1-Band Turingmaschine

(Q, Σ, I, q_o, F)

$I \subseteq Q \times \Sigma \times \Sigma \times QxL, R, S$

Konfiguration: (q, k, tape) q: Zustand k: Kopfpos tape: Bandinhalt

$(q, k, tape) \leftarrow (q', k', tape)$

$$falls(q, tape(k), tape'(k), q', m) \in I und k' = \begin{cases} k+1 & \text{falls m=R} \\ k-1 & \text{falls m=L} \\ k & \text{falls m=S} \end{cases}$$

tape: Zeichen an Pos k

2-Kopf-1-Band Turingmaschine

a) ersetze I durch

$I \subseteq Q \times \Sigma^2 \times \Sigma^2 \times Q \times L, R, S^2$

Konfiguration: (qq, i, j,tape)

Uebergangsrelation: $(q, i, j, tape) \leftarrow (q', i', j', tape')wenn$

$(q, (tape(i), tape(j)), (tape'(i'), tape'(j)), q', (m_1, m_2)) \in I$

$$i' = \begin{cases} i+1 & \text{falls } m_1 = R \\ i-1 & \text{falls } m_1 = L \\ i & \text{falls } m_1 = S \end{cases}$$

und

$$j' = \begin{cases} j+1 & \text{falls } m_2 = R \\ j-1 & \text{falls } m_2 = L \\ j & \text{falls } m_2 = S \end{cases}$$

b) Simulation von 2-Kopf Turing Maschine durch 1-Band Turingmaschine: Alphabet: $\Sigma' := \Sigma \times 0, 1, 2, 3$ 0:keine 1:Kopf1 2:Kopf2 3:beide

0.1. 2-1

Tm M hat Laufzeit f(n), wenn jede Eingabe$\times$in hoechstens f(|x|) Schritten akzeptiert oder abgelenhnt werid. $q_A \in F, q_R \in F, q_A \neq q_R$

$L_1 \in NPundL_2 \in NP, dannL_1 \cup L_2 \in NPundL_1 \cap L_2 \in NP$

- Sei M_1 und M_2 nicht def. Tm mit plynomiellen Laufzeiten und $L_1 = L(M_1)$ und $L_2 = L(M_2)$. Definiere M:

  1. Waehle nichtdeterministisch $b \in 1, 2$

  2. Fuehre M_b aus und akzeptiere bzw. lehne ab, wenn M_b akzeptiert bzw ablehen)

- schnittmenge: Definiere M:

  1. Fuehre M_1 auf die Eingabe aus, Wenn M_1 ablehnt, dann lehne ab.

  2. Sonst Fuehre M_2 auf die Eingabe aus. Das Ergebnis ist das Endergebnis.

  Laufzeit Polynomiell:

  - $L(M) <= L_1 \cap L_2$

  - $L_1 \cap L_2 <= L(M)$

0.2. 2-2

$L \subseteq \Sigma*$ ist in Np, wenn es ein Polynom p(x) und eine def. Pliynomzeit Tm M gibt , sod dass gilt $L = \{x \in \Sigma | \exists y \in 0, 1^{p(|x|)}$. M akzeptiert $y\$x\}$

NEXP $= \cup NTIME(2^{n^k})$

Spezialfall von nichtdet. Tm: Ersetze I durch

- : $\delta_0 Q \times \Sigma \leftarrow \Sigma \times Q \times L, R, S$

- : $\delta_1 Q \times \Sigma \leftarrow \Sigma \times Q \times L, R, S$

Gemeint ist Tm mit $I = \{(q, a, b, q', m) | (b, q', m) \in detla_0(q, a)$ oder $(b, q', m) \in detla_1(q, a) oder\}$

Beh. Jede Sprache in Np wird durch Tmin Spezialform und polynomieller Lafuzeit akkzeptiert.

Ein Baum mit Tiefe eins und 6 Blaettern wird zu einme Binaer Baum mit Tiefe 3 aber auch 6 Blaettern. Bild war ich zu faul zu zeichnen.

Beweis: Sei $L \in NP$, akteptiert durch Turingmaschine in spezialform und Laufzeit f(n).

Waehle p(n) := f(n). Waehle fuer M folgende Maschine:

1. Lies Eingabe y$x und schreibe y auf Band 2 und x auf Band 1.
2. Gehe Band 2 von links nach rechts durch und fuehre jeweils M mit δ_0 oder δ_1 (je nachdem ob auf Band 2 eine 0 oder eine 1 steht) einen Schritt aus.
3. Akzeptiere wenn M akzeptiert, lehne ab, wenn M ablehnt oder das Band zu Ende ist.

0.3. 10-November Baltt 4

0.3.1. Hennier und Stearns

Es gibt eine universelle Turing MAschine u, so dass gilt, M akyeptiert Eingabe x in t Schritten, genau dann wenn U akzeptiert Eingabe <code (M),x> in c*t*log t Schritten.

0.4. Aufgabe 4-1

- $Zeithierachisatz : P \subsetneq EXP$
- Wir wissen $P \subseteq NPundNP \subseteq EXP$
- Angenommen
  - $\neg(P \subsetneq NP)und\neg(NP \subsetneq EXP)$

- $PNPundNP = EXP$

- $\Rightarrow P = EXP$

0.5. Aufgabe 4-2

Bemerkung $L \in NP \Leftrightarrow \exists Polynompunddeterministische Polytime TmM, sodass L = \{x|\exists y \in \{0,1\}^{p(|x|)} Makzeptierty\$x\}$

Aufgabe:

- $L = \{x|\exists y\{0,1\}^{2^{|x|}} DieMaschineMakzeptierty\#x\}$
- Zeige: jede Sprache $L \in EXP$ lässt sich so fomrulieren
- Beweis: sei $L' \in EXP$ bezeugt durch Turingmaschine M' Laufzeit von M' ist $c \times 2^{n^k}$
- Konsturiere Turingmaschine M":
  1. Prüfe,dass Eingabe die Form y#x hat
  2. Schreibe x auf zweites Band
  3. Schreibe $2^{|y\#x|/2}$ viele Zeichen auf ein zweites Band
  4. Führe M' für $2^{|y\#x|/2}$ viele Schritte aus
- Laufzeit M" $\mathcal{O}(2^{n/2})$
- Simuliere M" durch 1-Band Turingmaschine M"'
- Laufzeit M"': $\mathcal{O}((2^{n/2})^2 = \mathcal{O}(2^n)$
- Spraceh von M"'
  - Betrachte Verhalten M"' bei Eingabe von y#x mit $|y| = 2^{|x|}$ M"' akzeptiert gdw M' die Eingabe x in $2^{|y\#x|/2} = 2^{2^{|x|}+1+|x|)/2}$ Schritten akteptiert
  - Es gibt n_0,so dass $c \times 2^{n^k} \leq 2^{2^n+1+n}$

- für alle $n \geq n_0$
- $\Rightarrow$ Für Eingaben der Länge $\geq n_0$ hat M"' das gleiche Akzeptantverhalten wie M'.
- Definiere M so, dass die Wörter die Länge $< n_0$ fest in der Übergangstabelle kodiert sind und dass sich M wie M"' verhält.

0.6. 4-3

Wenn jede unäre Sprache in Np auch in P liegt, dann folgt EXP = NEXP. E = NE

- $E \subseteq NE$
- $NE \subseteq E$
  - Sei $L \in NE$
  - Dann ist $L' = \{unr(x)|x \in L\} in NP$
  - Konstruiere nicht deterministische Turingmaschine M' mit L(M') = L':
    1. Dekodierunge $unr(x) \mapsto x$ Zeit $\mathcal{O}(n)$,grö0e von x ist $\mathcal{O}(logn)$ wobei $n = |Eingabe|$
    2. Lasse Ne-Tm für L laufen. Zeit $\mathcal{O}(2^{k \times logn} = \mathcal{O}(n^k)$
  - Nach Annahme folgt $L' \in P$. Sei M" eine deterministische Turingmaschine mit polynomieller Laufzeit für L'.
  - Definiere Turingmaschine M"' durch:
    1. Kodiere Eingabe $x \mapsto unr(x)$ Zeit: $\mathcal{O}(2^n)$, Gröse von unär(x) $\mathcal{O}(2^n)$
    2. Lasse M" laufen Zeit: $\mathcal{O}((2^n)^k) = (2^{n \times k}) \Rightarrow L \in EbezeuchtdurchM'''$.
- Es gilt $N = NE \Rightarrow Exp = NEXP$
- Beweis Aufgabe 2-2

0.7. 5-1

Sat ist Np-vollständig wenn man nur Klauseln der Form $(x_1 \vee x_2 \vee x_3 \cdots \vee x_n)$ und $(\neg x_1 \vee \neg x_2 \vee \neg x_3 \vee ..\neg x_n)E$ zulässt.

- Sat' ist in NP
- Sat' ist Np-schwer Z.z. $\forall L \in NP. \leq_p Sat'$ Es genügt zu zeigen $SAT \leq_p SAT'$ Beachte: $(x \Leftrightarrow z)$ ist äuqivalent zu KNF-Formel $= (\neg xv\neg y) \wedge (xvz)$
- Idee ersätze $(q \wedge (x_1 \wedge \neg x_2 \wedge x_3)$ durch $q \wedge (x_1 vzvx_3) \wedge (\neg z \neg x_2) \wedge (z \vee x_2)$ für eine frische Variable z. Beide Forlmen sind erfüllbarkeitsäuquivalent.
  1. Definiere f durch sukzesive Anwendung dieser Umformung.
  2. Es gibt $\forall q.q \in SAT \Leftrightarrow f(q) \in SAT$
  3. f ist berechenbar in linearer Zeit.

0.8. 5-2

0-1 Integeer Linear Programming.

i n NP

i n NP-schwer Reduktion von Sat'

3-Sat $\leq_p 0 - 1 - ILP$

- Für jede KLausel $(x_1 \vee \cdots \vee x_n)x_1 + \cdots + x_n \geq 1 \Leftrightarrow x_1 + \cdots + x_n - c_n - c_{n-1} = 1 c_i \in 0, 1 (C_1, \ldots, C_{n-1} sindneueVariablen)$
- Für jede Klausel $\neg x_1 \vee \cdots \vee \neg x_m$

  $-x_1 - \ldots x_m \geq 1 - m \Leftrightarrow -x_1 - \cdots - x_m + c_1 + \ldots c_m = 1$ (c_i sind neue variablen)
- Definiere f so dass eine gegebene Formel φ auf das folgendes Gleichungssystem abbildet.

- Eine Klausel$(x_1 \vee x_2 \vee x_3)$ wird zu $x_1 + x_2 + x_3 - c_1 - c_2 = 1$

- Eine KLausel$(\neg x_1 \vee \neg x_2 \vee \neg x_3)$ zu $-x_1 - x_2 - x_3 + c_1 + c_2 + c_3 = 1$

- zu Zeigen $\varphi \in Sat' \Leftrightarrow f(\varphi) \in \mathcal{O} - 1 - ILP \Rightarrow$ Sei η eine erfüllende Bedingung für φ. Setze $x_i = \begin{cases} 0 & \eta(x_i) = true \\ 1 & \eta(x_i) = false \end{cases} \Rightarrow$ Wert f[r die c_i finde um die Gleichen

- $Leftarrow$ Angenommen wir haben eine Lösung des gleichssystems f(p). Zeige φ erfüllbar $\eta(x_i) \begin{cases} \top & falls x_i := 1 \\ \bot & falls x_i := 0 \end{cases}$ ist einf. Bel.

Reduktion von Sat

- Wähle für jede aussagelogische Variable x zwei Variablen x_T und x_F und Gleichungssystem $x_T + x_F = 1$

- Klausel $(x_1 \vee \neg x_2 \vee x_3)$ wird zu

ka wies weiterging

0.9. 5-3

co-NP $= \{L|\overline{L} \in NP\}$

Wenn $L \subseteq \Sigma*$ Np-vollständig und $L \in$ co-NP, dann Np=co-NP

Beweis Sei L Np-vollständig und $L \in co - NP$ Zeige Np=co-NP

- $NP \subseteq$ co-NP Sei $L' \in NP$. Dann gilt $L' \leq_p L$ wegen Np-vollständig von L. D.h. $\forall x.x \in L' \Leftrightarrow f(x) \in L$ f[r eine Funktion f, die in polynomieller Zeit berechenbar ist

- $\Rightarrow \forall x.x \in \overline{L'} \Leftrightarrow f(x) \in \overline{L}$. Da $L \in co - NP$ gilt $\overline{L} \in NP$. Sei M eine Np-Turing-Maschiene für $\overline{L}$ Dann ist f;M eine NP-Turingmaschine für $\overline{L'} \Rightarrow \overline{L'} \in NP$.

- co-NP ⊆NP Sei L” ∈ co-NP d.h. $\overline{L''} \in$ NP Wederhole (_)* statt $L' \Rightarrow \overline{\overline{L''}} \in NP \Rightarrow L'' \in NP$

0.10. 6-1

$A \in P^B$ und $B \in P \Rightarrow A \in P$

0.11. 6-2

$A \in NP \cap coNP \Rightarrow NP^A = NP$

Sei $A \in NP$ und $A \in coNP$ (d.h. $\overline{A} \in NP$)

- $NP \subseteq NP^A$
- $NP^A \subseteq NP$

Sei $L \in NP^A$, d.h. L wird von einer nicht deterministischer Turingmaschine mit Zugriff auf A-Orcakel in Polynomieller Zeit akteptiert. Nach an Annahme existieren Np-Tm für A und für $\overline{A}$. Ersetze eine Orakelanfrage durch:

1. Rate, ob das Wort in A oder in $\overline{A}$ ist
2. Führe die Tm für A und $\overline{A}$ aus.
3. Wenn die Antwort dem geratenem Wert entspricht, dann fahre mit der Berechnung fort. Sonst ablehnen.

0.12. 6-3

Sei $A \in \Sigma*$ Da Beweis der Zeithierachisatz “relativiert”, d.h z.B. $D^A \subsetneq EXP^A$

Beweis Wir haben $P^A \subseteq DTIME^A(2^k), dennn^k \in O(2^n)$ für alle k.

Definiere $U = \{M|M$ kodiert eine TM mit A-Orcakel-Zugriff welcher die Eingabe M in höchstens $2^{|M|}$ Schritten akteptiert$\}$.

Es gibt eine Tm mit A-Oracel, die U in Zeit $O(2^{2n})$.

Zeige $U \notin DTIME^A(2^n)$ Ausgenommen $U \in DTIME^A(2^n)$. Dann ist auch $D(x) = ifU(x) = 1$ then 0 else 1

in $DTIME^A(2^n)$ Aber $O(D) = 1 \Leftrightarrow U(D) = 0 \Rightarrow D$ akzeptiert D nicht in $<= 2^n$ Schritten $D(D) = 0$ Das ist ein Wiederspruch.

0.13. 6-4

Problem Unabhängige Menge

$U = \{(G, k)|G$ hat unabhaengige Menge der Größe k $\}$ ist Np-vollständig.

Subproblem: Geg(G,k), finde Unabhängige Menge der Größe k.

0.13.1. Eine Lösung

1. Überprüfe, ob G eine Unabhängige Menge der Größe k hat, wenn nicht, Ausgabe nein
2. Solange G noch mehr als k Knoten hat:
   a) for $v \in V\{$
      i. Wenn $G\backslash\{v\}$ eine Unabhängige Mege der Größe k hat, denn $G := G\backslash\{v\}$
3. Die Knoten in G sind die gesuchte Unabhängige Menge.

0.13.2. Allgemeine Lösung

$L = \{x|\exists y \in \{0,1\}^{p(|x|)} M(x,y) = 1\}$ (Entscheidungsproblem)

Suchproblem Gegeben. x, berechne $y \in \{0,1\}^{\leq}$ mit $M(x,y) = 1$ Wenn L NP-vollständig, dann kann das Suchproblem auf das Entscheidungsporblem reduziert werden. D.h. es existiert eine TM für das Suchproblem mit L-Orakel.

Definiere: $L' = \{(x,b)|\exists y|by| \le p(|x|) \land M(x,by) = 1\}$ Wegen Np-vollständigkeit von L gilt $L' \le_p L$, d.h. es gibt DTIME-Fkt. f, mit $(x,b) \in L' \Leftrightarrow f(x,b) \in L$

Allgorithmus f[r Suchproblem:Eingabe x for i=1 to p(x)
$b_i = 0$
if $f(x, b_1 \dots b_i) \notin L$ then $b_i = 1$
enf for

Loesung ist $b_1 \dots b_{p(x)}$

0.14. 7-1

$\Sigma_0 = \Pi_0 = \Delta_0 = P$ $\Sigma_{i+1} = NP^{\Sigma_i}$ $\Delta_{i+1} = coNP^{\Sigma_i}$ $\Delta_{i+1} = p^{\Sigma_i}$

Charakterisierung

- $L \in \Sigma_i \Leftrightarrow$ Es gibt Polynome $p_1, \dots, p_n$ und eine Polynomialzeit TM M, so dass $x \in L \Leftrightarrow \exists x_1 \in \{0,1\}^{P_1(|x|)} \forall x_2 \in \{0,1\}^{P_2(|x|)} \forall x_3 \in \{0,1\}^{P_3(|x|)} \dots Q_i x_i \in \{0,1\}^{P_i(|x|)} M(x, x_1, x_2, \dots x_i$
- $L \in \Pi_i \Leftrightarrow$ Es gibt Polynome $p_1, \dots, p_n$ und eine Polynomialzeit TM M, so dass $x \in L \Leftrightarrow \forall x_1 \in \{0,1\}^{P_1(|x|)} \forall x_2 \in \{0,1\}^{P_2(|x|)} \dots Q_i x_i \in \{0,1\}^{P_i(|x|)} M(x, x_1, x_2, \dots x_i$

b

- $\Sigma_k \subseteq \Pi_k \Rightarrow \Sigma_k = \Pi_k$ Sei $\Sigma_k \subseteq$ Z.z $\Pi_k \subseteq \Sigma_k$ Sei $L \in \Pi_k = co\Sigma_k \Rightarrow \overline{L} \in \Sigma_k$ Nach Annahme $\overline{L} \in \Pi_k$ Wegen $\Pi_k = co\Sigma_k$ folgt $\overline{\overline{L}} \in \overline{Z}_k \Rightarrow L \in \Sigma_k$
- Zeige durch Indektion über $i \ge k\Sigma_i = \Pi_i = \Sigma_k$
  - I.A. $i = k$ Nach annahme gilt $\Sigma_k = \Pi_k \Rightarrow \Sigma_k = \Pi_k$
  - I.S. Sei $\Sigma_i = \Pi_i = \Sigma_k$ für eine $i \ge k$ Zeige $\Sigma_{i+1} = \Pi_{i+1} = \Sigma_k \dots$

0.15. 7-2

Nach 7-1 gezeigt $\Sigma_1 \subseteq \pi_1$ d.h. $NP \subseteq coNP \Rightarrow \leq_p \overline{Sat}$ wegen der Transitivität von $\leq_p$ Es gilt $\overline{Sat} \in CoNP$ Es gibt polynomlzeitberechenbare Funktionen f mit ...

0.16. 7-3

0.16.1. a

Äquivalenz $\forall_n$ wenn die eine Formel drue liefert genau dann liefert auch die andere Formel true.$\Rightarrow L \in \Pi_1$

0.16.2. b

- $\Pi_2 = P^{NP}$
  - Benutye Orakel für (G, k+1) um zu prüfen, ob es eine größere unabhängige Menge gibt.
  - Benutze Orakel für (G,k) um Existenz einer unabhängigen Menge der Größe k zu testen.

0.16.3. b

- Ein Knoten im Graph benötigt logarithmischen Platz. (log n Bits um Zahlen in 0,1,2,3 zu kodieren.
- Menge von Kknoten benötigt Platz k log n

Rightarrow Man kann alle Mengen von k Knoten durchprobieren und jeweils Unabhängigkeit testen.

0.17. 8-1

siehe Vorlesung, angeblich trivial

0.18. 8-2

Wie lang können Ableitungen $S \rightarrow *w$ sein? Ableitungen haben die folgenden Form $S \rightarrow \alpha_1^1 \rightarrow \alpha_2^1 \cdots \rightarrow \alpha_{n1}^1 \rightarrow \alpha_1^2 \rightarrow \alpha_2^2 \cdots \rightarrow \alpha_{n2}^2 \cdots \rightarrow \alpha_1^n \rightarrow \alpha_2^n \cdots \rightarrow \alpha_{nn}^n$

Wobei der Haufen mit α_x^1 sind Satzformen der Länge 1 mit Anzahl m, der Haufen α_x^2 sind Satzformen der Länge 2 mit Anzahl m^2 usw.

$\Rightarrow$ Länge $\leq m + m^2 \cdots + m^n \leq m^{n+1} \in 2^{O(c)}$

Algorithmus von Savitch: $reach(\alpha, \beta, k) =$

- if$k = 0$ then $\alpha = \beta$
- if$k = 1$ then $\alpha \rightarrow \beta$
- else for $\gamma \in (\Sigma \cup \Gamma)^n$:
  - if reach $(\alpha, \gamma, [\frac{k}{2}]) \wedge reach(\gamma, \beta, [\frac{k}{2}])$
  - then return true

Insgesamt Einge w Berechne reach(S,w,$2^{k \times irgendwas}$), wobei k und l so gewählt sind, dass $2^{k \times |w| + l}$ die Länge aller möglichen Zyklischen Ableitungen $S \rightarrow^* w$ beschränkt. Platzverbrauch: Rekursionstiefe: $log(2^{k \times |w| + l} = k \times |w| + l$

Strackfframe:

- Platz für $\alpha, \beta, \gamma \leq 3 \times |w|$
- Platz für $k \leq k \times |w| + l$

$\Rightarrow$ Isgesamt Rekusionstiefe $\times$ Stackframe $\in O(|w|^2)$

0.19. 8-3

Zeige: $Horn \leq_{log} \overline{CFGempty}$ und $\overline{CFGempty} \leq_{log} Horn$

0.20. 8-4

0.20.1. a

Der Beweis der NL Vollständigkeit von Reach kann man annehmen, dass der Konfigurationsgraph der TM kreisfrei ist, indem man die MAschine um ein Band mit einem Schrittzähler weritert.

0.20.2. b

Reduktion von Erreichbarkeit in Kreisfreiem Graphen. s von t in kreisfreiem Graph erreichbar g.d.w. G' = (V, $E \cup \{(t,s)\}$ kreis hat.

0.21. 10-1

Vorschläge:

- $A = QBF$
  $P^{QBF} = PSPACE$
  - $PSPACE \subseteq P^{QBF}$
    Sei $L \in PSPACE$. QBF ist $PSPACE$-vollständig $\Rightarrow$ Es gibt Polynomialzeitberechenbare Funktion f mit $x \in L \Leftrightarrow f(x) \in QBF$ Eine TM in $P^{Q}BF$ lässt sich ausgeben durch:
    * Berechne $f(x)$ und befrage das QBF-Orakel
    * Entsprechend der Antwort des Orakels wird angenommen oder Abgelehnt
  - $P^B \neq PSPACE^B$ // Aus Vorlesung: Es existiert B mit $P^B \subsetneq NP^B$
    Es gilt $NP^B \subseteq PSPACE^B$.

Das kann man so erklären: // $L \in NP^B$ genaudann wenn es Polynomp und P^B TM M gibt mit: $L = \{x|\exists y \in \{0,1\}^{|P(x)|}M...$ da hat er weggewischt.

0.22. 10-2

$QBF \in E$

$\phi, \xi := \top|\bot|x|\phi \wedge \xi|\neg\phi|\forall x.\phi|\exists x.\phi$
Ansatz $\forall x\phi \Leftrightarrow \phi[\bot x] \wedge \phi[\top x]$
$\exists x.\phi \Leftrightarrow \phi[\bot x] \vee \phi[\top x]$

Bei Eingabe ϕ werden alle Quantoren expandiert. Jede Expansion kann die Formel höchstens verdopplen und braucht lineare Zeit. Insgesamt höchstens n Verdoppleungen($n = |p|$) . Zeit: $2n + 4n + 8n + \cdots + 2^n * n \leq 2^{n+1}n \leq 2^{n+1} * 2^n = 2^{n+1} \leq 2^{3n}$ Am ende noch Auswerten in linearer Zeit. $\Rightarrow Zeitinsgesamt \not 2^{3n}$

Alternative:
eval(T, n) = T
eval($\phi \wedge \phi$,n) = if eval(ϕ,n) = $\bot$ then $\bot$ else eval(ϕ,n)
eval($\forall x.\phi$, n) = if eval(ϕ, n [$x := \bot$]) = $\bot$ then $\bot$ else eval(ϕ,n [$x := \top$])
...

Warum folgt nicht $PSPACE \subseteq E$? und schwupps war die Tafel wieder gewischt.

0.22.1. 10-3

Gegeben: Graph G mit Kntoen s und t
Gesicht: Länge des kürzestens PFades von s nach t (oder unendlich wenn keiner existiert).

Entscheidungsproblem $L = \{(G, s, t, n)|$ Der kürzeste PFad vons nach t hat die Länge n $\}$ zeige $L \in NL$:
Dazu:

- $L_n = \{(G, s, t, n)|$ Es gibt Pfad von s nach t der Länge n $\}$ ist in NL

Algorithmus: Beginne bei s. Rate die Nachfolge, solange bis n Schritte gemacht wurden. Akzeptiere, wenn am Ende t nicht erreicht ist.

- $\overline{L_n} \in NL$ $\overline{L_n} = \{(G, s, t, n)|$ Es gibt keinen Pfad von s nach t der Länge n $\}$

Damit $L = L_n \cap \overline{L_{n-1}}$. Da NLunter Schnitt abgeschlosse folgen $L \in NL$

Klasse $FNL|$:
Eine Funktion f ist in FNL falls gilt: Es gibt eine nicht det. TM M mit logarithmischem Platzverbrauch, so dass gilt: $f(x) = y$ genaudannwenn Die Maschine M hat bei Eingabe x einen akzeptierenden Lauf mit Ausgabe y.

Akzeptierende Definition: $FNL = FL^{NL}$
$FNL \subseteq FL^{NL}$:
Benutze als Orakel: $\{(x, i, c)|$ Bei Eingabe x ist c das i-te Zeichen in der Ausgabe $\}$.