- Für Typ 2-Grammatiken, d.h. kontextfreie Grammatiken, lässt sich das Wortproblem (Eingabe: ein Wort w Frage: Wird w von der Grammatik erzeugt?) mittels des CYK-Algorithmus in Zeit $O(n^3)$ lösen. Somit gilt für die Menge KFG aller kontextfreien Sprachen:

$$\text{KFG} \subseteq \text{DTIME}(n^3) \subsetneq \text{P}$$

Außerdem ist bekannt (hier ohne Beweis), dass

$$\text{KFG} \subseteq \text{SPACE}((\log n)^2)$$

(Lewis, Stearns, Hartmanis, 1965)

Bzgl. Typ 3-Grammatiken, d.h. reguläre Grammatiken, zeigen wir:

<u>Satz 4.28</u>

Für jedes $S: \mathbb{N} \to \mathbb{N}$ mit $S(n) = o(\log\log n)$ und jedes $L \in \text{SPACE}(S)$ gilt: L ist regulär.

Somit ist $\text{SPACE}(S) = \text{SPACE}(0) = \{L \subseteq \{0,1\}^* : L \text{ regulär}\}$.

(kurz: "$\text{SPACE}(o(\log\log n)) = \text{REG}$")

<u>Beweis:</u>

Sei $S: \mathbb{N} \to \mathbb{N}$ und sei $L \in \text{SPACE}(S)$.

Sei M eine $S(n)$-platzbeschränkte DTM, die L entscheidet.

Wegen "linearer Kompression" (Satz 4.9) können wir OBdA annehmen, dass M eine 2-Band DTM ist.

Behauptung 1:
Falls es ein $k \in \mathbb{N}$ gibt mit $S(n) \leq k$ f.a. $n \in \mathbb{N}$,
so ist L regulär.

Beweis:
Wegen $S(n) \leq k$ f.a. $n \in \mathbb{N}$, kann der gesamte Inhalt des Arbeitsbandes auch in einer konstanten Anzahl von Zuständen gespeichert werden.
Daher können wir OBdA annehmen, dass M eine 1-Band DTM mit read-only Eingabeband ist.
D.h. M ist ein sog. deterministischer 2-Wege-Automat.

Solche det. 2-Wege-Automaten lassen sich durch herkömmliche NFAs (ndet. endliche Automaten) simulieren. Details: Übung!

$\triangleright_{\text{Beh. 1}}$

Sei nun $S(n) = o(\log\log n)$.

Angenommen, L ist nicht regulär.

Wegen Beh. 1 muss es dann für jedes $k \in \mathbb{N}$ ein Wort $x^{(k)} \in \{0,1\}^*$ geben, bei dessen Eingabe M mehr als k Zellen des Arbeitsbandes benutzt.
Sei $x^{(k)}$ ein Wort minimaler Länge, für das dies gilt, und sei $n_k := |x^{(k)}|$ die Länge von $x^{(k)}$.

klar: $n_1 \leq n_2 \leq n_3 \leq \cdots$ und $n_k \xrightarrow[k \to \infty]{} \infty$.

Wir betrachten im Folgenden für jedes $x \in \{0,1\}^*$ die Berechnung $B_M(x)$, die als die Folge von Konfigurationen definiert ist, die M bei Eingabe x durchläuft.

Hier:

Jede Konfiguration C besteht aus

- dem aktuellen Zustand von M
- der aktuellen Kopfposition i auf dem Eingabeband
- dem aktuell auf dem Eingabeband gelesenen Symbol $x_i \in \{0,1,\triangleright,\triangleleft\}$
- der aktuellen Kopfposition auf dem Arbeitsband
- der aktuellen Beschriftung des Arbeitsbandes

Bei einer Eingabe x der Länge n gibt es für jede feste Kopfposition i auf dem Eingabeband höchstens

$$N_n := |Q| \cdot 4 \cdot S(n) \cdot |\Gamma|^{S(n)}$$

verschiedene Konfigurationen, wobei Q und Γ die Zustandsmenge und das Arbeitsalphabet von M sind.

Somit gibt es eine Zahl $d \in \mathbb{N}$ so dass

$(*)$ $N_n \leq 2^{d \cdot S(n)}$ f.a. hinreichend großen $n \in \mathbb{N}$ gilt

Für jede Position i auf dem Eingabeband sei

$C_{i,M}(x)$

die Teilfolge der Berechnung $B_M(x)$, die aus allen Konfigurationen besteht, bei denen der Kopf des Eingabebands auf Position i steht.

$C_{i,M}(x)$ wird <u>Crossing-Sequenz für Position i</u> genannt.

<u>klar:</u> In $C_{i,M}(x)$ kommt keine Konfiguration mehrfach vor, denn sonst würde die DTM M bei Eingabe x in eine Endlosschleife kommen — da M aber die Sprache L entscheidet muss M bei jeder Eingabe x irgendwann anhalten.

Somit gilt für die Länge $|C_{i,M}(x)|$ von $C_{i,M}(x)$:

$$|C_{i,M}(x)| \leq N_n \quad , \text{für } n := |x|$$

Außerdem gilt für jedes $\ell \in \mathbb{N}$:
Die Anzahl der möglichen Crossing-Sequenzen (für Pos. i) der Länge ℓ ist $\leq N_n^{\ell}$.

Somit gilt:

(**) Die Gesamtzahl möglicher Crossing-Sequenzen für jedes feste i ist

$$\leq \sum_{\ell=0}^{N_n} N_n^{\ell} < N_n^{N_n+1} = 2^{(\log N_n)\cdot(N_n+1)} \underset{(*)}{\leq} 2^{d\cdot S(n)\cdot(2^{d\cdot S(n)}+1)}$$

$$\leq 2^{2^{d'\cdot S(n)}}$$ für ein geeignetes $d' \in \mathbb{N}$ und alle hinreichend großen $n \in \mathbb{N}$.

Sei $C'_{i,M}(x)$ die Folge, die aus $C_{i,M}(x)$ entsteht, indem in jeder Konfiguration die Information über die Kopfposition i gelöscht wird.

<u>Behauptung 2:</u>

Sei $x \in \{0,1\}^*$, $n := |x|$ und sei $1 \leq i_1 < i_2 \leq n$ so dass $C'_{i_1,M}(x) = C'_{i_2,M}(x)$.

Dann gilt für $x' := x_1 \cdots x_{i_1} x_{i_2+1} \cdots x_n$ (d.h.: x' ist das Wort, das aus x entsteht, indem das Teilwort $x_{i_1+1} \cdots x_{i_2}$ gelöscht wird):

$B_M(x')$ ist die Konfigurationenfolge, die aus $B_M(x)$ entsteht, indem

- alle Konfigurationen, bei denen die Kopfposition auf dem Eingabeband eine Position aus $\{i_1+1, \ldots, i_2\}$ ist, gelöscht werden, und
- bei allen Konfigurationen, bei denen die Kopfposition auf dem Eingabeband eine Zahl der Form i_2+j, für $j \geq 1$ ist, diese Zahl ersetzt wird durch die Zahl i_1+j.

Insbes. gilt: M akzeptiert x' $\Longleftrightarrow$ M akzeptiert x.

<u>Beweis:</u> Übung!

Um den Beweis von Satz 4.28 zu beenden sei nun für jedes $k \in \mathbb{N}$ $x^{(k)} \in \{0,1\}^+$ ein Wort minimaler Länge, bei dessen Eingabe M mehr als k Zellen des Arbeitsbandes benutzt.

D.h. in $B_M(x^{(k)})$ gibt es mindestens eine Konfiguration C, für die die Beschriftung des Arbeitsbandes die Länge $>k$ hat.

Sei j die Position des Kopfes auf dem Eingabeband bei C.

Dann gilt:

(1) Die Crossing-Sequenzen $C'_{i,M}(x^{(k)})$ für alle $i \leq j$ müssen alle paarweise verschieden sein, und

(2) die Crossing-Sequenzen $C'_{i,M}(x^{(k)})$ für alle $i \geq j$ müssen alle paarweise verschieden sein.

Denn: Angenommen $C'_{i_1,M}(x^{(k)}) = C'_{i_2,M}(x^{(k)})$ mit $i_1 < i_2 \leq j$ oder $j \leq i_1 < i_2$. Dann gilt gemäß Behauptung 2 für $x' := x_1^{(k)} \dots x_{i_1}^{(k)} x_{i_2+1}^{(k)} \dots x_n^{(k)}$, dass $B_M(x')$ die Konfiguration C enthält (bzw. die Variante von C, bei der j ersetzt wurde durch $j - i_2 + i_1$).

D.h. insbesondere, dass M bei Eingabe x' mehr als k Zellen des Arbeitsbandes nutzt. Wegen $|x'| < |x^{(k)}|$ ist dies ein Widerspruch zur Wahl von $x^{(k)}$ als Wort minimaler Länge, bei dessen Eingabe M mehr als k Zellen des Arbeitsbandes nutzt.

Somit muss (1) und (2) gelten.

Insbesondere gibt es daher in $\{ C^j_{i,n}(x^{(k)}) : 1 \le i \le |x^{(k)}| = n_k \}$ mindestens $\frac{n_k}{2}$ verschiedene Elemente (für $j \ge \frac{n_k}{2}$ folgt dies aus (1); für $j \le \frac{n_k}{2}$ folgt dies aus (2))

Mit (**) folgt: $\frac{n_k}{2} \le 2^{2^{d' \cdot S(n_k)}}$

Somit: $n_k \le 2 \cdot 2^{2^{d' \cdot S(n_k)}} < 2^{2^{d'' S(n_k)}}$ für ein geeignetes $d'' \in \mathbb{N}$ und alle hinreichend großen n_k

und daher: $\log\log n_k \le d'' \cdot S(n_k)$,

also $S(n_k) \ge \frac{1}{d''} \cdot \log\log n_k$

$\lightning$ Widerspruch zu $S(n) = o(\log\log n)$

Somit muss L regulär sein, und Satz 4.28 ist bewiesen. $\square$

<u>Bemerkung 4.29</u>

Aus Satz 4.28 wissen wir, dass $SPACE(o(\log\log n))$ genau die regulären Sprachen enthält.

$SPACE(\log\log n)$ enthält auch nicht-reguläre Sprachen — z.B. die Sprache

$$L_{BIN} := \{ \langle bin_k(0), bin_k(1), \cdots, bin_k(2^k - 1) \rangle : k \in \mathbb{N} \}$$

wobei $bin_k(j)$ die Binärdarstellung der Länge k von j bezeichnet, für j mit $0 \le j < 2^k$.

(Dass L_{BIN} nicht regulär ist, folgt leicht aus dem Pumping-Lemma für reguläre Sprachen. Dass $L_{BIN} \in SPACE(\log\log n)$ ist,

Bemerkung 4.30

Durch Betrachtung von Crossing-Sequenzen kann man für einige konkrete Sprachen untere Schranken für den Platz- oder Zeitbedarf zum Entscheiden dieser Sprachen beweisen.

z.B. kann man für die Sprache

$$PAL := \{ w \in \{0,1\}^* : w \text{ ist ein Palindrom} \}$$

zeigen:

(1) $PAL \notin SPACE(o(\log n))$

(dh es gibt kein $S: \mathbb{N} \to \mathbb{N}$ mit $S(n) = o(\log n)$ so dass $PAL \in SPACE(S(n))$)

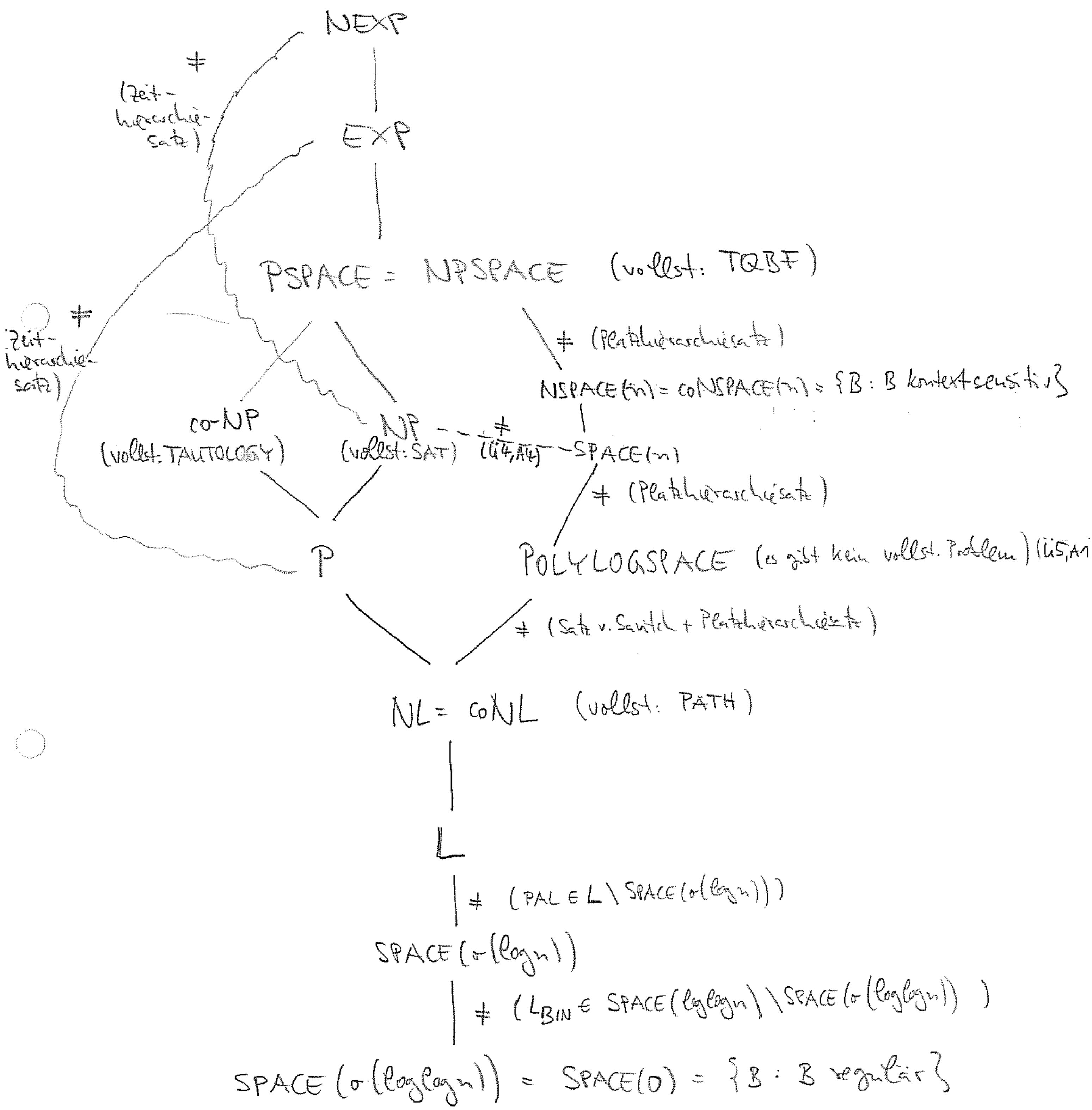
Zum Vergleich: $PAL \in L = SPACE(\log n)$

(2) PAL wird von keiner 1-Band DTM (mit Schreib-/Lesekopf auf dem Eingabeband) in $o(n^2)$ Schritten entschieden

Zum Vergleich: PAL kann in $O(n^2)$ Schritten von einer 1-Band DTM (mit Schreib-/Lesekopf auf dem Eingabeband) entschieden werden.
Und PAL kann in $O(n)$ Schritten von einer 2-Band DTM entschieden werden.

Details: Übung!

Überblick über die bisher betrachteten Komplexitätsklassen:



Kapitel 5: Die Polynomialzeit-Hierarchie und Alternierungen

5.1 Die Klasse Σ_2^P

Zur Erinnerung:

Das folgende Problem ist NP-vollständig:

$$\text{INDSET} = \{ \langle G, k \rangle : \text{ Graph } G \text{ besitzt eine unabhängige Menge der Größe } k \}$$

Frage 5.1:

Was ist mit den folgenden Problemen?

(a) $\text{EXACT-INDSET} := \{ \langle G, k \rangle : k \text{ ist die Größe der größten unabhängigen Menge von } G \}$

Klar: $\langle G, k \rangle \in \text{EXACT-INDSET} \iff$

$\exists S_1 \subseteq V(G) \quad \forall S_2 \subseteq V(G)$:

$|S_1| = k \wedge S_1$ ist eine unabhängige Menge $\wedge$

$(S_2$ ist eine unabhängige Menge $\rightarrow |S_2| \leq k)$

(b) MIN-EQ-DNF :=

$\{ \langle \varphi, k \rangle$: φ ist eine aussagenlogische Formel, $k \in \mathbb{N}$ s.d. für die kürzeste zu φ äquivalente Formel ψ in DNF gilt: $|\psi| = k$ $\}$

disjunktive Normalform

(Warum ist dieses Problem interessant?
— Weil das Erfüllbarkeitsproblem für DNF-Formeln in Polynomialzeit lösbar ist (siehe Vorlesung "Diskrete Modellierung") und man das SAT-Problem lösen kann, indem man eine gegebene CNF-Formel φ zunächst in eine äquivalente, möglichst kurze DNF-Formel ψ transformiert und dann diese Formel in Zeit poly($|\psi|$) auf Erfüllbarkeit testet.
Problem: Es ist bekannt, dass $|\psi|$ exponentiell größer sein kann als $|\varphi|$ (vgl. Vorlesung/Übung "Diskrete Modellierung")
)

Klar: $\langle \varphi, k \rangle \in$ MIN-EQ-DNF $\iff$

$\exists \psi_1 \ \forall \psi_2$:
ψ_1 in DNF $\wedge$ $\psi_1 \equiv \varphi$ $\wedge$ $|\psi_1| \leq k$ $\wedge$
$((\psi_2$ in DNF $\wedge$ $\psi_2 \equiv \varphi) \rightarrow |\psi_2| \geq k$)

Die beiden Probleme EXACT-INDSET und MIN-EQ-DNF gehören zur folgenden Klasse Σ_2^P :

Definition 5.2 (Σ_2^p – die 2te Stufe der Polynomialzeit-Hierarchie)

Die Klasse Σ_2^p besteht aus allen Sprachen $L \subseteq \{0,1\}^*$, für die es eine det. Polynomialzeit-TM M und ein Polynom $q: \mathbb{N} \to \mathbb{N}$ gibt, so dass f.a. $x \in \{0,1\}^*$ gilt:

$$x \in L \iff \exists u_1 \in \{0,1\}^{q(|x|)} \; \forall u_2 \in \{0,1\}^{q(|x|)} \; M(\langle x, u_1, u_2 \rangle) = 1.$$

Beachte: $\Sigma_2^p \supseteq NP$ und $\Sigma_2^p \supseteq coNP$.

5.2 Die Polynomialzeit-Hierarchie

Σ_2^p ist durch 2 wechselnde Quantoren ($\exists\forall$) definiert. Dies lässt sich natürlich für beliebige Zahlen k von wechselnden Quantoren verallgemeinern:

Definition 5.3 (Σ_k^p, Π_k^p und PH)

(a) Sei $k \in \mathbb{N}$. Die Klasse Σ_k^p besteht aus allen Sprachen $L \subseteq \{0,1\}^*$, für die es eine det. Polynomialzeit-TM M und ein Polynom $q: \mathbb{N} \to \mathbb{N}$ gibt, s.d. f.a. $x \in \{0,1\}^*$ gilt:

$$x \in L \iff \exists u_1 \in \{0,1\}^{q(|x|)} \; \forall u_2 \in \{0,1\}^{q(|x|)} \; \ldots \; Q_k u_k \in \{0,1\}^{q(|x|)}$$
$$M(\langle x, u_1, u_2, \ldots, u_k \rangle) = 1$$

wobei $Q = \begin{cases} \exists & \text{falls } k \text{ ungerade} \end{cases}$

Insbes.: $\Sigma_1^P = NP$, $\Sigma_0^P = P$

(b) Sei $k \in \mathbb{N}$ $\quad \Pi_k^P := \text{co}\Sigma_k^P \overset{\text{Def}}{=} \{L : \overline{L} \in \Sigma_k^P\}$.

Insbes: $\Pi_1^P = \text{coNP}$, $\Pi_0^P = P$.

Man sieht leicht, dass f.a. $L \subseteq \{0,1\}^*$ gilt: $L \in \Pi_k^P \iff$ es gibt eine det. Polynomialzeit-TM M und ein Polynom $q: \mathbb{N} \to \mathbb{N}$, s.d. f.a. $x \in \{0,1\}^*$ gilt:

$$x \in L \iff \forall u_1 \in \{0,1\}^{q(|x|)} \; \exists u_2 \in \{0,1\}^{q(|x|)} \cdots Q_k u_k \in \{0,1\}^{q(|x|)}$$

$$M(\langle x, u_1, u_2, \ldots, u_k \rangle) = 1$$

wobei $Q_k = \begin{cases} \forall & \text{falls } k \text{ ungerade} \\ \exists & \text{falls } k \text{ gerade und } k \neq 0. \end{cases}$

(c) Die <u>Polynomialzeit-Hierarchie</u> ist die Klasse

$$PH := \bigcup_{k \in \mathbb{N}} \Sigma_k^P$$

Anhand der Definition von Σ_k^P, Π_k^P und PH sieht man leicht, dass folgendes gilt (f.a. $k \in \mathbb{N}$):

$$\Sigma_k^P \subseteq \Pi_{k+1}^P \subseteq \Sigma_{k+2}^P \subseteq PH$$

und daher auch $PH = \bigcup_{k \in \mathbb{N}} \Pi_k^P$.

Für jedes $k \in \mathbb{N}$ definieren wir

$$\Delta_k^P := \Sigma_k^P \cap \Pi_k^P .$$

Klar:

$$\begin{array}{ccccc} & \subseteq & \Sigma_k^P & \subseteq & \\ \Delta_k^P & & & & \Delta_{k+1}^P \\ & \subseteq & \Pi_k^P & \subseteq & \end{array}$$

denn: $\Sigma_k^P \subseteq \Pi_{k+1}^P \cap \Sigma_{k+1}^P = \Delta_{k+1}^P$
und $\Pi_k^P \subseteq \Sigma_{k+1}^P \cap \Pi_{k+1}^P = \Delta_{k+1}^P$

Somit weisen die Stufen der Polynomialzeit-Hierarchie die folgende Inklusionsstruktur auf:

$$\begin{tikzcd}[arrows=dash]
 & PH \arrow[d] & \\
 & \vdots & \\
 & \Delta_{k+2}^P \arrow[dl] \arrow[dr] & \\
\Sigma_{k+1}^P \arrow[dr] & & \Pi_{k+1}^P \arrow[dl] \\
 & \Delta_{k+1}^P \arrow[dl] \arrow[dr] & \\
\Sigma_k^P \arrow[dr] & & \Pi_k^P \arrow[dl] \\
 & \Delta_k^P & \\
 & \vdots & \\
NP = \Sigma_1^P \arrow[dr] & & \Pi_1^P = coNP \arrow[dl] \\
 & \Delta_1^P = NP \cap coNP \arrow[d] & \\
\Sigma^P & \cdots & \Pi^P
\end{tikzcd}$$

Man vermutet, dass die Stufen alle verschieden sind, dh dass $\Sigma_{k-1}^P \neq \Sigma_k^P$ und $\Sigma_k^P \neq \Pi_k^P$

f.a. $k \geq 1$ gilt.

(<u>Beachte</u>: für $k=1$ besagt diese Vermutung gerade, dass $P \neq NP$ und $NP \neq coNP$ ist.)

Diese Vermutung wird oft in der Aussage "Die Polynomialzeit-Hierarchie ist strikt" bzw "Die Polynomialzeit-Hierarchie kollabiert nicht" zusammengefasst.

<u>Satz 5.4</u>

<u>(a)</u> Falls $P = NP$ ist, so gilt $PH = P$
(dh die Polynomialzeit-Hierarchie kollabiert zu ihrer 0-ten Stufe $P = \Sigma_0^P$)

<u>(b)</u> Für jedes $k \geq 1$ gilt:
Falls $\Sigma_k^P = \Pi_k^P$, so ist $PH = \Sigma_k^P$
(dh die Polynomialzeit-Hierarchie kollabiert zu ihrer k-ten Stufe Σ_k^P)

<u>Beweis:</u>

<u>(a)</u> Wir nehmen an, dass $P = NP$ ist und zeigen per Induktion nach k, dass f.a. $k \in \mathbb{N}$ gilt:

$\Sigma_k^P = P$.

$\underline{k=0}$: klar (da $\Sigma_0^P \overset{\text{Def}}{=} P$)

$\underline{k=1}$: $\Sigma_1^P = NP = P$ gemäß Annahme

$\underline{k \to k+1}$: Ind.annahme: $\Sigma_k^P = P$

(für $k \geq 1$) zu zeigen: $\Sigma_{k+1}^P = P$

Beweis: Gemäß Ind.annahme gilt $\Sigma_k^P = P$

Somit gilt auch $\Pi_k^P \overset{\text{Def}}{=} co\Sigma_k^P = coP = P$, also $(*)$: $\Pi_k^P = P$.

klar: $P \subseteq \Sigma_{k+1}^P$. Zum Beweis von $\Sigma_{k+1}^P \subseteq P$

sei nun $L \in \Sigma_{k+1}^P$. Zu zeigen: $L \in P$.

Wegen $L \in \Sigma_{k+1}^P$ gibt es eine det. Polynomialzeit-TM M und ein Polynom $q: \mathbb{N} \to \mathbb{N}$ s.d. f.a. $x \in \{0,1\}^*$ gilt:

$$x \in L \iff \exists u_1 \in \{0,1\}^{q(|x|)} \ \forall u_2 \in \{0,1\}^{q(|x|)} \ \cdots \ Q_{k+1} u_{k+1} \in \{0,1\}^{q(|x|)}$$
$$M(\langle x, u_1, u_2, \ldots, u_{k+1} \rangle) = 1$$

mit $Q_{k+1} = \begin{cases} \exists & \text{falls } k+1 \text{ ungerade} \\ \forall & \text{falls } k+1 \text{ gerade} \end{cases}$.

Sei $L' := \{ \langle x, u_1 \rangle : \forall u_2 \in \{0,1\}^{q(|x|)} \cdots Q_{k+1} u_{k+1} \in \{0,1\}^{q(|x|)}$
$M(\langle x, u_1, u_2, \ldots, u_{k+1} \rangle) = 1 \}$

klar: $L' \in \Pi_k^P$. Wegen $(*)$ gilt $\Pi_k^P = P$. Daher gibt es eine Polynomialzeit-TM M' s.d. f.a. $\langle x, u_1 \rangle$ gilt:

$$\langle x, u_1 \rangle \in L' \iff M'(\langle x, u_1 \rangle) = 1$$

Somit gilt f.a. $x \in \{0,1\}^*$

$$x \in L \iff \exists u_1 \in \{0,1\}^{q(|x|)} \quad \langle x, u_1 \rangle \in L'$$
$$\iff \exists u_1 \in \{0,1\}^{q(|x|)} \quad M'(\langle x, u_1 \rangle) = 1$$

Also ist $L \in NP$. Gemäß Voraussetzung ist $NP = P$, also $L \in P$. Somit ist $\Sigma_{k+1}^P \subseteq P$. □

(b): analog (Details: Übung!)

Vollständige Probleme für die einzelnen Stufen von PH

Definition 5.5

Sei K eine der Klassen PH, Σ_k^P, Π_k^P (für $k \geq 1$).
Eine Sprache $L \subseteq \{0,1\}^*$ heißt K-vollständig, falls gilt:
(1) $L \in K$ und
(2) L ist K-hart, dh f.a. $L' \in K$ gilt:
$L' \leq_p L$.

Beobachtung 5.6: (Vermutung: Es gibt kein PH-vollständiges Problem)
Falls es eine PH-vollständige Sprache L gibt, dann gibt es ein $k \in \mathbb{N}$ s.d. $PH = \Sigma_k^P$ (dh die Polynomialzeit-Hierarchie kollabiert zu ihrer k-ten Stufe Σ_k^P).

Beweis:

Sei $L \subseteq \{0,1\}^*$ PH-vollständig.

Wegen $L \in PH = \bigcup_{k \in \mathbb{N}} \Sigma_k^p$ gibt es ein $k \in \mathbb{N}$

s.d. $L \in \Sigma_k^p$ via TM M und Polynom q gemäß Def. 5.3.

Sei $L' \in PH$. Zu zeigen: $L' \in \Sigma_k^p$.

Da L PH-hart ist, ist $L' \leq_p L$ durch eine Polynomialzeit-Reduktion $f: \{0,1\}^* \to \{0,1\}^*$.

Somit gilt f.a. $x \in \{0,1\}^*$:

$$x \in L' \iff f(x) \in L$$
$$\iff \exists u_1 \in \{0,1\}^{q(|f(x)|)} \; \forall u_2 \in \{0,1\}^{q(|f(x)|)} \ldots Q_k u_k \in \{0,1\}^{q(|f(x)|)}$$
$$M(\langle f(x), u_1, u_2, \ldots, u_k \rangle) = 1$$

Wir können oBdA annehmen, dass $|f(x)| = |x|^c$ für ein geeignetes $c \in \mathbb{N}$ ist (da f in Polynomialzeit berechnet werden kann – und ggf unter Verwendung eines geeigneten Paddings). Da $f(x)$ in Polynomialzeit berechnet werden kann ist daher $L' \in \Sigma_k^p$.

$\square$

Beobachtung 5.7 (PH und PSPACE)

(a) $PH \subseteq PSPACE$

(b) Falls $PH = PSPACE$, so gibt es ein $k \in \mathbb{N}$ s.d. $PH = \Sigma_k^P$. (— Also gilt vermutlich: $PH \neq PSPACE$)

Beweis:

(a): Analog zum Nachweis, dass $NP \subseteq PSPACE$ und TQBF $\in$ PSPACE zeigt man für jedes $k \in \mathbb{N}$, dass $\Sigma_k^P \subseteq PSPACE$.

(b) Wir wissen, dass TQBF PSPACE-vollständig ist. Falls $PH = PSPACE$, so ist TQBF PH-vollständig. Somit folgt die Behauptung aus Beobachtung 5.6. □

Beobachtung 5.8 (Vollständige Probleme für Σ_k^P und Π_k^P)

(a) Für jedes $k \geq 1$ ist das folgende Problem Σ_k^P-vollständig

Σ_k-SAT $:= \{ \Phi : \Phi$ ist eine wahre QBF der Form $\exists \overline{u}_1 \forall \overline{u}_2 \cdots Q_k \overline{u}_k\, \varphi$, wobei $\overline{u}_1, \ldots, \overline{u}_k$ Listen von Variablen sind und φ eine aussagenlogische Formel über den Variablen $\overline{u}_1, \ldots, \overline{u}_k$ ist, und $Q_k = \begin{cases} \exists & \text{falls } k \text{ ungerade} \\ \forall & \text{falls } k \text{ gerade} \end{cases}$ $\}$

(b) Für jedes $k \geq 1$ ist das folgende Problem Π_k^P-vollständig:

Π_k-SAT $:= \{ \Phi : \Phi$ ist eine wahre QBF der Form $\forall \overline{u_1} \exists \overline{u_2} \cdots Q_k \overline{u_k} \, \varphi$, wobei $\overline{u_1}, \dots, \overline{u_k}$ Listen von Variablen sind und φ eine aussagenlogische Formel über den Variablen $\overline{u_1}, \dots, \overline{u_k}$ ist und $Q_k = \begin{cases} \forall & \text{falls } k \text{ ungerade} \\ \exists & \text{falls } k \text{ gerade} \end{cases}$ $\}$

Beweis:

(b) folgt leicht aus (a), da $\Pi_k^P = \text{co}\Sigma_k^P$.

zu (a): Σ_k-SAT $\in \Sigma_k^P$ folgt unmittelbar aus der Definition von Σ_k-SAT und Σ_k^P.

Die Σ_k^P-Härte von Σ_k-SAT lässt sich leicht aus dem Beweis des Satzes von Cook und Levin folgern.

Details: Übung!

Bemerkung 5.3

(a) Das Problem EXACT-INDSET (vgl. Frage 5.1 (a)) ist vermutlich nicht Σ_2^P-vollständig, da es bereits in $\Sigma_2^P \cap \Pi_2^P$ liegt.

(b) Das Problem MIN-EQ-DNF (vgl Frage 5.1(b))
ist Σ_2^P - vollständig (hier ohne Beweis)

5.3 Charakterisierung der PH durch Orakel-TMen

Zur Erinnerung (vgl Kapitel 3.3)

- M eine Orakel-TM, $O \subseteq \{0,1\}^*$, $x \in \{0,1\}^*$ $\Rightarrow$ schreibe $M^O(x)$, um die Ausgabe von M bei Eingabe x mit Orakel O zu bezeichnen
- NP^O: die Klasse aller Sprachen $L \subseteq \{0,1\}^*$, die durch eine ndet. Orakel-TM mit Orakel O in polynomiell vielen Schritten entschieden werden können

Satz 5.10

Für jedes $k \geq 2$ gilt: $\Sigma_k^P = NP^{\Sigma_{k-1}\text{-SAT}}$.

Beweis: Wir zeigen hier die Aussage für $k=2$
(die allgemeine Aussage für $k \geq 2$ lässt sich analog beweisen)

Zu zeigen: $\Sigma_2^P = NP^{\Sigma_1\text{-SAT}}$.

"$\subseteq$": Sei $L \in \Sigma_2^P$. D.h. sei M eine det. Polynomialzeit-TM und sei $q: \mathbb{N} \to \mathbb{N}$ ein Polynom s.d. f.a. $x \in \{0,1\}^*$ gilt: