FH Wedel

Die Struktur der Menge der Primzahlen

Nikolas Rajkovic
102129
Informatik

Inhaltsverzeichnis

Abbildungsverzeichnis ..... *II*

Einleitung *1*

Primzahlen allgemein *1*

Primzahlsatz 2

Satz von Euklid 2

Primzahlzwillinge 2

Berechnungen von πx für große x-Werte 4

Muster und Verfahren zur Berechnung einer Primzahl *4*

Vollkommene Zahlen **4**

Mersennesche Primzahlen **5**

Fermatsche Primzahlen **5**

Grenzen der Faktorisierung und Primzahlerkennung **5**

Kongruenz 6

Kleiner Fermatscher Satz **6**

Fermat-Test **6**

Russische Bauernmethode 7

Performance-Vergleich mit Faktorisierung **8**

Primzahlen in der Praxis: Public Key Crypto-System *9*

RSA-Kryptosystem **10**

RSA-Kryptosystem Beispiel **11**

Fazit *12*

Literaturverzeichnis ..... *III*

Abbildungsverzeichnis

Abbildung 1: Anzahl der Primzahlen im Bereich 1 bis 50 000 3

Abbildung 2: Differenz zwischen $\boldsymbol{\pi}(\boldsymbol{x})$ und glatten Approximationen von Legendre, Gauß und Riemann. 3

Abbildung 3: Verschlüsselung mit öffentlichem Schlüssel und Entschlüsselung mit privatem Schlüssel 10

Einleitung

In dieser Ausarbeitung beschäftige ich mich mit dem Thema Primzahlen, Muster der Primzahlen, Verfahren zur Berechnung einer Primzahl und einem praktischen Anwendungsfall von Primzahlen, dem sogenannten RSA-Kryptosystem.

Als Grundlage für diese Ausarbeitung dient der Artikel „Primzahlen: Theorie und Anwendung“ von Don Zagier.

Primzahlen allgemein

Eine natürliche Zahl p heißt prim bzw. Primzahl, wenn sie genau zwei natürliche Teiler hat (nämlich 1 und p).

Die Primzahlen 2,3,5,7,… sind eins der ältesten Themen, welches von Mathematikern studiert wird. Wir haben viele Intuitionen über das Verhalten der Primzahlen und viele als „sicher“ geltende Vermutungen über die Primzahlen. Dennoch haben wir große Probleme solche Thesen zu beweisen. Letztendlich liegt das daran, dass man glaubt, Primzahlen verhalten sich „pseudo-zufällig“ und folgen keinem einfachen Muster. Es gibt viele Möglichkeiten zu zeigen, dass ein Muster existiert… aber wie zeigt man das Fehlen eines Musters?

Die Theorie der Primzahlen ist voll von Paradoxen. Don Zagier nennt in seinem Artikel vier vermeintliche Widersprüche:

1. Die Primzahlen besitzen in hohem Maße die Eigenschaft der Gesetzmäßigkeit, gleichzeitig aber und genauso stark die der Willkür.
2. Man kann leicht erkennen, ob eine gegebene große Zahl prim ist, dies aber schwer beweisen.
3. Es ist möglich festzustellen, ob eine gegebene Zahl prim ist oder nicht, ohne ihre Faktoren zu kennen.

4. Man kann die Primzahlen unter einer gegebenen Grenze x zählen, ohne die einzelnen zu kennen.

Im Folgenden möchte ich mich vor allem mit den Punkten 2 bis 4 beschäftigen.

Primzahlsatz

Es sei $\pi(x) \coloneqq \{Die\ Anzahl\ der\ Primzahlen\ p\ mit\ p \leq n\}$

Dann gilt $\pi(n) \sim \frac{n}{\log n}$

~ bedeutet in diesem Kontext, dass der Limes des Quotienten der beiden Seiten für $n \to \infty$ gegen 1 geht.

→ eine zufällige k-stellige Zahl ist etwa mit der Wahrscheinlichkeit $\frac{1}{k \log 10}$ prim.

Satz von Euklid

Es gibt unendlich viele Primzahlen.

Primzahlzwillinge

Eine hier zu erwähnende besondere Struktur der Primzahlen sind die sogenannten „Primzahlzwillinge“. Primzahlzwillinge sind Paare der Form (p, p+2), wobei beide prim sind.

Beispiele sind 3 und 5, 5 und 7, …

Man kann ebenso zeigen, dass es unendlich viele Primzahlzwillinge gibt.

Die folgenden zwei Abbildungen sollen den von Zagier erwähnten ersten Punkt verdeutlichen. Abb 1. zeigt das linear wirkende Wachstum der Funktion $\pi(x) = Anzahl\ der\ Primzahlen$ unterhalb einer Zahl x. Abb 2. zeigt die Abweichung von $\pi(x)$ zu glatten Approximationen von Legendre, Gauss und Riemann im Bereich bis 10 Millionen. Als Beispiel lautet die Approximation von Gauss $\pi(x) \approx \frac{x}{\log x - 1}$,

welche sogar um ein vielfaches genauer ist, als der sogenannte Primzahlsatz $\pi(x) \approx \frac{x}{\log x}$ (dieser hat eine Abweichung von 44158 bei x= 10 000 000).

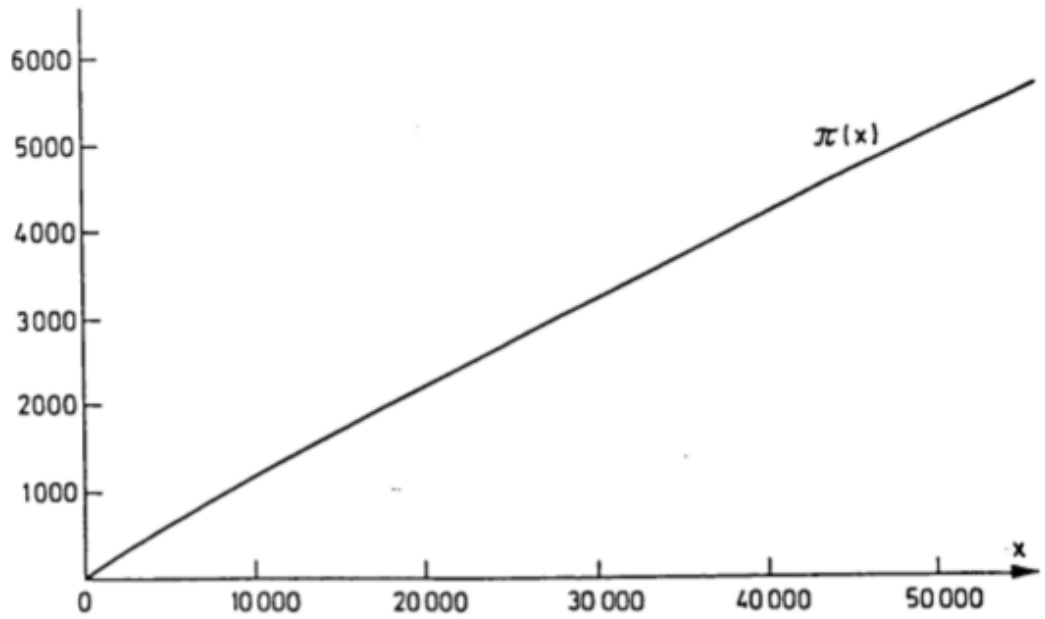

Fig. 1. $\pi(x)$, die Anzahl der Primzahlen $\leq x$, im Bereich $1 \leq x \leq 50\,000$

Abbildung 1: Anzahl der Primzahlen im Bereich 1 bis 50 000

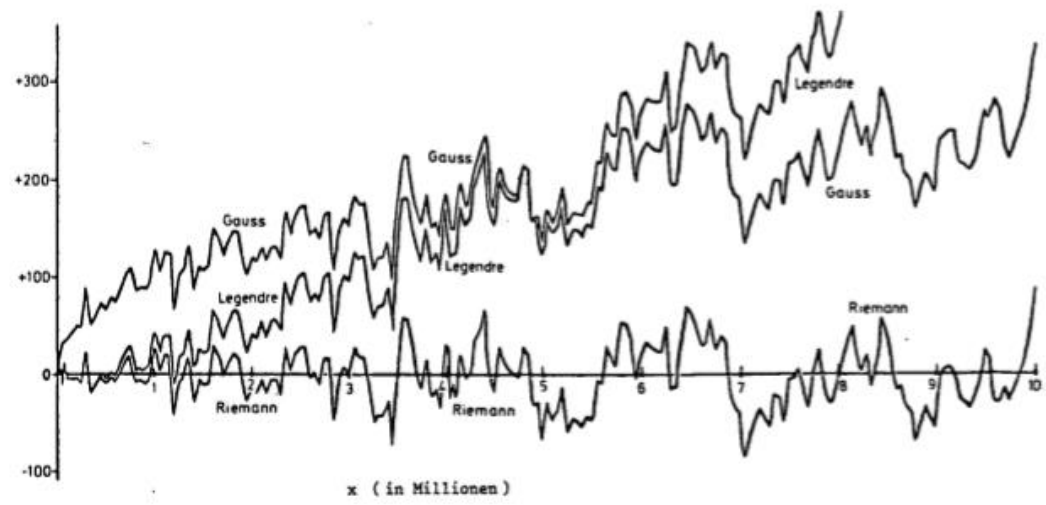

Abbildung 2: Differenz zwischen $\pi(x)$ und glatten Approximationen von Legendre, Gauß und Riemann.

Berechnungen von $\pi(x)$ für große x-Werte

Um den 4. „Widerspruch“ zu illustrieren, möchte ich einige historische Versuche der Berechnung von $\pi(x)$ für große Werte von x, ohne alle Primzahlen einzeln zu bestimmen, erwähnen.

1885 hat Meißel eine erste solche Methode entwickelt, mit der er den Wert $\pi(1\,000\,000\,000) = 50847478$ errechnete. Wie man heute weiß, hat er sich allerdings vertan und es gibt tatsächlich 50847534 Primzahlen unter einer Milliarde.

1958 berechnete Lehmer mit einer weiterentwickelten Variante der Meißelschen Methode den Wert $\pi(10\,000\,000\,000) = 455052512$. Er hat sich allerdings auch vertan und es gibt nur 455052511 Primzahlen unter zehn Milliarden.

1985 haben schließlich Lagarias, Miller und Odlyzko mit Hilfe einer neuen Methode und einer sehr großen Rechenanlage den Wert $\pi(40\,000\,000\,000\,000\,000) = 1.075.292.778.753.150$ errechnet. Ob dieser Wert nun genau ist, ist bis heute unbekannt.

Muster und Verfahren zur Berechnung einer Primzahl

In dem Hauptteil dieser Ausarbeitung möchte ich nun einige Primzahltypen vorstellen, die in der Vergangenheit eine große Rolle gespielt haben. Außerdem möchte ich mich den Behauptungen 2 und 3 von Zagier befassen, also mit der Frage, wie man für eine gegebene große Zahl herausfinden kann, ob es sich um eine Primzahl handelt, und dies ohne ihre Faktoren zu kennen.

Vollkommene Zahlen

Eine vollkommene Zahl ist eine Zahl, deren Summe ihrer echten Teiler entspricht.
Ein Beispiel wäre die Zahl 28, da 28 = 1+2+4+7+14.
Vollkommene Zahlen waren vor allem in der Antike wegen ihrer vermeintlichen mystischen Eigenschaften sehr beliebt.

Euklid zeigte, dass die Zahl $2^{n-1}p$, insofern $p = 2^n - 1$ prim ist, stets vollkommen ist.

Mersennesche Primzahlen

Mersennesche Primzahlen sind nach dem französischen Priester Mersenne benannt und haben die Gestalt $2^n - 1$.

Mersenne behauptete 1644, dass die Zahl $2^n - 1$ für $n = 2,3,5,7,11,13,17,19,31,67,127\ und\ 257$ prim ist.

Seine Liste war allerdings nicht vollständig bzw. hatte ein paar Fehler:

- 67 sollte 61 heißen
- 89 und 107 fehlen
- Die Zahl $2^{257} - 1$ ist nicht prim

Bis heute sind 50 Mersennesche Primzahlen bekannt.

Fermatsche Primzahlen

1650 behauptete Fermat, dass die Zahl 2^{2^k} immer prim ist. ($2^1 + 1 = 3, 2^2 + 1 = 5, 2^4 + 1 = 17, 2^8 + 1 = 257, 2^{16} + 1 = 65537$)

Seine Behauptung war allerdings recht unglücklich, da schon für k=5 die Zahl $2^{32} + 1$ den kleinen Faktor 641 hat, wie Euler bemerkte.

Grenzen der Faktorisierung und Primzahlerkennung

1876 bewies E. Lucas, dass $2^{127} - 1$ prim ist.

Diese 39-stellige Zahl blieb 76 Jahre lang die größte bekannte Primzahl. Erst 1952 mit der Einführung der elektronischen Rechenmaschinen wurde sie durch die 678-stellige Zahl $2^{2281} - 1$ übertroffen.

Der Weltrekord stieg mehrmals, 1971 zum Beispiel wurde die Zahl $2^{19937} - 1$(6.002 Ziffern) gefunden.

Am 26.12.2017 wurde die 50. Bekannte Mersenne-Primzahl $2^{77.232.917} - 1$ (23.249.425 Ziffern) gefunden.

In entgegengesetzter Richtung hat man bewiesen, dass die 20. Fermatsche Primzahl

F_{20} (315.653 Ziffern) **nicht** prim ist, allerdings ohne auch nur einen Faktor gefunden zu haben.

Aber wie kann man so große Zahlen nun eigentlich auf Primalität testen?

Dazu benutzen wir den Fermatschen Satz.
Um diesen aufstellen zu können, benötigen wir den Begriff der Kongruenz:

Kongruenz

Zwei Zahlen a und b sind kongruent modulo m, wenn sie nach Division durch m den selben Rest haben.

Bsp.: $87 \equiv 587\ modulo\ 100$.

Kleiner Fermatscher Satz

Ist p eine Primzahl und x eine beliebige nicht durch p teilbare Zahl, so ist die (p-1)-te Potenz von x modulo p zu eins kongruent.

Anders ausgedrückt:
ist p prim, dann gilt $x^{p-1} \equiv 1\ (mod\ p)$, für alle nicht durch p teilbaren Zahlen x.

Bsp.: $p = 5$: $x^4 \equiv 1\ mod\ 5$

$$1^4 = 1,\ 2^4 = 16, 3^4 = 81, 4^4 = 256$$

Fermat-Test

Eine Variante des Fermatschen Satzes lautet:

$$Ist\ p\ prim, dann\ ist\ die\ \frac{p-1}{2} - te\ Potenz$$
$$von\ x\ ist\ immer\ kongruent\ 1\ oder\ p-1\ modulo\ p.$$

$$x^{\frac{p-1}{2}} \equiv 1\ oder\ p-1\ mod\ p$$

→ so kann man den kleinen Fermatschen Satz nun als Test auf Primalität benutzen.

Wenn wir für gegebenes p eine nicht durch p teilbare Zahl x finden, für die die fermatsche Behauptung $x^{p-1} \equiv 1 \; mod \; p$ nicht gilt, kann p nicht prim gewesen sin.

Bsp.: Wir wollen prüfen ob die Zahl p=15 prim ist:

$$2^{14} = 16384 = 15 * 1092 + 4 \equiv 4 \; mod \; 15 \not\equiv 1 \; mod \; 15$$

→ 15 kann nicht prim sein.

Auf den ersten Blick sieht die Methode recht umständlich aus, da die Zahl x^{p-1} sehr groß sein wird.

Dazu benutzen wir die sogenannte „russische Bauernmethode“, welche eine solche Berechnung schnell durchführt.

Russische Bauernmethode

Dient der Multiplikation größerer Zahlen. Sukzessives Quadrieren und Berechnen der Reste. Ermittle die Kongruenzklassen der Zahlen $x^2, x^4 = (x^2)^2, x^8 = (x^4)^2, x^{16} = (x^8)^2, \ldots$ modulo p, und so dann x^{p-1} mit Hilfe der binären Darstellung von p-1 als Produkt von Zahlen x^{2^r} zu berechnen.

Beispiel: Wir wollen die Primalität von p=101 prüfen.

Dazu überprüfen wir, ob $2^{50} \equiv 1 \; oder \; 100 \; mod \; 101$:

$$2^1 = 2, 2^2 = 4, 2^4 = 4^2 = 16, 2^8 = 16^2 = 256 \equiv 54 \; mod \; 101,$$
$$2^{16} \equiv 54^2 \; mod \; 101, = 2916 \equiv 88 \; mod \; 101, 2^{32} \equiv 88^2 \; mod \; 101$$
$$= 7744 \equiv 68 \; mod \; 101$$

Und da 50 = 32+16+2:

$$2^{50} \equiv 2^{32} * 2^{16} * 2^2 \; mod \; 101 \equiv 68 * 88 * 4 \; mod \; 101 = 23936 \equiv 100 \; mod \; 101.$$

Auch wenn dies auf den ersten Blick aufwändig wirkt, nimmt der Aufwand bei größeren Zahlen nur sehr langsam zu.

So berechnet man für die 78-stellige Zahl $p = 2^{257} - 1 =$
231584178474632390847141970017375815706539969331281128078915168015826259279871

die Zahl $3^{\frac{p-1}{2}}$ mit nur ungefähr 500 Multiplikationen (auf dem Laptop etwa 8 Sekunden) und findet

$3^{\frac{p-1}{2}} \equiv$
$37912010207938437192741606119904315123014364869874654175066341228527222315865$
Modulo p

→ Da das Ergebnis weder 1 noch p-1 ist, haben wir bewiesen, dass p zusammengesetzt ist, ohne auch nur einen Faktor zu kennen.

Also:

Besteht eine Zahl den Fermat-Test nicht, ist sie nicht prim. Die Umkehrung gilt allerdings nicht. (In gewissen Fällen gilt partielle Umkehrung)

Bsp.: $2^{\frac{p-1}{2}}$ ist für $p = 2^{257} - 1$ kongruent 1 modulo p

Für Nicht-Primzahlen p schlägt $x^{\frac{p-1}{2}} \equiv 1\ oder\ p - 1$ mod p für mindestens die Hälfte aller x fehlt. Eine Ausnahme bilden die sogenannten „Pseudoprimzahlen“. Diese haben die Eigenschaft, dass sie den Fermattest für alle x bestehen, dennoch keine Primzahlen sind. Der Fermat-Test dient also vor allem dazu, die Primalität einer Zahl zu widerlegen. Man kann den Fermat-Test natürlich so oft mit zufällig gewählten Werten für x wiederholen, bis die zu untersuchende Zahl durchgefallen ist oder mit an Sicherheit grenzender Wahrscheinlichkeit als prim erkannt worden ist.

Performance-Vergleich mit Faktorisierung

Wie zuvor gezeigt ist es recht leicht, die Primalität einer sehr großen Zahl zu bestätigen oder zu widerlegen. Ist die Zahl allerdings zusammengesetzt, so können diese Faktoren nur sehr schwer ermittelt werden.

Wie im vorigen Beispiel demonstriert, lässt sich mit dem Fermat-Test innerhalb von 8 Sekunden auf einem Laptop überprüfen, ob eine 78-stellige Zahl prim ist.

Auf einem Großrechner sind so auch mehrere hundert Ziffern im Bruchteil einer Sekunde möglich.

Im Vergleich dazu stellt die folgende Tabelle approximativ die Rechenzeit zur Faktorisierung auf den größten, heute verfügbaren, Computern mit den schnellsten heute verfügbaren Methoden dar:

Anzahl der Ziffern	*50*	*70*	*90*	*100*
Typische Rechenzeit	1 Minute	7 Stunden	4 Tage	1 Monat

Wie man klar erkennen kann, existiert ein riesiger Performance-Unterschied zwischen dem Erkennen einer Primzahl und dem Faktorisieren einer Primzahl. Genau dieser Unterschied ist die Basis für die Verwendung von Primzahlen in der Kryptographie.

Primzahlen in der Praxis: Public Key Crypto-System

Ein Anwendungsfall von großen Primzahlen in der Praxis ist das sogenannte Public Key Crypto-System.

Jeder Benutzer A des Systems ist mit einem Verfahren ausgestattet, welches Nachrichten x in andere umwandelt. Dieses Verfahren ist öffentlich bekannt (öffentlicher Schlüssel), das heißt jeder Benutzer hat Zugang zu ihm. Ebenso gibt es ein Verfahren, welches die verschlüsselten Nachrichten zurück in x verwandelt. Dieses Verfahren ist allerdings nur dem Benutzer A bekannt (privater Schlüssel). Anderen Nutzern ist diese Rückwandlung also nicht, bzw. nur mit sehr großem

Aufwand möglich.

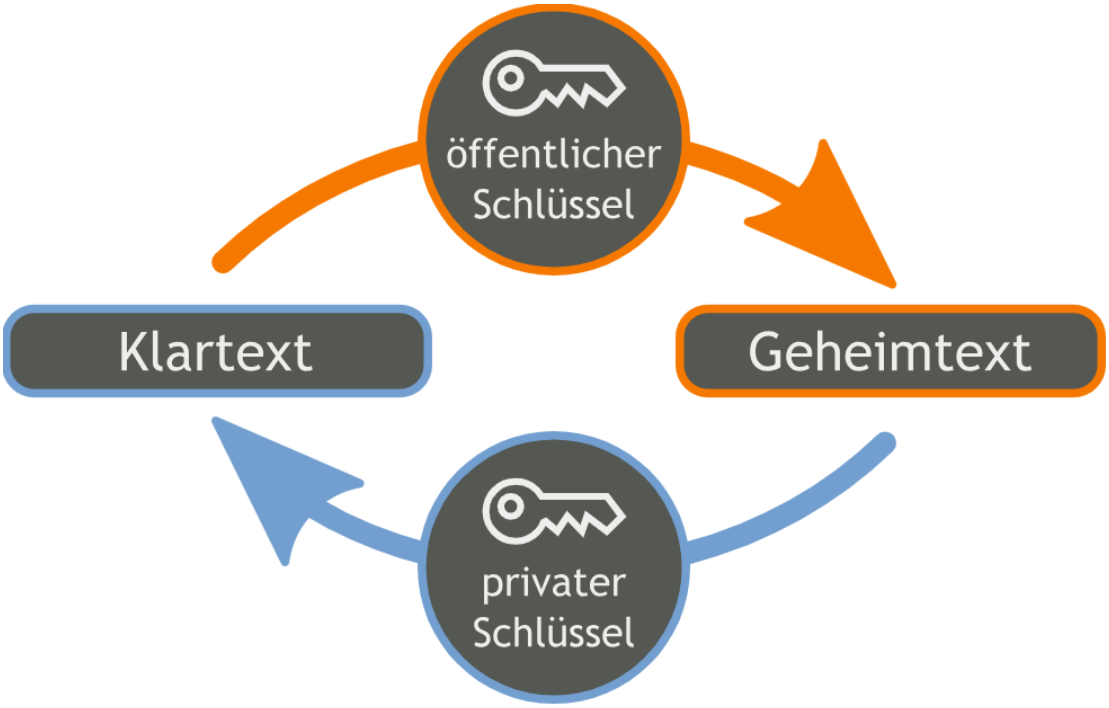

Abbildung 3: Verschlüsselung mit öffentlichem Schlüssel und Entschlüsselung mit privatem Schlüssel

RSA-Kryptosystem

Eine Variante des Public Key Crypto-Systems, welche Primzahlen als Schlüssel einsetzt, ist das sogenannte RSA-Kryptosystem. Der öffentliche Schlüssel und der private Schlüssel sind in diesem Fall jeweils ein Zahlenpaar (e,N) und (d,N). Erzeugt werden diese durch folgendes Verfahren:

- Wähle 2 Primzahlen $p \neq q$ gleicher Größenordnung
- Berechne RSA_Modul: N = $p * q$
- Berechne Eulersche φ-Funktion von N:

$$\varphi(N) = (p-1) * (q-1)$$

- Wähle eine zu $\varphi(N)$ teilerfremde Zahl e, für die gilt $1 < e < \varphi(N)$
- Berechne d als multiplikativ Inverses von Modulo $\varphi(N)$

Also $e * d \equiv 1 \; mod \; \varphi(N)$

Eine Nachricht x wird nun mit dem öffentlichen Schlüssel (e,N) verschlüsselt:

$$c \equiv x^e \; mod \; N.$$

Eine Nachricht c wird mit dem pri

Eine verschlüsselte Nachricht c wird mit dem privaten Schlüssel (d,N) entschlüsselt: $x \equiv c^d \; mod \; N$.

RSA-Kryptosystem Beispiel

- $p = 29 \; q = 31 \; \rightarrow N = 899$
- Berechne $\varphi(N) = (p-1) * (q-1) = 28 * 30 = 840$
- Wähle e teilerfremd zu 840: $e = 19$
- Berechne d mit $e * d \equiv 1 \; mod \; \varphi(N)$

  Bzw. $e * d + k * \varphi(N) = 1$

  → mit erweitertem euklidischem Algorithmus: $d = 619 \; k = -14$
- Damit haben wir den öffentlichen Schlüssel (19,899) und den privaten Schlüssel (619,899)
- Nachricht $x = 21$: Verschlüsseln mit $c \equiv x^e \; mod \; N$, also $21^{19} \; mod \; 899 = 230$
- Entschlüsseln der öffentlichen Nachricht $c = 230$:

$$x \equiv c^d \; mod \; N, also \; 230^{619} \; mod \; 899 = 21$$

Fazit

Wir haben einen kleinen Einblick in die Welt der Primzahlen gewinnen können. Wir haben festgestellt, dass es unendlich viele Primzahlen gibt, welche sowohl eine gewisse Gesetzmäßigkeit aufweisen, sich in vielen Fällen jedoch „pseudo-zufällig“ verhalten.

Wir haben verschiedene Arten von Primzahlen kennengelernt, wie die Mersennschen oder Fermatschen Primzahlen.

Es ist möglich extrem große Zahlen auf Primalität zu untersuchen, ohne ihre Faktoren zu kennen. Als eine von vielen Möglichkeiten haben wir dazu den Fermat-Satz kennengelernt. Der Fermat-Test ist zudem deutlich performanter als die Faktorisierung einer Zahl. Dieser gewaltige Unterschied zwischen der Bestimmung einer hohen Primzahl und der Faktorisierung einer großen Zahl, ist der Grund, weshalb sich Primzahlen sehr gut in der Kryptographie, wie zum Beispiel dem RSA-Kryptosystem, einsetzen lassen.

Literaturverzeichnis

Don Zagier: *Primzahlen: Theorie und Anwendung*, White paper, Max-Planck-Institut für Mathematik, Bonn.

Terence Tao: *Structure and randomness in the prim numbers*, University of California, Los Angeles.

Dr. Karin Halupczok(2013): *Struktur und Zufall in der Menge der Primzahlen*, LVM, Münster.

Wikipedia(2019), https://de.wikipedia.org/w/index.php?title=RSA-Kryptosystem&oldid=185894336, letzter Abruf am 27.02.2019.

Wikipedia(2019), https://de.wikipedia.org/w/index.php?title=Asymmetrisches_Kryptosystem&oldid=185639001, letzer Abruf am 27.02.2019.