Theoretische Informatik

Martin Gropp

2005 – 2007

Teil I.

1. Eigenschaften von Relationen

- reflexiv $\forall a : aRa$
- irreflexiv $\forall a : \neg aRa$
- symmetrisch $\forall a, b : aRb \Rightarrow bRa$
- antisymmetrisch $\forall a, b : aRb \wedge bRa \Rightarrow a = b$
- asymmetrisch $\forall a, b : aRb \Rightarrow \neg bRa$
- transitiv $\forall a, b, c : aRb \wedge bRc \Rightarrow aRc$
- Äquivalenzrelation: reflexiv, transitiv, symmetrisch
- Ordnungsrelation: reflexiv, transitiv, antisymmetrisch
- strikte Ordnungsrelation: irreflexiv, transitiv

2. Eigenschaften von Graphen

- (stark) zusammenhängend
- Minimalgrad: $\delta(G) = \min\{d_g(v) \mid v \in V\}$
- Planarer Graph: $|E| \leq 3|V| - 6$
- Handshake-Lemma
  $\sum \deg(v) = 2 \cdot |E|$ (jede Kante „schüttelt“ zwei Knoten)
- Satz von Kuratowski: Ein endlicher Graph ist genau dann planar, wenn er keinen Teilgraphen enthält, der durch Erweiterung von K_5 oder $K_{3,3}$ entstanden ist. Erweiterung bedeutet hier das beliebig oft wiederholbare (auch nullmalige) Einfügen von neuen Knoten auf Kanten.

- Eulerscher Polyedersatz: Sei G ein zusammenhängender, ebener Graph mit v Knoten, e Kanten und s Flächen.

  $$s = e - v + 2$$

- Graph $G = (V, E)$ ist bipartit $\Leftrightarrow \exists V_1, V_2 : V = V_1 \dot\cup V_2$, so dass es keine Kanten innerhalb der Teilmengen gibt

3. Automaten

3.1. Mealy

Ausgabe an Zustandsübergänge gekoppelt.

$M = (Z, \Sigma, \Delta, \delta : Z \times \Sigma \to Z, \lambda : Z \times \Sigma \to \Delta, z_0 \in Z)$

Z Zustände, Σ Eingabemenge, Δ Ausgabemenge, δ Überführungsfunktion, λ Ausgabefunktion, z_0 Startzustand

3.2. Moore

Ausgabe an Zustände gekoppelt.

$M = (Z, \Sigma, \Delta, \delta : Z \times \Sigma \to Z, \lambda : Z \to \Delta, z_0 \in Z)$

Z Zustände Σ Eingabemenge, Δ Ausgabemenge, δ Überführungsfunktion, λ Ausgabefunktion, z_0 Startzustand

3.3. DEA

$M = (Z, \Sigma, \delta : Z \times \Sigma \to Z, z_0 \in Z, F \subseteq Z)$

Z Zustände, Σ Eingabemenge, δ Überführungsfunktion, z_0 Startzustand, F Endzustände

4. Reguläre Sprachen

4.1. Pumping-Lemma für reguläre Sprachen

Satz. *Sei $L \subseteq \Sigma^*$ eine reguläre Sprache. Dann gibt es eine Zahl n, so dass gilt:*
Für alle Wörter $x \in L$ mit $|x| \geq n$ gibt es Wörter $u, v, w \in \Sigma^*$, so dass:

- $x = uvw$
- $|v| \geq 1$
- $|uv| \leq n$
- *für alle $i \in \{0, 1, 2, 3, \dots\}$ gilt $uv^iw \in L$*

Also: Um zu zeigen, dass L nicht regulär ist, wähle $x \in L$ in Abhängigkeit von „freiem“ n und zeige, dass für jede Zerlegung $x = uvw$ $\exists i : uv^iw \notin L$.

4.2. Satz von Myhill-Nerode

Satz. $L \subseteq \Sigma^*$ *ist regulär* $\Leftrightarrow$

es gibt eine rechts-invariante Äquivalenzrelation R *mit endlichem Index, so dass* L *die Vereinigung einiger Äquivalenzklassen von* R *ist* $\Leftrightarrow$

der Index von R_L *ist endlich*

Beispiel: $L = \{a^n b^n \mid n \geq 1\}$ ist nicht regulär.
Äquivalenzklassen von R_L: $[ab] = L$, $[a^2b] = \{a, a^2b, a^3b^2, \dots\}$, $\dots$, $[a^k b], \dots$
$\Rightarrow$ Index nicht endlich $\Rightarrow$ L nicht regulär.

4.3. Sonstiges

Die regulären Sprachen sind unter *Vereinigung*, *Produkt*, *Stern*, *Komplement* und *Schnitt* abgeschlossen.

Falls $M = (Z, \Sigma, \delta, z_0, E)$ eine DEA ist, dann ist $M = (Z, \Sigma, \delta, z_0, Z \setminus E)$ ein DEA der die Komplementsprache erkennt.

4.4. Minimalautomaten

Definition. *Sei* $L \subseteq \Sigma^*$ *eine reguläre Sprache.*
Ein Minimalautomat ist ein Automat mit der geringsten Anzahl von Zuständen, der L *akzeptiert.*

Konstruktion:

- Stelle eine Tabelle für alle Paare von Zuständen $\{z, z'\}$ mit $z \neq z'$ auf.
- Markiere alle $\{z, z'\}$ mit $z \in F$ und $z' \notin F$.
- Markiere $\{z, z'\}$, falls es ein $a \in \Sigma$ gibt, so dass $\{\delta(z, a), \delta(z', a)\}$ markiert ist.
- Wenn keine weitere Markierung mehr möglich ist, kann man alle nicht markierten Paare zu einem Paar verschmelzen.

5. Chomsky-Hierarchie

Für alle Regeln $w_1 \to w_2$:

1. Kontextsensitiv
   $|w_1| \leq |w_2|$ (Ausnahme $S \to \varepsilon$)
2. Kontextfrei
   w_1 ist einzelne Variable
3. Regulär
   $w_2 \in \Sigma \cup \Sigma V$

Abschlusseigenschaften:

	$\cap$	$\cup$	$\setminus$	$\cdot$	$\star$	C	$\mathcal{R}$
Typ 3	•	•	•	•	•	•	•
Typ 2	$\circ^*$	•	–	•	•	–	•
Typ 1	•	•	•	•	•	•	•
Typ 0	•	•	–	•	•	–	•

$\mathcal{R}$: Reverse
$*$: mit regulären Sprachen

6. Sonstiges

Bell-Zahlen (Anzahl der Partitionen einer Menge):

$$B_{n+1} = \sum_{k=0}^{n} \binom{n}{k} \cdot B_{n-k}$$

$$m|n \;\Rightarrow\; \frac{n}{m}|n$$

Teil II.

7. Turing-Maschinen

$DTM = (Q, \Sigma, \Gamma, \square, q_0, F \subseteq Q, \delta : Q \times \Gamma \to Q \times \Gamma \times \{R, L, N\})$

Q Zustände, Σ Eingabealphabet, Γ Bandalphabet, $\square$ Blank-Symbol ($\in \Gamma, \notin \Sigma$), q_0 Startzustand, F akz. Endzustände, δ Überführungsfunktion

$NTM : \delta : Q \times \Gamma \to \mathcal{P}(Q \times \Gamma \times \{R, L, N\}$

8. Satz von Rice

$\mathcal{R} = \{f \mid f : \{0,1\}^* \to \{0,1\}^*$ ist berechenbar $\}$
$Prog(S) = \{\langle M \rangle \mid M$ berechnet eine Funktion $f \in S\}$

Satz. *Satz von Rice*
$S \subset \mathcal{R},\ \emptyset \neq S \neq \mathcal{R} \;\Rightarrow\; Prog(S)$ *nicht entscheidbar*

($Prog(\emptyset)$ und $Prog(\mathcal{R})$ sind durch Syntaxanalyse entscheidbar)

9. Wichtige NP-Probleme

- Erfüllbarkeit (3-...)
  SAT = {⟨Φ⟩ | Φ ist erfüllbare KNF }
- Clique
  CLIQUE = {⟨G, k⟩ | G ist ein Graph, der einen vollständigen Teilgraphen der Größe k enthält }
- Hamilton-Kreis
  HC = {⟨G⟩ | Graph G enthält Kreis, der jeden Knoten genau einmal enthält }
- Handlungsreisender
  TSP = {⟨G, c, k⟩ | c-gewichteter Graph G enthält einen HC mit Gewicht $\leq k$}
- 3-Färbbarkeit
  3COL = {⟨G⟩ | G ist drei-färbbar }
- Vertex Cover
  VC = {⟨G, k⟩ | G ist ein Graph, $k \in \mathbb{N}$, es gibt k Knoten in G, so dass jede Kante von G zu mind. einem der k Knoten inzident ist }
- Binary Programming
  BP = {⟨$A, \mathbf{b}$⟩ | A ist eine $m \times n$-Matrix mit ganzzahligen Einträgen, und es gibt einen 0-1-Vektor $\mathbf{x} \in \{0,1\}^n$ mit $A\mathbf{x} \leq \mathbf{b}$}

10. Kontextfreie Sprachen

10.1. Pumping-Lemma für kontextfreie Sprachen

Satz. *Sei $L \subseteq \Sigma^*$ eine kontextfreie Sprache. Dann gibt es eine Zahl n, so dass gilt: Für alle Wörter $z \in L$ mit $|z| \geq n$ gibt es Wörter $u, v, w, x, y \in \Sigma^*$, so dass:*

- $z = uvwxy$
- $|vx| \geq 1$
- $|vwx| \leq n$
- *für alle* $i \in \{0, 1, 2, 3, \dots\}$ *gilt* $uv^iwx^iy \in L$

Um L nicht kontextfrei zu zeigen: Wähle z und i, zeige $z_i \notin L$

10.2. Chomsky-Normalform

Nur $A \to a$, $A \to BC$ oder $S \to \varepsilon$.

Überführung einer Grammatik in Chomsky-Normalform:

- ε-Regeln $E_0 = \{A \mid (A \to \varepsilon) \in P\}$
  $E_i = \{A \mid A \to B_1 \dots B_k \wedge \forall j \in \{1 \dots k\} : B_j \in E_{i-1}$
  Alle ε-Regeln entfernen, für $A \to w$ alle Regeln einfügen, die durch Weglassen von Variablen aus E_{i_0} entstehen (außer $\to \varepsilon$).

- Kettenregeln $A \to B$ Graph mit Knoten ≙ Variablen, Kanten ≙ Kettenregeln:
  Ersetze in den starken Zusammenhangskomponenten die Variablen durch eine davon.
  Ersetze „von unten nach oben“ die Kettenregeln $A \to B$ durch $A \to$ alle rechten Seiten von B.
- Für alle $a \in \Sigma$: Regel $A_a \to a$ einführen und a durch A_a ersetzen (außer wenn dadurch neue Kettenregeln entstehen)
- Ersetze $A \to B_1 \dots B_k$ durch $A \to B_1C_1,\ C_1 \to B_2C_2,\ \dots\ C_{k-2} \to B_{k-1}B_k$

10.3. CYK-Algorithmus

Entscheidet für kontextfreie Grammatik G in Chomsky-Normalform in polynomieller Zeit, ob $w = w_1 \dots w_n \in L(G)$.

Gesucht: für alle $1 \leq i \leq j \leq n$: $V(i,j) = \{A \mid A \overset{*}{\to} w_i \dots w_j\}$

$V(i,i) = \{A \mid A \to w_i\}$
$V(i,j) = \{A \mid A \to BC,\ B \in V(i,k),\ C \in V(k+1,j),\ 1 \leq k \leq j-1\}$

10.4. Kellerautomat

$M = (Q, \Sigma, \Gamma, \delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to \mathcal{P}(Q \times \Gamma^*), q_0, Z_0, F)$

Q Zustände, Σ Eingabealphabet, Γ Kelleralphabet, δ Überführungsfunktion, q_0 Startzustand, Z_0 Kellergrundsymbol, F akzeptierende Endzustände

Teil III.

11. Verschiedene Formeln & Auswendiglernzeugs

Harm. Reihe:

$$\sum_{i=1}^{n} \frac{1}{i} \sim \log n$$

Geometr. Reihe:

$$\sum_{i=0}^{n} \alpha^i = \frac{1-\alpha^{n+1}}{1-\alpha} \quad (\alpha \neq 1;\ \text{konv. für } \alpha < 1)$$

Potenzsumme:

$$\sum_{i=1}^{n} i^k \sim n^{k+1}$$

Binomialkoeffizienten:

$$\binom{n}{k} = \frac{n!}{(n-k)!k!} \sim n^k$$

Stirling:

$$n! \approx \sqrt{2\pi n}(\frac{n}{e})^n$$

$$\rightsquigarrow \log n! \in \Theta(n \cdot \log n)$$

Catalan-Zahlen (Anzahl Binärbäume mit n inneren Knoten / mit $n+1$ Blättern):

$$c_n = \frac{1}{n+1}\binom{2n}{n} = \frac{(2n)!}{(n+1)!n!} \in \Theta(\frac{4^n}{n^{3/2}})$$

Noch ein paar:

$$\sum_{i=0}^{n}\binom{n}{i} = 2^n \quad \sum_{i=0}^{n} i\binom{n}{i} = n \cdot 2^{n-1} \quad \sum_{i=0}^{n} i \cdot 2^i = (n-1)2^{n+1} + 2 \quad \sum_{i=0}^{n} \log i = \log n! \in \Theta(n \log n)$$

Komplexitäten:

	best	avg	worst
Mergesort	$n \log n$	$n \log n$	$n \log n$
Heapsort	$n \log n$	$n \log n$	$n \log n$
Quicksort	$n \log n$	$n \log n$	n^2
Quickselect	n	n	n^2

Jedes (Vergl.-)Sortierverfahren (worst case): $n \log n - n \log e + O(\log n) \approx n \log n - 1.44n$

Schnelle Exponentiation a^n : $\lceil \log n \rceil + \#_1(n) - 2$ („Länge + $\#_1 - 2$“)

Satz. *Mastertheorem (Variante I)*

$$T(n) = \sum_{i=1}^{m} T(\alpha_i \cdot n) + \Theta(n^k)$$

$$T(n) \in \begin{cases} \Theta(n^k) & \sum \alpha_i^k < 1 \\ \Theta(n^k \cdot \log n) & \sum \alpha_i^k = 1 \\ \Theta(n^c) & \sum \alpha_i^k > 1 \\ & \textit{mit } c : \sum \alpha_i^c = 1 \quad (bzw. n^{\log_b a}) \end{cases}$$

Satz. *Mastertheorem (Variante II)*

$$T(n) = a \cdot T(\frac{n}{b}) + f(n), \quad \textit{mit } f(n) = n^k \cdot \log^i n$$

$$T(n) \in \begin{cases} \Theta(f(n)) & \log_b a < k \\ \Theta(f(n) \cdot \log n) & \log_b a = k \\ \Theta(n^{\log_b a}) & \log_b a > k \end{cases}$$

12. Multiplizieren

- Klassische Methode: $O(n^2)$
- Karatsuba: $x \cdot y$ mit $x = x_{2n} \dots x_1 = x_h \cdot b^n + x_l,\ y = y_{2n} \dots y_1 = y_h \cdot b^n + y_l$

  $$x \cdot y = (x_h \cdot b^n + x_l) \cdot (y_h \cdot b^n + y_l) =$$
  $$\underline{x_h y_h} \cdot b^{2n} + (x_h y_l + x_l y_h) \cdot b^n + \underline{x_l y_l}$$

  Mit $x_h y_l + x_l y_h = \underline{(x_h + x_l)(y_h + y_l)} - \underline{x_h y_h} - \underline{x_l y_l}$:

  $$x \cdot y = \underline{x_h y_h} \cdot b^{2n} + (\underline{(x_h + x_l)(y_h + y_l)} - \underline{x_h y_h} - \underline{x_l y_l}) \cdot b^n + x_l y_l$$

  $\Rightarrow$ 3 Multiplikationen!
- Schönhage-Strassen (FFT): $O(n \log n \log\log n)$

13. Binärbäume

$$e(t) = i(t) + 1$$
$$h(t) \le i(t)$$
$$e(t) \le 2^{\bar{h}(t)}$$

Innere Weglänge:

$$w_i(t) = \sum_{a \in I(t)} h(a,t) \quad \bar{w}_i(t) = \frac{w_i(t)}{i(t)}$$

Äußere Weglänge:

$$w_e(t) = \sum_{a \in E(t)} h(a,t) \quad \bar{w}_e(t) = \frac{w_e(t)}{e(t)}$$

14. Sortieren

14.1. Inversionen und Permutationen

Satz. (i,j) *ist Inversion von* $L = [L_1, L_2, \dots, L_n] \ \Leftrightarrow\ 1 \le i < j < n \wedge L_i > L_j$

Inversionen von $[101, 115, 30, 63, 47, 20]$:
$(1,3),(1,4),(1,5),(1,6),(2,3),(2,4),(2,5),(2,6),(3,6),(4,5),(4,6),(5,6)$

Satz. *Inversionsvektoren („Wie viele links davon sind größer?“)*

$$l_j(\sigma) = \#\{i < j \mid \sigma_i > \sigma_j\}$$
$$l(\sigma) = (l_1(\sigma), \dots, l_n(\sigma))$$

$l([4,2,1,3]) = (0,1,2,1)$

Satz. *Links-Rechts-Maxima*

$$lrm(\sigma) = \#\{1 \le i \le len(\sigma) \mid \forall j < i : \sigma[j] < \sigma[i]\}$$

$\mathrm{lrm}([\underline{33}, 12, \underline{57}, \underline{61}, 44, 28, 61, \underline{72}, 49, \underline{93}, 12, 66]) = 5$

Durchschn. Inversionen v. Perm.: $\frac{n(n-1)}{4}$

14.2. Sortieralgorithmen

- Selection Sort: wiederholt kleinstes „unsortiertes“ Element auswählen und anfügen

$$\sum_{i=0}^{n-1} i = \frac{n(n-1)}{2} \in \Theta(n^2)$$

- Insertion Sort: wiederholt nächstes „unsortierte“ Element an der richtigen Stelle einfügen

$\Theta(n^2)$

- Mergesort: Liste teilen, Teillisten rek. sortieren, lin. $(n_1 + n_2 - 1)$ Zusammenfügen

$$T(n) = 2 \cdot T(\frac{n}{2}) + \Theta(n) \quad \in \Theta(n \cdot \log n)\ (\alpha_1 = \alpha_2 = \frac{1}{2} \Rightarrow \sum \alpha_i = 1)$$

- Heapsort: Liste als Binärbaum/Max-Heap.

$\Theta(n \log n)$

- Quicksort: Pivot-Element p, Teillisten $L_1 < p$ und $L_2 > p$ rekursiv sortieren

avg: $\Theta(n \log n)$ worst: $\Theta(n^2)$

15. Codierung

(n, K, d)-Code: $C \subseteq \mathbb{B}^n,\ K = \#C,\ d = d_{min}(C)$
Coderate von C : $R(C) = \frac{1}{n} \cdot \log \#C$

15.1. Lineare Codes $[n, k, d]$

C ist Unterraum von $\mathbb{B}^n$.
Generatormatrix $k \times n$: $x \cdot G = c$
Kontrollmatrix $(n-k) \times n$: $H \cdot c^T = 0$
Generatormatrix G und Kontrollmatrix H können folgendermaßen umgeformt werden:

$$G' = [E|A] \quad H' = [A^T|E]$$

H ist Kontrollmatrix zu G gdw

$$H \cdot G^T = 0 \ \wedge\ H \text{ hat Rang } n-k$$

Satz. *Syndrom*

$$\mathbf{s}^T = H \cdot \mathbf{b}^T = H \cdot \mathbf{c}^T \oplus H \cdot \mathbf{f}^T = H \cdot \mathbf{f}^T$$

(Codevektoren haben Syndrom $\mathbf{0}^T$).

Syndromtabelle:

- Nebenklassen $C + v_i$ mit $1 < \|v_i\| <= \lfloor d/2 \rfloor$ korrigierbar
- Restliche Syndrome: nicht korrigierbare Fehler

Satz. *Sphere Packing Bound ($[n, k]$-Code, der t Fehler korrigiert)*

$$\sum_{i=0}^{t} (q-1)^i \leq q^{n-k}$$

(Perfekter Code: =; Spezialfall $q = 2, t = 1:\ 2^{n-k} = 1 + n$)

Satz. *Ungleichung von Kraft (r: Größe Alphabet ($\hookrightarrow$ r-ary tree), l_i: Wortlängen)*

$$\sum_{i=1}^{q} r^{-l_i} \leq 1 \quad \textit{(Baum!)}$$

Satz. *Entropie*

$$H_r(\mathcal{S}) = \sum_{i=1}^{q} p_i \cdot \log(\frac{1}{p_i})$$

$$H_r(p) = -p \log p - \bar{p} \log \bar{p}$$

Satz. *Shannon's First Theorem (Source S^n with avg. word length L_n/n)*

$$H_r(\mathcal{S}^n) \leq L_n \leq 1 + H_r(\mathcal{S}^n) \ \Leftrightarrow$$

$$H_r(\mathcal{S}) \leq \frac{L_n}{n} \leq \frac{1}{n} + H_r(\mathcal{S})$$

$$\lim_{n \to \infty} \frac{L_n}{n} = H_r(\mathcal{S})$$

„By encoding $\mathcal{S}^n$ with n sufficiently large, one can find uniquely decodable r-ary encodings of a source $\mathcal{S}$ with average word-lengths arbitrarily close to the entropy $H_r(\mathcal{S})$.“

Strehl ($\bar{h}$ mittlere Länge):

$$H(p) \leq \bar{h}(t, p) < H(p) + 1$$

Satz. *Mehrere unabhängige Quellen $\mathcal{S}_i$*

$$H_r(\mathcal{S}_1 \times \cdots \times \mathcal{S}_n) = H_r(\mathcal{S}_1) + \cdots + H_r(\mathcal{S}_n)$$

16. Modulare Arithmetik

16.1. Invertierbare Elemente / Einheiten

Definition. *Invertierbarkeit*
a ist invertierbar in $\mathbb{Z}_n$ $\Leftrightarrow\ \exists b : a \cdot_n b \equiv 1 \mod n$

(b: Bezout! $a \cdot b + n \cdot c = 1$)

Definition. *Menge der invertierbaren Elemente*
$U^n = \mathbb{Z}_n^* = \{a \in \{1, 2, ...n-1\} \mid ggT(a, n) = 1\}$

16.2. φ-Funktion

Definition. $\varphi(n)$
„die Anzahl der zu n teilerfremden natürlichen Zahlen, die nicht größer als n sind"
$\varphi(n) := \#\mathbb{Z}_n^*$

$n = \sum_{d|n} \varphi(d)$

p prim : $\varphi(p^e) = p^{e-1}(p-1)$ $(e > 0)$

Für Primfaktoren $p_i^{e_i}$ von n:

$$\varphi(n) = \prod_{i=1}^{k} p_i^{e_i-1}(p_i - 1) = n \cdot \prod_{p|n,\ p\ prim} (1 - \tfrac{1}{p})$$

Euler-Fermat: $a^{\varphi(n)} \equiv 1 \mod n$, falls a, n teilerfremd

16.3. Ordnung

Definition. *Ordnung eines Elements*
$ord_g(a) = \min\{n \mid a^n \equiv 1 \mod g\}$

Die Ordnung eines Elements teilt die Gruppenordnung ($\varphi(n)$)!

Berechnung der Ordnung in isomorpher Struktur ($\mathbb{Z}_a \times \mathbb{Z}_b \cong \mathbb{Z}_{a\cdot b}$ mit $a, b, \cdots \in \mathbb{P}$):

$$\text{ord}_a(x) = p,\ \text{ord}_b(x) = q \quad \Rightarrow \quad \text{ord}_{a\cdot b}(x) = \text{kgV}(p, q)$$

Rechenregeln:
$a^k = a^{k \bmod ord_n(a)}$
$a^{ord_n(a)} \equiv 1 \mod n$
$a^k = a^{k \bmod \varphi(n)}$

Definition. *Zyklische Gruppe*
G ist zykl. Gruppe $\Leftrightarrow \exists a \in G : G = \{a^k \mid k \in \mathbb{Z}\}$

16.4. EEA, CRT etc.

$$z \equiv a \mod x$$
$$z \equiv b \mod y$$

Kongruenz lösbar $\Leftrightarrow a \equiv b \mod \text{ggT}(x, y)$

Satz. *Binärer Euklidischer Algorithmus*

$ggT(2a, b) = ggT(a, b)$
$ggT(2a, 2b) = 2ggT(a, b)$
$ggT(a, b) = ggT(a - b, b)$ (*falls* $a > b$)

17. Primzahlen

$2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101$
$\pi(n) = \#\{p \le n \mid p \text{ Primzahl}\} \sim \frac{n}{\ln n}$

17.1. Primzahltests

- Teilbarkeit: n prim $\Rightarrow a \not| n$
- Euklid: n prim $\Rightarrow$ ggT$(n, a) = 1$
- Fermat: n prim $\Rightarrow a^{n-1} \equiv 1 \mod n$ (ord$(n) = n - 1$!)
- Miller-Rabin-Test
  - $t, u : 2^t \cdot u = N - 1$, u ungerade
  - Wähle $a \in \mathbb{Z}_N$ mit ggT$(a, N) = 1$
  - Wiederholtes Quadrieren: $a^u \rightsquigarrow a^{2u} \rightsquigarrow \cdots \rightsquigarrow a^{2^t u} = a^{N-1}$ (schnelle Exponentiation!)

  N ist *keine* Primzahl, wenn:
  - die 1 nicht in der Folge auftaucht, *oder*
  - die 1 auftaucht, aber mit Vorgänger -1
- Lucas-Kriterium

$$n \in \mathbb{P} \Leftrightarrow \exists a \in \mathbb{Z}_n^* : \text{ord}_n(a) = n - 1$$
$$\Leftrightarrow a^{n-1} \equiv 1 \mod n \ \wedge\ \forall p : (p \in \mathbb{P} \wedge p|(n-1)) \rightarrow a^{(n-1)/p_i} \not\equiv 1 \mod n$$

  Primzahlzertifikat:
  - Angabe einer Faktorisierung $p - 1 = p_1^{\alpha_1} \cdot \cdots \cdot p_n^{\alpha_n}$
  - Zertifikate für die Primzahlen p_i
  - Angabe einer positiven Zahl $a < n$ entspr. Lucas-Kriterium:

$$a^{n-1} \equiv 1 \mod n \ \wedge\ \forall p_i : a^{(n-1)/p} \not\equiv 1 \mod n$$

18. Public-Key-Kryptographie

18.1. RSA

- wähle zwei Primzahlen p, q
- berechne $n = p \cdot q$ und $\varphi(n)$ $(= (p-1)(q-1))$
- wähle eine zufällige ungerade Zahl d mit $1 < d < \varphi(n)$ und ggT$(\varphi(n), d) = 1$
- berechne $e = d^{-1} \mod \varphi(n)$ $(d \cdot e \equiv 1 \mod \varphi(n) \rightsquigarrow$ eea$(\varphi(n), d))$
- öffentlicher Schlüssel: (e, n)
- privater Schlüssel: (d, n)
- Verschlüsseln: $C = \mathcal{N}^e \mod n$
- Entschlüsseln: $\mathcal{N} = C^d \mod n$

18.2. Sonstige

Diffie-Hellman (Key Exchange), Shamir, ElGamal