Grundbegriffe:
Alphabet $\sum$ = endliche Menge, z.B. {0,1}, ASCII, ...
Wort über $\sum$ = endliche Folge von Zeichen aus $\sum$, z.B. 010
Leeres Wort: ϵ
w^n: $w^0 = \epsilon$ und $w^{n+1} = ww^n$
Konkatenation: u und $v \to uv$
Wortlänge: $|w|$
Alle Wörter über $\sum$: $\sum^*$
(Formale) Sprache: $L \subseteq \sum^*$
Sprachen:Regulär $\subset$ kontextfrei $\subset$ entscheidbar $\subset$ alle formalen Sprache

Operationen:
Konkatenation: $\{ab, b\}\{a, bb\} = \{aba, abbb, ba, bbb\}$
Spiegelung: $w = a_1 \dots a_n \to w^R := a_n \dots a_1$
A^n: $\{ab, ba\}^2 = \{abab, abba, baab, baba\}$
A^*: $\{01\}^* = \{\epsilon, 01, 0101, \dots\} \neq \{0, 1\}^*$
A^+: AA^* mit $\epsilon \in A$

Rechenregeln:
$\emptyset A = \emptyset$
$\{\epsilon\}A = A$
$A(B \cup C) = AB \cup AC$
$(A \cup B)C = AC \cup BC$
i.A.: $A(B \cap C) \neq AB \cap AC$
$A^*A^* = A^*$

Deterministische endliche Automaten:
· endliche Zustandsmenge Q
· endlichse Alphabet $\sum$
· totale Übergangsfunktion $\delta : Q \times \sum \to Q$
· Startzustand $q_0 \in Q$
· Endzustände $F \subseteq Q$
$\widehat{\delta}(q, \epsilon) = q$
$\widehat{\delta}(q, aw) = \widehat{\delta}(\delta(q, a), w)$ für $a \in \sum$, $w \in \sum^*$
Jeder Weg ist eindeutig
reguläre Sprache $\Leftrightarrow$ von DFA akzeptiert

Nichtdeterministische endliche Automaten:
Übergangsfunktion: $\delta : Q \times \sum \to P(Q)$ mit $P(Q) = 2^Q$
$\overline{\delta} : P(Q) \times \sum \to P(Q)$
$\overline{\delta}(S, a) := \bigcup_{q \in S} \delta(q, a)$
$\widehat{\overline{\delta}} : P(Q) \times \sum^* \to P(Q)$
$\widehat{\overline{\delta}}(S, w)$ = Menge aller von w erreichbaren Zustände Der Automat "rät" den richtigen Weg, da Wege nicht eindeutig
Jeder von NFA aktzeptierte Sprache ist als DFA darstellbar

NFA mit ϵ-Übergängen:
ϵ-Übergänge verbinden Knoten ohne ein neues Zeichen einzulesen
$\epsilon \notin \sum$
Übergänge: RE $\to$ ϵ-NFA $\to$ NFA $\to$ DFA $\to$ RE und NFA $\to$ RE

Reguläre Ausdrücke:
Ausdrücke: $\emptyset, \epsilon$ für jedes $a \in \sum$ ist a ein Ausdruck
Wenn α und β reguläre Ausdrücke sind, dann auch $\alpha\beta$, $\alpha|\beta$ und α^*
Bindungsstärke: * > Konkatenation > |
Zugehörige Sprache: rekursiv definiert durch
· $L(\emptyset) = \emptyset$:
· $L(\epsilon) = \{\epsilon\}$:
· $L(\alpha) = \{\alpha\}$:
· $L(\alpha\beta) = L(\alpha)L(\beta)$:
· $L(\alpha|\beta) = L(\alpha) \cup L(\beta)$:
· $L(\alpha^*) = L(\alpha)^*$:

Rechnen mit regulären Ausdrücken
Zwei reguläre Ausdrücke sind äquivalent gdw sie die gleiche Sprache darstellen
$\alpha \equiv \beta \Leftrightarrow L(\alpha) = L(\beta)$
$\epsilon \equiv \emptyset^*$ aber $\epsilon \neq \emptyset^*$
· $\emptyset|\alpha \equiv \alpha|\emptyset \equiv \alpha$
· $\emptyset\alpha \equiv \alpha\emptyset \equiv \emptyset$
· $\epsilon\alpha \equiv \alpha\epsilon \equiv \alpha$
· $\emptyset^* \equiv \epsilon$
· $\epsilon^* \equiv \epsilon$
Assoziativität:
· $(\alpha|\beta)|\gamma \equiv \alpha|(\beta|\gamma)$
· $(\alpha\beta)\gamma \equiv \alpha(\beta\gamma)$
Kommutativität:
· $\alpha|\beta \equiv \beta|\alpha$
Distributivität:
· $\alpha(\beta|\gamma) \equiv \alpha\beta|\alpha\gamma$
· $(\alpha|\beta)\gamma \equiv \alpha\gamma|\beta\gamma$
Idempotenz:
· $\alpha|\alpha \equiv \alpha$

Lemma
$\epsilon|\alpha\alpha^* \equiv \alpha^*$
$\alpha^*\alpha \equiv \alpha\alpha^*$
$(\alpha^*)^* \equiv \alpha^*$
Wenn M, M_1, M_2 Mengen von REs:
· $L[M_1M_2] = L[M_1]L[M_2]$
· $L[M_1 \cup M_2] = L[M_1] \cup L[M_2]$
· $L[M^*] = (L[M])^*$

Pumping Lemma
Zeigt ob eine Sprache nicht regulär ist
Sei $R \subseteq \sum^*$ regulär.Dann gibt es ein $n > 0$, so dass sich jedes $z \in R$ mit $|z| \geq n$ so in $z = uvw$ zerlegen lässt, dass
· $v \neq \epsilon$
· $|uv| \leq n$, und
· $\forall i \geq 0. uv^iw \in R$

Entscheidbarkeit
Eine Eigenschaft ist entscheidbar gdw. die zugehörige Menge entscheidbar ist
Reguläre Sprachen sind in linearer Zeit $O(|w|)$ entscheidbar
Eingabe: Wort+Automat $\to$ entscheidbar in $O(|Q|^2|w|)$
NFA:
· Leerheitsproblem: $O(|Q|^2|\sum|)$
· Äquivalenzproblem: $O(2^{|Q_1|+|Q_2|})$
DFA:
· Leerheitsproblem: $O(|Q||\sum|)$
· Endlichkeitsproblem: entscheidbar
· Äquivalenzproblem: $O(|Q_1||Q_2|)$

Automaten und Gleichungssysteme
Ardens Lemma: $X = AX \cup B \Rightarrow X = A^*B$
Umwandeln von DFA $\to$ regulärer Ausdruck:
· X_i für alle Zustände
· $X_1 = aX_2|bX_3$ falls $X_1 \to X_2 = a$ und $X_1 \to X_3 = b$
· $X_3 = aX_2|bX_3|\epsilon$ falls X_3 zusätzlich ein Endzustand
· auflösen nach X_i
$\begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} aX_1 & bX_2 \\ cX_1 & dX_2 \end{pmatrix}$
$A^* := A^0 + A^1 + A^2 + \dots$
$A(i, j)$ = Alle Wege von i nach j der Länge 1
$A^n(i, j)$ = Alle Wege von i nach j der Länge n
$A^*(i, j)$ = Alle Wege von i nach j endlicher Länge

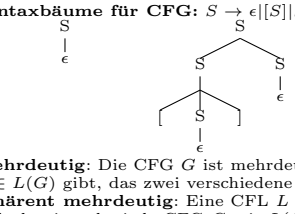
Minimieren endlicher Automaten
· Entferne alle von q_0 nicht erreichbaren Zustände
· Äquivalente Zustände berechnen
· p und q sind unterscheidbar $\Leftrightarrow \widehat{\delta}(p, w) \in F$ und $\widehat{\delta}(q, w) \notin F$
· Alle Äquivalenten Zustände zusammenfassen
Berechnung äquivalenter Zustände eines DFA
for all $p \in F, q \in Q \backslash F$
  do markiere $\{p, q\}$
while $\exists$ unmarkiertes $\{p, q\} \exists a \in \sum . \{\delta(p, a), \delta(q, a)\}$ ist markiert
  do markiere $\{p, q\}$
Äquivalenzrelation
Eine Relation $\approx \subseteq A \times A$ ist eine Äquivalenzrelation falls
· Reflexivität: $\forall a \in A. a \approx a$
· Symmetrie: $\forall a, b \in A. a \approx b \Rightarrow b \approx a$
· Transitivität: $\forall a, b, c \in A. a \approx b \wedge b \approx c \Rightarrow a \approx c$
Quotientenautomat
$A/\equiv := (Q/\equiv, \sum, \delta', [q_0]_\equiv, F/\equiv)$
$\delta'([p]_\equiv, a) := [\delta(p, a)]_\equiv$ A ist ein minimaler DFA für $L(A)$ falls A keine unerreichbaren Zustände enthält

Kanonischer Minimalautomat
$M_L := (\sum^* / \equiv_L, \sum, \delta_L, [\epsilon]_{\equiv_L}, F_L)$
mit $\delta_L([w]_{\equiv_L}, a) := [wa]_{\equiv_L}$ und $F_L := \{[w]_{\equiv_L} | w \in L\}$

De Morgan'sche Gesetze
· $\neg(a \wedge b) = \neg a \vee \neg b$
· $\neg(a \vee b) = \neg a \wedge \neg b$

Kontextfreie Sprachen
Eine kontextfreie Grammatik $G = (V, \sum, P, S)$ erzeugt die Sprache $L(G) := \{w \in \sum^* | S \to_G^* w\}$
CFG: Kontextfreie Grammatik
CFL: Kontextfreie Sprache
Arithmetische Ausdrücke:
· <Expr> $\to$ <Term>
· <Expr> $\to$ <Expr> + <Term>
· <Term> $\to$ <Factor>
· <Term> $\to$ <Term> * <Factor>
· <Factor> $\to$ <a>
· <Factor> $\to$ (<Expr>)
Nicht entscheidbar für CFGs
· Äquivalenz $L(G_1) = L(G_2)$?
· Schnittproblem $L(G_1) \cap L(G_2) = \emptyset$?
· Regularität: $L(G)$ regulär?
· Mehrdeutigkeit:Ist G mehrdeutig?
· V endliche Menge, die Nichtterminalzeichen (oder Variablen)
· $\sum$ ist ein Alphabet, die Terminalzeichen, disjunkt von V
· $P \subseteq V \times (V \cup \sum)^*$ eine endliche Menge, die Produktion, und
· $S \in V$, ist das Startsymbol
Konventionen
· $A, B, C, \dots$ sind Nichtterminale
· $a, b, c, \dots$ (und Sonderzeichen wie $+, *, \dots$) sind Terminale
· $\alpha, \beta, \gamma, \dots \in (V \cup \sum)^*$
· Produktionen: $A \to \alpha$ statt $(A, \alpha) \in P$
· Statt $A \to \alpha_1, A \to \alpha_2, A \to \alpha_3$ schreibt man $A \to \alpha_1|\alpha_2|\alpha_3$,
Ableitungsrelationen $\to_G$ auf Wörter über $V \cup \sum$: $\alpha \to_G \beta$ gdw es eine Regel $A \to \gamma$ in P gibt, und Wörter α_1, α_2, so dass $\alpha = \alpha_1 A \alpha_2$ und $\beta = \alpha_1\gamma\alpha_2$
Rechtslinear: $A \to aB$ oder $A \to \epsilon$
Linkslinear: $A \to Ba$ oder $A \to \epsilon$
Vorkommen $\#_a(w)$:=Anzahl der a's in w
$u \prec w :\Leftrightarrow \exists v. uv = w$
Balanciert: Anzahl aller vorkommenden Zeichen gleich

Syntaxbäume für CFG: $S \to \epsilon|[S]|SS$



Mehrdeutig: Die CFG G ist mehrdeutig, gdw es ein $w \in L(G)$ gibt, das zwei verschiedene Syntaxbäume hat.
Inhärent mehrdeutig: Eine CFL L heißt inhärent mehrdeutig, gdw jede CFG G mit $L(G) = L$ mehrdeutig ist

Chomsky-Normalform (CN)
Eine kontextfreie Grammatik G ist in Chomsky-Normalform gdw alle Produktionen eine der Formen $A \to a$ oder $A \to BC$ haben
ϵ-Produktion: $A \to \epsilon$
Ketten-Produktion: $A \to B$
Konstruktion einer CN
· Eliminiere alle ϵ-Produktionen
· Eliminiere alle Ketten-Produktionen
· Füge für jedes $a \in \sum$, das in einer rechten Seite der Länge ≥ 2 vorkommt, ein neues Nichtterminal A_a zu V hinzu, ersetze a in allen rechten Seiten der Länge ≥ 2 durch A_a, und füge $A_a \to a$ zu P hinzu.
· Ersetze jede Produktion der Form $A \to B_1B_2 \dots B_k (k \geq 3)$ durch $A \to B_1C_2, C_2 \to B_2C_3, \dots, C_{k-1} \to B_{k-1}B_k$, wobei $C_2, \dots, C_{k-1}$ neue Nichtterminale sind.

Greibach-Normalform
Eine CFG ist in Greibach-Normalform, falls jede Produktion von der Form $A \to aA_1 \dots A_n$ ist.

Pumping-Lemma (CFL)
Für jede kontextfreie Sprache L gibt es ein $n \geq 1$, so dass sich jedes Wort $z \in L$ mit $|z| \geq n$ zerlegen lässt in $z = uvwxy$ mit
· $vx \neq \epsilon$,
· $|vwx| \leq n$, und
· $\forall i \in N. uv^iwx^iy \in L$.

Algorithmen für CFG
Ein Symbol $X \in V \cup \sum$ ist
· **nützlich** gdw es eine Ableitung $S \to_G^* w \in \sum^*$ gibt, in der X vorkommt
· **erzeugend** gdw es eine Ableitung $X \to_G^* w \in \sum^*$ gibt
· **erreichbar** gdw es eine Ableitung $S \to_G^* \alpha X \beta$ gibt
Nützliche Symbole sind erzeugend und erreichbar, aber nicht notwendigerweise umgekehrt.

Cocke-Younger-Kasami-Algorithmus (CYK)
Entscheidet das Wortproblem für kontextfreie Grammatiken in Chomsky-Normalform in $O(|w|^3)$.
Definition: $V_{ij} := \{A \in V | A \to_G^* a_i \dots a_j\}$ für $i \leq j$

Chomsky Hierarchie
Typ 0:
Typ 1: falls $|\alpha| \leq |\beta|$
Typ 2: falls G vom Typ 1 ist und $\alpha \in V$
Typ 3: falls G vom Typ 2 ist und $\beta \in \sum \cup \sum V$

Berechenbarkeit
$f : N^k \to N$ ist intuitiv berechenbar, wenn es einen Algorithmus gibt, der bei Eingabe $(n_1, \dots, n_k) \in N^k$
· nach endlich vielen Schritten mit Ergebnis $f(n_1, \dots, n_k)$ hält, falls $f(n_1, \dots, n_k)$ definiert ist,
· und nicht terminiert, falls $f(n_1, \dots, n_k)$ nicht definiert ist.
Eine Funktion $f : A \to B$ ist
· **total** gdw $f(a)$ für alle $a \in A$ definiert ist
· **partiell** gdw $f(a)$ auch undefiniert sein kann
· **echt partiell** gdw sie nicht total ist

Turingmaschinen
Eine TM ist ein 7-Tupel $M = (Q, \sum, \Gamma, \delta, q_0\Box, F)$
· Q ist eine endliche Menge von Zuständen
· $\sum$ ist eine endliche Menge, das Eingabealphabet
· Γ ist eine endliche Menge, das Bandalphabet, mit $\sum \subset \Gamma$
· $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R, N\}$ ist die Übergangsfunktion
· δ darf partielle sein
· $q_0 \in Q$ ist der Startzustand
· $\Box \in \Gamma \backslash \sum$ ist das Leerzeichen
· $F \subseteq Q$ ist die Menge der akzeptierenden oder Endzustände
nichtdeterministische TM
$\Leftrightarrow \delta : Q \times \Gamma \to P(Q \times \gamma \times \{L, R, N\})$
Ablauf: $\delta(q, a) = (q', b, d)$ M befindet sich im Zustand q und liest a ein. Danach geht M in den Zustand q', überschreibt a mit b und bewegt den Schreib-/Lesekopf nach rechts (falls $d = R$), nach links (falls $d = L$), oder nicht (falls $d = N$).
Konfiguration: Ein Tripel $(\alpha, q, \beta) \in \Gamma^* \times Q \times \Gamma^*$
Startkonfiguration: Bei Eingabe $w \in \sum^*$: (ϵ, q_0, w)
$(\alpha, q, \beta) =$
$\begin{cases} (\alpha, q', c\ \text{rest}(\beta)), \text{falls } d = N \\ (\alpha c, q', \text{rest}(\beta)), \text{falls } d = R \\ (\text{butlast}(\alpha), q', \text{last}(\alpha)c\ \text{rest}(\beta)), \text{falls } d = L \end{cases}$
Für $a \in \Gamma$ und $w \in \Gamma^*$ gilt:
· first(aw)=a first(ϵ)=$\Box$
· rest(aw)=w rest(ϵ)=ϵ
· last(wa)=a last(ϵ)=$\Box$
· butlast(wa)=w butlast(ϵ)=ϵ
Von TM akzeptierte Sprache: $L(M) = \{w \in \sum^* | \exists q \in F, \alpha, \beta \in \Gamma^*. (\epsilon, q_0, w) \to_M^* (\alpha, q, \beta)\}$
akzeptierte Sprache $\Leftrightarrow$ Typ-0-Sprachen Chomsky Hierarchie
Turing-berechenbar:
· Eine Funktion $f : N^k \to N$ ist Turing-berechenbar, gdw es eine TM M gibt, so dass für alle $n_1, \dots, n_k, m \in N$ gilt $f(n_1, \dots, n_k) = m \Leftrightarrow \exists r \in F. (\epsilon, q_0, \text{bin}(n_1)\#\text{bin}(n_2)\#\dots\#\text{bin}(n_k)) \to_M^* (\Box \dots \Box, r, \text{bin}(m)\Box \dots \Box)$
· Eine Funktion $f : \sum^* \to \sum^*$ ist Turing-berechenbar, gdw es eine TM M gibt, so dass für alle $u, v \in \sum^*$ gilt $f(u) = v \Leftrightarrow \exists r \in F. (\epsilon, q_0, u) \to_M^* (\Box, \dots \Box, r, v\Box, \dots \Box)$
Hintereinanderschaltung:
$M := (Q_1 \cup Q_2, \sum, \Gamma_1 \cup \Gamma_2, \delta, q_1, \Box, F_2)$

LOOP-, WHILE- & GOTO-Berechenbarkeit

Aufbau: Loop=PR $\subset$ total $\subset$ berechenbar = TM = WHILE = GOTO = μR
Modifizierte Differenz:
$m \dot{-} n = \begin{cases} m-n & \text{falls } m \geq n \\ 0 & \text{sonst} \end{cases}$
LOOP-Semantik: LOOP x_i DO P END
Führe P genau n mal aus, wobei n der Anfangswert von x_i ist. Zuweisungen an x_i in P ändern die Anzahl der Schleifendurchläufe nicht!
LOOP-berechenbar: gdw es ein LOOP-Programm P gibt, das für $n_1, \ldots, n_k$ terminiert
LOOP-berechenbar: $\Rightarrow$ totale Funktion
WHILE-Programme: LOOP-Programme erweitert um WHILE-Schleifen: $P \rightarrow$ WHILE $X \neq 0$ DO P END
GOTO-Programm: Sequenzen von markierten Anweisungen $M_1 : A_1; M_2 : A_2; \ldots; M_k : A_k$
Übersetzungen:
· LOOP $\rightarrow$ WHILE $\rightarrow$ TM $\rightarrow$ GOTO
· WHILE $\leftrightarrow$ GOTO

Primitiv rekursive Funktionen (PR)
Definition: Die Menge der PR ist die folgende induktiv definierte Teilmenge aller Funktionen $N^k \rightarrow N, k \geq 0$
· Die konstante Funktion: 0
· Die Nachfolgerfunktion: $s(n) = n + 1$
· Die Projektionsfunktion:
$\pi_i^k : N^k \rightarrow N, 1 \leq i \leq k : \pi_i^k(x_1, \ldots, x_k) = x_i$
Aus den Funktionen g und h mit $\overline{x} = (x_1, \ldots, x_n)$ wird die Funktion
· $f(0, \overline{x}) = g(\overline{x})$
· $f(m + 1x\overline{x}) = h(f(m, \overline{x}), m, \overline{x})$
· Die Basisfunktionen $0, s$ und π_i^k sind PR
· Sind g und h_i PR, dann auch
$f(\overline{x}) = g(h_1(\overline{x}), \ldots, h_k(\overline{x}))$
primitiv-rekursive Funktion $\Rightarrow$ totale Funktion
beschränkter max-Operator:
$\max\{x \leq n | P(x)\} =: q(n)$ **beschränkter**
Existenzquantor: $\exists x \leq n.P(x) =: Q(x)$

PR = LOOP
Hauptproblem: Kodierung aller Variablen eines LOOP-Programms in einer Zahl
Cantorsche Paarungsfunktion: Bijektion zwischen N^2 und N:
$c(x, y) := \binom{x+y+1}{2} + x = (x + y)(x + y + 1)/2 + x$
PR-berechenbare Fkt $\Leftrightarrow$ LOOP-berechenbare Fkt

μ-rekursive Funktionen
Mit PR kann man nur beschränkt suchen $n, \ldots, 0$ und nicht unbeschränkt $0, \ldots$
Der μ-Operator ermöglicht $f(n) = \ldots f(n + 1) \ldots$
Notation: $f(n) = \perp$ bedeutet "$f(n)$ ist undefiniert"
$f : N^{k+1} \rightarrow N$ und $\mu f : N^k \rightarrow N : \overline{x} \rightarrow$
$\begin{cases} min\{n \in N | f(n, \overline{x}) = 0\} & \text{falls } \exists n \text{ und } \forall m \leq n : f(m, \overline{x}) \neq \perp \\ \perp & \text{sonst} \end{cases}$
Menge der μ-rekursiven Funktionen: Die kleinste Teilmenge aller Funktionen, die die Basisfunktionen $(0, +1, \pi_i^k)$ enthält und alle Funktionen die man durch wiederholte Anwendung erhalten kann
μ-berechenbare Fkt $\Leftrightarrow$ WHILE-berechenbare Fkt

Ackermann-Funktion
· $a(0, n) = n + 1$
· $a(m + 1, 0) = a(m, 1)$
· $a(m + 1, n + 1) = a(m, a(m + 1, n))$

Entscheidbarkeit und das Halteproblem
Die Menge $A \subseteq N$ ist entscheidbar gdw ihre charakteristische Funktion $\chi_A(x) := \begin{cases} 1 & \text{falls } x \in A \\ 0 & \text{falls } x \notin A \end{cases}$
$M[w]$**:** "Maschine M mit Eingabe w"
$M[w] \downarrow$**:** $M[w]$ terminiert

Spezielles Halteproblem:
Das Spezielle Halteproblem ist nicht entscheidbar
· **Gegeben:** Ein Wort $w \in \{0, 1\}^*$
· **Problem:** Hält M_w bei Eingabe w?
· Als Menge: $K := \{w \in \{0, 1\}^* | M_w[w] \downarrow\}$

Allgemeines Halteproblem:
Das Halteproblem ist nicht entscheidbar
· **Gegeben:** Wörter $w, x \in \{0, 1\}^*$
· **Problem:** Hält M_w bei Eingabe x?
· Als Menge: $H := \{w\#x | M_w[x] \downarrow\}$

Semi-Entscheidbarkeit
Eine Menge $A \subseteq N$ heißt semi-entscheidbar (s-e) gdw $\chi_A(x) := \{1 \quad \text{falls } x \in A \perp \quad \text{falls } x \notin A$ **Rekursiv aufzählbar:** Eine Menge A ist rekursiv aufzählbar, gdw $A = \emptyset$ oder es eine berechenbare totale Fkt $f : N \rightarrow A$ gibt, so dass $A = \{f(0), f(1), f(2), \ldots\}$ **Rekursiv aufzählbar** $\Leftrightarrow$ s-e

---

Satz von Rice
Sei F eine Menge berechenbarer Funktionen. Es gelte weder $F = \emptyset$ noch $F =$ alle ber. Funkt. ("F nicht trivial") Dann ist unentscheidbar, ob die von einer gegebenen TM M_w berechnete Funktion Element F ist, dh ob $\varphi_w \in F$.

Satz von Rice-Shapiro
Sei F eine Menge berechenbarer Funktionen. Ist $C_F := \{w | \varphi_w \in F\}$ semi-entscheidbar, so gilt für alle berechenbaren $f : f \in F \Leftrightarrow$ es gibt eine endlich Teilfunktion $g \subseteq f$ mit $g \in F$.

Das Postsche Korrespondenzproblem (PCP)
· **Gegeben:** Eine endliche Folge $(x_1, y_1), \ldots, (x_k, y_k)$, wobei $x_i, y_i \in \sum^+$
· **Problem:** Gibt es eine Folge von Indizes $i_1, \ldots, i_n \in \{1, \ldots, k\}$, $n > 0$, mit $x_{i_1} \ldots x_{i_n} = y_{i_1} \ldots y_{i_n}$?
Modifiziertes PCP, MPCP: Gibt es eine Lösung mit $i_1 = 1$?

Unentscheidbare CFG Probleme
Für DFAs: ist fast alles entscheidbar
Für TM: ist fast nichts entscheidbar
Für CFGs: ist manches entscheidbar
Folgende Probleme sind unentscheidbar:
· Ist $L(G_1) \cap L(G_2) = \emptyset$?
· Ist $|L(G_1) \cap L(G_2)| = \infty$?
· Ist $L(G_1) \cap L(G_2)$ kontextfrei?
· Ist $L(G_1) \subset L(G_2)$?
· Ist $L(G_1) = L(G_2)$?

Komplexitätstheorie
Komplexitätsklasse P
Sind die Sprachen, bei denen $w \in L$ schnell entschieden werden kann
DTM: deterministische Mehrband-TM
NTM: nichtdeterministische Mehrband-TM
time$_M(w)$**:** Anzahl der Schritte bis die DTM $M[w]$ hält
Komplexitätsklasse NP
Sind die Sprachen, bei denen ein Zertifikat für $w \in L$ schnell verifiziert werden kann.
NP die Klasse der Sprachen, die von einer NTM in polynomieller Zeit akzeptiert werden.
$\text{ntime}_M(w) :=$
$\begin{cases} \text{minimale Anzahl der Schritte bis NTM } M[w] \text{ akzeptiert} \\ \quad \text{falls } w \notin L(M) \\ 0 \quad \text{falls } w \notin L(M) \end{cases}$
Sei M eine DTM mit $L(M) \subseteq \{w\#c | w \in \sum^*, c \in \Delta^*\}$
· Falls $w\#c \in L(M)$, so heißt c Zertifikat für w
· M ist ein polynomiell beschränkter Verifikator für die Sprache $\{w \in \sum^* | \exists c \in \Delta^*.w\#c \in L(M)\}$, falls es ein Polynom p gibt, so dass $\text{time}_M(w\#c) \leq p(|w|)$.
Übersicht: P $\subset$ NP $\subset$ LOOP $\subset$ entscheidbar $\subset$ semi-entscheidbar $\subset$ Alle Sprachen
NP-Vollständig
Wenn $A \subseteq \sum^*$ und $B \subseteq \Gamma^*$, dann heißt A polynomiell reduzierbar auf B, gdw es eine totale und von einer DTM in polynomieller Zeit berechenbare Funktion $f : \sum^* \rightarrow \Gamma^*$ gibt, so dass für alle $w \in \sum^*$ gilt:
$w \in A \Leftrightarrow f(w) \in B$
NP-Hart: Eine Sprache L heißt NP-hart gdw $A \leq_p L$ für alle $A \in NP$
NP-Vollständig: Eine Sprache L heißt NP-vollständig gdw L NP-hart ist und $L \in$ NP

Aussagenlogik
· Formeln: $F \rightarrow \neg F | (F \wedge F) | (F \vee F) | X$
· Variablen: $X \rightarrow x | y | z | \ldots$
$\wedge$ bindet stärker als $\vee$
Belegung: $\sigma = \{x \mapsto 0, y \mapsto 1, z \mapsto 0, \ldots\}$
Erfüllbar: Eine Formel F ist erfüllbar, wenn es eine Belegung σ gibt, mit $\sigma(F) = 1$
Entscheidbarkeit von SAT: in $O(f(n))$
Berechnung einer erf. Bel.: in $O(n \cdot (f(n) + n))$
Reduzierung der Berechnung einer erfüllenden Belegung auf SAT:
if $F \notin$ SAT **then** output("nicht lösbar")
else
  for $i := 1$ **to** k **do**
    if $F[x_i := 0] \in$ SAT **then** $b := 0$ **else** $b := 1$
    output(x_i "=" b)
    $F := F[x_i := b]$
wobei $F[x := b] = F$ mit x ersetzt durch b.
· Eine Formel ist in **Konjunktiver Normalform (KNF):** gdw sie eine Konjunktion von Klauseln ist:
$K_1 \wedge \ldots \wedge K_n$
· Eine **Klausel** ist eine Disjunktion von Literalen:
$L_1 \vee \ldots \vee L_m$
· Ein **Literal** ist eine (evtl. negierte) Variable
· Eine Formel ist in 3KNF gdw jede Klausel ≤ 3 Literale enthält

---

Mengenüberdeckung
Gegeben: Teilmengen $T_1, \ldots, T_n \subseteq M$ einer endlichen Menge M und eine Zahl $k \leq n$
Gegeben: Gibt es $i_1, \ldots, i_k \in \{1, \ldots, n\}$ mit $M = T_{i_1} \cup \cdots \cup T_{i_k}$?
Minimierungsproblem: Finde das kleinste k, so dass M von k der Teilmengen überdeckt wird.
Berechnung:
if $(\overrightarrow{T}, M, k) \notin$ MÜ **then** output("nicht lösbar")
else
  Ü $:= \emptyset$
  for $i := 1$ **to** n **do**
    if $(\overrightarrow{T} - T_i, M, k) \in$ MÜ
    then $\overrightarrow{T} := \overrightarrow{T} - T_i$
    else Ü $:=$ Ü $\cup \{T_i\}$
