für alle $L' \in NP$ gilt $L' \propto L$

Die Relation $\propto$ ist transitativ: aus $L_1 \propto L_2$ und $L_2 \propto L_3$ folgt $L_1 \propto L_3$.

Falls $L_1, L_2 \in NP$, $L_1 \propto L_2$ und L_1 ist NP-Vollständig, dann ist auch L_2 NP-vollständig.

Ist $\Pi \in P$?
Zeige $\Pi \propto \Pi'$ für bekanntes $\Pi' \in P$
oder
Konstruiere DTM die Π löst.

Ist $\Pi \in NP$?
Konstruiere DTM die Π löst
(Zeige $\Pi \propto \Pi'$ für bekanntes $\Pi' \in NP$)

Ist Π NP-Schwer
Zeige $\Pi' \propto \Pi$ für bekanntes NP-Schweres Π'

Problem SAT Geg.: Menge U von Variablen, Menge C von Klauseln über U

Frage: Existiert eine Wahrheitsbelegung von U, so dass jede Klausel in C erfüllt wird?

Satz von Cook: SAT ist NP-vollständig.

Approximationsalgorithmen (für Minimierungsproblem ↓)

Absolute Approximation (additiver Fehler): $A(I) \leq OPT(I) + k$ | $A(I) \geq OPT(I) - k$

Relative Approximation (multiplikativer Fehler): $A(I) \leq OPT(I) \cdot k$ | $A(I) \geq \frac{OPT(I)}{k}$

Instanz I optimale Lösung von Wert $OPT(I)$

Algorithmus A liefert Lösung mit Wert $A(I)$

Relativer Approximationsalgorithmus (relative Gütegarantie)
Sei Π ein Optimierungsproblem. Ein polynomieller Algorithmus A, der für jedes $I \in D_\Pi$ einen Wert $A(I)$ liefert mit $R_A(I) \leq K$, wobei $K \geq 1$ eine Konstante

$$R_A(I) := \begin{cases} \frac{A(I)}{OPT(I)} & \text{für Minimierungsprobleme} \\ \frac{OPT(I)}{A(I)} & \text{für Maximierungsprobleme} \end{cases}$$

Wir brauchen für
Minimierungsprobleme
eine obere Schranke für $A(I)$
eine untere Schranke für $OPT(I)$ sodass

$A(I) \leq X$ und $OPT(I) \geq Y$

$\Rightarrow A(I) \leq X = \frac{X \cdot Y}{Y} \leq \frac{X}{Y} \cdot OPT(I)$

Maximierungsprobleme
eine untere Schranke für $A(I)$
eine obere Schranke für $OPT(I)$ sodass

$A(I) \geq X$ und $OPT(I) \leq Y$

$\Rightarrow A(I) \geq X = \frac{X \cdot Y}{Y} \geq \frac{X}{Y} \cdot OPT(I)$

Bestmögliche Approximationsgüte: für polynomialen Approximationsalgorithmus A ist

$$R_A^\infty := \inf \{ r \geq 1 \mid \text{es gibt ein } N_0 > 0, \text{ sodass } R_A(I) \leq r \text{ für alle } I \text{ mit } OPT(I) \geq N_0 \}$$

Approximationsschemata für ein Optimierungsproblem Π ist eine Familie von Algorithmen $\{A_\varepsilon \mid \varepsilon > 0\}$, sodass für alle $\varepsilon > 0$ gilt $R_{A_\varepsilon} \leq 1 + \varepsilon$

Approximationsschemata PTAS hat A_ε polynomielle Laufzeit in $|I|$ ($O(n^{\frac{1}{\varepsilon}})$)
FPTAS hat A_ε polynomiale Laufzeit in $|I|$ und $\frac{1}{\varepsilon}$ ($O(\frac{1}{\varepsilon} \cdot n)$)

Grammatiken $G = (\Sigma, V, S, R)$ mit $V \cap \Sigma = \emptyset$, $S \in V$, $R = \{(l,r) \mid l \in (V \cup \Sigma)^+, r \in (V \cup \Sigma)^*\}$
(Terminalsymbole, Nichtterminale, Startsymbol, Ableitungsregeln)

Chomsky-Hierarchie

		Bsp		
Typ 0 (semi-entscheidbar)	Wortproblem semi-entscheidbar	$G=(\Sigma,V,S,R)$ R beliebig	universelle Sprache	NTM akzeptiert L. DTM akzeptiert L.
entscheidbar				
Typ 1 (kontextsensitiv)	NP-Schwer	$u \to v$, $\lvert u\rvert \leq \lvert v\rvert$ $u \in V^+$, $S \notin v$ $S \to \varepsilon$	$L=\{a^i b^i c^i \mid i \geq 1\}$	NTM mit Platzbedarf in n erkennt Wörter der Länge n in L $\Rightarrow$ NTAPE(n)
Typ 2 (kontextfrei)	polynomiell	$A \to v$, $A \in V$ v beliebig	$L=\{a^i b^i \mid i \geq 1\}$	CYK-Alg. erkennt L in polynom. Zeit. $\Rightarrow$ Chomsky-Normalform	NPDA
Typ 3 (regulär) (rechtslinear)	linear	$A \to v$, $A \in V$ $v \in \{\varepsilon\} \cup \Sigma \cdot V$	$L=\{a^i \mid i \geq 1\}$	NEA erkennt L. DEA erkennt L.	

Linksableitung (Rechtsableitung) ist eine Ableitung, bei der in jedem Schritt die linkeste (rechteste) Variable abgeleitet wird.

Eindeutige kontextfreie Grammatik G: jedes Wort $w \in L(G)$ hat genau einen Syntaxbaum.

Eindeutige kontextfreie Sprache L: eine eindeutige Grammatik G mit $L(G) = L$

Chomsky-Normalform: kontextfreie Grammatik wo Alle Regeln der Form $A \to BC$ oder $A \to a$ ($A, B, C \in V$, $a \in \Sigma$)
Erweiterte Chomsky-Normalform: $S' \to \varepsilon \mid S$

Jede kontextfreie Grammatik kann in eine äquivalente Grammatik in erweiterter Chomsky-Normalform überführt werden.

Schritt 1: ersetze alle $a \in \Sigma$ in Regeln durch neue Variablen Y_a und füge $Y_a \to a$ hinzu.

Schritt 2: ersetze $A \to B_1 \dots B_m$ mit $m > 2$ durch $A \to B_1 C_1$, $C_i \to B_{i+1} C_{i+1}$ für $1 \leq i \leq m-2$, $C_{m-2} \to B_{m-1} B_m$

Schritt 3: Phase 1: Finde die Menge V' aller Variablen A für die $A \overset{*}{\to} \varepsilon$ existiert

Phase 2: Streiche alle Regeln $A \to \varepsilon$; Für $A \to BC$ (behalten) füge ein $A \to B$ falls $C \in V'$, $A \to C$ falls $B \in V'$

Schritt 4: Phase 1: Finde alle Kreise $A_1 \to A_2 \to \dots \to A_n \to A_1$. Ersetze alle A_i durch A_n (rechts und links)

Phase 2: Für jede Regel $A \to B$ und jede Regel $B \to C$ füge Regel $A \to C$ hinzu. Lösche Regel $A \to B$

Falls $S \overset{*}{\to} \varepsilon$ existierte füge Startsymbol S' mit Regeln $S' \to S \mid \varepsilon$ hinzu.

Der CYK-Algorithmus entscheidet für eine Chomsky-Normalform G in $O(|R| \cdot n^3)$ ob ein Wort $w \in L(G)$ ist — Anzahl Regeln $|R|$, $n = |w|$

~~Überprüfe für jede Regel $(A \to BC) \in R$ und jedes k, ob $\dots$~~

Bsp.

$S \to Z_a S \mid Z_a C \mid S Z_a \mid d \mid Z_d E \mid c \mid Z_c F$
$D \to d \mid Z_d E$
$C \to d \mid Z_d E \mid c$
$Z_a \to a \quad Z_c \to c \quad Z_d \to d$
$E \to DD \quad F \to SZ_d$

Input $w = w_1 \dots w_n$ aus Σ^*
Variable $A \in V$ ist in $V_{i,i+j}$ gdw. $A \overset{*}{\Rightarrow} w_i \dots w_{i+j}$
w ist in $L(G) \Leftrightarrow S \in V_{1n}$

Pumping-Lemma für kontextfreie Sprachen

Sei L eine kontextfreie Sprache. Dann existiert eine Zahl $n \in \mathbb{N}$, sodass für jedes Wort $z \in L$ mit $|z| \geq n$ eine Darstellung

$z = uvwxy$ mit $|vwx| \leq n$, $vx \neq \varepsilon$

existiert, bei der auch $uv^i wx^i y \in L$ ist für alle $i \in \mathbb{N}_0$.

Aussage: $\exists n \; \forall z \in L, |z| \geq n \; \exists uvwxy = z, |vwx| \leq n, vx \neq \varepsilon \; \forall i \in \mathbb{N}_0 : uv^i wx^i y \in L$

Widerlegen: $\forall n \; \exists z \in L, |z| \geq n \; \forall uvwxy = z, |vwx| \leq n, vx \neq \varepsilon \; \exists i \in \mathbb{N}_0 : uv^i wx^i y \notin L$

Bsp. $L = \{a^i b^i c^i \mid i \geq 1\}$ ist nicht kontextfrei.

Beweis
"$\forall$" Betrachte beliebiges $n \in \mathbb{N}$
"$\exists$" Wähle $z = a^n b^n c^n$. Beachte: $|z| = 3n \geq n$ und $z \in L$
"$\forall$" Betrachte beliebige Zerlegung $z = uvwxy$, $|vwx| \leq n$, $vx \neq \varepsilon$
"$\exists$" Wähle $i = 0$
Fall 1: vwx enthält kein c, dann ist $uv^0 wx^0 y = a^r b^s c^n \notin L$ weil $r < n$ oder $s < n$.
Fall 2: vwx enthält kein a, dann ist $uv^0 wx^0 y = a^n b^r c^s \notin L$ weil $r < n$ oder $s < n$.

Ogden's Lemma für kontextfreie Sprachen

Sei L eine kontextfreie Sprache. Dann existiert eine Zahl $n \in \mathbb{N}$, sodass für jedes Wort $z \in L$ mit $|z| \geq n$ gilt:
Wenn wir in z mindestens n Buchstaben markieren, so existiert eine Darstellung
$z = uvwxy$
in der von den mindestens n markierten Buchstaben
- höchstens n zu vwx gehören und
- mindestens einer zu vx gehört

bei der auch $uv^i wx^i y \in L$ ist für alle $i \in \mathbb{N}_0$.

Aussage: $\exists n$. $\forall z \in L$, $|z| \geq n$, mind. n Markierungen
$\exists u,v,w,x,y = z$, vwx höchst. n Markierungen, vx mind. 1 Markierung
$\forall i \in \mathbb{N}_0$: $uv^i wx^i y \in L$

Widerlegen: $\forall n$. $\exists z \in L$, $|z| \geq n$, mind. n Markierungen
$\forall u,v,w,x,y = z$, vwx höchst. n Markierungen, vx mind. 1 Markierung
$\exists i \in \mathbb{N}_0$: $uv^i wx^i \notin L$

Bsp. $L = \{a^i b^i c^i \mid i \geq 1\}$ ist nicht kontextfrei:

"$\forall$" Betrachte beliebiges $n \in \mathbb{N}$
"$\exists$" Wähle $z = a^{n+1} b^{n+1} c^{n+1}$ und markiere alle b. (Beachte: $|z| \geq n$, $z \in L$, mind. n Markierungen)
"$\forall$" Betrachte beliebige Zerlegung $z = uvwxy$, sodass
vwx höchstens n Markierungen, vx mindestens 1 Markierung hat
"$\exists$" Wähle $i = 0$
Da vwx kein a oder kein c hat, gilt $uv^0 wx^0 y \notin L$.

Die Klasse der kontextfreien Sprachen ist abgeschlossen bzgl. Vereinigung, Konkatination und Kleeneschem Abschluss, aber nicht bzgl. Komplement und Durchschnitt.
Lässt sich in polynomieller Zeit berechnen.

Nutzlose Variablen: Variable A ist nutzlos, falls es keine Ableitung $S \to^* w \in \Sigma^*$ gibt, in der A vorkommt

Schritt 1: Füge alle $A \in V$ mit $A \to w$ für ein $w \in \Sigma^*$ in Q und V' ein.
Entferne der Reihe nach jedes Element A aus Q
- Ersetze jede Regel $B \to \alpha A \beta$ mit $\alpha, \beta \in (V \cup \Sigma)^*$ durch die Regel $B \to \alpha w \beta$, wobei $w \in \Sigma^*$
- Wenn dabei eine Regel der Form $B \to w'$, $w' \in \Sigma^*$, entsteht und $B \notin V'$, füge B in Q und V' ein

Schritt 2: Starte mit $V'' = \{S\}$
Füge zu allen Regeln $A \to \alpha B \beta$ mit $\alpha, \beta \in (V' \cup \Sigma)^*$, $A \in V''$, $B \in V'$ die Variable B in V'' ein.
Wiederhole bis sich V'' nicht mehr ändert.

Fazit: Am Ende ist V'' die Menge aller nützlichen Variablen.

Für eine kontextfreie Grammatik G kann (in polynomieller Zeit) entschieden werden, ob $L(G) = \emptyset$ ist.

(NPDA) Zustandsalphabet Eingabealphabet Stack-Alphabet Anfangszustand Übergangsrelation
Siebenertupel $(Q, \Sigma, \Gamma, q_0 \in Q, \delta: Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to 2^{Q \times \Gamma^*}, F \subseteq Q)$ Endzustände

$\delta(q,a,Z) \subseteq \{(q,\gamma) \mid q \in Q, \gamma \in \Gamma^*\}$
$\delta(q,\varepsilon,Z) \subseteq \{(q,\gamma) \mid q \in Q, \gamma \in \Gamma^*\}$

Deterministisch (DPDA), falls $|\delta(q,a,Z)| + |\delta(q,\varepsilon,Z)| \leq 1$ für alle $q \in Q, a \in \Sigma, Z \in \Gamma$

Zu einem NPDA, der Sprache L durch einen akzeptierenden Endzustand akzeptiert, kann ein NPDA konstruiert werden, der L mit leerem Stack akzeptiert.

Für eine kontextfreie Grammatik G kann ein NPDA konstruiert werden, der L(G) mit leerem STACK akzeptiert.

Akzeptanz durch leeren Stack: $\exists$ Folge von Konfigurationen $(q_0, w, Z_0) \overset{*}{\to} (q, \varepsilon, \varepsilon)$ mit $q \in Q$

Akzeptanz durch Endzustände: $\exists$ Folge von Konfigurationen $(q_0, w, Z_0) \overset{*}{\to} (f, \varepsilon, \alpha)$ mit $f \in F, \alpha \in \Gamma^*$

Informationstheorie
- Quellkodierung: verlustfreie / verlustbehaftete Datenkompression
- Kanalkodierung: Schutz vor Übertragungsfehlern durch Redundanz über gestörte Kanäle / Fehlerkorrektur
- Kryptographie: Informationssicherheit; Widerstand gegen unbefugtes Lesen / Verändern

Informationsquelle X liefert Zeichen $i \in \Sigma$ mit Wahrscheinlichkeit p_i

Die Information I_p von Wahrscheinlichkeit p (zur Basis b) ist $I_p = \log_b\left(\frac{1}{p}\right) = -\log_b(p)$ (immer b=2)

Rechenregeln Logarithmus

$\log_a(1) = 0 \qquad \log_a(x \cdot y) = \log_a(x) + \log_a(y) \qquad \log_a\left(\frac{1}{x}\right) = -\log_a(x)$

Basiswechsel: $\log_a(x) = \frac{\log_b(x)}{\log_b(a)}$

Entropie ist ein Maß für den mittleren Informationsgehalt pro Zeichen einer Quelle
... Länge unter der ein String nicht komprimiert werden kann
Kolmogorov-Komplexität: Länge eines kürzesten Programms, das String ausgibt
Entropie ist untere Schranke für Kolmogorov-Komplexität.

Die Entropie (zur Basis 2) einer diskreten Zufallsvariable X mit Ergebnissen (Zeichen) in Σ und Wahrscheinlichkeiten $p(a) > 0$ für $a \in \Sigma$, ist definiert durch

$$H(X) = \sum_{a \in \Sigma} p(a) \log_2\left(\frac{1}{p(a)}\right).$$

Bemerkung $H(X) = \sum_{a \in \Sigma} p(a) \cdot I_p(a)$ Entropie der erwarteten Information einer Auswahl von X

Es gilt immer $H(X) \geq 0$ Es gilt $H(X) = 0 \Leftrightarrow p(a) = 1$ für ein $a \in \Sigma$

Präfix-Codes: kein Codewort ist Anfang eines anderen Codeworts; braucht keine Trennzeichen
kann als Baum dargestellt werden

Tiefe $d_T(v)$ eines Knotens v in einem (Kodierungs-)Baum T ist die Anzahl der Kanten auf einem kürzesten Weg von der Wurzel zu v.

Gegeben: Kodierung für Alphabet Σ mit Wahrscheinlichkeit p_i für $i \in \Sigma$
Codewortlänge n_i für $i \in \Sigma$
zugehörigen Kodierungsbaum T

Dann ist ~~mittlere Codewortlänge~~

$$\bar{n} = \sum_{i \in \Sigma} p_i n_i = \sum_{v \in \Sigma} p_v d_T(v)$$

Quellenkodierungstheorem: Codes mit variabler Länge erlaubt; nützlich häufige Zeichen mit kurzen Wörtern kodieren

Shannon's Quellenkodierungstheorem
Sei X eine diskrete endliche Zufallsvariable mit Entropie $H(X)$.
Weiter sei ein Präfix-Code für X mit einem Codealphabet aus D Zeichen und minimaler mittlerer Codewortlänge $\bar{n}$ gegeben. Dann gilt

$$\frac{H(X)}{\log_2 D} \leq \bar{n} < \frac{H(X)}{\log_2 D} + 1$$

Shannon-Fano Kodierung $(c_1, \dots, c_k)$

Zeichen	Wahrscheinlichkeit	Codewort
0	15,91%	000
1	13,88%	001
2	11,25%	010
3	9,96%	011
4	7,96%	1000
5	6,69%	1001
6	5,63%	1010
7	4,73%	1011
8	3,98%	11000
9	2,34%	11001
10	2,81%	11010
11	2,37%	11011
12	1,99%	111000

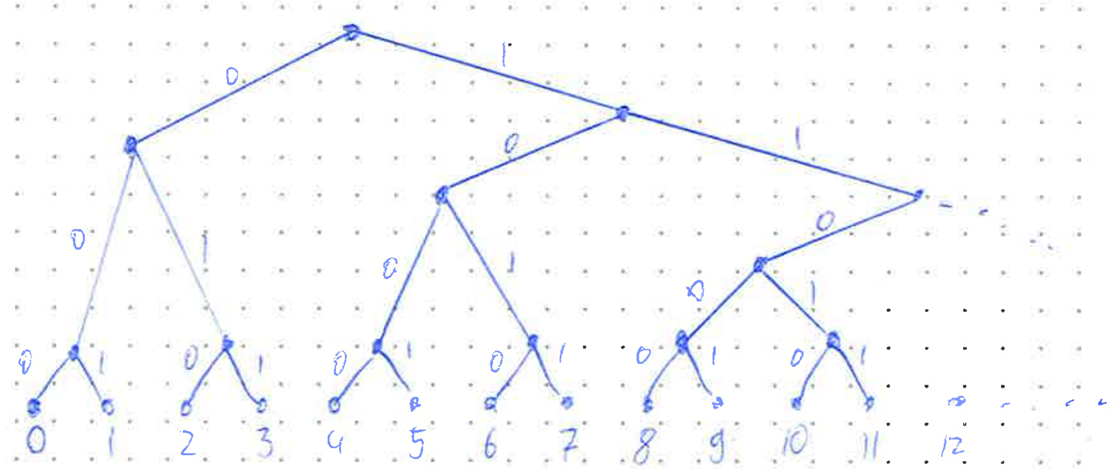
Eingabe: Zeichenliste $Z = (z_1, \dots, z_k)$ mit Wahrscheinlichkeiten $p_1, \dots, p_k$

Wenn $k = 1$ return $(c_1 = \varepsilon)$; exit
Sortiere Zeichen Z absteigend nach Wkt p_i (d.h. $p_1 \geq \dots \geq p_k$)
Trenne Z in
$Z_1 \leftarrow (z_1, \dots, z_l)$
$Z_2 \leftarrow (z_{l+1}, \dots, z_k)$
sodass $|\sum_{i=1}^{l} p_i - \sum_{i=l+1}^{k} p_i|$ minimal ist.

$(c_1, \dots, c_l) \leftarrow (0s_1, \dots, 0s_l)$ mit $(s_1, \dots, s_l) \leftarrow$ ShannonFano(Z_1)

$(c_{l+1}, \dots, c_k) \leftarrow (1s_{l+1}, \dots, 1s_k)$ mit $(s_{l+1}, \dots, s_k) \leftarrow$ ShannonFano(Z_2)

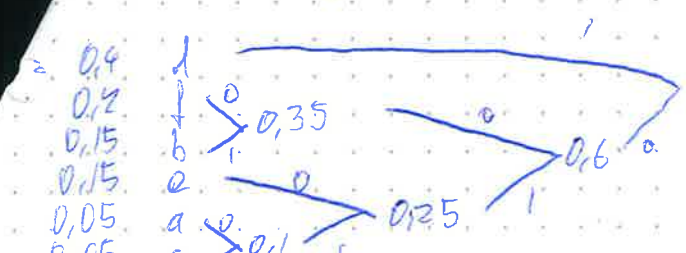
return $(c_1, \dots, c_k)$



Mittlere Codewortlänge von ShannonFano muss nicht optimal sein.

Huffman-Kodierung

Eingabe: Zeichen $1, \dots, n$ mit Wahrscheinlichkeiten $p_1, \dots, p_n$



Zeichen c hat Code 0111
Zeichen e hat Code 010
Zeichen d hat Code 1

Menge $Q = \{1, \dots, n\}$
Füge alle Zeichen aus Q als Blätter in Baum T ein
For $i = 1, \dots, n-1$
- Erzeuge neuen Knoten z für T
- $u \leftarrow$ extrahiere Element x aus Q mit p_x minimal
- Bestimme u als linken Nachfolger von z
- $v \leftarrow$ extrahiere Element x aus Q mit p_x minimal
- Bestimme v als rechten Nachfolger von z
- Wahrscheinlichkeit p_z von z ist $p_u + p_v$
- Füge z in Q ein.

$r \leftarrow$ extrahiere letztes Element aus Q
return r (r ist Wurzel von T)

Der Huffman-Algorithmus berechnet einen Kodierungsbaum mit minimaler mittlerer Codewortlänge.

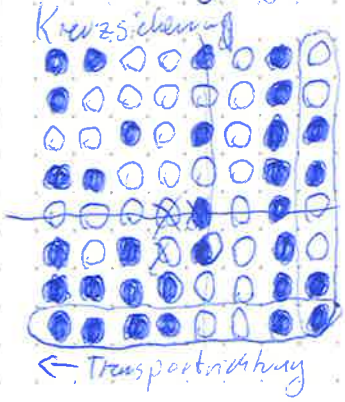
Lauflängenkodierung: Kodiere nicht die Bildpunkte, sondern den Abstand zwischen zwei gemalten Bildpunkten.

- Paritätscodes

⊕	0	1
0	0	1
1	1	0

RS232: 7 Bit Daten, 1 Bit Parität: $b_8 = b_1 \oplus b_2 \oplus b_3 \oplus b_4 \oplus b_5 \oplus b_6 \oplus b_7$
ein Bit-Fehler wird erkannt, kann aber nicht korrigiert rekonstruiert werden

Kreuzsicherung



Dient zum Schutz gegen Doppelfehler
Erklärung am Beispiel Lochkarte
Alle 1,2,3-fachen Fehler sind erkennbar / rekonstruierbar
Ab 4 Fehlern nicht zwingend erkennbar

Paritätscodes: Gegeben ein Alphabet $\Sigma = \{0, 1, \dots, s\}$ und eine Zahl $q \geq s+1$.
Ein Code mit fester Länge n zum Alphabet Σ ist ein Paritätscode, wenn für jedes Codewort $a_1 a_2 \dots a_n$ gilt: $(a_1 + a_2 + \dots + a_n) \bmod q = 0$

- Paritätscodes enthalten höchstens $\frac{1}{q} |\Sigma|^n$ der möglichen $|\Sigma|^n$ Codewörter
- Paritätscodes "verlängern" die Codewörter nur um Faktor $\frac{n+1}{n}$
- Jeder Paritätscode erkennt Einzelfehler

Paritätscode mit Gewichten (gegen Vertauschungsfehler):
Wir führen zusätzlich ganzzahlige Gewichte $w_1, \dots, w_{n-1}$ ein, sodass
$(w_1 \cdot a_1 + w_2 \cdot a_2 + \dots + w_{n-1} \cdot a_{n-1} + a_n) \bmod q = 0$
Zusatzbedingung: Alle Gewichte w_i müssen teilerfremd zu q sein.

Ein Paritätscode mit Gewichten erkennt die Vertauschung an den Stellen i und j, falls die Zahl $w_i - w_j$ teilerfremd zu q ist.

Faltungs-Codes: Codeworte können beliebig lang sein. Die Zeichen sind von Vorgeschehen abhängig.

Block-Codes: Codeworte festen Länge. Aufeinanderfolgende Blöcke werden unabhängig voneinander kodiert.

Hamming-Distanz: Teilmenge $C \subseteq \Sigma^n$ für ein $n \in \mathbb{N}$. Falls $\#C = 1$ heißt C trivial, weil es nur ein Codewort gibt.
zwischen x und $y \in \{0,1\}^n$ ist $d(x,y) := \#\{i \mid i = 1,\dots,n, x_i \neq y_i\}$
Anzahl der Zeichen in x, die sich von denen in y unterscheiden.

Hamming-Kugel $B_r(x)$ um x mit Radius r ist die Menge aller Worte y mit $d(x,y) \leq r$

Sei eine Kodierung C gegeben und y ein empfangenes Wort. Die Maximum-Likelihood-Decoding dekodiert y als dasjenige Codewort $x \in C$, für das $d(x,y)$ minimal wird.

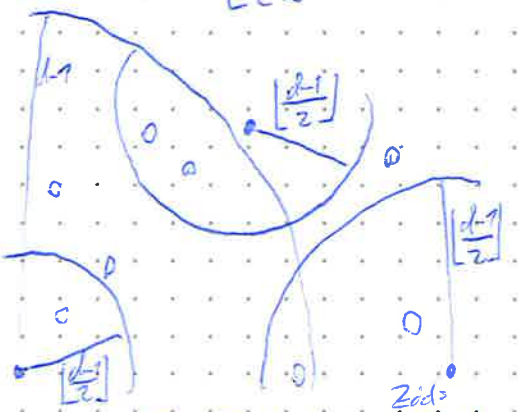
Minimaldistanz eines nichttrivialen Block-Codes C ist $m(C) := \min_{c_1, c_2 \in C, c_1 \neq c_2} d(c_1, c_2)$

Bsp. Ziel: Finde Codeworte C in $\{0,1\}^n$ mit $m(C)$ groß, also paarweise großer Hamming-Distanz

Zeichen	Codewort C
A	00000
H	10011
L	11100
O	01111

Maximum-Likelihood-Decoding
- Empfänger dekodiert 00010 als A, weil $d(00010, 00000) = 1$
- Minimaldistanz ist $m(C) = \min\{3,3,4,4,3,3\} = 3$.
- Es werden 2 Fehler erkannt
- Es kann 1 Fehler korrigiert werden.

Ein Block-Code C mit Minimaldistanz $m(C) = d$ kann bis zu $d-1$ Fehler erkennen und bis zu $\left\lfloor \frac{d-1}{2} \right\rfloor$ Fehler korrigieren.



Finden eines besten Block-Codes: Finde Codeworte C in $\{0,1\}^n$ mit $m(C)$ groß, also paarweise großer Hamming-Distanz

Modellierung als Graph
Sei $Q_n = (\{0,1\}^n, E)$ der n-dimensionale Hyperwürfel mit $xy \in E \Leftrightarrow d(x,y) = 1$
Sei Q_n^{d-1} die $(d-1)$-te Potenz von Q_n
$xy \in E(Q_n^{d-1}) \Leftrightarrow d(x,y) = \text{dist}_{Q_n}(x,y) \leq d-1$
Es ~~existieren~~
Es existiert ein Block-Code C mit $k = |C|$ Worten und $m(C) \geq d$ genau dann wenn Q_n^{d-1} enthält eine unabhängige Menge der Größe k