

# Pseudozufallsgeneratoren

vorläufige Version  
2. Juni 2000

*Anyone who considers arithmetical  
methods of producing random digits  
is, of course, in a state of sin.*

— John von Neumann (1951)

## 1. Pseudozufällige Ensembles

Um asymptotische Aussagen treffen zu können, betrachten wir Folgen von Wahrscheinlichkeitsverteilungen:

**Definition 3.1** (Ensemble). *Sei  $\ell(n)$  ein Polynom. Wir nennen eine Folge von Wahrscheinlichkeitsverteilungen  $(X_n)_{n \in \mathbb{N}}$ , wobei  $X_n$  eine Verteilung auf  $\{0, 1\}^{\ell(n)}$  ist, ein Ensemble vom Typ  $\ell(n)$ .*

YAO [Yao82] bezeichnet den Typ  $\ell(n)$  als Längenfunktion des Ensembles. Im weiteren verwenden wir die gleiche Notation sowohl für die Wahrscheinlichkeitsverteilung als auch für gemäß dieser Verteilung verteilten Zufallsvariablen. Sei  $U_n$  die Gleichverteilung auf  $\{0, 1\}^n$ .

In der klassischen Sichtweise ist ein Pseudozufallsgenerator ein Algorithmus, der deterministisch zu einem zufälligen Startwert eine Folge von Zahlen ausgibt, die sich ähnlich einer Folge unabhängiger, uniform verteilter Zufallsvariablen verhalten soll. Um diese Eigenschaft zu untersuchen, entstanden im Laufe der Zeit zahlreiche Tests [Knuth97]. Ein Generator, dessen Ausgabe die bekannten Standardtests besteht, gilt nach klassischer Sichtweise als Pseudozufallsgenerator. Dieses Vorgehen in der klassischen Theorie ist nicht systematisch: Zwar kann ein Generator alle bekannten Tests bestehen, dennoch ist nicht auszuschließen, dass die Ausgabefolge einen anderen Test nicht besteht. Für kryptographische Anwendungen von Pseudozufallsgeneratoren ist ein Sicherheitsbeweis Voraussetzung: Der Generator besteht jeden effizient durchführbaren Test.

**Definition 3.2** (Statistischer Test). *Ein statistischer Test (Unterscheider<sup>1</sup>)  $\mathcal{D}$  ist ein probabilistischer Polynomialzeit-Algorithmus mit  $\{0, 1\}$ -Ausgabe. Zum Ensemble  $(X_n)_{n \in \mathbb{N}}$  vom Typ  $\ell(n)$  setze:*

$$\Delta_{\mathcal{D}}(X_n) := \text{Ws}[\mathcal{D}(X_n) = 1] - \text{Ws}[\mathcal{D}(U_{\ell(n)}) = 1],$$

wobei die Wahrscheinlichkeit über die Eingabe und die internen Münzwürfe des Algorithmus'  $\mathcal{D}$  gebildet wird. Das Ensemble  $(X_n)_{n \in \mathbb{N}}$  besteht den statistischen Test  $\mathcal{D}$  mit Toleranz  $\delta(n) \geq 0$ , falls ein  $n_0 \in \mathbb{N}$  existiert, so dass für alle  $n \geq n_0$  gilt:  $|\Delta_{\mathcal{D}}(X_n)| < \delta(n)$ .

Besteht das Ensemble  $(X_n)_{n \in \mathbb{N}}$  den statistischen Test  $\mathcal{D}$  bei Toleranz  $\delta(n)$  nicht, existieren unendlich viele  $n$  mit  $|\Delta_{\mathcal{D}}(X_n)| \geq \delta(n)$ .

**Definition 3.3** (Pseudozufälliges Ensemble). *Ein Ensemble heißt pseudozufällig, wenn es jeden statistischen Test mit vernachlässigbarer<sup>2</sup> Toleranz besteht.*

Wir hatten bereits den Begriff der *polynomiellen Ununterscheidbarkeit* kennen gelernt. Zwei Ensembles  $(X_n)_{n \in \mathbb{N}}$  und  $(Y_n)_{n \in \mathbb{N}}$  des gleich Typs  $\ell(n)$  sind polynomiell ununterscheidbar, falls für jeden statistischen Test  $\mathcal{D}$  der Absolutbetrag von

$$\Delta_{\mathcal{D}}(X_n, Y_n) = \text{Ws}[\mathcal{D}(X_n) = 1] - \text{Ws}[\mathcal{D}(Y_n) = 1]$$

vernachlässigbar ist. Ein Ensemble  $(X_n)_{n \in \mathbb{N}}$  des Typs  $\ell(n)$  ist pseudozufällig, wenn es von der uniformen Verteilung  $(U_{\ell(n)})_{n \in \mathbb{N}}$  polynomiell ununterscheidbar ist.

Ein (polynomiell erzeugbares<sup>3</sup>) pseudozufälliges Ensemble ist auch dann von echten Zufallsbits polynomiell ununterscheidbar, wenn der statistische Test polynomiell viele Eingaben erhält. Zu  $X_n$  sei  $X_n^{\times m}$  die Verteilung von  $m$  unabhängigen, gemäß  $X_n$  verteilten Zufallsvariablen. Zu Ensemble  $(X_n)_{n \in \mathbb{N}}$  und einem Polynom  $m := m(n)$  nennen wir  $(X_n^{\times m})_{n \in \mathbb{N}}$  das  *$m$ -fache Produkt-Ensemble*.

**Lemma 3.4.** *Sei  $(X_n)_{n \in \mathbb{N}}$  ein polynomiell erzeugbares Ensemble vom Typ  $\ell(n)$  und  $m := m(n)$  ein Polynom. Wenn das Produkt-Ensemble  $(X_n^{\times m})_{n \in \mathbb{N}}$  einen statistischen Test  $\mathcal{D}_m$  mit Toleranz  $\delta(n)$  nicht besteht, dann existiert ein statistischer Test  $\mathcal{D}$ , den  $(X_n)_{n \in \mathbb{N}}$  mit Toleranz  $\frac{\delta(n)}{m(n)}$  nicht besteht. Dieser*

<sup>1</sup>engl. Distinguisher

<sup>2</sup>Eine Funktion  $\delta : \mathbb{N} \rightarrow [0, 1]$  heißt vernachlässigbar, wenn sie kleiner als jeder polynomieller Bruchteil wird, d.h. wenn für jedes Polynom  $p$  ein  $n_0 \in \mathbb{N}$  existiert, so dass für alle  $n \geq n_0$  gilt:  $\delta(n) < \frac{1}{p(n)}$ .

<sup>3</sup>Ein Ensemble  $(X_n)_{n \in \mathbb{N}}$  heißt *polynomiell erzeugbar*, wenn es einen probabilistischen Polynomialzeit-Algorithmus gibt, der bei Eingabe von  $1^n$  ( $n$  Einsen) ein gemäß  $X_n$  verteiltes Element erzeugt.

Test hat die Laufzeit  $|\mathcal{D}| = |\mathcal{D}_m| + m \cdot |X_n| + \mathcal{O}(1)$ , wobei  $|X_n|$  die Zeit zum Erzeugen eines gemäß  $X_n$  verteilten Elementes ist.

**Beweis.** Übungsaufgabe. Wende das Hybrid-Argument aus dem Beweis von Satz 3.11 auf die hybriden Verteilungen

$$H_n^{(j)} := U_{\ell(n)}^{\times j} X_n^{\times (m-j)} \quad \text{für } j := j(n) = 0, 1, \dots, \ell(n).$$

(die ersten  $j$  Werte sind uniform und die letzten  $m - j$  Werte sind gemäß  $X_n$  verteilt) an.  $\square$

Wenn ein Ensemble einen statistischen Test  $\mathcal{D}$  mit Toleranz  $\delta$  nicht besteht, so existiert ein Unterscheider für das  $m$ -fach Produkt-Ensemble: Rufe  $\mathcal{D}$  mit dem ersten Teil der Eingabe auf:

**Satz 3.5.** Sei  $(X_n)_{n \in \mathbb{N}}$  ein polynomiell erzeugbares Ensemble und  $m := m(n)$  ein Polynom. Genau dann ist  $(X_n)_{n \in \mathbb{N}}$  pseudozufällig, wenn das  $m$ -fach Produkt-Ensemble  $(X_n^{\times m})_{n \in \mathbb{N}}$  pseudozufällig ist.

**Definition 3.6** (Pseudozufallsgenerator). Ein deterministischer Polynomialzeit-Algorithmus  $G : \{0, 1\}^* \rightarrow \{0, 1\}^*$  heißt (kryptographisch sicherer) Pseudozufallsgenerator vom Typ  $\ell(n)$ , wenn:

- (1) Für alle  $n \in \mathbb{N}$  gilt  $G(\{0, 1\}^n) \subseteq \{0, 1\}^{\ell(n)}$ .
- (2) Für alle  $n \in \mathbb{N}$  gilt  $\ell(n) \geq n + 1$ .
- (3) Das Ensemble  $(G(U_n))_{n \in \mathbb{N}}$  vom Typ  $\ell(n)$  ist pseudozufällig.

Da Algorithmus  $G$  polynomielle Laufzeit hat, ist  $\ell : \mathbb{N} \rightarrow \mathbb{N}$  durch ein Polynom beschränkt. Für Eigenschaft 3 nehmen wir an, dass der Algorithmus des Generators  $G$  öffentlich ist, d.h.  $(G(U_n))_{n \in \mathbb{N}}$  ist polynomiell erzeugbar.

Die Forderungen 1 und 2 sichern, dass die Eingabe, der sog. *Seed*, um mindestens ein Bit expandiert wird. Aus einem Pseudozufallsgenerator  $G$  des Typs  $n + 1$  erhält man einen Generator beliebigen Typs, indem man  $G$  iterativ jeweils auf die ersten  $n$  Bits der Ausgabe anwendet und das zusätzliche Bit ausgibt:

**Satz 3.7.** Sei  $G$  ein Pseudozufallsgenerator des Typs  $n + 1$  und  $\ell(n)$  ein Polynom. Dann ist  $G_{\ell(n)}$  mit

$$G_{\ell(n)}(\mathbf{u}) := (x_1, \dots, x_{\ell(n)}) \in \{0, 1\}^{\ell(n)}$$

gegeben durch die Rekursion  $(\mathbf{u}_{i+1}, x_{i+1}) := G(\mathbf{u}_i) \in \{0, 1\}^{\ell(n)} \times \{0, 1\}$  mit  $\mathbf{u}_0 := \mathbf{u}$  ein Pseudozufallsgenerator vom Typ  $\ell(n)$ .

**Beweis.** Übungsaufgabe. Wende das Hybrid-Argument aus dem Beweis von Satz 3.11 auf die hybriden Verteilungen  $H_n^{(j)} := U_j G_{\ell(n)-j}$  für  $j = j(n) := 0, 1, \dots, \ell(n)$  an.  $\square$

## 2. Vorhersagbarkeit

Ein Polynomialzeit-Algorithmus zur Vorhersage eines Bits nennen wir Prediktor. Analog zum statistischen Test definiere:

**Definition 3.8** (Prediktor zur Vorhersage). *Ein Prediktor  $\mathcal{P}$  zur Vorhersage (nach rechts) des Ensembles  $(X_n)_{n \in \mathbb{N}}$  vom Typ  $\ell(n)$  ist ein probabilistischer Polynomialzeit-Algorithmus mit  $\{0, 1\}$ -Ausgabe, der von der Eingabe  $(i, \mathbf{x}) \in [0, \ell(n)] \times \{0, 1\}^{\ell(n)}$  nur die ersten  $i$  Bits von  $\mathbf{x}$  liest. Der Prediktor  $\mathcal{P}$  hat bei der Vorhersage den Vorteil  $\epsilon(n) \geq 0$ , wenn für unendlich viele  $n$  gilt:*

$$\text{Ws} \left[ \mathcal{P}(I_{\ell(n)}, X_n) = \text{bit}_{I_{\ell(n)}+1}(X_n) \right] \geq \frac{1}{2} + \epsilon(n)$$

wobei  $I_{\ell(n)}$  die Gleichverteilung auf  $[0, \ell(n)]$  sei.

Für eine fixierte Bitstelle  $i = i(n)$  spricht man von einem Prediktor  $\mathcal{P}_i(X_n) := \mathcal{P}(i, X_n)$  zur Vorhersage des  $(i+1)$ -ten Bits. Falls ein Prediktor  $\mathcal{P}_i$  zur Vorhersage des  $(i+1)$ -ten Bits mit Vorteil  $\epsilon(n)$  existiert, ist dies auch ein Prediktor  $\mathcal{P}$  mit Vorteil mindestens  $\frac{\epsilon(n)}{\ell(n)}$ , denn mit Wahrscheinlichkeit  $\frac{1}{\ell(n)}$  gilt  $I_{\ell(n)} = i$ . Umgekehrt folgt aus einem allgemeinen Prediktor  $\mathcal{P}$ , dass eine Bitposition  $i := i(n)$  existiert, so dass der Prediktor  $\mathcal{P}_i$  das  $(i+1)$ -te Bit mit gleichem Vorteil  $\epsilon(n)$  vorhersagt, denn ein Algorithmus  $\mathcal{P}_0, \dots, \mathcal{P}_{\ell(n)-1}$  hat mindestens den durchschnittlichen Vorteil  $\epsilon(n)$ .

**Definition 3.9** (Unvorhersagbares Ensemble). *Ein Ensemble heißt unvorhersagbar, wenn jeder Prediktor zur Vorhersage nur einen nachlässigbaren Vorteil hat.*

Während YAO [Yao82] pseudozufällige Ensembles über statistische Tests definiert hat, haben BLUM und MICALI [BM84] pseudozufällige Ensembles mit Hilfe von Prediktoren sowie der Unvorhersagbarkeit charakterisiert. Wir zeigen ein Resultat von YAO, dass die beiden Eigenschaften äquivalent sind.

**Lemma 3.10.** *Sei  $\mathcal{P}$  ein Prediktor mit Vorteil  $\epsilon(n)$  bei der Vorhersage des Ensembles  $(X_n)_{n \in \mathbb{N}}$  vom Typ  $\ell(n)$ . Dann existiert ein statistischer Test  $\mathcal{D}$  mit Laufzeit  $|\mathcal{D}| = |\mathcal{P}| + \mathcal{O}(1)$ , den das Ensemble  $(X_n)_{n \in \mathbb{N}}$  mit  $\epsilon(n)$ -Toleranz nicht besteht.*

**Beweis.** Mit Hilfe des Prediktors  $\mathcal{P}$  konstruieren wir einen statistischen Test  $\mathcal{D}$ , der entscheidet, ob die Eingabe  $\mathbf{x} = (x_1, \dots, x_{\ell(n)}) \in \{0, 1\}^{\ell(n)}$  gemäß  $X_n$  oder uniform verteilt ist:

- (1) Wähle zufälliges  $i \in_{\mathbb{R}} [0, \ell(n)]$ .
- (2) Falls  $\mathcal{P}(i, \mathbf{x}) \stackrel{!}{=} x_{i+1}$ , gib 1 aus, anderenfalls 0.

Sollte  $\mathbf{x}$  gemäß  $X_n$  verteilt sein, sagt der Prediktor mit Wahrscheinlichkeit  $\frac{1}{2} + \epsilon(n)$  das Bit korrekt voraus. Falls  $\mathbf{x}$  uniform in  $\{0, 1\}^{\ell(n)}$  verteilt ist, stimmt die Ausgabe des Prediktors mit Wahrscheinlichkeit  $\frac{1}{2}$  mit dem Zufallsbit  $x_{i+1}$  überein:

$$\begin{aligned}\Delta_{\mathcal{D}}(X_n) &= \text{Ws} [\mathcal{D}(X_n) = 1] - \text{Ws} [\mathcal{D}(U_{\ell(n)}) = 1] \\ &= \frac{1}{2} + \epsilon(n) - \frac{1}{2} \\ &= \epsilon(n).\end{aligned}$$

Das Ensemble  $(X_n)_{n \in \mathbb{N}}$  besteht den statistischen Test  $\mathcal{D}$  mit Toleranz  $\epsilon(n)$  nicht.  $\square$

**Lemma 3.11.** *Sei  $(X_n)_{n \in \mathbb{N}}$  ein Ensemble vom Typ  $\ell(n)$ , das einen statistischen Test  $\mathcal{D}$  mit  $\delta(n)$ -Toleranz nicht besteht. Dann existieren eine Bitposition  $i = i(n)$  mit  $0 \leq i < \ell(n)$  und ein Prediktor  $\mathcal{P}_i$  zur Vorhersage des  $(i + 1)$ -ten Bits mit Vorteil  $\epsilon(n) := \frac{\delta(n)}{\ell(n)}$  und Laufzeit  $|\mathcal{P}_i| = |\mathcal{D}| + \mathcal{O}(\ell(n))$ .*

**Beweis.** Wir können o.B.d.A. annehmen, dass  $\Delta_{\mathcal{D}}(X_n) \geq 0$  für unendlich viele  $n$  gilt (sonst invertiere die Ausgabe des statistischen Tests  $\mathcal{D}$ ) und betrachten im weiteren nur diese  $n$ . Zur Vereinfachung der Notation setze  $\ell := \ell(n)$  und  $\delta := \delta(n)$ .

Der Beweis basiert auf der Hybrid-Methode<sup>4</sup>, einer für die Unterscheidung von Verteilungen üblichen Beweistechnik. Wir definieren eine Folge hybrider Verteilungen  $H_n^{(0)}, \dots, H_n^{(\ell)}$ . Sei  $H_n^{(j)}$  folgende Verteilung auf  $\{0, 1\}^{\ell}$ : Wähle ein zufälliges Element  $(x_1, \dots, x_{\ell}) \in \{0, 1\}^{\ell}$  gemäß der Verteilung  $X_n$  und ersetze die Bits  $x_{j+1}, x_{j+2}, \dots, x_{\ell}$  durch zufällige Bits für  $j := 0, \dots, \ell$ . Wir betrachten die Ensembles  $(H_n^{(j)})_{n \in \mathbb{N}}$  des Typs  $\ell$ :

$$\underbrace{H_n^{(0)}, H_n^{(1)}, H_n^{(2)}, \dots, H_n^{(\ell-2)}, H_n^{(\ell-1)}}_{=U_{\ell}}, \underbrace{H_n^{(\ell)}}_{=X_n}$$

Bei der Hybrid-Methode folgert man aus der Möglichkeit, die beiden äußeren Verteilungen, nämlich  $H_n^{(0)}$  und  $H_n^{(\ell)}$ , zu unterscheiden, dies auch für zwei benachbarte, hybride Verteilungen zu können. Setze:

$$(1) \quad p_j := \text{Ws} \left[ \mathcal{D}(H_n^{(j)}) = 1 \right] \quad \text{für } j = 0, \dots, \ell.$$

<sup>4</sup>lat. hybrid: von zweierlei Herkunft

Nach Voraussetzung gilt für folgende Teleskopsumme:

$$\begin{aligned}
 (2) \quad \sum_{j=0}^{\ell-1} (p_{j+1} - p_j) &= p_1 - p_0 + p_2 - p_1 + \cdots + p_\ell - p_{\ell-1} \\
 &= p_\ell - p_0 \\
 &= \text{Ws}[\mathcal{D}(X_n) = 1] - \text{Ws}[\mathcal{D}(U_\ell) = 1] \\
 &\geq \delta
 \end{aligned}$$

Mindestens einer der  $\ell$  Summanden  $p_{j+1} - p_j$  muss größer oder gleich  $\frac{\delta}{\ell}$  sein, denn anderenfalls wäre die Summe in (2) kleiner als  $\delta$ . Es existiert ein  $i \in [0, \ell)$  mit

$$(3) \quad p_{i+1} - p_i \geq \frac{\delta}{\ell}.$$

Konstruiere einen Prediktor  $\mathcal{P}_i$  zur Vorhersage des  $(i+1)$ -ten Bits des Ensembles  $(X_n)_{n \in \mathbb{N}}$ :

- (1) Wähle  $u_{i+1}, \dots, u_\ell \in_{\mathbb{R}} \{0, 1\}$  zufällig
- (2) Falls  $\mathcal{D}(x_1, \dots, x_i, u_{i+1}, \dots, u_\ell) = 1$ , gib  $u_{i+1}$ , sonst  $1 - u_{i+1}$  aus.

Wir analysieren den Vorteil des Prediktors  $\mathcal{P}_i$ . Es gilt für gemäß  $X_n$  verteiltes  $(x_1, \dots, x_\ell) \in \{0, 1\}^\ell$  und zufällige Bits  $u_{i+1}, \dots, u_\ell \in_{\mathbb{R}} \{0, 1\}$

$$p_{i+1} \stackrel{(1)}{=} \text{Ws}[\mathcal{D}(x_1, \dots, x_i, u_{i+1}, \dots, u_\ell) = 1 \mid x_{i+1} = u_{i+1}],$$

da  $(x_1, \dots, x_{i+1}, u_{i+2}, \dots, u_\ell)$  gemäß der hybriden Verteilung  $H_n^{(i+1)}$  verteilt ist. Setze:

$$q_{i+1} := \text{Ws}[\mathcal{D}(x_1, \dots, x_i, u_{i+1}, \dots, u_\ell) = 1 \mid x_{i+1} \neq u_{i+1}]$$

Die Bitfolge  $(x_1, \dots, x_i, u_{i+1}, \dots, u_\ell)$  ist gemäß der hybriden Verteilung  $H_n^{(i)}$  verteilt. Da mit Wahrscheinlichkeit  $\frac{1}{2}$  das Zufallsbit  $u_{i+1}$  mit  $x_{i+1}$  übereinstimmt, gilt:

$$(4) \quad p_i \stackrel{(1)}{=} \text{Ws}[\mathcal{D}(x_1, \dots, x_i, u_{i+1}, \dots, u_\ell) = 1] = \frac{1}{2}(p_{i+1} + q_{i+1})$$

Für die Analyse des Vorteils des Prediktors  $\mathcal{P}_i$  unterscheiden wir zwei Fälle, nämlich ob das zufällige Bit  $u_{i+1}$  gleich  $x_{i+1}$  ist oder nicht:

- Falls  $u_{i+1}$  gleich  $x_{i+1}$  ist, sagt  $\mathcal{P}_i$  genau dann das Bit  $x_{i+1}$  richtig voraus, wenn der statistische Test  $\mathcal{D}$  den Wert 1 liefert.
- Falls  $u_{i+1}$  ungleich  $x_{i+1}$  ist, also  $x_{i+1} = 1 - u_{i+1}$ , sagt  $\mathcal{P}_i$  genau dann das Bit  $x_{i+1}$  richtig voraus, wenn der statistische Test  $\mathcal{D}$  den Wert 0 liefert.

Wir erhalten als Erfolgswahrscheinlichkeit des Prediktors  $\mathcal{P}_i$  (Die Eingabe  $(x_1, \dots, x_\ell)$  sei gemäß  $X_n$  verteilt):

$$\begin{aligned} & \text{Ws}[\mathcal{P}_i(x_1, \dots, x_\ell) = x_{i+1}] \\ &= \frac{1}{2} \cdot \text{Ws}[\mathcal{D}(x_1, \dots, x_i, u_{i+1}, \dots, u_\ell) = 1 \mid u_{i+1} = x_{i+1}] \\ & \quad + \frac{1}{2} \cdot \text{Ws}[\mathcal{D}(x_1, \dots, x_i, u_{i+1}, \dots, u_\ell) = 0 \mid u_{i+1} \neq x_{i+1}] \\ &= \frac{1}{2} p_{i+1} + \frac{1}{2} (1 - q_{i+1}) \\ &= \frac{1}{2} + \frac{1}{2} (p_{i+1} - q_{i+1}) \end{aligned}$$

Aus Gleichung (4) und Abschätzung (3) folgt

$$\frac{1}{2} (p_{i+1} - q_{i+1}) = p_{i+1} - \frac{1}{2} (p_{i+1} + q_{i+1}) = p_{i+1} - p_i \geq \frac{\delta}{\ell},$$

so dass der Prediktor  $\mathcal{P}_i$  bei der Vorhersage des  $(i+1)$ -ten Bits des Ensembles  $(X_n)_{n \in \mathbb{N}}$  den Vorteil  $\epsilon(n) := \frac{\delta}{\ell}$  hat.  $\square$

Aus Lemma 3.11 erhalten wir ein Prediktor  $\mathcal{P}$  zur Vorhersage des Ensembles  $(X_n)_{n \in \mathbb{N}}$  mit Vorteil  $\frac{\delta(n)}{\ell(n)^2}$ .

**Satz 3.12** (Yao 1982). *Ein Ensemble ist genau dann (nach rechts) unvorhersagbar, wenn es pseudozufällig ist.*

Der Beweis folgt unmittelbar aus Lemma 3.10 und 3.11. YAO hat diesen Satz in seinem Vortrag auf der FOCS-Konferenz 1982 vorgestellt, er ist jedoch nicht im Artikel [Yao82] veröffentlicht.

Analog zur Vorhersage eines Ensembles  $(X_n)_{n \in \mathbb{N}}$  vom Typ  $\ell(n)$  nach rechts definiert man die Vorhersagbarkeit nach links. Ein Prediktor  $\mathcal{P}^{\text{li}}$  liest von der Eingabe  $(i, \mathbf{x})$  neben  $i$  nur die Bits  $x_i, \dots, x_{\ell(n)}$  und sagt das  $(i-1)$ -te Bit voraus. Zeige: Ein Ensemble ist genau dann nach rechts unvorhersagbar, wenn es nach links unvorhersagbar ist.

**Satz 3.13.** *Ein Ensemble ist genau dann nach rechts unvorhersagbar, wenn es nach links unvorhersagbar ist.*

**Beweis.** Übungsaufgabe. Sei  $(X_n)_{n \in \mathbb{N}}$  ein Ensembles vom Typ  $\ell(n)$  und  $(X_n^{\text{SP}})_{n \in \mathbb{N}}$  die Folge der Verteilungen  $X_n^{\text{SP}}$ , die entstehen, dreht man die Bitfolge eines gemäß  $X_n$  verteilten Bitstrings um. Falls  $(X_n^{\text{SP}})_{n \in \mathbb{N}}$  einen statistischen Test  $\mathcal{D}$  nicht besteht, existiert ein Test, den  $(X_n)_{n \in \mathbb{N}}$  bei gleicher Toleranz ebenfalls nicht besteht: Man dreht die Reihenfolge der Eingabebits  $X_n$  um und wendet  $\mathcal{D}$  an.  $\square$

Ein Ensemble  $(X_n)_{n \in \mathbb{N}}$  heißt *shift-symmetrisch*, wenn jedes Bit von  $X_n$  identisch verteilt ist. Für shift-symmetrisch Ensembles können wir die Aussage von Lemma 3.11 auf Seite 5 verschärfen:

**Satz 3.14.** Sei  $(X_n)_{n \in \mathbb{N}}$  ein shift-symmetrisches Ensemble vom Typ  $\ell(n)$ , das einen statistischen Test  $\mathcal{D}$  mit  $\delta(n)$ -Toleranz nicht besteht. Dann existiert ein Prediktor  $\mathcal{P}_2^{\text{li}}$  zur Vorhersage des ersten Bits nach links mit Vorteil  $\frac{\delta(n)}{\ell(n)}$ .

**Beweis.** Übungsaufgabe. Sei  $\mathcal{P}^{\text{li}}$  ein Prediktor zur Vorhersage nach links.  $\mathcal{P}_2^{\text{li}}$  geht bei Eingabe  $\mathbf{x} \in \{0, 1\}^{\ell(n)}$  wie folgt vor:

- (1) Wähle zufälliges  $i \in_{\mathbb{R}} [2, \ell(n) + 1]$
- (2) Setze  $y_1 := \dots := y_{i-1} := 0$  und  $y_j := x_j$  für  $j = i, \dots, \ell(n)$ .
- (3) Gib  $\mathcal{P}^{\text{li}}(i, \mathbf{y})$  aus.

Da das Ensemble  $(X_n)_{n \in \mathbb{N}}$  shift-symmetrisch ist, hängt die Verteilung der Bits von  $\mathbf{y}$ , die  $\mathcal{P}^{\text{li}}$  liest, nicht von  $i$  ab.  $\square$

### 3. Allgemeine Konstruktion von Pseudozufallsgeneratoren

In diesem Abschnitt konstruieren wir (unter kryptographischen Annahmen) Pseudozufallsgeneratoren. Die Konstruktion beruht auf der allgemeinen Annahmen, dass sogenannte One-Way-Funktionen bzw. -Permutationen existieren.

Informell ist eine One-Way-Funktion eine effizient berechenbare Funktion, die im Durchschnitt schwierig zu invertieren ist.

**Definition 3.15** (One-Way-Funktion und -Permutation). Eine effizient berechenbare Funktion  $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$  heißt One-Way-Funktion, wenn für jeden probabilistischen Polynomialzeit-Algorithmus (PPTA)  $A$  die Wahrscheinlichkeit

$$\text{Ws} [A(1^n, f(x)) \in f^{-1}(f(x))]$$

(über die Wahl von  $x \in_{\mathbb{R}} \{0, 1\}^n$  und  $A$ 's interne Münzwürfe) vernachlässigbar ist. Gilt zusätzlich  $f(\{0, 1\}^n) = \{0, 1\}^n$  für alle  $n \in \mathbb{N}$ , dann heißt  $f$  One-Way-Permutation.

Warum geben wir  $A$  als zusätzliche Eingabe den Wert  $1^n$ ? Der Grund ist, dass sich die polynomielle Laufzeitbeschränkung von  $A$  auf die Eingabelänge bezieht. Folglich wäre bei Weglassen von  $1^n$  die Funktion

$$f(x) = \text{ersten } \log n \text{ Bits von } x \in \{0, 1\}^n$$

eine One-Way-Funktion (da das Hinschreiben eines Urbildes bereits exponentielle Zeit benötigt), obwohl intuitiv Urbilder leicht zu finden sind.

Während die One-Way-Funktion  $f$  in Definition 3.15 einen unendlichen Argumentbereich hat, sind die bisher kennengelernten Kandidaten wie die RSA-Funktion  $\text{RSA}_{N,e}(x) = x^e \bmod N$  und die Diskrete-Exponentiation

$\text{EXP}_{p,g}(x) = g^x \bmod p$  indizierte Mengen von Funktionen: Jede Funktion  $f_i$  für Index  $i$  hat nur einen endlichen Argumentbereich, der von  $i$  abhängen kann (beispielsweise  $\text{RSA}_{N,e} : \mathbb{Z}_N^* \rightarrow \mathbb{Z}_N^*$  oder  $\text{EXP}_{p,g} : \mathbb{Z}_{p-1} \rightarrow \langle g \rangle$ ). Die One-Way-Eigenschaft besagt in diesem Fall, dass die Wahrscheinlichkeit

$$\text{Ws} [A(1^n, i, f_i(x)) \in f_i^{-1}(f_i(x))]$$

über  $A$ 's Münzwürfe, die Wahl des Index  $i$  (für Sicherheitsparameter  $n$ , z.B. ein RSA-Modul der Bitlänge  $n$ ) und die Wahl von  $x$  uniform aus dem Argumentbereich von  $f_i$  vernachlässigbar ist. Wir verzichten hier allerdings auf eine weitere Betrachtung solcher Mengen von Funktionen und beschränken uns im folgenden auf den einfachen Fall einer Funktion  $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$  mit unendlichem Bereich.

Unser Ziel ist es, aus einer One-Way-Permutation einen Pseudozufallsgenerator zu bauen. Dazu zeigen wir, dass man aus dem Urbild  $x$  des Wertes  $f(x)$  einer One-Way-Permutation  $f$  ein pseudozufälliges Bit "erzeugen" kann. Dieses Bit soll nur mit Wahrscheinlichkeit ungefähr  $\frac{1}{2}$  aus  $f(x)$  vorhersagbar sein:

**Definition 3.16** (Hardcore-Prädikat). *Eine effizient berechenbare Funktion  $B : \{0, 1\}^* \rightarrow \{0, 1\}$  heißt Hardcore-Prädikat für die Funktion  $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$ , wenn für jeden probabilistischen Polynomialzeit-Algorithmus  $P$  die Wahrscheinlichkeit*

$$\text{Ws} [P(1^n, f(x)) = B(x)] - \frac{1}{2}$$

*vernachlässigbar ist, wobei die Wahrscheinlichkeit über die Wahl von  $x \in_R \{0, 1\}^n$  und  $P$ 's interne Münzwürfe gebildet wird.*

Die Definition ist resistent gegen "schlechte" Vorhersager, die noch unter der Ratewahrscheinlichkeit von  $\frac{1}{2}$  liegen: Angenommen, es gebe einen Algorithmus  $P$ , der das Hardcore-Prädikat mit Wahrscheinlichkeit  $\frac{1}{2} - \frac{1}{p(n)}$  für ein Polynom  $p(n)$  und unendliche viele  $n \in \mathbb{N}$  bestimmt. Dann sagt Algorithmus  $P'$ , der  $P$  simuliert und dann das Ausgabebit von  $P$  invertiert, das Hardcore-Prädikat mit Wahrscheinlichkeit  $\frac{1}{2} + \frac{1}{p(n)}$  unendlich oft vorher. Wir können uns daher im folgenden auf "gute" Vorhersager beschränken, die die Ratewahrscheinlichkeit  $\frac{1}{2}$  übertreffen.

Da man den Wert eines Hardcore-Prädikats quasi nur raten kann, erhalten wir aus einer One-Way-Permutation mit zugehörigem Hardcore-Prädikat einen Pseudozufallsgenerator vom Typ  $\ell(n) = n + 1$ :

**Satz 3.17.** *Sei  $f$  eine One-Way-Permutation und  $B$  ein Hardcore-Prädikat für  $f$ . Dann ist  $G$  definiert durch  $G(x) = (f(x), B(x))$  ein Pseudozufallsgenerator vom Typ  $\ell(n) = n + 1$ .*

**Beweis.** Der Beweis folgt unmittelbar aus der Charakterisierung von Generatoren mittels der Nicht-Vorhersagbarkeit nach rechts: Für  $i = 1, \dots, n$  kann kein Algorithmus aus den ersten  $i - 1$  Bits von  $f(x)$  das  $i$ -te Bit vorhersagen, da dieses Bit —unabhängig von den ersten  $i - 1$  Bits— uniform verteilt ist ( $f$  ist eine Permutation!). Das  $(n + 1)$ -te Bit ist aus den ersten  $n$  Bits nicht vorhersagbar, da es ein Hardcore-Prädikat darstellt. Ferner ist  $G$  effizient berechenbar, da  $f$  und  $B$  es sind.  $\square$

Unser Ziel ist daher die Konstruktion von Hardcore-Prädikaten. One-Way Funktionen sind zwar schwierig zu invertieren, dies besagt allerdings nicht, das einzelne Bits des Urbildes schwierig zu bestimmen sind. Der folgende Ansatz von Goldreich und Levin [GL89] verwendet eine zufällige Gewichtung über das Urbild in Form des inneren Produkts  $\langle x, r \rangle = \sum_i x_i r_i \bmod 2$ .

**Satz 3.18.** *Sei  $f$  eine One-Way-Funktion. Dann gilt für jeden probabilistischen Polynomialzeit-Algorithmus  $P$ , dass*

$$\text{Ws}[P(1^n, f(x), r) = \langle x, r \rangle] = \frac{1}{2}$$

(über  $x, r \in_R \{0, 1\}^n$  und  $P$ 's Zufallsbits) vernachlässigbar ist.

Wir können diese Konstruktion als Funktion  $g(x, r) = (f(x), r)$  auffassen. Offensichtlich ist  $g$  eine One-Way-Funktion, da das Invertieren von  $g$  das Invertieren von  $f$  bedingt. Ferner ist  $g$  genau dann eine One-Way-Permutation, wenn  $f$  diese Eigenschaft hat. Damit besagt Satz 3.18, dass jede One-Way-Funktion in eine Funktion mit Hardcore-Prädikat  $B(x, r) = \langle x, r \rangle$  abgewandelt werden kann. Man sagt in der Kryptographie vereinfachend, dass jede One-Way-Funktion ein Hardcore-Prädikat besitzt.

**Beweis.** Angenommen, es gebe einen PPTA  $P$ , der  $\langle x, r \rangle$  mit Wahrscheinlichkeit  $\frac{1}{2} + \frac{1}{p(n)}$  für ein Polynom und unendliche viele  $n \in \mathbb{N}$  vorhersagt. Wir konstruieren daraus einen effizienten Algorithmus  $A$ , der im Widerspruch zur One-Way-Eigenschaft die Funktion  $f$  mit nicht vernachlässigbarer Wahrscheinlichkeit invertiert. Es sei  $N \subseteq \mathbb{N}$  die Menge der  $n$ , für die  $P$  mit Wahrscheinlichkeit  $\frac{1}{2} + \frac{1}{p(n)}$  korrekt vorhersagt. Wir beschränken uns im folgenden auf diese  $n \in N$ .

Zur Erläuterung nehmen wir zunächst an, dass  $P$  mit Wahrscheinlichkeit 1 das innere Produkt vorhersagt. Um das  $i$ -te Bit des Urbildes eines Wertes zu bestimmen, genügt es in diesem Fall,  $P$  auf  $(1^n, f(x), r)$  und  $(1^n, f(x), r \oplus e_i)$  für  $e_i = (0, \dots, 0, 1, 0, \dots, 0)$  laufen zu lassen. Genau dann, wenn sich  $P$ 's Antworten unterscheiden, ist das  $i$ -te Bit 1, denn

$$\langle x, r \oplus e_i \rangle = \langle x, r \rangle \oplus \langle x, e_i \rangle = \langle x, r \rangle \oplus x_i$$

Auf diese Weise kann man sukzessive alle Bits von  $x$  bestimmen und invertiert damit  $f(x)$  erfolgreich.

Problem:  $P$  könnte beispielsweise für Werte  $r$  und  $r \oplus e_i$  genau einmal richtig und einmal falsch antworten bzw. beidesmal richtig, so dass wir  $x_i$  nicht bestimmen können. Die Lösungsidee ist, dass wir  $f(x)$  festhalten, und Algorithmus  $P$  auf mehreren Werten  $r_1, \dots, r_m$  für  $(1^n, f(x), r_j \oplus e_i)$  für alle  $i$  abfragen. Die Werte  $r_j$  werden wir zusätzlich so erzeugen, dass wir alle Werte  $\langle x, r_j \rangle$  kennen. Dann liefert uns  $P$  eine Folge von Bits, die —jeweils “ge-xor-t” mit  $\langle x, r_j \rangle$ — zum richtigen Wert  $x_i$  tendieren.

Wir zeigen zunächst, dass wir einen genügend großen Anteil der  $x$  fixieren können, ohne die Erfolgswahrscheinlichkeit von  $P$  wesentlich zu verringern:

BEHAUPTUNG: Für alle  $n \in \mathbb{N}$  existiert eine Menge  $S_n \subseteq \{0, 1\}^n$  der Grösse  $\frac{1}{2^{p(n)}} \cdot 2^n$ , so dass für alle  $x_0 \in S_n$  gilt:

$$s(x_0) := \text{Ws}[P(1^n, f(x_0), r) = \langle x_0, r \rangle] \geq \frac{1}{2} + \frac{1}{2^{p(n)}}$$

wobei die Wahrscheinlichkeit über  $r$  und  $P$ 's Zufallsbits gebildet wird.

BEWEIS: Es gilt  $\mathbb{E}_x[s(x)] = \sum_{x_0} s(x_0) \text{Ws}[x = x_0] \geq \frac{1}{2} + \frac{1}{p(n)}$ . Markovs Ungleichung für nicht-negative Zufallsvariablen

$$\text{Ws}[Z \geq c] < \frac{\mathbb{E}[Z]}{c}$$

ergibt daher

$$\begin{aligned} \text{Ws}_x \left[ s(x) \leq \frac{1}{2} + \frac{1}{2^{p(n)}} \right] &= \text{Ws}_x \left[ 1 - s(x) \geq \frac{1}{2} - \frac{1}{2^{p(n)}} \right] \\ &< \frac{\mathbb{E}_x[1 - s(x)]}{\frac{1}{2} - \frac{1}{2^{p(n)}}} \\ &\leq \frac{\frac{1}{2} - \frac{1}{p(n)}}{\frac{1}{2} - \frac{1}{2^{p(n)}}} \\ &\leq 1 - \frac{1}{2^{p(n)}} \end{aligned}$$

Dies zeigt die Behauptung, da  $x$  uniform in  $\{0, 1\}^n$  gewählt wird.  $\square$

Wir bedingen im folgenden, dass  $x \in S_n$ , d.h. die Vorhersagewahrscheinlichkeit über die Wahl von  $r$  und  $P$ 's Zufallsbits sei mindestens  $\frac{1}{2} + \frac{1}{2^{p(n)}}$ . Es bleibt zu zeigen, wie und wieviele Werte  $r_1, \dots, r_m$  wir generieren. Wir können i.a. die  $r_j$  nicht unabhängig und uniform wählen, da wir dann die inneren Produkte  $\langle x, r_j \rangle$  gegebenenfalls nicht erraten können. Wir erzeugen daher die  $r_i$  als paarweise unabhängige Werte aus einer kleineren Menge von Vektoren.

Sei  $\ell := \lceil \log(2n \cdot p(n)^2 + 1) \rceil$ . Algorithmus  $A$  wählt uniform und unabhängig  $s_1, \dots, s_\ell \in_R \{0, 1\}^n$  und  $c_1, \dots, c_\ell \in_R \{0, 1\}$  und berechnet für jede nicht-leere Teilmenge  $J \subseteq \{1, \dots, \ell\}$  den  $n$ -Bit-String  $r_J := \bigoplus_{j \in J} s_j$  sowie das Bit  $b_J := \bigoplus_{j \in J} c_j$ . Idee: wenn alle  $c_j$  richtig als  $c_j = \langle x, s_j \rangle$  geraten werden (was mit Wahrscheinlichkeit  $\frac{1}{2np(n)+1}$  geschieht), dann gilt auch  $\langle x, r_J \rangle = b_J$  für alle  $J$ .

Betrachte ein festes  $i \in \{1, \dots, n\}$ . Für jedes  $J$  berechnet Algorithmus  $A$  den Wert  $y_J = b_J \oplus P(1^n, f(x), r_J \oplus e_i)$  und rät das  $i$ -te Bit von  $x$  als

$$z_i = \text{MAJORITY}(\{y_J \mid J\}) = \begin{cases} 1 & \text{falls } \sum_J y_J \geq \frac{1}{2}(2^\ell - 1) \\ 0 & \text{sonst} \end{cases}$$

Algorithmus  $A$  gibt  $z = z_1 \dots z_n$  aus.

Bleibt zu zeigen, dass  $A$  mit genügend großer Wahrscheinlichkeit erfolgreich ist. Betrachte den Fall, dass  $n \in N$  und  $x \in S_n$ . Wir zeigen, dass dann für ein festes  $i$  die Mehrheitsentscheidung korrekt ist, gegeben dass die  $c_j$  richtig geraten wurden und somit  $b_J = \langle x, r_J \rangle$  gilt:

BEHAUPTUNG: Über die Wahl der  $s_1, \dots, s_\ell$  und  $P$ 's Zufallsbits gilt:

$$\text{Ws} \left[ \#\{J \mid \langle x, r_J \rangle \oplus P(1^n, f(x), r_J \oplus e_i) = x_i\} > \frac{1}{2}(2^\ell - 1) \right] > 1 - \frac{1}{2n}$$

BEWEIS: Offensichtlich ist jeder Wert  $r_J \oplus e_i \in \{0, 1\}^n$  uniform verteilt, und  $r_J \oplus e_i, r_{J'} \oplus e_i$  sind für  $J \neq J'$  paarweise unabhängig. Sei  $\xi_J$  die 0-1-Zufallsvariable, die genau dann 1 ist, wenn  $x_i = y_J$ . Aus Chebychevs Ungleichung für paarweise unabhängige 0-1-Zufallsvariablen  $Z_1, \dots, Z_m$  mit identischer Verteilung

$$\text{Ws} \left[ \left| \sum (Z_i - \mathbb{E}[Z_1]) \right| > m\delta \right] < \frac{\text{Var}[Z_1]}{m\delta^2}$$

folgt

$$\begin{aligned} \text{Ws} \left[ \sum \xi_J \leq \frac{1}{2}(2^\ell - 1) \right] &= \text{Ws} \left[ \sum \xi_J \leq \left( \frac{1}{2} + \frac{1}{2p(n)} \right) (2^\ell - 1) - \frac{1}{2p(n)}(2^\ell - 1) \right] \\ &\leq \text{Ws} \left[ \left| \sum \left( \xi_J - \left( \frac{1}{2} + \frac{1}{2p(n)} \right) \right) \right| \geq \frac{1}{2p(n)}(2^\ell - 1) \right] \\ &< \frac{\text{Var}[\xi_{\{1, \dots, \ell\}}]}{\left( \frac{1}{2p(n)} \right)^2 \cdot (2^\ell - 1)} \\ &\leq \frac{\frac{1}{4}}{\left( \frac{1}{2p(n)} \right)^2 \cdot (2np^2(n))} \\ &\leq \frac{1}{2n} \end{aligned}$$

wobei wir ausgenutzt haben, dass die Varianz einer 0-1-Zufallsvariablen maximal  $\frac{1}{4}$  ist.  $\square$

Mit der “Union Bound”  $\text{Ws}[\exists i : E_i] \leq \sum_i \text{Ws}[E_i]$  folgt, dass  $A$  mit Wahrscheinlichkeit mindestens  $\frac{1}{2}$  für alle  $i = 1, \dots, n$  korrekte Mehrheitsentscheidungen trifft. Insgesamt invertiert  $A$  daher  $f(x)$  mit Wahrscheinlichkeit mindestens

$$\frac{1}{2p(n)} \cdot \frac{1}{2^l} \cdot \frac{1}{2} \geq \frac{1}{8n \cdot p^3(n) + 1}$$

für unendlich viele  $n$ , nämlich wenn  $x \in S_n$ , wenn die  $c_j$  richtig geraten werden, und falls die Mehrheitsentscheidungen alle korrekt sind. Offensichtlich hat  $A$  polynomielle Laufzeit.  $\square$

Folglich ist  $B(x, r) = \langle x, r \rangle$  ein Hardcore-Prädikat für die One-Way-Funktion  $g(x, r) = (f(x), r)$ , d.h. wir können aus jeder One-Way-Permutation mit Satz 3.17 einen Pseudozufallsgenerator konstruieren. Mit dem Resultat über die Expansion der Ausgabelänge von Generatoren (Satz 3.7) folgt, dass man aus der One-Way-Permutation  $g$  einen sicheren Generator

$$G(x, r) = \langle f^{(\ell(n)-1)}(x), r \rangle, \langle f^{(\ell(n)-2)}(x), r \rangle, \dots, \langle x, r \rangle$$

vom Typ  $\ell(n)$  für jedes Polynom  $\ell(n)$  erhält. Dabei sei  $f^{(i)}(x) = f(\dots(f(x)))$  die  $i$ -fache Komposition von  $f$ . Dieser Satz wurde zunächst in einem speziellen Kontext von Blum und Micali [BM84] bewiesen:

**Satz 3.19** (Blum, Micali 1984). *Sei  $f$  eine One-Way-Permutation und  $B$  ein Hardcore-Prädikat für  $f$ . Sei  $\ell(n)$  ein Polynom. Dann ist  $G_{\ell(n)}$  mit*

$$G_{\ell(n)}(\mathbf{u}) := (x_1, \dots, x_{\ell(n)}) \in \{0, 1\}^{\ell(n)}$$

gegeben durch die Rekursion  $\mathbf{u}_{i+1} := f(\mathbf{u}_i) \in \{0, 1\}^n$ ,  $x_{i+1} := B(\mathbf{u}_i) \in \{0, 1\}$  mit  $\mathbf{u}_0 := \mathbf{u}$  ein Pseudozufallsgenerator vom Typ  $\ell(n)$ .

Der Generator  $(f(x), B(x))$  aus Satz 3.17 bzw. 3.19 benötigt eine One-Way-Permutation  $f$ . Allgemeiner kann man zeigen, dass man bereits aus jeder One-Way-Funktion einen Pseudozufallsgenerator konstruieren kann; die oben angegebene Hardcore-Prädikat-Konstruktion kann man nicht unmittelbar anwenden, da in diesem Fall der Wert  $f(x)$  nicht uniform in  $\{0, 1\}^n$  verteilt sein muss. Die Vorgehensweise beruht jedoch ebenfalls auf Hardcore-Prädikaten [HILL99].

**Fakt 3.20** (Håstad, Impagliazzo, Levin, Luby, 1999). *One-Way-Funktionen existieren genau dann, wenn es Pseudozufallsgeneratoren gibt.*



---

# Literaturverzeichnis

- [BBS86] L. BLUM, M. BLUM und M. SHUB: *A simple unpredictable Random Number Generator*, SIAM Journal on Computing, Band 15, Nr. 2, Seiten 364–383, 1986.
- [BM84] M. BLUM und S. MICALI: *How to generate cryptographically strong Sequences of Pseudo-Random Bits*, SIAM Journal on Computing, Band 13, Nr. 4, Seiten 850–864, 1984.
- [GL89] O. GOLDBREICH und L.A. LEVIN: *A Hard Core Predicate for any One Way Function*, Proceedings of the 21.st Symposium on Theory of Computing (STOC), ACM Press, New York, Seiten 25–32, 1989.
- [G98] O. GOLDBREICH: **Foundations of Cryptography**, Fragments of a Book, Version 2.03, 1998.
- [HILL99] J.HÅSTAD, R.IMPAGLIAZZO, L.LEVIN, M.LUBY: *Construction of a Pseudorandom Generator from any One-Way Function*, SIAM Journal on Computing, Vol. 28, Number 4, pp. 1364–1396, 1999.
- [Knuth97] D.E. KNUTH: **The Art of Computer Programming**, Band II: Seminumerical Algorithms, 3. Auflage, Addison Wesley, Reading, 1997.
- [Yao82] A. YAO: *Theory and Application of Trapdoor Functions*, 23.rd Annual IEEE Symposium on Foundation of Computer Science (FOCS), IEEE Computer Society Press, Los Alamitos, Seiten 80–91, 1982.