

# Randomisierte Algorithmen

# Randomisierte Algorithmen

Ein randomisierter Algorithmus  $A$  wählt in jedem Schritt eine von mehreren Optionen gleichwahrscheinlich aus.

- (a) Die Wahrscheinlichkeit  $\text{pr}[B]$  einer Berechnung  $B$  von  $A$  ist das Produkt der Wahrscheinlichkeiten über alle Schritte der Berechnung.
- (b) Die Akzeptanz-Wahrscheinlichkeit für eine Eingabe  $x$  ist  
die Summe der Wahrscheinlichkeiten akzeptierender Berechnungen für  $x$ .
- (c) Die von  $A$  akzeptierte Sprache  $L(M)$  ist

$$L(A) = \{x \in \{0, 1\}^* : A \text{ akzeptiert } x \text{ mit } W\text{-keit mindestens } \frac{1}{2}\}.$$

# Randomisierte Algorithmen

Ein randomisierter Algorithmus  $A$  wählt in jedem Schritt eine von mehreren Optionen gleichwahrscheinlich aus.

- (a) Die Wahrscheinlichkeit  $\text{pr}[B]$  einer Berechnung  $B$  von  $A$  ist das Produkt der Wahrscheinlichkeiten über alle Schritte der Berechnung.
- (b) Die Akzeptanz-Wahrscheinlichkeit für eine Eingabe  $x$  ist  
die Summe der Wahrscheinlichkeiten akzeptierender Berechnungen für  $x$ .
- (c) Die von  $A$  akzeptierte Sprache  $L(A)$  ist

$$L(A) = \{x \in \{0, 1\}^* : A \text{ akzeptiert } x \text{ mit W-keit mindestens } \frac{1}{2}\}.$$

Für  $0 \leq p < q \leq 1$  heißt  $A$  ein  $(p, q)$ -Algorithmus, wenn  $A$  jede Eingabe mit Wahrscheinlichkeit höchstens  $p$  oder mindestens  $q$  akzeptiert.

# Randomisierte Algorithmen: Beispiele

- Randomisierte Algorithmen
  - ▶ Quicksort mit zufälliger Pivot-Wahl
  - ▶ Randomisierung für geometrische Probleme
    - ★ Volumenmessung von konvexen Mengen
    - ★ das „Closest-Point-Problem“
  - ▶ Randomisierte Primzahltests:  
Die Algorithmen von Rabin und Solovay-Strassen
  - ▶ Anwendungen im Entwurf von Online-Algorithmen:  
Randomisierung gegen die Zukunft
- Randomisierte Datenstrukturen:  
Bloom-Filter, randomisierte Skip-Listen, Treaps, universelles Hashing
- Anwendungen in der Kryptographie:  
Zufällige Wahl eines öffentlichen Schlüssels
- Randomisierung in Beweissystemen:  
Interaktive Beweise und probabilistically-checkable-proofs

- (a) Für die Funktion  $t : \mathbb{N} \rightarrow \mathbb{N}$  und alle Paare  $(p, q)$  mit  $0 \leq p < q \leq 1$  ist

$$\text{RTIME}_{p,q}(t) := \{ L \subseteq \{0, 1\}^* \quad : \quad \text{es gibt einen } (p, q)\text{-Algorithmus } A \\ \text{mit } L(A) = L \text{ und } t_A = \mathcal{O}(t) \}$$

die Komplexitätsklasse aller in Zeit  $\mathcal{O}(t)$  berechenbaren Sprachen.

- (b) Die Klasse  $\text{RTIME}_{p,q}^*(t)$  wird analog definiert, allerdings wird die Worst-Case-Laufzeit durch die erwartete Laufzeit ersetzt.

- (c) Die Komplexitätsklasse

$$\text{BPP} := \bigcup_{k \in \mathbb{N}} \text{RTIME}_{\frac{1}{3}, \frac{2}{3}}(n^k)$$

(*Bounded-Error-Probabilistic-Polynomial-Time*) ist die Klasse aller mit beschränktem Fehler, in polynomieller Zeit akzeptierbaren Sprachen.

Weitere fundamentale Klassen:

- 1  $\text{WeakBPP} := \bigcup_{k \in \mathbb{N}} \text{RTIME}_{\frac{1}{2}-n^{-k}, \frac{1}{2}+n^{-k}}(n^k)$  und  
 $\text{StrongBPP} := \bigcap_{r \in \mathbb{N}} \bigcup_{k \in \mathbb{N}} \text{RTIME}_{2^{-nr}, 1-2^{-nr}}(n^k)$
- 2  $\text{ZPP} := \bigcup_{k \in \mathbb{N}} \text{RTIME}_{0,1}^*(n^k)$  (*Zero-Error-Probabilistic-Polynomial-Time*) die Klasse aller fehlerfrei durch randomisierte Algorithmen in polynomieller **erwarteter** Laufzeit akzeptierbaren Sprachen,
- 3  $\text{RP} := \bigcup_{k \in \mathbb{N}} \text{RTIME}_{0, \frac{1}{2}}(n^k)$  (*Randomized-Polynomial-Time*) die Klasse aller durch randomisierte Algorithmen  $A$  in polynomieller Zeit akzeptierbaren Sprachen, wobei  $A$  beim Verwerfen fehlerfrei sein muss,
- 4  $\text{coRP} := \bigcup_{k \in \mathbb{N}} \text{RTIME}_{\frac{1}{2}, 1}(n^k)$  (*co-Randomized-Polynomial-Time*) die Klasse aller durch randomisierte Algorithmen  $A$  in polynomieller Zeit akzeptierbaren Sprachen, wobei  $A$  beim Akzeptieren fehlerfrei sein muss,
- 5  $\text{PP} := \bigcup_{k \in \mathbb{N}} \text{RTIME}_{\frac{1}{2}, \frac{1}{2}}(n^k)$  (*Probabilistic-Polynomial-Time*) die Klasse aller durch randomisierte Algorithmen in polynomieller Zeit akzeptierbaren Sprachen, wobei die Akzeptanzwahrscheinlichkeit für keine Eingabe genau  $\frac{1}{2}$  betragen darf.

Die folgenden Inklusionen gelten:

- (a)  $P \subseteq ZPP = RP \cap coRP \subseteq RP \subseteq NP \subseteq PP \subseteq PSPACE \subseteq EXP$  und  
 $coRP \subseteq coNP$ .
- (b)  $RP, coRP \subseteq BPP \subseteq PP$ .

# Bounded Error Probabilistic Polynomial Time

(a) **Amplifizierung:**  $\text{WeakBPP} = \text{BPP} = \text{StrongBPP}$ .

Eine schwach von  $\frac{1}{2}$  weg-getrennte Fehlerwahrscheinlichkeit ist ausreichend, um „fast-fehlerfrei“ zu rechnen.

(b)  $\text{BPP} \subset \text{P/poly}$

Randomisierte Algorithmen lassen sich effizient derandomisieren, wenn **nicht-uniforme** Algorithmen erlaubt sind.

(c)  $\text{BPP} \subseteq \Sigma_2^P \cap \Pi_2^P$

Sprachen mit effizienten randomisierten Algorithmen liegen in der polynomiellen Hierarchie.

(a) **Amplifizierung:**  $\text{WeakBPP} = \text{BPP} = \text{StrongBPP}$ .

Eine schwach von  $\frac{1}{2}$  weg-getrennte Fehlerwahrscheinlichkeit ist ausreichend, um „fast-fehlerfrei“ zu rechnen.

(b)  $\text{BPP} \subset P/\text{poly}$

Randomisierte Algorithmen lassen sich effizient derandomisieren, wenn **nicht-uniforme** Algorithmen erlaubt sind.

(c)  $\text{BPP} \subseteq \Sigma_2^P \cap \Pi_2^P$

Sprachen mit effizienten randomisierten Algorithmen liegen in der polynomiellen Hierarchie.

Später: Wenn es eine Sprache  $L \in E$  gibt, so dass  $L$  nur durch Schaltkreise der Größe  $2^{\Omega(n)}$  berechenbar ist, dann folgt

$$P = \text{BPP}.$$

Randomisierung in der Kryptographie, im Entwurf von Online-Algorithmen oder im Entwurf von Beweissystemen ist hingegen nicht durch Determinismus ersetzbar.

# Fehlerreduktion

Es gelte  $L = L(A)$  für einen randomisierten Algorithmus  $A$  mit

- Laufzeit  $t_A(n) = \mathcal{O}(n^k)$  und
- Schwellenwerten  $\frac{1}{2} - n^{-k}, \frac{1}{2} + n^{-k}$  für die Akzeptanzwahrscheinlichkeiten.

Es gelte  $L = L(A)$  für einen randomisierten Algorithmus  $A$  mit

- Laufzeit  $t_A(n) = \mathcal{O}(n^k)$  und
- Schwellenwerten  $\frac{1}{2} - n^{-k}, \frac{1}{2} + n^{-k}$  für die Akzeptanzwahrscheinlichkeiten.

Ein „starker“ Algorithmus  $A^*$  für  $L$ :

- $A^*$  simuliert  $A$  auf Eingabe  $w$  genau  $\ell$ -mal
- und akzeptiert  $x$  wenn  $A$  mindestens  $\lceil \frac{\ell}{2} \rceil$ -mal akzeptiert.

Es gelte  $L = L(A)$  für einen randomisierten Algorithmus  $A$  mit

- Laufzeit  $t_A(n) = \mathcal{O}(n^k)$  und
- Schwellenwerten  $\frac{1}{2} - n^{-k}, \frac{1}{2} + n^{-k}$  für die Akzeptanzwahrscheinlichkeiten.

Ein „starker“ Algorithmus  $A^*$  für  $L$ :

- $A^*$  simuliert  $A$  auf Eingabe  $w$  genau  $\ell$ -mal
- und akzeptiert  $x$  wenn  $A$  mindestens  $\lceil \frac{\ell}{2} \rceil$ -mal akzeptiert.

Sei  $X_i \in \{0, 1\}$  die Zufallsvariable mit  $X_i = 1$  genau dann, wenn  $w$  in der  $i$ ten Simulation akzeptiert wird. Für  $\mu = \sum_{i=1}^{\ell} \mathbb{E}[X_i]$  folgt

$$\text{pr}\left[\left|\sum_{i=1}^{\ell} X_i - \mu\right| \geq \beta \cdot \mu\right] \leq \exp(-\Omega(\beta^2 \cdot \mu))$$

mit der Chernoff-Schranke.

**Fall 1:**  $x \in L$ . Dann ist  $\mu \geq \ell(\frac{1}{2} + n^{-k})$  und  $1 - \frac{\ell}{2\mu} \geq \frac{\ell}{\mu} \cdot n^{-k} \geq n^{-k}$  folgt.

- Setze  $\beta := 1 - \frac{\ell}{2\mu}$ , und es ist  $\mu \cdot (1 - \beta) = \frac{\ell}{2}$ .
- Des weiteren ist  $\beta \geq n^{-k}$ .

**Fall 1:**  $x \in L$ . Dann ist  $\mu \geq \ell(\frac{1}{2} + n^{-k})$  und  $1 - \frac{\ell}{2\mu} \geq \frac{\ell}{\mu} \cdot n^{-k} \geq n^{-k}$  folgt.

- Setze  $\beta := 1 - \frac{\ell}{2\mu}$ , und es ist  $\mu \cdot (1 - \beta) = \frac{\ell}{2}$ .
- Des Weiteren ist  $\beta \geq n^{-k}$ .

Wenn  $A^*$  verwirft, dann sind höchstens  $\frac{\ell}{2}$  Berechnungen akzeptierend  $\implies$

$$\text{pr}[A^* \text{ verwirft } x] \leq \text{pr}\left[\sum_{i=1}^{\ell} X_i \leq \frac{\ell}{2}\right] =$$

**Fall 1:**  $x \in L$ . Dann ist  $\mu \geq \ell(\frac{1}{2} + n^{-k})$  und  $1 - \frac{\ell}{2\mu} \geq \frac{\ell}{\mu} \cdot n^{-k} \geq n^{-k}$  folgt.

- Setze  $\beta := 1 - \frac{\ell}{2\mu}$ , und es ist  $\mu \cdot (1 - \beta) = \frac{\ell}{2}$ .
- Des weiteren ist  $\beta \geq n^{-k}$ .

Wenn  $A^*$  verwirft, dann sind höchstens  $\frac{\ell}{2}$  Berechnungen akzeptierend  $\implies$

$$\begin{aligned} \text{pr}[A^* \text{ verwirft } x] &\leq \text{pr}\left[\sum_{i=1}^{\ell} X_i \leq \frac{\ell}{2}\right] = \text{pr}\left[\sum_{i=1}^{\ell} X_i \leq \mu \cdot (1 - \beta)\right] \\ &\leq \end{aligned}$$

**Fall 1:**  $x \in L$ . Dann ist  $\mu \geq \ell(\frac{1}{2} + n^{-k})$  und  $1 - \frac{\ell}{2\mu} \geq \frac{\ell}{\mu} \cdot n^{-k} \geq n^{-k}$  folgt.

- Setze  $\beta := 1 - \frac{\ell}{2\mu}$ , und es ist  $\mu \cdot (1 - \beta) = \frac{\ell}{2}$ .
- Des weiteren ist  $\beta \geq n^{-k}$ .

Wenn  $A^*$  verwirft, dann sind höchstens  $\frac{\ell}{2}$  Berechnungen akzeptierend  $\implies$

$$\begin{aligned} \text{pr}[A^* \text{ verwirft } x] &\leq \text{pr}\left[\sum_{i=1}^{\ell} X_i \leq \frac{\ell}{2}\right] = \text{pr}\left[\sum_{i=1}^{\ell} X_i \leq \mu \cdot (1 - \beta)\right] \\ &\leq \exp(-\Omega(\beta^2 \cdot \mu)) \leq \end{aligned}$$

**Fall 1:**  $x \in L$ . Dann ist  $\mu \geq \ell(\frac{1}{2} + n^{-k})$  und  $1 - \frac{\ell}{2\mu} \geq \frac{\ell}{\mu} \cdot n^{-k} \geq n^{-k}$  folgt.

- Setze  $\beta := 1 - \frac{\ell}{2\mu}$ , und es ist  $\mu \cdot (1 - \beta) = \frac{\ell}{2}$ .
- Des weiteren ist  $\beta \geq n^{-k}$ .

Wenn  $A^*$  verwirft, dann sind höchstens  $\frac{\ell}{2}$  Berechnungen akzeptierend  $\implies$

$$\begin{aligned} \text{pr}[A^* \text{ verwirft } x] &\leq \text{pr}\left[\sum_{i=1}^{\ell} X_i \leq \frac{\ell}{2}\right] = \text{pr}\left[\sum_{i=1}^{\ell} X_i \leq \mu \cdot (1 - \beta)\right] \\ &\leq \exp(-\Omega(\beta^2 \cdot \mu)) \leq \exp(-\Omega(\frac{\ell}{n^{2k}})), \end{aligned}$$

denn  $\beta^2 \cdot \mu \geq n^{-2k} \cdot \frac{\ell}{2}$  folgt aus  $\beta \geq n^{-k}$  und  $\mu \geq \frac{\ell}{2}$

**Fall 1:**  $x \in L$ . Dann ist  $\mu \geq \ell(\frac{1}{2} + n^{-k})$  und  $1 - \frac{\ell}{2\mu} \geq \frac{\ell}{\mu} \cdot n^{-k} \geq n^{-k}$  folgt.

- Setze  $\beta := 1 - \frac{\ell}{2\mu}$ , und es ist  $\mu \cdot (1 - \beta) = \frac{\ell}{2}$ .
- Des weiteren ist  $\beta \geq n^{-k}$ .

Wenn  $A^*$  verwirft, dann sind höchstens  $\frac{\ell}{2}$  Berechnungen akzeptierend  $\implies$

$$\begin{aligned} \text{pr}[A^* \text{ verwirft } x] &\leq \text{pr}\left[\sum_{i=1}^{\ell} X_i \leq \frac{\ell}{2}\right] = \text{pr}\left[\sum_{i=1}^{\ell} X_i \leq \mu \cdot (1 - \beta)\right] \\ &\leq \exp(-\Omega(\beta^2 \cdot \mu)) \leq \exp(-\Omega(\frac{\ell}{n^{2k}})), \end{aligned}$$

denn  $\beta^2 \cdot \mu \geq n^{-2k} \cdot \frac{\ell}{2}$  folgt aus  $\beta \geq n^{-k}$  und  $\mu \geq \frac{\ell}{2} \implies$  Für  $\ell = \Omega(n^{2k+r})$  wird  $A^*$  die Eingabe  $x$  mit Wahrscheinlichkeit höchstens  $2^{-n^r}$  verworfen.

**Fall 1:**  $x \in L$ . Dann ist  $\mu \geq \ell(\frac{1}{2} + n^{-k})$  und  $1 - \frac{\ell}{2\mu} \geq \frac{\ell}{\mu} \cdot n^{-k} \geq n^{-k}$  folgt.

- Setze  $\beta := 1 - \frac{\ell}{2\mu}$ , und es ist  $\mu \cdot (1 - \beta) = \frac{\ell}{2}$ .
- Des weiteren ist  $\beta \geq n^{-k}$ .

Wenn  $A^*$  verwirft, dann sind höchstens  $\frac{\ell}{2}$  Berechnungen akzeptierend  $\implies$

$$\begin{aligned} \text{pr}[A^* \text{ verwirft } x] &\leq \text{pr}\left[\sum_{i=1}^{\ell} X_i \leq \frac{\ell}{2}\right] = \text{pr}\left[\sum_{i=1}^{\ell} X_i \leq \mu \cdot (1 - \beta)\right] \\ &\leq \exp(-\Omega(\beta^2 \cdot \mu)) \leq \exp(-\Omega(\frac{\ell}{n^{2k}})), \end{aligned}$$

denn  $\beta^2 \cdot \mu \geq n^{-2k} \cdot \frac{\ell}{2}$  folgt aus  $\beta \geq n^{-k}$  und  $\mu \geq \frac{\ell}{2} \implies$  Für  $\ell = \Omega(n^{2k+r})$  wird  $A^*$  die Eingabe  $x$  mit Wahrscheinlichkeit höchstens  $2^{-n^r}$  verworfen.

**Fall 2:**  $x \notin L$ . Die Argumentation verläuft analog.

# Nicht-uniforme Derandomisierung,

$$\text{BPP} \subset \text{P/poly}$$

# Nicht-uniforme Derandomisierung: $BPP \subset P/poly$

Es ist  $BPP = StrongBPP$ .

Sei  $L \in BPP$ . Dann ist  $L = L(A)$  für einen randomisierten Algorithmus  $A$  mit Laufzeit  $t_A = poly(n)$  und Fehlerwahrscheinlichkeit höchstens  $2^{-2n}$ .

# Nicht-uniforme Derandomisierung: $BPP \subset P/poly$

Es ist  $BPP = StrongBPP$ .

Sei  $L \in BPP$ . Dann ist  $L = L(A)$  für einen randomisierten Algorithmus  $A$  mit Laufzeit  $t_A = poly(n)$  und Fehlerwahrscheinlichkeit höchstens  $2^{-2n}$ .

- $A$  mit Eingabe  $x$  lässt sich als *deterministischer* Algorithmus auffassen, der die Eingaben  $x$  und  $y =$  eine Folge von Zufallsbits erhält.
- Für jede Eingabe  $x \in \{0, 1\}^n$  wird  $A$  für einen Anteil von höchstens  $2^{-2n}$  aller Zufallsstrings  $y$  die falsche Antwort liefern.

# Nicht-uniforme Derandomisierung: $BPP \subset P/poly$

Es ist  $BPP = StrongBPP$ .

Sei  $L \in BPP$ . Dann ist  $L = L(A)$  für einen randomisierten Algorithmus  $A$  mit Laufzeit  $t_A = poly(n)$  und Fehlerwahrscheinlichkeit höchstens  $2^{-2n}$ .

- $A$  mit Eingabe  $x$  lässt sich als *deterministischer* Algorithmus auffassen, der die Eingaben  $x$  und  $y =$  eine Folge von Zufallsbits erhält.
- Für jede Eingabe  $x \in \{0, 1\}^n$  wird  $A$  für einen Anteil von höchstens  $2^{-2n}$  aller Zufallsstrings  $y$  die falsche Antwort liefern.
  - ▶ Für wie viele Zufallsstrings erhält  $A$  die falsche Antwort auf *irgendeiner* Eingabe in  $\{0, 1\}^n$ ? Für höchstens den Anteil

$2^n$

# Nicht-uniforme Derandomisierung: $BPP \subset P/poly$

Es ist  $BPP = StrongBPP$ .

Sei  $L \in BPP$ . Dann ist  $L = L(A)$  für einen randomisierten Algorithmus  $A$  mit Laufzeit  $t_A = poly(n)$  und Fehlerwahrscheinlichkeit höchstens  $2^{-2n}$ .

- $A$  mit Eingabe  $x$  lässt sich als *deterministischer* Algorithmus auffassen, der die Eingaben  $x$  und  $y =$  eine Folge von Zufallsbits erhält.
- Für jede Eingabe  $x \in \{0, 1\}^n$  wird  $A$  für einen Anteil von höchstens  $2^{-2n}$  aller Zufallsstrings  $y$  die falsche Antwort liefern.
  - ▶ Für wie viele Zufallsstrings erhält  $A$  die falsche Antwort auf *irgendeiner* Eingabe in  $\{0, 1\}^n$ ? Für höchstens den Anteil

$$2^n \cdot 2^{-2n} = 2^{-n} < 1$$

aller Zufallsstrings.

# Nicht-uniforme Derandomisierung: $BPP \subset P/poly$

Es ist  $BPP = StrongBPP$ .

Sei  $L \in BPP$ . Dann ist  $L = L(A)$  für einen randomisierten Algorithmus  $A$  mit Laufzeit  $t_A = poly(n)$  und Fehlerwahrscheinlichkeit höchstens  $2^{-2n}$ .

- $A$  mit Eingabe  $x$  lässt sich als *deterministischer* Algorithmus auffassen, der die Eingaben  $x$  und  $y =$  eine Folge von Zufallsbits erhält.
- Für jede Eingabe  $x \in \{0, 1\}^n$  wird  $A$  für einen Anteil von höchstens  $2^{-2n}$  aller Zufallsstrings  $y$  die falsche Antwort liefern.
  - ▶ Für wie viele Zufallsstrings erhält  $A$  die falsche Antwort auf *irgendeiner* Eingabe in  $\{0, 1\}^n$ ? Für höchstens den Anteil

$$2^n \cdot 2^{-2n} = 2^{-n} < 1$$

aller Zufallsstrings.

- ▶ Es gibt einen Zufallsstring  $y_n$  für den  $A$  auf *allen* Eingaben der Länge  $n$  die richtige Klassifikation bestimmt.

# Nicht-uniforme Derandomisierung: $BPP \subset P/poly$

Es ist  $BPP = StrongBPP$ .

Sei  $L \in BPP$ . Dann ist  $L = L(A)$  für einen randomisierten Algorithmus  $A$  mit Laufzeit  $t_A = poly(n)$  und Fehlerwahrscheinlichkeit höchstens  $2^{-2n}$ .

- $A$  mit Eingabe  $x$  lässt sich als *deterministischer* Algorithmus auffassen, der die Eingaben  $x$  und  $y =$  eine Folge von Zufallsbits erhält.
- Für jede Eingabe  $x \in \{0, 1\}^n$  wird  $A$  für einen Anteil von höchstens  $2^{-2n}$  aller Zufallsstrings  $y$  die falsche Antwort liefern.
  - ▶ Für wie viele Zufallsstrings erhält  $A$  die falsche Antwort auf *irgendeiner* Eingabe in  $\{0, 1\}^n$ ? Für höchstens den Anteil

$$2^n \cdot 2^{-2n} = 2^{-n} < 1$$

aller Zufallsstrings.

- ▶ Es gibt einen Zufallsstring  $y_n$  für den  $A$  auf *allen* Eingaben der Länge  $n$  die richtige Klassifikation bestimmt.
- $L$  wird von einem nicht-uniformen, deterministischen Algorithmus erkannt, der für Eingaben der Länge  $n$  mit  $y_n$  als Zufallsstring arbeitet.  $\square$

# BPP und die polynomielle Hierarchie

BPP ist unter Komplementbildung abgeschlossen  $\implies$  Zeige:  $BPP \subseteq \Sigma_2^P$ .

- $BPP = \text{StrongBPP} \implies$  für  $L \in BPP$  gibt es einen randomisierten Algorithmus  $A$  mit Fehlerwahrscheinlichkeit höchstens  $2^{-n}$ .

BPP ist unter Komplementbildung abgeschlossen  $\implies$  Zeige:  $BPP \subseteq \Sigma_2^P$ .

- $BPP = \text{StrongBPP} \implies$  für  $L \in BPP$  gibt es einen randomisierten Algorithmus  $A$  mit Fehlerwahrscheinlichkeit höchstens  $2^{-n}$ .
- Simuliere  $A$  durch einen effizienten  $\Sigma_2^P$ -Algorithmus  $A^*$ .

Für Eingabe  $x$

- ▶ rät  $A^*$  in der Ratephase „Zufallstrings“  $y_1, \dots, y_k$  und
- ▶ setzt in der Verifikationsphase einen String  $y$  „dagegen“.
  - ★  $A^*$  akzeptiert die Eingabe  $x : \iff$  Es gibt Zufallstrings  $y_1, \dots, y_k$ , so dass es für jedes  $y$  mindestens ein  $i$  gibt und  $A(x, y \oplus y_i)$  akzeptiert

$A^*$  versucht mit der Wahl von  $y$  sicherzustellen, dass nicht ungerechtfertigter Weise akzeptiert wird.

$A^*$  akzeptiert die Eingabe  $x$  :  $\iff$  Es gibt Zufallstrings  $y_1, \dots, y_k$ , so dass es für jedes  $y$  mindestens ein  $i$  gibt und  $A(x, y \oplus y_i)$  akzeptiert

**Fall 1:**  $x \in L$ . Sage, dass  $y_i$  falsch für  $x$  und  $y$  ist, wenn  $A(x, y \oplus y_i)$  verwirft.

- Es ist  $\text{pr}_{y_i}[y_i \text{ ist falsch für } x \text{ und } y] \leq \frac{1}{2}$ .
- Würfel die  $y_i$  unabhängig voneinander aus  $\implies$

$$\text{pr}_{y_1, \dots, y_k}[\text{alle } y_i \text{ sind falsch für } x \text{ und } y] \leq 2^{-k}.$$

$A^*$  akzeptiert die Eingabe  $x$  :  $\iff$  Es gibt Zufallstrings  $y_1, \dots, y_k$ , so dass es für jedes  $y$  mindestens ein  $i$  gibt und  $A(x, y \oplus y_i)$  akzeptiert

**Fall 1:**  $x \in L$ . Sage, dass  $y_i$  falsch für  $x$  und  $y$  ist, wenn  $A(x, y \oplus y_i)$  verwirft.

- Es ist  $\text{pr}_{y_i} [y_i \text{ ist falsch für } x \text{ und } y] \leq \frac{1}{2}$ .
- Würfel die  $y_i$  unabhängig voneinander aus  $\implies$

$$\text{pr}_{y_1, \dots, y_k} [\text{alle } y_i \text{ sind falsch für } x \text{ und } y] \leq 2^{-k}.$$

Sei  $|x| = n$  und  $A$  arbeite in Laufzeit  $t_A(n)$ . Dann folgt

$$\text{pr}_{y_1, \dots, y_k} [\text{es gibt } y, \text{ so dass alle } y_i \text{ falsch für } x \text{ und } y \text{ sind}] \leq 2^{t_A(n) - k}.$$

$A^*$  akzeptiert die Eingabe  $x$  :  $\iff$  **Es gibt** Zufallstrings  $y_1, \dots, y_k$ , so dass es **für jedes**  $y$  mindestens ein  $i$  gibt und  $A(x, y \oplus y_i)$  akzeptiert

**Fall 1:**  $x \in L$ . Sage, dass  $y_i$  **falsch für  $x$  und  $y$**  ist, wenn  $A(x, y \oplus y_i)$  verwirft.

- Es ist  $\text{pr}_{y_i}[y_i \text{ ist falsch für } x \text{ und } y] \leq \frac{1}{2}$ .
- Würfel die  $y_i$  unabhängig voneinander aus  $\implies$

$$\text{pr}_{y_1, \dots, y_k}[\text{alle } y_i \text{ sind falsch für } x \text{ und } y] \leq 2^{-k}.$$

Sei  $|x| = n$  und  $A$  arbeite in Laufzeit  $t_A(n)$ . Dann folgt

$$\text{pr}_{y_1, \dots, y_k}[\text{es gibt } y, \text{ so dass alle } y_i \text{ falsch für } x \text{ und } y \text{ sind}] \leq 2^{t_A(n) - k}.$$

Wähle  $k := t_A(n) + 1$ : **Es gibt** Zufallstrings  $y_1, \dots, y_k$ , so dass es **für jedes**  $y$  mindestens ein  $i$  gibt und  $A(x, y \oplus y_i)$  akzeptiert  $\implies$

$A^*$  akzeptiert  $x$ .

$A^*$  akzeptiert die Eingabe  $x : \iff$  Es gibt Zufallstrings  $y_1, \dots, y_k$ , so dass es für jedes  $y$  mindestens ein  $i$  gibt und  $A(x, y \oplus y_i)$  akzeptiert

**Fall 2:**  $x \notin L$ . Sage, dass  $y_i$  falsch für  $x$  und  $y$  ist, wenn  $A(x, y \oplus y_i)$  akzeptiert.

Die Strings  $y_1, \dots, y_k$  seien beliebig. Da Fehlerwahrscheinlichkeit  $\leq 2^{-n}$ , gilt

$$\text{pr}_y[y_i \text{ ist falsch für } x \text{ und } y] \leq 2^{-n},$$

für jedes  $i \leq k$

$A^*$  akzeptiert die Eingabe  $x : \iff$  Es gibt Zufallstrings  $y_1, \dots, y_k$ , so dass es für jedes  $y$  mindestens ein  $i$  gibt und  $A(x, y \oplus y_i)$  akzeptiert

**Fall 2:**  $x \notin L$ . Sage, dass  $y_i$  falsch für  $x$  und  $y$  ist, wenn  $A(x, y \oplus y_i)$  akzeptiert.

Die Strings  $y_1, \dots, y_k$  seien beliebig. Da Fehlerwahrscheinlichkeit  $\leq 2^{-n}$ , gilt

$$\text{pr}_y[y_i \text{ ist falsch für } x \text{ und } y] \leq 2^{-n},$$

für jedes  $i \leq k \implies$

$$\text{pr}_y[\text{es gibt } i \text{ und } y_i \text{ ist falsch für } x \text{ und } y] \leq k \cdot 2^{-n}$$

$A^*$  akzeptiert die Eingabe  $x : \iff$  Es gibt Zufallstrings  $y_1, \dots, y_k$ , so dass sie für jedes  $y$  mindestens ein  $i$  gibt und  $A(x, y \oplus y_i)$  akzeptiert

**Fall 2:**  $x \notin L$ . Sage, dass  $y_i$  falsch für  $x$  und  $y$  ist, wenn  $A(x, y \oplus y_i)$  akzeptiert.

Die Strings  $y_1, \dots, y_k$  seien beliebig. Da Fehlerwahrscheinlichkeit  $\leq 2^{-n}$ , gilt

$$\text{pr}_y[y_i \text{ ist falsch für } x \text{ und } y] \leq 2^{-n},$$

für jedes  $i \leq k \implies$

$$\text{pr}_y[\text{es gibt } i \text{ und } y_i \text{ ist falsch für } x \text{ und } y] \leq k \cdot 2^{-n} < 1,$$

denn  $k$  ist polynomiell in  $n$ .

$A^*$  akzeptiert die Eingabe  $x$  :  $\iff$  Es gibt Zufallstrings  $y_1, \dots, y_k$ , so dass es für jedes  $y$  mindestens ein  $i$  gibt und  $A(x, y \oplus y_i)$  akzeptiert

**Fall 2:**  $x \notin L$ . Sage, dass  $y_i$  falsch für  $x$  und  $y$  ist, wenn  $A(x, y \oplus y_i)$  akzeptiert. Die Strings  $y_1, \dots, y_k$  seien beliebig. Da Fehlerwahrscheinlichkeit  $\leq 2^{-n}$ , gilt

$$\text{pr}_y[y_i \text{ ist falsch für } x \text{ und } y] \leq 2^{-n},$$

für jedes  $i \leq k \implies$

$$\text{pr}_y[\text{es gibt } i \text{ und } y_i \text{ ist falsch für } x \text{ und } y] \leq k \cdot 2^{-n} < 1,$$

denn  $k$  ist polynomiell in  $n$ .

Für alle  $y_1, \dots, y_k$  gibt es  $y$ , so dass  $A(x, y \oplus y_i)$  für jedes  $i$  verwerfend ist  $\implies$

$A^*$  verwirft Eingabe  $x$ .

# Randomisierte Reduktionen

Im „*Promise-Problem*“ UNIQUE-SAT wird versprochen, dass eine Formel  $\psi$  in konjunktiver Normalform **genau eine** erfüllende Belegung  $\alpha$  besitzt.

Bestimme  $\alpha$ .

Wie schwierig ist UNIQUE-SAT? Eine Sprachenversion von UNIQUE-SAT:

*Für eine KNF  $\psi$  ist zu bestimmen, ob  $\psi$  eindeutig erfüllbar oder unerfüllbar ist. Ist  $\psi$  mehrdeutig erfüllbar, dann ist jede Antwort korrekt.*

Im „*Promise-Problem*“ UNIQUE-SAT wird versprochen, dass eine Formel  $\psi$  in konjunktiver Normalform **genau eine** erfüllende Belegung  $\alpha$  besitzt.

Bestimme  $\alpha$ .

Wie schwierig ist UNIQUE-SAT? Eine Sprachenversion von UNIQUE-SAT:

*Für eine KNF  $\psi$  ist zu bestimmen, ob  $\psi$  eindeutig erfüllbar oder unerfüllbar ist. Ist  $\psi$  mehrdeutig erfüllbar, dann ist jede Antwort korrekt.*

Es gibt eine effizient berechenbare, randomisierte Reduktion  $T$ , die KNF-Formeln  $\psi$  mit  $n$  Variablen auf KNF-Formeln  $T(\psi)$  mit den folgenden Eigenschaften abbildet:

- 1 Wenn  $\psi$  zu KNF-SAT gehört, dann wird  $T(\psi)$  genau eine erfüllende Belegung mit Wahrscheinlichkeit mindestens  $\frac{8}{n}$  besitzen.
- 2 Wenn  $\psi$  nicht zu KNF-SAT gehört, dann wird  $T(\psi)$  mit Wahrscheinlichkeit 1 unerfüllbar sein.

# Wie arbeitet $T$ ?

Die KNF  $\psi$  besitze genau  $n$  Variablen.

1.  $T$  wählt  $i \in \{0, 1, \dots, n-1\}$  zufällig aus:

- ▶  $T$  „wettet“ darauf, dass die Anzahl erfüllender Belegungen mindestens  $2^i$  aber weniger als  $2^{i+1}$  beträgt.
- ▶ Sollte  $\phi$  erfüllbar sein, dann wird  $T$  diese Wette natürlich nur mit W-keit  $\frac{1}{n}$  gewinnen, aber eine kleine Gewinnwahrscheinlichkeit ist akzeptabel.

# Wie arbeitet $T$ ?

Die KNF  $\psi$  besitze genau  $n$  Variablen.

1.  $T$  wählt  $i \in \{0, 1, \dots, n-1\}$  zufällig aus:

- ▶  $T$  „wettet“ darauf, dass die Anzahl erfüllender Belegungen mindestens  $2^i$  aber weniger als  $2^{i+1}$  beträgt.
- ▶ Sollte  $\phi$  erfüllbar sein, dann wird  $T$  diese Wette natürlich nur mit W-keit  $\frac{1}{n}$  gewinnen, aber eine kleine Gewinnwahrscheinlichkeit ist akzeptabel.

2. Für  $m := i + 2$  wird eine zufällige Hashfunktion

$$h(x) = A \cdot x + b \text{ mit } A \in \mathbb{Z}_2^{m \times n} \text{ und } b \in \mathbb{Z}_2^m$$

ausgewürfelt. Für  $A, b$ , die Belegung  $x$  und *Hilfsvariablen*  $y$  ist

$$T(\psi)(x, y) := \psi(x) \wedge \phi(x, A, b, y).$$

# T funktioniert, weil ...

Annahme:  $\psi$  besitze  $b$  erfüllende Belegungen mit  $2^{m-2} \leq b \leq 2^{m-1}$ .

$U_x$  ist das Ereignis, dass  $x$  die einzige erfüllende Belegung mit  $h(x) = 0$  ist.

Mit welcher Wahrscheinlichkeit wird  $U_x$  für irgendeine Belegung  $x$  erfüllt?

$$\text{pr}[U_x] \geq \text{pr}_{h \in \mathcal{A}_{m,n}}[h(x) = 0] - \sum_{x', \psi(x') \text{ ist wahr}} \text{pr}_{h \in \mathcal{A}_{m,n}}[h(x) = h(x') = 0]$$

# T funktioniert, weil ...

Annahme:  $\psi$  besitze  $b$  erfüllende Belegungen mit  $2^{m-2} \leq b \leq 2^{m-1}$ .

$U_x$  ist das Ereignis, dass  $x$  die einzige erfüllende Belegung mit  $h(x) = 0$  ist.

Mit welcher Wahrscheinlichkeit wird  $U_x$  für irgendeine Belegung  $x$  erfüllt?

$$\begin{aligned} \text{pr}[U_x] &\geq \text{pr}_{h \in \mathcal{A}_{m,n}}[h(x) = 0] - \sum_{x', \psi(x') \text{ ist wahr}} \text{pr}_{h \in \mathcal{A}_{m,n}}[h(x) = h(x') = 0] \\ &\geq \frac{1}{2^m} - \end{aligned}$$

# T funktioniert, weil ...

Annahme:  $\psi$  besitze  $b$  erfüllende Belegungen mit  $2^{m-2} \leq b \leq 2^{m-1}$ .

$U_x$  ist das Ereignis, dass  $x$  die einzige erfüllende Belegung mit  $h(x) = 0$  ist.

Mit welcher Wahrscheinlichkeit wird  $U_x$  für irgendeine Belegung  $x$  erfüllt?

$$\begin{aligned} \text{pr}[U_x] &\geq \text{pr}_{h \in \mathcal{A}_{m,n}}[h(x) = 0] - \sum_{x', \psi(x') \text{ ist wahr}} \text{pr}_{h \in \mathcal{A}_{m,n}}[h(x) = h(x') = 0] \\ &\geq \frac{1}{2^m} - \frac{2^{m-1}}{2^{2m}} \end{aligned}$$

# T funktioniert, weil ...

Annahme:  $\psi$  besitze  $b$  erfüllende Belegungen mit  $2^{m-2} \leq b \leq 2^{m-1}$ .

$U_x$  ist das Ereignis, dass  $x$  die einzige erfüllende Belegung mit  $h(x) = 0$  ist.

Mit welcher Wahrscheinlichkeit wird  $U_x$  für irgendeine Belegung  $x$  erfüllt?

$$\begin{aligned} \text{pr}[U_x] &\geq \text{pr}_{h \in \mathcal{A}_{m,n}}[h(x) = 0] - \sum_{x', \psi(x') \text{ ist wahr}} \text{pr}_{h \in \mathcal{A}_{m,n}}[h(x) = h(x') = 0] \\ &\geq \frac{1}{2^m} - \frac{2^{m-1}}{2^{2m}} \geq \frac{1}{2^{m+1}}. \end{aligned}$$

# T funktioniert, weil ...

Annahme:  $\psi$  besitze  $b$  erfüllende Belegungen mit  $2^{m-2} \leq b \leq 2^{m-1}$ .

$U_x$  ist das Ereignis, dass  $x$  die einzige erfüllende Belegung mit  $h(x) = 0$  ist.

Mit welcher Wahrscheinlichkeit wird  $U_x$  für irgendeine Belegung  $x$  erfüllt?

$$\begin{aligned} \text{pr}[U_x] &\geq \text{pr}_{h \in \mathcal{A}_{m,n}}[h(x) = 0] - \sum_{x', \psi(x') \text{ ist wahr}} \text{pr}_{h \in \mathcal{A}_{m,n}}[h(x) = h(x') = 0] \\ &\geq \frac{1}{2^m} - \frac{2^{m-1}}{2^{2m}} \geq \frac{1}{2^{m+1}}. \end{aligned}$$

Also folgt

$$\text{pr}[\exists x \in \{0, 1\}^n U_x] = \sum_{x, \psi(x) \text{ ist wahr}} \text{pr}[U_x]$$

# T funktioniert, weil ...

Annahme:  $\psi$  besitze  $b$  erfüllende Belegungen mit  $2^{m-2} \leq b \leq 2^{m-1}$ .

$U_x$  ist das Ereignis, dass  $x$  die einzige erfüllende Belegung mit  $h(x) = 0$  ist.

Mit welcher Wahrscheinlichkeit wird  $U_x$  für irgendeine Belegung  $x$  erfüllt?

$$\begin{aligned} \text{pr}[U_x] &\geq \text{pr}_{h \in \mathcal{A}_{m,n}}[h(x) = 0] - \sum_{x', \psi(x') \text{ ist wahr}} \text{pr}_{h \in \mathcal{A}_{m,n}}[h(x) = h(x') = 0] \\ &\geq \frac{1}{2^m} - \frac{2^{m-1}}{2^{2m}} \geq \frac{1}{2^{m+1}}. \end{aligned}$$

Also folgt

$$\text{pr}[\exists x \in \{0, 1\}^n U_x] = \sum_{x, \psi(x) \text{ ist wahr}} \text{pr}[U_x] \geq \frac{2^{m-2}}{2^{m+1}} \geq \frac{1}{8}.$$

# T funktioniert, weil ...

Annahme:  $\psi$  besitze  $b$  erfüllende Belegungen mit  $2^{m-2} \leq b \leq 2^{m-1}$ .

$U_x$  ist das Ereignis, dass  $x$  die einzige erfüllende Belegung mit  $h(x) = 0$  ist.

Mit welcher Wahrscheinlichkeit wird  $U_x$  für irgendeine Belegung  $x$  erfüllt?

$$\begin{aligned} \text{pr}[U_x] &\geq \text{pr}_{h \in \mathcal{A}_{m,n}}[h(x) = 0] - \sum_{x', \psi(x') \text{ ist wahr}} \text{pr}_{h \in \mathcal{A}_{m,n}}[h(x) = h(x') = 0] \\ &\geq \frac{1}{2^m} - \frac{2^{m-1}}{2^{2m}} \geq \frac{1}{2^{m+1}}. \end{aligned}$$

Also folgt

$$\text{pr}[\exists x \in \{0, 1\}^n U_x] = \sum_{x, \psi(x) \text{ ist wahr}} \text{pr}[U_x] \geq \frac{2^{m-2}}{2^{m+1}} \geq \frac{1}{8}.$$

- Ist  $\psi$  erfüllbar, dann ist  $T(\psi)$  mit W-keit  $\frac{1}{8n}$  erfüllbar.
- Ist  $\psi$  unerfüllbar, dann ist natürlich auch  $T(\psi)$  unerfüllbar. □

# Zusammenfassung

- 1 Randomisierung besitzt eine drastische Fehlerreduktion, **wenn** der Fehler merklich kleiner als  $\frac{1}{2}$  ist.
  - ▶  $\text{WeakBPP} = \text{StrongBPP}$ ,
  - ▶ **aber**  $\text{NP} \subseteq \text{PP}$ .
- 2 Es ist  $\text{BPP} \subseteq \text{P/poly}$ . Die Inklusion  $\text{NP} \subseteq \text{BPP}$  ist nicht plausibel, denn
  - ▶  $\text{NP} \subseteq \text{BPP} \subseteq \text{P/poly}$ .
  - ▶ Das Theorem von Karp und Lipton besagt, dass die polynomielle Hierarchie  $\text{PH}$  zu  $\Sigma_2^{\text{P}}$  kollabiert, wenn  $\text{NP} \subseteq \text{P/poly}$  gilt.
- 3 Vermutlich gilt sogar  $\text{P} = \text{BPP}$ : Siehe dazu die folgende Diskussion über Pseudo-Random-Generatoren. Randomisierung zeigt aber seine Stärke
  - ▶ im Entwurf einfacher und schneller Offline-Algorithmen,
  - ▶ im Entwurf von Online-Algorithmen,
  - ▶ in der Kryptographie (Wahl des öffentlichen Schlüssels)
  - ▶ in interaktiven Beweisen (später).
- 4 Mit einer randomisierten Reduktion kann die Schwierigkeit von Promise-Problemen wie etwa UNIQUE-SAT nachgewiesen werden.