

## Script generated by TTT

Title: Seidl: Theoretische\_Informatik  
(08.07.2013)

Date: Mon Jul 08 10:16:34 CEST 2013

Duration: 89:02 min

Pages: 127

### Zwei zentrale Komplexitätsklassen:

**P** = die von einer DTM in polynomieller Zeit lösbaren Probleme  
= die „leichten“ Probleme (*feasible* problems)

**NP** = die von einer NTM in polynomieller Zeit lösbaren Probleme  
In exponentieller Zeit lösbar (s. Simulation NTM durch DTM)

Zentrale Frage:

¿  $P = NP$  ?

Ist wichtig

- weil viele *praktisch relevante* Such- und Optimierungsprobleme in NP liegen

### Zwei zentrale Komplexitätsklassen:

**P** = die von einer DTM in polynomieller Zeit lösbaren Probleme  
= die „leichten“ Probleme (*feasible* problems)

**NP** = die von einer NTM in polynomieller Zeit lösbaren Probleme  
In exponentieller Zeit lösbar (s. Simulation NTM durch DTM)

Zentrale Frage:

¿  $P = NP$  ?

### Zwei zentrale Komplexitätsklassen:

**P** = die von einer DTM in polynomieller Zeit lösbaren Probleme  
= die „leichten“ Probleme (*feasible* problems)

**NP** = die von einer NTM in polynomieller Zeit lösbaren Probleme  
In exponentieller Zeit lösbar (s. Simulation NTM durch DTM)

Zentrale Frage:

¿  $P = NP$  ?

Ist wichtig

- weil viele *praktisch relevante* Such- und Optimierungsprobleme in NP liegen
- und alle schnell lösbar wären wenn eines schnell lösbar wäre.

### Überblick:

- 1 Die Komplexitätsklassen P und NP

### Überblick:

- 1 Die Komplexitätsklassen P und NP
- 2 NP-Vollständigkeit
- 3 SAT: Ein NP-vollständiges Problem

### Überblick:

- 1 Die Komplexitätsklassen P und NP
- 2 NP-Vollständigkeit

### 5.1 Die Komplexitätsklasse P

Berechnungsmodelle:

DTM = deterministische Mehrband-TM

NTM = nichtdeterministische Mehrband-TM

## 5.1 Die Komplexitätsklasse P

Berechnungsmodelle:

DTM = deterministische Mehrband-TM

NTM = nichtdeterministische Mehrband-TM

### Definition 5.1

$time_M(w)$  := Anzahl der Schritte bis die DTM  $M[w]$  hält

## 5.1 Die Komplexitätsklasse P

Berechnungsmodelle:

DTM = deterministische Mehrband-TM

NTM = nichtdeterministische Mehrband-TM

### Definition 5.1

$time_M(w)$  := Anzahl der Schritte bis die DTM  $M[w]$  hält  
 $\in \mathbb{N} \cup \{\infty\}$

Sei  $f : \mathbb{N} \rightarrow \mathbb{N}$  eine totale Funktion.

## 5.1 Die Komplexitätsklasse P

Berechnungsmodelle:

DTM = deterministische Mehrband-TM

NTM = nichtdeterministische Mehrband-TM

### Definition 5.1

$time_M(w)$  := Anzahl der Schritte bis die DTM  $M[w]$  hält  
 $\in \mathbb{N} \cup \{\infty\}$

## 5.1 Die Komplexitätsklasse P

Berechnungsmodelle:

DTM = deterministische Mehrband-TM

NTM = nichtdeterministische Mehrband-TM

### Definition 5.1

$time_M(w)$  := Anzahl der Schritte bis die DTM  $M[w]$  hält  
 $\in \mathbb{N} \cup \{\infty\}$

Sei  $f : \mathbb{N} \rightarrow \mathbb{N}$  eine totale Funktion.

Die Klasse der in Zeit  $f(n)$  entscheidbaren Sprachen:

## 5.1 Die Komplexitätsklasse P

Berechnungsmodelle:

DTM = deterministische Mehrband-TM

NTM = nichtdeterministische Mehrband-TM

### Definition 5.1

$time_M(w) :=$  Anzahl der Schritte bis die DTM  $M[w]$  hält  
 $\in \mathbb{N} \cup \{\infty\}$

Sei  $f : \mathbb{N} \rightarrow \mathbb{N}$  eine totale Funktion.

Die Klasse der in Zeit  $f(n)$  entscheidbaren Sprachen:

$TIME(f(n)) := \{A \subseteq \Sigma^* \mid \exists \text{DTM } M. A = L(M) \wedge$   
 $\forall w \in \Sigma^*. time_M(w) \leq f(|w|)\}$

## 5.1 Die Komplexitätsklasse P

Berechnungsmodelle:

DTM = deterministische Mehrband-TM

NTM = nichtdeterministische Mehrband-TM

### Definition 5.1

$time_M(w) :=$  Anzahl der Schritte bis die DTM  $M[w]$  hält  
 $\in \mathbb{N} \cup \{\infty\}$

Sei  $f : \mathbb{N} \rightarrow \mathbb{N}$  eine totale Funktion.

Die Klasse der in Zeit  $f(n)$  entscheidbaren Sprachen:

$TIME(f(n)) := \{A \subseteq \Sigma^* \mid \exists \text{DTM } M. A = L(M) \wedge$   
 $\forall w \in \Sigma^*. time_M(w) \leq f(|w|)\}$

Merke:

- $n$  ist implizit die Länge der Eingabe
- Die DTM entscheidet die Sprache  $A$  in  $\leq f(n)$  Schritten

## 5.1 Die Komplexitätsklasse P

Berechnungsmodelle:

DTM = deterministische Mehrband-TM

NTM = nichtdeterministische Mehrband-TM

### Definition 5.1

$time_M(w) :=$  Anzahl der Schritte bis die DTM  $M[w]$  hält  
 $\in \mathbb{N} \cup \{\infty\}$

Sei  $f : \mathbb{N} \rightarrow \mathbb{N}$  eine totale Funktion.

Die Klasse der in Zeit  $f(n)$  entscheidbaren Sprachen:

$TIME(f(n)) := \{A \subseteq \Sigma^* \mid \exists \text{DTM } M. A = L(M) \wedge$   
 $\forall w \in \Sigma^*. time_M(w) \leq f(|w|)\}$

Merke:

- $n$  ist implizit die Länge der Eingabe

Wir betrachten nur Polynome mit Koeffizienten  $a_k, \dots, a_0 \in \mathbb{N}$ :

$$p(n) = a_k n^k + \dots + a_1 n + a_0$$

Wir betrachten nur Polynome mit Koeffizienten  $a_k, \dots, a_0 \in \mathbb{N}$ :

$$p(n) = a_k n^k + \dots + a_1 n + a_0$$

Definition 5.2

$$P := \bigcup_{p \text{ Polynom}} \text{TIME}(p(n))$$

Wir betrachten nur Polynome mit Koeffizienten  $a_k, \dots, a_0 \in \mathbb{N}$ :

$$p(n) = a_k n^k + \dots + a_1 n + a_0$$

Definition 5.2

$$P := \bigcup_{p \text{ Polynom}} \text{TIME}(p(n))$$

Damit gilt auch

$$P = \bigcup_{k \geq 0} \text{TIME}(O(n^k))$$

wobei

$$\text{TIME}(O(f)) := \bigcup_{g \in O(f)} \text{TIME}(g)$$

Wir betrachten nur Polynome mit Koeffizienten  $a_k, \dots, a_0 \in \mathbb{N}$ :

$$p(n) = a_k n^k + \dots + a_1 n + a_0$$

Definition 5.2

$$P := \bigcup_{p \text{ Polynom}} \text{TIME}(p(n))$$

Damit gilt auch

$$P = \bigcup_{k \geq 0} \text{TIME}(O(n^k))$$

Wir betrachten nur Polynome mit Koeffizienten  $a_k, \dots, a_0 \in \mathbb{N}$ :

$$p(n) = a_k n^k + \dots + a_1 n + a_0$$

Definition 5.2

$$P := \bigcup_{p \text{ Polynom}} \text{TIME}(p(n))$$

Damit gilt auch

$$P = \bigcup_{k \geq 0} \text{TIME}(O(n^k))$$

wobei

$$\text{TIME}(O(f)) := \bigcup_{g \in O(f)} \text{TIME}(g)$$

### Beispiel 5.3

- $\{ww^R \mid w \in \Sigma^*\} \in TIME(O(n^2)) \subseteq P$

### Beispiel 5.3

- $\{ww^R \mid w \in \Sigma^*\} \in TIME(O(n^2)) \subseteq P$
- $\{(G, w) \mid G \text{ ist CFG} \wedge w \in L(G)\} \in P$

### Beispiel 5.3

- $\{ww^R \mid w \in \Sigma^*\} \in TIME(O(n^2)) \subseteq P$
- $\{(G, w) \mid G \text{ ist CFG} \wedge w \in L(G)\} \in P$
- $\{(G, w) \mid G \text{ ist Graph} \wedge w \text{ ist Pfad in } G\} \in P$

### Beispiel 5.3

- $\{ww^R \mid w \in \Sigma^*\} \in TIME(O(n^2)) \subseteq P$
- $\{(G, w) \mid G \text{ ist CFG} \wedge w \in L(G)\} \in P$
- $\{(G, w) \mid G \text{ ist Graph} \wedge w \text{ ist Pfad in } G\} \in P$
- $\{bin(p) \mid p \text{ Primzahl}\} \in P$

### Beispiel 5.3

- $\{ww^R \mid w \in \Sigma^*\} \in TIME(O(n^2)) \subseteq P$
- $\{(G, w) \mid G \text{ ist CFG} \wedge w \in L(G)\} \in P$
- $\{(G, w) \mid G \text{ ist Graph} \wedge w \text{ ist Pfad in } G\} \in P$
- $\{bin(p) \mid p \text{ Primzahl}\} \in P$

Beweis durch Angabe eines Algorithmus mit Komplexität  $O(n^k)$ .

### Beispiel 5.3

- $\{ww^R \mid w \in \Sigma^*\} \in TIME(O(n^2)) \subseteq P$
- $\{(G, w) \mid G \text{ ist CFG} \wedge w \in L(G)\} \in P$
- $\{(G, w) \mid G \text{ ist Graph} \wedge w \text{ ist Pfad in } G\} \in P$
- $\{bin(p) \mid p \text{ Primzahl}\} \in P$

Beweis durch Angabe eines Algorithmus mit Komplexität  $O(n^k)$ .

### Bemerkungen

- $O(n \log n) \subset O(n^2)$
- $n^{\log n}, 2^n \notin O(n^k)$  für alle  $k$

### Beispiel 5.3

- $\{ww^R \mid w \in \Sigma^*\} \in TIME(O(n^2)) \subseteq P$
- $\{(G, w) \mid G \text{ ist CFG} \wedge w \in L(G)\} \in P$
- $\{(G, w) \mid G \text{ ist Graph} \wedge w \text{ ist Pfad in } G\} \in P$
- $\{bin(p) \mid p \text{ Primzahl}\} \in P$

Beweis durch Angabe eines Algorithmus mit Komplexität  $O(n^k)$ .

### Bemerkungen

- $O(n \log n) \subset O(n^2)$

### Beispiel 5.3

- $\{ww^R \mid w \in \Sigma^*\} \in TIME(O(n^2)) \subseteq P$
- $\{(G, w) \mid G \text{ ist CFG} \wedge w \in L(G)\} \in P$
- $\{(G, w) \mid G \text{ ist Graph} \wedge w \text{ ist Pfad in } G\} \in P$
- $\{bin(p) \mid p \text{ Primzahl}\} \in P$

Beweis durch Angabe eines Algorithmus mit Komplexität  $O(n^k)$ .

### Bemerkungen

- $O(n \log n) \subset O(n^2)$
- $n^{\log n}, 2^n \notin O(n^k)$  für alle  $k$
- Beweis  $A \notin P$  meist schwierig

### Beispiel 5.3

- $\{ww^R \mid w \in \Sigma^*\} \in TIME(O(n^2)) \subseteq P$
- $\{(G, w) \mid G \text{ ist CFG} \wedge w \in L(G)\} \in P$
- $\{(G, w) \mid G \text{ ist Graph} \wedge w \text{ ist Pfad in } G\} \in P$
- $\{bin(p) \mid p \text{ Primzahl}\} \in P$

Beweis durch Angabe eines Algorithmus mit Komplexität  $O(n^k)$

### Bemerkungen

- $O(n \log n) \subset O(n^2)$
- $n^{\log n}, 2^n \notin O(n^k)$  für alle  $k$
- Beweis  $A \notin P$  meist schwierig

- Warum P und nicht (zB)  $O(n^{17})$ ?  
Um robust bzgl Maschinenmodell zu sein:

- Warum P und nicht (zB)  $O(n^{17})$ ?

- Warum P und nicht (zB)  $O(n^{17})$ ?  
Um robust bzgl Maschinenmodell zu sein:  
1-Band DTM braucht  $O(t^2)$  Schritte  
um  $t$  Schritte einer  $k$ -Band DTM zu simulieren.



- Warum P und nicht (zB)  $O(n^{17})$ ?  
Um robust bzgl Maschinenmodell zu sein:  
1-Band DTM braucht  $O(t^2)$  Schritte  
um  $t$  Schritte einer  $k$ -Band DTM zu simulieren.  
Fast alle bekannten „vernünftigen“ Maschinenmodelle lassen sich von einer DTM in polynomieller Zeit simulieren.

- Warum P und nicht (zB)  $O(n^{17})$ ?  
Um robust bzgl Maschinenmodell zu sein:  
1-Band DTM braucht  $O(t^2)$  Schritte  
um  $t$  Schritte einer  $k$ -Band DTM zu simulieren.  
Fast alle bekannten „vernünftigen“ Maschinenmodelle lassen sich von einer DTM in polynomieller Zeit simulieren.  
Offen: Quantencomputer

- Warum TM? Historisch.

- Warum P und nicht (zB)  $O(n^{17})$ ?  
Um robust bzgl Maschinenmodell zu sein:  
1-Band DTM braucht  $O(t^2)$  Schritte  
um  $t$  Schritte einer  $k$ -Band DTM zu simulieren.  
Fast alle bekannten „vernünftigen“ Maschinenmodelle lassen sich von einer DTM in polynomieller Zeit simulieren.  
Offen: Quantencomputer

- Warum P und nicht (zB)  $O(n^{17})$ ?  
Um robust bzgl Maschinenmodell zu sein:  
1-Band DTM braucht  $O(t^2)$  Schritte  
um  $t$  Schritte einer  $k$ -Band DTM zu simulieren.  
Fast alle bekannten „vernünftigen“ Maschinenmodelle lassen sich von einer DTM in polynomieller Zeit simulieren.  
Offen: Quantencomputer


- Warum TM? Historisch.  
Ebenfalls möglich: (zB) WHILE.  
Aber zwei mögliche Kostenmodelle:

- Warum P und nicht (zB)  $O(n^{17})$ ?  
Um robust bzgl Maschinenmodell zu sein:  
1-Band DTM braucht  $O(t^2)$  Schritte  
um  $t$  Schritte einer  $k$ -Band DTM zu simulieren.  
Fast alle bekannten „vernünftigen“ Maschinenmodelle lassen sich von einer DTM in polynomieller Zeit simulieren.  
Offen: Quantencomputer
- Warum TM? Historisch.  
Ebenfalls möglich: (zB) WHILE.  
Aber zwei mögliche Kostenmodelle:  
Uniform  $x_i := x_j + n$  kostet 1 Zeiteinheit.

## 5.2 Die Komplexitätsklasse NP

Intuitiv:

- NP die Klasse der Sprachen, die von einer NTM in polynomieller Zeit akzeptiert werden.

- Warum P und nicht (zB)  $O(n^{17})$ ?  
Um robust bzgl Maschinenmodell zu sein:  
1-Band DTM braucht  $O(t^2)$  Schritte  
um  $t$  Schritte einer  $k$ -Band DTM zu simulieren.  
Fast alle bekannten „vernünftigen“ Maschinenmodelle lassen sich von einer DTM in polynomieller Zeit simulieren.  
Offen: Quantencomputer
- Warum TM? Historisch.  
Ebenfalls möglich: (zB) WHILE.  
Aber zwei mögliche Kostenmodelle:  
Uniform  $x_i := x_j + n$  kostet 1 Zeiteinheit.  
 Logarithmisch  $x_i := x_j + n$  kostet  $\log x_j$  Zeiteinheiten.

## 5.2 Die Komplexitätsklasse NP

Intuitiv:

- NP die Klasse der Sprachen, die von einer NTM in polynomieller Zeit akzeptiert werden.
- Dh: Eine Sprache  $A$  liegt in NP gdw es ein Polynom  $p(n)$  und eine NTM  $M$  gibt, so dass

## 5.2 Die Komplexitätsklasse NP

Intuitiv:

- NP die Klasse der Sprachen, die von einer NTM in polynomieller Zeit akzeptiert werden.
- Dh: Eine Sprache  $A$  liegt in NP gdw es ein Polynom  $p(n)$  und eine NTM  $M$  gibt, so dass
  - 1  $L(M) = A$  und

Definition 5.4

$$ntime_M(w) := \begin{cases} \text{minimale Anzahl der Schritte} & \text{falls } w \in L(M) \\ \text{bis NTM } M[w] \text{ akzeptiert} & \\ 0 & \text{falls } w \notin L(M) \end{cases}$$

## 5.2 Die Komplexitätsklasse NP

Intuitiv:

- NP die Klasse der Sprachen, die von einer NTM in polynomieller Zeit akzeptiert werden.
- Dh: Eine Sprache  $A$  liegt in NP gdw es ein Polynom  $p(n)$  und eine NTM  $M$  gibt, so dass
  - 1  $L(M) = A$  und
  - 2 für alle  $w \in A$  kann  $M[w]$  in  $\leq p(|w|)$  Schritten akzeptieren, dh einen Endzustand erreichen.

Definition 5.4

$$ntime_M(w) := \begin{cases} \text{minimale Anzahl der Schritte} & \text{falls } w \in L(M) \\ \text{bis NTM } M[w] \text{ akzeptiert} & \\ 0 & \text{falls } w \notin L(M) \end{cases}$$

Sei  $f : \mathbb{N} \rightarrow \mathbb{N}$  eine totale Funktion.

$$NTIME(f(n)) := \{A \subseteq \Sigma^* \mid \exists \text{NTM } M. A = L(M)\}$$

### Definition 5.4

$$ntime_M(w) := \begin{cases} \text{minimale Anzahl der Schritte} & \text{falls } w \in L(M) \\ \text{bis NTM } M[w] \text{ akzeptiert} & \\ 0 & \text{falls } w \notin L(M) \end{cases}$$

Sei  $f : \mathbb{N} \rightarrow \mathbb{N}$  eine totale Funktion.

$$NTIME(f(n)) := \{A \subseteq \Sigma^* \mid \exists \text{NTM } M. A = L(M) \wedge \forall w \in \Sigma^*. ntime_M(w) \leq f(|w|)\}$$

### Definition 5.4

$$ntime_M(w) := \begin{cases} \text{minimale Anzahl der Schritte} & \text{falls } w \in L(M) \\ \text{bis NTM } M[w] \text{ akzeptiert} & \\ 0 & \text{falls } w \notin L(M) \end{cases}$$

Sei  $f : \mathbb{N} \rightarrow \mathbb{N}$  eine totale Funktion.

$$NTIME(f(n)) := \{A \subseteq \Sigma^* \mid \exists \text{NTM } M. A = L(M) \wedge \forall w \in \Sigma^*. ntime_M(w) \leq f(|w|)\}$$

$$NP := \bigcup_{p \text{ Polynom}} NTIME(p(n))$$

### Definition 5.4

$$ntime_M(w) := \begin{cases} \text{minimale Anzahl der Schritte} & \text{falls } w \in L(M) \\ \text{bis NTM } M[w] \text{ akzeptiert} & \\ 0 & \text{falls } w \notin L(M) \end{cases}$$

Sei  $f : \mathbb{N} \rightarrow \mathbb{N}$  eine totale Funktion.

$$NTIME(f(n)) := \{A \subseteq \Sigma^* \mid \exists \text{NTM } M. A = L(M) \wedge \forall w \in \Sigma^*. ntime_M(w) \leq f(|w|)\}$$

$$NP := \bigcup_{p \text{ Polynom}} NTIME(p(n))$$

### Bemerkungen:

- $P \subseteq NP$

### Definition 5.4

$$ntime_M(w) := \begin{cases} \text{minimale Anzahl der Schritte} & \text{falls } w \in L(M) \\ \text{bis NTM } M[w] \text{ akzeptiert} & \\ 0 & \text{falls } w \notin L(M) \end{cases}$$

Sei  $f : \mathbb{N} \rightarrow \mathbb{N}$  eine totale Funktion.

$$NTIME(f(n)) := \{A \subseteq \Sigma^* \mid \exists \text{NTM } M. A = L(M) \wedge \forall w \in \Sigma^*. ntime_M(w) \leq f(|w|)\}$$

$$NP := \bigcup_{p \text{ Polynom}} NTIME(p(n))$$

### Bemerkungen:

- $P \subseteq NP$
- Seit etwa 1970 ist offen ob  $P = NP$ .

Bemerkungen zur Definition von NP:

Akzeptierende NTM  $M$

- braucht für  $w \notin L(M)$  nicht zu halten.

Bemerkungen zur Definition von NP:

Akzeptierende NTM  $M$

- braucht für  $w \notin L(M)$  nicht zu halten.
- kann für  $w \in L(M)$  auch beliebig lange Berechnungsfolgen haben.

Äquivalente Definition NP' von NP:

Die NTM  $M[w]$  muss nach maximal  $p(|w|)$  Schritten halten.

Bemerkungen zur Definition von NP:

Akzeptierende NTM  $M$

- braucht für  $w \notin L(M)$  nicht zu halten.
- kann für  $w \in L(M)$  auch beliebig lange Berechnungsfolgen haben.

Bemerkungen zur Definition von NP:

Akzeptierende NTM  $M$

- braucht für  $w \notin L(M)$  nicht zu halten.
- kann für  $w \in L(M)$  auch beliebig lange Berechnungsfolgen haben.

Äquivalente Definition NP' von NP:

Die NTM  $M[w]$  muss nach maximal  $p(|w|)$  Schritten halten.

NP'  $\subseteq$  NP: Klar.

Bemerkungen zur Definition von NP:

Akzeptierende NTM  $M$

- braucht für  $w \notin L(M)$  nicht zu halten.
- kann für  $w \in L(M)$  auch beliebig lange Berechnungsfolgen haben.

Äquivalente Definition NP' von NP:

Die NTM  $M[w]$  muss nach maximal  $p(|w|)$  Schritten halten.

$NP' \subseteq NP$ : Klar.

$NP \subseteq NP'$ : Falls  $A \in NP$  mit Polynom  $p$  und NTM  $M$ , so kann man NTM  $M'$  konstruieren mit  $L(M') = A$ , so dass  $M'[w]$  immer innerhalb von polynomieller Zeit hält.

- 1 Eingabe  $w$
- 2 Schreibe  $p(|w|)$  auf ein getrenntes Band („timer“)
- 3 Simuliere  $M[w]$ , aber dekrementiere timer nach jedem Schritt.
- 4 Wird timer=0, ohne dass  $M$  gehalten hat, halte in einem nicht-Endzustand („timeout“)

### Beispiel 5.5

- Ein Euler-Kreis ist ein geschlossener Pfad in einem (ungerichteten) Graphen, der jede Kante genau einmal enthält.
- Ein Graph hat einen Euler-Kreis gdw er zusammenhängend ist, und jeder Knoten geraden Grad hat.

### Beispiel 5.5

- Ein Euler-Kreis ist ein geschlossener Pfad in einem (ungerichteten) Graphen, der jede Kante genau einmal enthält.

### Beispiel 5.5

- Ein Euler-Kreis ist ein geschlossener Pfad in einem (ungerichteten) Graphen, der jede Kante genau einmal enthält.
- Ein Graph hat einen Euler-Kreis gdw er zusammenhängend ist, und jeder Knoten geraden Grad hat.
- Beide Eigenschaften sind in polynomieller Zeit von einer DTM überprüfbar.

### Beispiel 5.6

- Ein Hamilton-Kreis ist ein geschlossener Pfad in einem (ungerichteten) Graphen, der jeden Knoten genau einmal enthält.

### Beispiel 5.6

- Ein Hamilton-Kreis ist ein geschlossener Pfad in einem (ungerichteten) Graphen, der jeden Knoten genau einmal enthält.
- HAMILTON :=  $\{G \mid G \text{ hat Hamilton-Kreis}\} \in \text{NP}$  mit folgendem Algorithmus der Art „Rate und prüfe“:

### Beispiel 5.6

- Ein Hamilton-Kreis ist ein geschlossener Pfad in einem (ungerichteten) Graphen, der jeden Knoten genau einmal enthält.
- HAMILTON :=  $\{G \mid G \text{ hat Hamilton-Kreis}\} \in \text{NP}$

### Beispiel 5.6

- Ein Hamilton-Kreis ist ein geschlossener Pfad in einem (ungerichteten) Graphen, der jeden Knoten genau einmal enthält.
- HAMILTON :=  $\{G \mid G \text{ hat Hamilton-Kreis}\} \in \text{NP}$  mit folgendem Algorithmus der Art „Rate und prüfe“:
  - ① Rate eine Permutation der Knoten des Graphen.

### Beispiel 5.6

- Ein Hamilton-Kreis ist ein geschlossener Pfad in einem (ungerichteten) Graphen, der jeden Knoten genau einmal enthält.
- HAMILTON :=  $\{G \mid G \text{ hat Hamilton-Kreis}\} \in \text{NP}$  mit folgendem Algorithmus der Art „Rate und prüfe“:
  - ① Rate eine Permutation der Knoten des Graphen.
  - ② Prüfe, ob diese Permutation ein Hamilton-Kreis ist.

Viele Probleme sind von der Art dass

- schwer ist, zu entscheiden, ob sie lösbar sind,

### Beispiel 5.6

- Ein Hamilton-Kreis ist ein geschlossener Pfad in einem (ungerichteten) Graphen, der jeden Knoten genau einmal enthält.
- HAMILTON :=  $\{G \mid G \text{ hat Hamilton-Kreis}\} \in \text{NP}$  mit folgendem Algorithmus der Art „Rate und prüfe“:
  - ① Rate eine Permutation der Knoten des Graphen.
  - ② Prüfe, ob diese Permutation ein Hamilton-Kreis ist.Das Raten ist in polynomieller Zeit von einer NTM machbar.  
Das Prüfen ist in polynomieller Zeit von einer DTM machbar.

Vermutung: HAMILTON  $\notin$  P, da man keinen substanziell besseren Algorithmus kennt, als alle Permutationen auszuprobieren.

Viele Probleme sind von der Art dass

- schwer ist, zu entscheiden, ob sie lösbar sind,
- leicht ist, zu entscheiden, ob eine Lösungsvorschlag eine Lösung ist.



Viele Probleme sind von der Art dass

- schwer ist, zu entscheiden, ob sie lösbar sind,
- leicht ist, zu entscheiden, ob eine Lösungsvorschlag eine Lösung ist.

#### Definition 5.7

Sei  $M$  eine DTM mit  $L(M) \subseteq \{w\#c \mid w \in \Sigma^*, c \in \Delta^*\}$ .

Viele Probleme sind von der Art dass

- schwer ist, zu entscheiden, ob sie lösbar sind,
- leicht ist, zu entscheiden, ob eine Lösungsvorschlag eine Lösung ist.

#### Definition 5.7

Sei  $M$  eine DTM mit  $L(M) \subseteq \{w\#c \mid w \in \Sigma^*, c \in \Delta^*\}$ .

- Falls  $w\#c \in L(M)$ , so heißt  $c$  **Zertifikat** für  $w$ .
- $M$  ist ein **polynomiell beschränkter Verifikator** für die Sprache  $\{w \in \Sigma^* \mid \exists c \in \Delta^*. w\#c \in L(M)\}$

Viele Probleme sind von der Art dass

- schwer ist, zu entscheiden, ob sie lösbar sind,
- leicht ist, zu entscheiden, ob eine Lösungsvorschlag eine Lösung ist.

#### Definition 5.7

Sei  $M$  eine DTM mit  $L(M) \subseteq \{w\#c \mid w \in \Sigma^*, c \in \Delta^*\}$ .

- Falls  $w\#c \in L(M)$ , so heißt  $c$  **Zertifikat** für  $w$ .

Viele Probleme sind von der Art dass

- schwer ist, zu entscheiden, ob sie lösbar sind,
- leicht ist, zu entscheiden, ob eine Lösungsvorschlag eine Lösung ist.

#### Definition 5.7

Sei  $M$  eine DTM mit  $L(M) \subseteq \{w\#c \mid w \in \Sigma^*, c \in \Delta^*\}$ .

- Falls  $w\#c \in L(M)$ , so heißt  $c$  **Zertifikat** für  $w$ .
- $M$  ist ein **polynomiell beschränkter Verifikator** für die Sprache  $\{w \in \Sigma^* \mid \exists c \in \Delta^*. w\#c \in L(M)\}$  falls es ein Polynom  $p$  gibt, so dass  $time_M(w\#c) \leq p(|w|)$ .

NB:

In Zeit  $p(n)$  kann  $M$  maximal die ersten  $p(n)$  Zeichen von  $c$  lesen.

Viele Probleme sind von der Art dass

- schwer ist, zu entscheiden, ob sie lösbar sind,
- leicht ist, zu entscheiden, ob eine Lösungsvorschlag eine Lösung ist.

#### Definition 5.7

Sei  $M$  eine DTM mit  $L(M) \subseteq \{w\#c \mid w \in \Sigma^*, c \in \Delta^*\}$ .

- Falls  $w\#c \in L(M)$ , so heißt  $c$  **Zertifikat** für  $w$ .
- $M$  ist ein **polynomiell beschränkter Verifikator** für die Sprache  $\{w \in \Sigma^* \mid \exists c \in \Delta^*. w\#c \in L(M)\}$  falls es ein Polynom  $p$  gibt, so dass  $\text{time}_M(w\#c) \leq p(|w|)$ .

NB:

In Zeit  $p(n)$  kann  $M$  maximal die ersten  $p(n)$  Zeichen von  $c$  lesen.  
Daher genügt für  $w$  ein Zertifikat der Länge  $\leq p(|w|)$ .

#### Beispiel 5.9 (RUCKSACK)

**Gegeben:** Zahlen  $a_1, \dots, a_n \in \mathbb{N}$  und  $c \in \mathbb{N}$ .

Viele Probleme sind von der Art dass

- schwer ist, zu entscheiden, ob sie lösbar sind,
- leicht ist, zu entscheiden, ob eine Lösungsvorschlag eine Lösung ist.

#### Definition 5.7

Sei  $M$  eine DTM mit  $L(M) \subseteq \{w\#c \mid w \in \Sigma^*, c \in \Delta^*\}$ .

- Falls  $w\#c \in L(M)$ , so heißt  $c$  **Zertifikat** für  $w$ .
- $M$  ist ein **polynomiell beschränkter Verifikator** für die Sprache  $\{w \in \Sigma^* \mid \exists c \in \Delta^*. w\#c \in L(M)\}$  falls es ein Polynom  $p$  gibt, so dass  $\text{time}_M(w\#c) \leq p(|w|)$ .

NB:

In Zeit  $p(n)$  kann  $M$  maximal die ersten  $p(n)$  Zeichen von  $c$  lesen.  
Daher genügt für  $w$  ein Zertifikat der Länge  $\leq p(|w|)$ .

#### Beispiel 5.8 (HAMILTON)

Zertifikat: Knotenpermutation. In polynomieller Zeit verifizierbar.

#### Beispiel 5.9 (RUCKSACK)

**Gegeben:** Zahlen  $a_1, \dots, a_n \in \mathbb{N}$  und  $c \in \mathbb{N}$ .

**Problem:** Gibt es  $R \subseteq \{1, \dots, n\}$  mit  $\sum_{i \in R} a_i = c$ ?

### Beispiel 5.9 (RUCKSACK)

Gegeben: Zahlen  $a_1, \dots, a_n \in \mathbb{N}$  und  $c \in \mathbb{N}$ .

Problem: Gibt es  $R \subseteq \{1, \dots, n\}$  mit  $\sum_{i \in R} a_i = c$ ?

$$\text{RUCKSACK} := \{ \text{bin}(a_1) \# \dots \# \text{bin}(a_n) \# \text{bin}(c) \mid \\ \exists R \subseteq \{1, \dots, n\}. \sum_{i \in R} a_i = c \}$$

### Beispiel 5.9 (RUCKSACK)

Gegeben: Zahlen  $a_1, \dots, a_n \in \mathbb{N}$  und  $c \in \mathbb{N}$ .

Problem: Gibt es  $R \subseteq \{1, \dots, n\}$  mit  $\sum_{i \in R} a_i = c$ ?

$$\text{RUCKSACK} := \{ \text{bin}(a_1) \# \dots \# \text{bin}(a_n) \# \text{bin}(c) \mid \\ \exists R \subseteq \{1, \dots, n\}. \sum_{i \in R} a_i = c \}$$

RUCKSACK  $\in$  NP:

Zertifikat: Indexmenge  $R$ . In polynomialer Zeit verifizierbar.

### Beispiel 5.9 (RUCKSACK)

Gegeben: Zahlen  $a_1, \dots, a_n \in \mathbb{N}$  und  $c \in \mathbb{N}$ .

Problem: Gibt es  $R \subseteq \{1, \dots, n\}$  mit  $\sum_{i \in R} a_i = c$ ?

$$\text{RUCKSACK} := \{ \text{bin}(a_1) \# \dots \# \text{bin}(a_n) \# \text{bin}(c) \mid \\ \exists R \subseteq \{1, \dots, n\}. \sum_{i \in R} a_i = c \}$$

RUCKSACK  $\in$  NP:

Zertifikat: Indexmenge  $R$ . In polynomialer Zeit verifizierbar.

Merke: Der Nachweis  $A \in$  NP ist meistens einfach.

### Satz 5.10

$A \in$  NP gdw es gibt polynomial beschränkten Verifikator für  $A$ .

### Satz 5.10

$A \in NP$  gdw es gibt polynomiell beschränkten Verifikator für  $A$ .

Beweis:

„ $\Rightarrow$ “:

Sei  $A \in NP$ . Dh es gibt NTM  $N$ , die  $A$  in Zeit  $p(n)$  akzeptiert.

### Satz 5.10

$A \in NP$  gdw es gibt polynomiell beschränkten Verifikator für  $A$ .

Beweis:

„ $\Rightarrow$ “:

Sei  $A \in NP$ . Dh es gibt NTM  $N$ , die  $A$  in Zeit  $p(n)$  akzeptiert.

Ein Zertifikat für  $w \in A$  ist die Folge der benutzten Transitionen

$\delta(q, a) \ni (q', a', d)$  einer akzeptierenden Berechnungsfolge von

$N[w]$  mit  $\leq p(n)$  Schritten.

Ein polynomiell beschränkter Verifikator für  $L(N)$ :

### Satz 5.10

$A \in NP$  gdw es gibt polynomiell beschränkten Verifikator für  $A$ .

Beweis:

„ $\Rightarrow$ “:

Sei  $A \in NP$ . Dh es gibt NTM  $N$ , die  $A$  in Zeit  $p(n)$  akzeptiert.

Ein Zertifikat für  $w \in A$  ist die Folge der benutzten Transitionen

$\delta(q, a) \ni (q', a', d)$  einer akzeptierenden Berechnungsfolge von

$N[w]$  mit  $\leq p(n)$  Schritten.

### Satz 5.10

$A \in NP$  gdw es gibt polynomiell beschränkten Verifikator für  $A$ .

Beweis:

„ $\Rightarrow$ “:

Sei  $A \in NP$ . Dh es gibt NTM  $N$ , die  $A$  in Zeit  $p(n)$  akzeptiert.

Ein Zertifikat für  $w \in A$  ist die Folge der benutzten Transitionen

$\delta(q, a) \ni (q', a', d)$  einer akzeptierenden Berechnungsfolge von

$N[w]$  mit  $\leq p(n)$  Schritten.

Ein polynomiell beschränkter Verifikator für  $L(N)$ :

- 1 Eingabe  $w\#c$
- 2 Simuliere  $N[w]$ , gesteuert durch die Transitionen in  $c$ .

### Satz 5.10

$A \in NP$  gdw es gibt polynomiell beschränkten Verifikator für  $A$ .

Beweis:

„ $\Rightarrow$ “:

Sei  $A \in NP$ . Dh es gibt NTM  $N$ , die  $A$  in Zeit  $p(n)$  akzeptiert. Ein Zertifikat für  $w \in A$  ist die Folge der benutzten Transitionen  $\delta(q, a) \ni (q', a', d)$  einer akzeptierenden Berechnungsfolge von  $N[w]$  mit  $\leq p(n)$  Schritten.

Ein polynomiell beschränkter Verifikator für  $L(N)$ :

- 1 Eingabe  $w\#c$
- 2 Simuliere  $N[w]$ , gesteuert durch die Transitionen in  $c$ .
- 3 Überprüfe dabei, ob die Transition in  $c$  jeweils zu  $N$  und zur augenblicklichen Konfiguration von  $N$  passt.

Beweis (Forts.):

„ $\Leftarrow$ “:

Sei  $M$  ein polynomiell (durch  $p$ ) beschränkter Verifikator für  $A$ . Wir bauen eine polynomiell beschränkte NTM  $N$  mit  $L(M) = A$ :

- 1 Eingabe  $w$ .
- 2 Schreibe hinter  $w$  zuerst  $\#$  und dann ein nichtdeterministisch gewähltes Wort  $c \in \Delta^*$ ,  $|c| = p(|w|)$ :

### Satz 5.10

$A \in NP$  gdw es gibt polynomiell beschränkten Verifikator für  $A$ .

Beweis:

„ $\Rightarrow$ “:

Sei  $A \in NP$ . Dh es gibt NTM  $N$ , die  $A$  in Zeit  $p(n)$  akzeptiert. Ein Zertifikat für  $w \in A$  ist die Folge der benutzten Transitionen  $\delta(q, a) \ni (q', a', d)$  einer akzeptierenden Berechnungsfolge von  $N[w]$  mit  $\leq p(n)$  Schritten.

Ein polynomiell beschränkter Verifikator für  $L(N)$ :

- 1 Eingabe  $w\#c$
- 2 Simuliere  $N[w]$ , gesteuert durch die Transitionen in  $c$ .
- 3 Überprüfe dabei, ob die Transition in  $c$  jeweils zu  $N$  und zur augenblicklichen Konfiguration von  $N$  passt.
- 4 Akzeptiere, falls  $c$  in einen Endzustand führt.

Beweis (Forts.):

„ $\Leftarrow$ “:

Sei  $M$  ein polynomiell (durch  $p$ ) beschränkter Verifikator für  $A$ . Wir bauen eine polynomiell beschränkte NTM  $N$  mit  $L(M) = A$ :

- 1 Eingabe  $w$ .
- 2 Schreibe hinter  $w$  zuerst  $\#$  und dann ein nichtdeterministisch gewähltes Wort  $c \in \Delta^*$ ,  $|c| = p(|w|)$ :  
**for**  $i := 1, \dots, p(|w|)$  **do**  
    schreibe ein Zeichen aus  $\Delta$  und gehe nach rechts

### Beweis (Forts.):

„ $\Leftarrow$ “:

Sei  $M$  ein polynomiell (durch  $p$ ) beschränkter Verifikator für  $A$ .

Wir bauen eine polynomiell beschränkte NTM  $N$  mit  $L(M) = A$ :

- ① Eingabe  $w$ .
- ② Schreibe hinter  $w$  zuerst  $\#$  und dann ein nichtdeterministisch gewähltes Wort  $c \in \Delta^*$ ,  $|c| = p(|w|)$ :  
  **for**  $i := 1, \dots, p(|w|)$  **do**  
    schreibe ein Zeichen aus  $\Delta$  und gehe nach rechts
- ③ Führe  $M$  aus (mit Eingabe  $w\#c$ ).

### Beweis (Forts.):

„ $\Leftarrow$ “:

Sei  $M$  ein polynomiell (durch  $p$ ) beschränkter Verifikator für  $A$ .

Wir bauen eine polynomiell beschränkte NTM  $N$  mit  $L(M) = A$ :

- ① Eingabe  $w$ .
- ② Schreibe hinter  $w$  zuerst  $\#$  und dann ein nichtdeterministisch gewähltes Wort  $c \in \Delta^*$ ,  $|c| = p(|w|)$ :  
  **for**  $i := 1, \dots, p(|w|)$  **do**  
    schreibe ein Zeichen aus  $\Delta$  und gehe nach rechts
- ③ Führe  $M$  aus (mit Eingabe  $w\#c$ ).

Nach Annahme gilt  $time_M(w\#c) \leq p(|w|)$ .

Damit braucht  $N[w] O(p(|w|))$  Schritte. □

### Beweis (Forts.):

„ $\Leftarrow$ “:

Sei  $M$  ein polynomiell (durch  $p$ ) beschränkter Verifikator für  $A$ .

Wir bauen eine polynomiell beschränkte NTM  $N$  mit  $L(M) = A$ :

- ① Eingabe  $w$ .
- ② Schreibe hinter  $w$  zuerst  $\#$  und dann ein nichtdeterministisch gewähltes Wort  $c \in \Delta^*$ ,  $|c| = p(|w|)$ :  
  **for**  $i := 1, \dots, p(|w|)$  **do**  
    schreibe ein Zeichen aus  $\Delta$  und gehe nach rechts
- ③ Führe  $M$  aus (mit Eingabe  $w\#c$ ).

Nach Annahme gilt  $time_M(w\#c) \leq p(|w|)$ .

### Fazit:

- $P$  sind die Sprachen, bei denen  $w \in L$  schnell **entschieden** werden kann.

**Fazit:**

- P sind die Sprachen, bei denen  $w \in L$  schnell **entschieden** werden kann.
- NP sind die Sprachen, bei denen ein Zertifikat für  $w \in L$  schnell **verifiziert/überprüft** werden kann.

**Fazit:**

- P sind die Sprachen, bei denen  $w \in L$  schnell **entschieden** werden kann.
- NP sind die Sprachen, bei denen ein Zertifikat für  $w \in L$  schnell **verifiziert/überprüft** werden kann.

Intuition:

Es ist leichter, eine Lösung zu verifizieren als zu finden.

Aber:

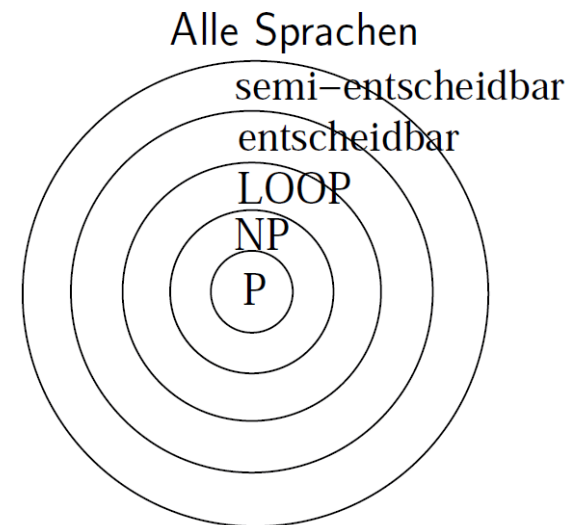
Noch wurde von keiner Sprache bewiesen, dass sie in  $NP \setminus P$  liegt.

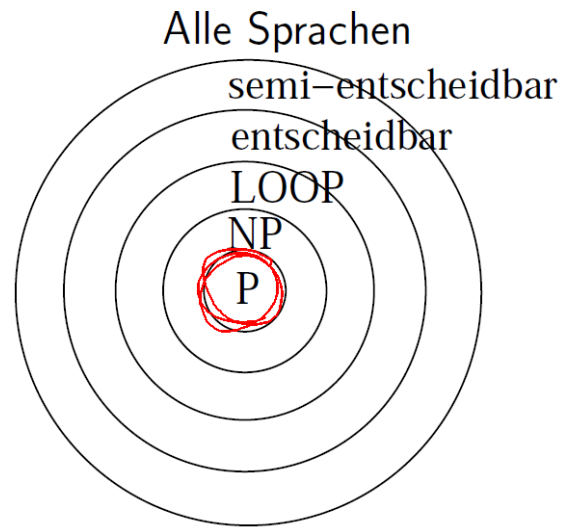
**Fazit:**

- P sind die Sprachen, bei denen  $w \in L$  schnell **entschieden** werden kann.
- NP sind die Sprachen, bei denen ein Zertifikat für  $w \in L$  schnell **verifiziert/überprüft** werden kann.

Intuition:

Es ist leichter, eine Lösung zu verifizieren als zu finden.

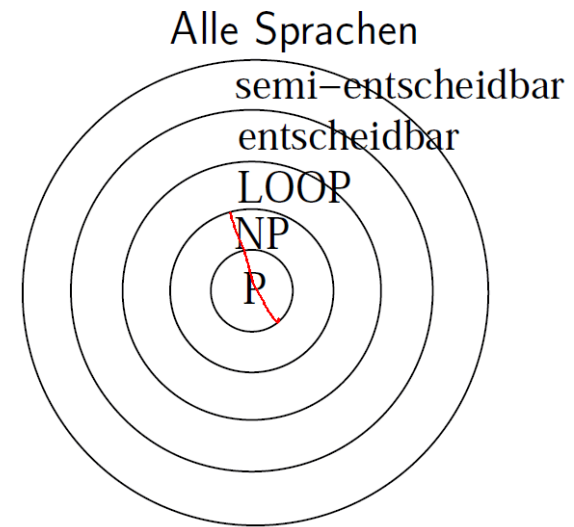




Nur die P-NP-Grenze ist offen.

**Satz 5.11**

Ist  $A \in TIME(f)$  und ist  $f$  LOOP-berechenbar, dann ist  $A$  LOOP-entscheidbar, dh  $\chi_A$  ist LOOP-berechenbar.



Nur die P-NP-Grenze ist offen.

**Satz 5.11**

Ist  $A \in TIME(f)$  und ist  $f$  LOOP-berechenbar, dann ist  $A$  LOOP-entscheidbar, dh  $\chi_A$  ist LOOP-berechenbar.

**Beweis:**

Sei  $M$  eine DTM mit  $L(M) = A$  und  $time_M(w) \leq f(|w|)$ .



### Satz 5.11

Ist  $A \in TIME(f)$  und ist  $f$  LOOP-berechenbar, dann ist  $A$  LOOP-entscheidbar, dh  $\chi_A$  ist LOOP-berechenbar.

#### Beweis:

Sei  $M$  eine DTM mit  $L(M) = A$  und  $time_M(w) \leq f(|w|)$ .

$M \mapsto GOTO \mapsto WHILE$ :

### Satz 5.11

Ist  $A \in TIME(f)$  und ist  $f$  LOOP-berechenbar, dann ist  $A$  LOOP-entscheidbar, dh  $\chi_A$  ist LOOP-berechenbar.

#### Beweis:

Sei  $M$  eine DTM mit  $L(M) = A$  und  $time_M(w) \leq f(|w|)$ .

$M \mapsto GOTO \mapsto WHILE$ :

```
pc := 1;
WHILE pc ≠ 0 DO
  IF pc = 1 THEN P1 ELSE
  ...
  IF pc = k THEN Pk ELSE pc := 0
END
```

wobei jeder Schleifendurchlauf einen Schritt von  $M$  simuliert.

### Satz 5.11

Ist  $A \in TIME(f)$  und ist  $f$  LOOP-berechenbar, dann ist  $A$  LOOP-entscheidbar, dh  $\chi_A$  ist LOOP-berechenbar.

#### Beweis:

Sei  $M$  eine DTM mit  $L(M) = A$  und  $time_M(w) \leq f(|w|)$ .

$M \mapsto GOTO \mapsto WHILE$ :

```
pc := 1;
WHILE pc ≠ 0 DO
  IF pc = 1 THEN P1 ELSE
  ...
  IF pc = k THEN Pk ELSE pc := 0
END
```

### Satz 5.11

Ist  $A \in TIME(f)$  und ist  $f$  LOOP-berechenbar, dann ist  $A$  LOOP-entscheidbar, dh  $\chi_A$  ist LOOP-berechenbar.

#### Beweis:

Sei  $M$  eine DTM mit  $L(M) = A$  und  $time_M(w) \leq f(|w|)$ .

$M \mapsto GOTO \mapsto WHILE$ :

```
pc := 1;
WHILE pc ≠ 0 DO
  IF pc = 1 THEN P1 ELSE
  ...
  IF pc = k THEN Pk ELSE pc := 0
END
```

wobei jeder Schleifendurchlauf einen Schritt von  $M$  simuliert.

Es kann maximal  $f(|w|)$  viele Schleifendurchläufe geben.

### Satz 5.11

Ist  $A \in \text{TIME}(f)$  und ist  $f$  LOOP-berechenbar, dann ist  $A$  LOOP-entscheidbar, dh  $\chi_A$  ist LOOP-berechenbar.

#### Beweis:

Sei  $M$  eine DTM mit  $L(M) = A$  und  $\text{time}_M(w) \leq f(|w|)$ .

$M \mapsto \text{GOTO} \mapsto \text{WHILE}$ :

```
pc := 1;
WHILE pc ≠ 0 DO
  IF pc = 1 THEN P1 ELSE
    ...
  IF pc = k THEN Pk ELSE pc := 0
END
```

wobei jeder Schleifendurchlauf einen Schritt von  $M$  simuliert.

Es kann maximal  $f(|w|)$  viele Schleifendurchläufe geben.

Ersetze **WHILE**  $pc \neq 0$  **DO** durch  $i := f(|w|)$ ; **LOOP**  $i$  **DO**.

Da  $+$  und  $*$  LOOP-berechenbar sind, folgt:

### Korollar 5.12

Alle Sprachen in  $P$  sind LOOP-entscheidbar.

### Satz 5.11

Ist  $A \in \text{TIME}(f)$  und ist  $f$  LOOP-berechenbar, dann ist  $A$  LOOP-entscheidbar, dh  $\chi_A$  ist LOOP-berechenbar.

#### Beweis:

Sei  $M$  eine DTM mit  $L(M) = A$  und  $\text{time}_M(w) \leq f(|w|)$ .

$M \mapsto \text{GOTO} \mapsto \text{WHILE}$ :

```
pc := 1;
WHILE pc ≠ 0 DO
  IF pc = 1 THEN P1 ELSE
    ...
  IF pc = k THEN Pk ELSE pc := 0
END
```

wobei jeder Schleifendurchlauf einen Schritt von  $M$  simuliert.

Es kann maximal  $f(|w|)$  viele Schleifendurchläufe geben.

Ersetze **WHILE**  $pc \neq 0$  **DO** durch  $i := f(|w|)$ ; **LOOP**  $i$  **DO**.

Berechnung wie vorher, aber evtl mit zusätzlichen Durchläufen mit  $pc = 0$ . □

Da  $+$  und  $*$  LOOP-berechenbar sind, folgt:

### Korollar 5.12

Alle Sprachen in  $P$  sind LOOP-entscheidbar.

Da sich jede polynomiell zeitbeschränkte NTM (Zeit  $O(p(n))$ )

Da + und \* LOOP-berechenbar sind, folgt:

#### Korollar 5.12

*Alle Sprachen in P sind LOOP-entscheidbar.*

Da sich jede polynomiell zeitbeschränkte NTM (Zeit  $O(p(n))$ ) in eine exponentiell zeitbeschränkte DTM (Zeit  $O(c^{p(n)})$ )

Da + und \* LOOP-berechenbar sind, folgt:

#### Korollar 5.12

*Alle Sprachen in P sind LOOP-entscheidbar.*

Da sich jede polynomiell zeitbeschränkte NTM (Zeit  $O(p(n))$ ) in eine exponentiell zeitbeschränkte DTM (Zeit  $O(c^{p(n)})$ ) umwandeln lässt und  $c^{p(n)}$  LOOP-berechenbar ist, folgt:

Da + und \* LOOP-berechenbar sind, folgt:

#### Korollar 5.12

*Alle Sprachen in P sind LOOP-entscheidbar.*

Da sich jede polynomiell zeitbeschränkte NTM (Zeit  $O(p(n))$ ) in eine exponentiell zeitbeschränkte DTM (Zeit  $O(c^{p(n)})$ ) umwandeln lässt und  $c^{p(n)}$  LOOP-berechenbar ist, folgt:

#### Korollar 5.13

*Alle Sprachen in NP sind LOOP-entscheidbar.*

Da + und \* LOOP-berechenbar sind, folgt:

#### Korollar 5.12

*Alle Sprachen in P sind LOOP-entscheidbar.*

Da sich jede polynomiell zeitbeschränkte NTM (Zeit  $O(p(n))$ ) in eine exponentiell zeitbeschränkte DTM (Zeit  $O(c^{p(n)})$ ) umwandeln lässt und  $c^{p(n)}$  LOOP-berechenbar ist, folgt:

#### Korollar 5.13

*Alle Sprachen in NP sind LOOP-entscheidbar.*

### 5.3 NP-Vollständigkeit

### 5.3 NP-Vollständigkeit

- 1 Polynomielle Reduzierbarkeit  $\leq_p$
- 2 NP-vollständige Probleme = härteste Probleme in NP,

### 5.3 NP-Vollständigkeit

- 1 Polynomielle Reduzierbarkeit  $\leq_p$

### 5.3 NP-Vollständigkeit

- 1 Polynomielle Reduzierbarkeit  $\leq_p$
- 2 NP-vollständige Probleme = härteste Probleme in NP, alle anderen Probleme in NP darauf polynomiell reduzierbar
- 3 Satz: SAT ist NP-vollständig

### 5.3 NP-Vollständigkeit

- 1 Polynomielle Reduzierbarkeit  $\leq_p$
- 2 NP-vollständige Probleme = härteste Probleme in NP, alle anderen Probleme in NP darauf polynomiell reduzierbar
- 3 Satz: SAT ist NP-vollständig

#### Definition 5.14

Sei  $A \subseteq \Sigma^*$  und  $B \subseteq \Gamma^*$ .

Dann heißt  $A$  **polynomiell reduzierbar** auf  $B$ ,  $A \leq_p B$ ,

#### Definition 5.14

Sei  $A \subseteq \Sigma^*$  und  $B \subseteq \Gamma^*$ .

#### Definition 5.14

Sei  $A \subseteq \Sigma^*$  und  $B \subseteq \Gamma^*$ .

Dann heißt  $A$  **polynomiell reduzierbar** auf  $B$ ,  $A \leq_p B$ ,  
gdw es eine totale und von einer DTM in polynomieller Zeit  
berechenbare Funktion  $f : \Sigma^* \rightarrow \Gamma^*$  gibt,

#### Definition 5.14

Sei  $A \subseteq \Sigma^*$  und  $B \subseteq \Gamma^*$ .

Dann heißt  $A$  **polynomiell reduzierbar** auf  $B$ ,  $A \leq_p B$ ,  
gdw es eine totale und von einer DTM in polynomieller Zeit  
berechenbare Funktion  $f : \Sigma^* \rightarrow \Gamma^*$  gibt, so dass für alle  $w \in \Sigma^*$

$$w \in A \iff f(w) \in B$$

#### Lemma 5.16

*Die Klassen  $P$  und  $NP$  sind unter polynomieller Reduzierbarkeit  
nach unten abgeschlossen:*

#### Definition 5.14

Sei  $A \subseteq \Sigma^*$  und  $B \subseteq \Gamma^*$ .

Dann heißt  $A$  **polynomiell reduzierbar** auf  $B$ ,  $A \leq_p B$ ,  
gdw es eine totale und von einer DTM in polynomieller Zeit  
berechenbare Funktion  $f : \Sigma^* \rightarrow \Gamma^*$  gibt, so dass für alle  $w \in \Sigma^*$

$$w \in A \iff f(w) \in B$$

Da  $q(p(n))$  ein Polynom ist falls  $p$  und  $q$  Polynome sind:

#### Lemma 5.15

*Die Relation  $\leq_p$  ist transitiv.*

#### Definition 5.14

Sei  $A \subseteq \Sigma^*$  und  $B \subseteq \Gamma^*$ .

Dann heißt  $A$  **polynomiell reduzierbar** auf  $B$ ,  $A \leq_p B$ ,  
gdw es eine totale und von einer DTM in polynomieller Zeit  
berechenbare Funktion  $f : \Sigma^* \rightarrow \Gamma^*$  gibt, so dass für alle  $w \in \Sigma^*$

$$w \in A \iff f(w) \in B$$

Da  $q(p(n))$  ein Polynom ist falls  $p$  und  $q$  Polynome sind:

#### Lemma 5.15

*Die Relation  $\leq_p$  ist transitiv.*

### Lemma 5.16

Die Klassen  $P$  und  $NP$  sind unter polynomieller Reduzierbarkeit nach unten abgeschlossen:

$$A \leq_p B \in P/NP \implies A \in P/NP$$

Beweis:

- Sei  $A \leq_p B$  mittels  $f$ , die von DTM  $M_f$  berechnet wird. Polynom  $p$  begrenzt Rechenzeit von  $M_f$ .

### Lemma 5.16

Die Klassen  $P$  und  $NP$  sind unter polynomieller Reduzierbarkeit nach unten abgeschlossen:

$$A \leq_p B \in P/NP \implies A \in P/NP$$

Beweis:

- Sei  $A \leq_p B$  mittels  $f$ , die von DTM  $M_f$  berechnet wird.

### Lemma 5.16

Die Klassen  $P$  und  $NP$  sind unter polynomieller Reduzierbarkeit nach unten abgeschlossen:

$$A \leq_p B \in P/NP \implies A \in P/NP$$

Beweis:

- Sei  $A \leq_p B$  mittels  $f$ , die von DTM  $M_f$  berechnet wird. Polynom  $p$  begrenzt Rechenzeit von  $M_f$ .
- Sei  $B \in P$  mittels DTM  $M$ .

### Lemma 5.16

Die Klassen  $P$  und  $NP$  sind unter polynomieller Reduzierbarkeit nach unten abgeschlossen:

$$A \leq_p B \in P/NP \implies A \in P/NP$$

Beweis:

- Sei  $A \leq_p B$  mittels  $f$ , die von DTM  $M_f$  berechnet wird. Polynom  $p$  begrenzt Rechenzeit von  $M_f$ .
- Sei  $B \in P$  mittels DTM  $M$ . Polynom  $q$  begrenzt Rechenzeit von  $M$ .

Damit ist  $M_f; M$  polynomiell zeitbeschränkt in  $|w|$ :

### Lemma 5.16

Die Klassen  $P$  und  $NP$  sind unter polynomieller Reduzierbarkeit nach unten abgeschlossen:

$$A \leq_p B \in P/NP \implies A \in P/NP$$

Beweis:

- Sei  $A \leq_p B$  mittels  $f$ , die von DTM  $M_f$  berechnet wird. Polynom  $p$  begrenzt Rechenzeit von  $M_f$ .
- Sei  $B \in P$  mittels DTM  $M$ . Polynom  $q$  begrenzt Rechenzeit von  $M$ .

Damit ist  $M_f; M$  polynomiell zeitbeschränkt in  $|w|$ :

- $M_f[w]$  macht  $\leq p(|w|)$  Schritte.
- Ausgabe  $f(w)$  von  $M_f$  hat Länge  $\leq |w| + p(|w|)$ .
- $M$  macht  $\leq q(|f(w)|) \leq q(|w| + p(|w|))$  Schritte

### Lemma 5.16

Die Klassen  $P$  und  $NP$  sind unter polynomieller Reduzierbarkeit nach unten abgeschlossen:

$$A \leq_p B \in P/NP \implies A \in P/NP$$

Beweis:

- Sei  $A \leq_p B$  mittels  $f$ , die von DTM  $M_f$  berechnet wird. Polynom  $p$  begrenzt Rechenzeit von  $M_f$ .
- Sei  $B \in P$  mittels DTM  $M$ . Polynom  $q$  begrenzt Rechenzeit von  $M$ .

Damit ist  $M_f; M$  polynomiell zeitbeschränkt in  $|w|$ :

- $M_f[w]$  macht  $\leq p(|w|)$  Schritte.
- Ausgabe  $f(w)$  von  $M_f$  hat Länge  $\leq |w| + p(|w|)$ .

### Lemma 5.16

Die Klassen  $P$  und  $NP$  sind unter polynomieller Reduzierbarkeit nach unten abgeschlossen:

$$A \leq_p B \in P/NP \implies A \in P/NP$$

Beweis:

- Sei  $A \leq_p B$  mittels  $f$ , die von DTM  $M_f$  berechnet wird. Polynom  $p$  begrenzt Rechenzeit von  $M_f$ .
- Sei  $B \in P$  mittels DTM  $M$ . Polynom  $q$  begrenzt Rechenzeit von  $M$ .

Damit ist  $M_f; M$  polynomiell zeitbeschränkt in  $|w|$ :

- $M_f[w]$  macht  $\leq p(|w|)$  Schritte.
- Ausgabe  $f(w)$  von  $M_f$  hat Länge  $\leq |w| + p(|w|)$ .
- $M$  macht  $\leq q(|f(w)|) \leq q(|w| + p(|w|))$  Schritte ( $q$  monoton)

Daher macht  $(M_f; M)[w]$  maximal  $p(|w|) + q(|w| + p(|w|))$  Schritte, ein Polynom in  $|w|$ .

Analog:  $A \leq_p B \in NP \implies A \in NP$

□