Seminar der WE AℓZAGK

WiSe 2010/11

Glatte Zahlen

von Sonja Riedel

Mail: sriedel@math.uni-bremen.de

1 Motivation

Glatte Zahlen sind, grob gesagt, Zahlen, die nur kleine Primfaktoren besitzen. Sie werden in vielen zahlentheoretischen Algorithmen verwendet, da sie...

- ... eine einfache (multiplikative) Struktur haben,
- ... leicht zu erkennen sind,
- ... überraschend zahlreich sind.

Damit verbessern sie die Rechenzeit enorm. Besonders beim Finden von Quadraten in Faktorisierungsverfahren sind sie quasi unverzichtbar.

Ich werde in dieser Ausarbeitung erst ein paar Anwendungsbeispiele erläutern und mich dann mit dem Finden von glatten Zahlen und der Wahl geeigneter Schranken (für die Größe der Primfaktoren) beschäftigen. Beim Bestimmen der Schranke wirken zwei Kräfte gegeneinander. Bei einer kleinen Schranke ist zwar die Matrix, deren Zeilen sich durch die glatten Zahlen ergeben und mit deren Hilfe wir ein Quadrat bestimmen wollen, ebenso klein. Aber es zerfallen auch nur wenige Zahlen wie gewünscht und man muss viele Zahlen betrachten, um genügend viele glatte unter ihnen zu finden. Bei einer großen Schranke sind glatte Zahlen häufig und einfach zu finden. Aber man vergrößert, durch die größere Faktorbasis, auch das zu lösende Gleichungssystem und erhöht damit die Rechenzeit stark.

Es sei darauf hingewiesen, dass ich mich beim Erstellen meiner Ausarbeitung stark an den Texten von Pomerance [1] und Granville [2] orientiert habe.

2 Anwendungen

2.1 Definitionen

Eine Zahl n heißt *y-glatt*, wenn alle ihre Primfaktoren kleiner gleich y sind. Bezeichne $p_1 = 2 < p_2 = 3 < p_3 < ....$ die Folge der Primzahlen, dann wird k so gewählt, dass p_k die größte Primzahl kleiner gleich y ist. Also ist $k = \pi(y)$ die Anzahl der Primzahlen kleiner gleich y.

Sei $S(x, y) = \{n \in \mathbb{N} | \ n \leq x \text{ ist } y\text{-glatt}\}$ die Menge der y-glatten Zahlen kleiner gleich x und sei $\Psi(x, y) = \#S(x, y)$ die Anzahl der y-glatten Zahlen kleiner gleich x.

2.2 Quadratisches Sieb

Wir wollen eine Zahl $n \in \mathbb{N}$ faktorisieren. Ohne Beschränkung der Allgemeinheit nehmen wir an, dass n ungerade ist. Nun suchen wir nach $x, y \in \mathbb{N}$, so dass

$$x^2 - y^2 = (x + y)(x - y) = n. \tag{1}$$

Es ist leicht ersichtlich, dass sich jede ungerade Zahl als Differenz zweier Quadrate schreiben lässt (spätestens bei $k^2 - (k-1)^2 = 2k - 1$). Beginnend mit $x = \lceil n^{1/2} \rceil$ betrachtet man also die Funktion $f(x) = x^2 - n$, bis man ein Quadrat für $f(x)$ erhält. Soweit war das die ursprüngliche Faktorisierungsmethode von Fermat.
Nun hat man aber festgestellt, dass sich die Rechenzeit des Algorithmus noch deutlich verkürzen lässt. Dazu ein Beispiel:

Für $n = 1649$ gilt:

x	$f(x) = x^2 - n$	Primfaktorzerlegung von $f(x)$
41	32	2^5
42	115	$5 \cdot 2^3$
43	200	$2^3 \cdot 5^2$
...	...	...
57	1600	$5^2 \cdot 2^6$

Wir würden also mit der ursprünglichen Methode nach 16 Schritten ans Ziel gelangen. Nämlich

$$1649 = 57^2 - 40^2 = (57 + 40)(57 - 40) = 90 \cdot 17.$$

Wenn man aber genau hinsieht, stellt man fest, dass man schon nach den ersten drei Schritten ein Quadrat erhält. Nämlich

$$(41 \cdot 43)^2 \equiv 2^8 \cdot 5^2 \equiv (2^4 \cdot 5)^2 \pmod{n}.$$

Somit haben wir Lösungen für

$$x^2 \equiv y^2 \pmod{n}. \tag{2}$$

Dabei soll gelten, dass

$$x \not\equiv \pm y \pmod{n}. \tag{3}$$

Wobei (2) äquivalent dazu ist, dass n die Differenz der Quadrate von x und y teilt, und (3) dazu, dass n weder $x + y$ noch $x - y$ teilt. Sind (2) und (3) erfüllt, erhalten wir mit $1 < \mathrm{ggT}(n, x \pm y) < n$ zwei nicht-triviale Teiler von n.
In unserem Beispiel sind (2) und (3) erfüllt. Also sind

$$\begin{aligned} \mathrm{ggT}(41 \cdot 43 - 80, 1649) &= \mathrm{ggT}(34, 1649) = 17 \\ \text{und } \mathrm{ggT}(41 \cdot 43 + 80, 1649) &= \mathrm{ggT}(194, 1649) = 97 \end{aligned}$$

nicht-triviale Teiler, und wir erhalten damit unsere Zerlegung $1649 = 17 \cdot 97$. Zwar liefert nicht jede quadratische Kongruenz echte Teiler, aber im Schnitt liefert jede zweite eine echte Faktorisierung.

Kennt man die Primfaktorzerlegung der Zahlen $f(x)$, so kann man die Teilfolge, deren Produkt eine Quadratzahl ist, leicht erkennen. Man muss lediglich die Exponenten mod 2 betrachten. Dies ist genau der Punkt, an dem glatte Zahlen ins Spiel kommen, beim Bestimmen der Teilfolge. Unsere Überlegung dazu ist, dass Zahlen mit kleinen Primfaktoren wahrscheinlicher in dieser Teilfolge enthalten sind.
Wählen wir nun als Schranke y, wobei $k = \pi(y) = \#\{p \leq y \mid p \text{ prim}\}$ die Anzahl der Primzahlen kleiner gleich y ist. Nun betrachten wir die Matrix, deren Spalten aus den Exponenten-Vektoren mod 2 unserer y-glatten Zahlen bestehen. Dann ist es unser Ziel, durch Linearombination der Zeilen die Nullzeile zu erhalten. Spätestens bei $k+1$ y-glatten Zahlen ist dies definitiv möglich. Durch Gauß-Elimination erhalten wir dann die gewünschte Teilfolge. Liefert die Teilfolge triviale Teiler, muss man den letzten Schritt wiederholen und eine neue Teilfolge suchen.

2.3 Test von Lenstra

Mithilfe des Algorithmus von Lenstra können wir herausfinden, ob eine vermutliche Primzahl keine Quadrate enthält. Und zwar innerhalb *polynomial time*, d.h. die Rechenzeit wächst höchstens polynomial, gerechnet in der Bitlänge der Eingangszahl.

Behauptung

Gilt für ein $n \in \mathbb{N}$, dass $a^{n-1} \equiv 1 \pmod{n}$ für alle $a < \log^2 n$, dann ist n quadratfrei.

Beweis

Indirekt: Sei p eine ungerade Primzahl mit p^2 teilt n.
Nun gilt für alle $a < 4\log^2 p$ ($\leq \log^2 n$), dass

$$a^{n-1} \equiv 1 \equiv a^{p(p-1)} \pmod{p^2} \tag{4}$$

(wobei die erste Kongruenz nach Voraussetzung gilt und die zweite, da
$\mathbb{Z}_{p^2}^* = \mathbb{Z}_{p^2} - \{0, p, ..., (p-1)p\}$, somit gilt also $|\mathbb{Z}_{p^2}^*| = p(p-1)$).
Da $\text{ggT}(p, n-1) = 1$ gilt, existieren $l, k \in \mathbb{Z}$ mit $kp + l(n-1) = 1$. Also

$$a^{p-1} = (a^{p-1})^{l(n-1)+kp} = (a^{n-1})^{l(p-1)} \cdot (a^{p(p-1)})^k \overset{(4)}{\equiv} 1 \pmod{p^2}. \tag{5}$$

Mit (5) gilt also für alle $(4\log^2 p)$-glatten Zahlen m, dass $m^{p-1} \equiv 1 \pmod{p^2}$. Da $\mathbb{Z}_{p^2}^*$ zyklisch ist, existiert mit einem Erzeuger b von $\mathbb{Z}_{p^2}^*$ die Darstellung $m = b^c$ mit

$c \in \{1, ..., p(p-1)\}$ für $m \in \mathbb{Z}_{p^2}^*$. Dann ist $m^{p-1} = b^{c(p-1)} \equiv 1 \pmod{p^2}$ äquivalent dazu, dass p die Zahl c teilt. Also kann c (und damit auch m) höchstens $p-1$ unterschiedliche Werte annehmen. Also erhalten wir mit $p-1$ eine obere Schranke für $\Psi(p^2, 4\log^2 p)$ (Menge der $(4\log^2 p)$-glatten Zahlen kleiner gleich p^2).
Eine untere Schranke erhalten wir mithilfe der Abschätzung (vergleiche [2])

$$\Psi(x, \log^2 x) \geq \sqrt{x}. \tag{6}$$

Mit (6) gilt auch

$$\Psi(p^2, 4\log^2 p) = \Psi(p^2, \log^2 p^2) \overset{(6)}{\geq} p.$$

Also ist $p - 1 \geq \Psi(p^2, 4\log^2 p) \geq p$, und dies ist ein Widerspruch. □

3 Umgang mit glatten Zahlen

3.1 Finden der glatten Zahlen

(i) Die einfachste Methode, um y-glatte Zahlen bis zu einer Grenze x zu finden, ist es wohl, das Sieb des Eratosthenes zu nutzen. Das heißt den einfachsten Primzahltest, bei dem man die Zahlen von 2 bis x betrachtet und dann anfangs alle echten Vielfachen von 2 streicht, dann die nächste unmarkierte Zahl betrachtet und deren Vielfache streicht und so weiter (bis man bei $\sqrt{x}$ ankommt). Nur interessieren wir uns hier aber für die besonders markierten Zahlen, da diese viele kleine Primfaktoren enthalten.

Betrachte man alle Primpotenzen p^j kleiner gleich x mit $p \leq y$. Und anstatt nun jede p^j Zahl zu streichen , ersetze sie durch ihren Quotient nach Division durch p. Die y-glatten Zahlen sind die Zahlen, an deren Stelle nach diesem Prozess nur noch eine 1 steht. Die Rechenzeit dieses Verfahrens ist ungefähr $x \log\log y$, dies folgt mit Hilfe des Satzes von Mertens. Dieser besagt

$$\sum_{p \leq t} \frac{1}{p} = \log\log t + C + O(\frac{1}{\log t}),$$

wobei $C \approx 0,2614972...$ die Meissel-Mertens-Konstante ist.

Nun könnte man auch annehmen, dass *trial division* eine gute Methode wäre, um zu erkennen, ob eine Zahl n y-glatt ist. Also die Faktorisierungsmethode, in der man die Zahl n auf die Teilbarkeit durch die aufeinander folgenden Primzahlen testet. Aber dennoch braucht man, obwohl glatte Zahlen weit entfernt vom *worst case* von *trial division* sind, ungefähr y Schritte pro Zahl. Damit beträgt die Rechenzeit von *trial division* ungefähr xy. Dies ist verglichen mit einer Rechenzeit von $x \log\log y$ ein drastischer Unterschied.

(ii) Sei $y = x^{\frac{1}{u}}$ mit $u \geq 1$. So bietet sich zum Beispiel $u = 2$, d.h. $y = x^{\frac{1}{2}}$ als Grenze beim Sieb des Eratosthenes an. Um y-glatte Zahlen im Intervall $(x, x+z)$ mit $z \leq x$ aus $\mathbb{N}$ zu finden, definiere man $a[i] := 0$ für $1 \leq i \leq z$ (wobei $a[i]$ $x+i$ entspricht). Nun bestimme man für alle aufeinander folgenden Primpotenzen $p^j \leq x+z$ mit $p \leq y$ das kleinste i, so dass $x+i$ von p^j geteilt wird. Addiere nun $\log p$ zu den entsprechenden $a[i]$, $a[i+p^j]$, $a[i+2p^j]$ und so weiter bis zum Ende der Folge. Ist

$$a[i] \geq \log x \tag{7}$$

nach diesem Prozess, so ist $x+i$ y-glatt, da

$$x + i = m \cdot \prod_{k=1}^{n} p_k = m \cdot e^{a[i]} \tag{8}$$

für $m \in \mathbb{N}$ und p_k wie oben. Dann gilt mit (8), dass

$$m = \frac{x+i}{\prod_{k=1}^{n} p_k} \overset{(7)}{\leq} \frac{x+i}{x} = 1 + \frac{i}{x} \leq 2.$$

Da nun $y < 2$ als Grenze keinen Sinn macht, ist $m = 1$ und $x+i$ somit y-glatt. Die Rechenzeit dieses Algorithmus ist nach Granville (5.7 in [2]) durch $z \log\log y + uy$ beschränkt.

Betrachten wir einmal ein Beispiel, um diesen Algorithmus besser zu verstehen. Sei $x = 100, z = 10, u = 2$. Also untersuchen wir die Zahlen im Intervall (100,110) darauf, ob sie 10-glatt sind. Dafür schauen wir, ob die Primzahlen $2, 3, 5, 7$ und ihre Potenzen $2^2, 2^3, 2^4, 2^5, 2^6, 3^2, 3^3, 3^4, 5^2, 7^2$ die Zahlen $100 + i$, wobei $1 \leq i \leq 10$, teilen.

2 teilt $102 = 100 + 2$, also addieren wir $\log 2$ zu $a[2]$, $a[4]$, $a[6]$, $a[8]$ und erhalten $a[2] = a[4] = a[6] = a[8] = \log 2$. Da 2^2 ein Teiler von $104 = 100 + 4$ ist, addieren wir zu $a[4]$, $a[8]$ nochmals $\log 2$ und erhalten $a[4] = a[8] = \log 2 + \log 2$. Dies führen wir fort und erhalten:

$$\begin{aligned}
2^3|104 &: a[4] = \log 2 + \log 2 + \log 2;\\
3|102 &: a[2] = \log 2 + \log 3,\ a[5] = \log 3, a[8] = \log 2 + \log 2 + \log 3;\\
3^2|108 &: a[8] = \log 2 + \log 2 + \log 3 + \log 3;\\
3^3|108 &: a[8] = \log 2 + \log 2 + \log 3 + \log 3 + \log 3;\\
5|105 &: a[5] = \log 3 + \log 5;\\
7|105 &: a[7] = \log 3 + \log 5 + \log 7.
\end{aligned}$$

Die Potenzen $2^4, 2^5, 2^6, 3^4, 5^2, 7^2$ teilen keine der Zahlen zwischen 100 und 110.

i	$a[i]$	$e^{a[i]}$
1	0	1
2	$\log 2$	2
3	0	1
4	$3\log 2 = \log 8$	8
5	$\log 3 + \log 5 + \log 7 = \log 105$	105
6	$\log 2$	2
7	0	1
8	$2\log 2 + 3\log 3 = \log 108$	108
9	0	1

Wie wir in unserer Tabelle ablesen können, sind nur $a[5] \geq \log 100$ und $a[8] \geq \log 100$ und damit sind 105 und 108 die einzigen 10-glatten Zahlen zwischen 100 und 110.

(iii) Nun interessieren uns aber nicht die y-glatten Zahlen bis x, sondern die y-glatten Werte des Polynoms $f(x) = x^2 - n$. Das ist aber keine große Hürde, da $f(x + mp^j) = x^2 + 2xmp^j + m^2p^{2j} - n \equiv x^2 - n \pmod{p^j}$. Damit können wir das Argument aus (i) wieder anwenden.

Ebenso kann man weiterhin die Methode aus (ii) anwenden. Denn die Idee für glatte Werte von Polynomen funktioniert sehr ähnlich, da $p|f(x)$ genau dann, wenn $p|f(x + p)$.

3.2 Wahl der Schranke

Wie vorher erwähnt, ist die Wahl der „Glätte-Schranke“ y ein Optimierungsproblem. Ist y klein, benötigt man zwar nur wenig y-glatte Zahlen, und die Matrix, mit der man arbeitet, ist klein. Doch muss man viele Werte unserer Folge $x^2 - n$ betrachten, um eine ausreichende Zahl an y-glatten Zahlen zu erhalten.
Ist andererseits y groß, sind die y-glatten Zahlen in $x^2 - n$ häufig. Aber man benötigt auch eine entsprechend größere Anzahl an y-glatten Zahlen, und damit wird auch die Matrix, mit der wir arbeiten, ziemlich groß, was den Rechenaufwand stark erhöht.

Bestimmen der „besten“ Schranke y für $y^u = x$. Dann ist

$$\frac{x}{\Psi(x, x^{\frac{1}{u}})}$$

(ungefähr, falls $x \notin \mathbb{N}$) der erwartete Wert an zufälligen Versuchen, bis man in $[1, x]$ eine y-glatte Zahl erhält. Und wir brauchen ca. $\pi(y)$ y-glatte Zahlen, wobei die Rechenzeit pro Zahl ungefähr $\log\log y$ beträgt, dies folgt unter Verwendung des Satzes von Mertens.

Also beträgt die erwartete Rechenzeit

$$\varphi(x,u) = \frac{\pi(y)\log\log(y)\cdot x}{\Psi(x,y)}. \tag{9}$$

Wir wollen diese minimieren. Vorweg sei angemerkt, dass alle folgenden Abschätzungen unter den Annahmen, dass x und u gegen unendlich gehen, gemacht werden. Nun machen wir die groben Abschätzungen

$$\pi(y)\log\log y \approx y = x^{\frac{1}{u}} \tag{10}$$

und

$$\frac{x}{\Psi(x, x^{\frac{1}{u}})} \approx u^u. \tag{11}$$

Die Abschätzung (10) folgt mithilfe des Primzahl-Satzes (siehe [3]), der besagt

$$\pi(y) \sim \frac{y}{\log y}. \tag{12}$$

Es gilt dann nämlich, dass

$$\pi(y)\log\log y \overset{(12)}{\sim} y\cdot\tfrac{\log\log y}{\log y} = y^{1+o(1)}.$$

Für (11) sei auf (6) auf S. 77 in [1] verwiesen.
Mit diesen Abschätzungen erhalten wir für die Rechenzeit (9), dass

$$\varphi(x,u) \approx x^{\frac{1}{u}}\cdot u^u. \tag{13}$$

Das Minimum von $\varphi(x,u)$ ist gleich dem von $\log\varphi(x,u)$, also erhalten wir:

$$\phi(x,u) = \log\varphi(x,u) = \tfrac{1}{u}\log x + u\log u.$$

Nun minimieren wir $\phi(x,u)$, d.h.

$$\frac{\partial\phi(x,u)}{\partial u} = -\tfrac{1}{u^2}\log x + \log u + 1 = 0 \Longleftrightarrow u^2(\log u + 1) = \log x. \tag{14}$$

Logarithmieren von (14) ergibt:

$$\tfrac{1}{2}\log\log x = \log u + \tfrac{1}{2}\log(\log u + 1) \sim \log u. \tag{15}$$

Wobei wir die letzte Abschätzung machen dürfen, da für $u \to \infty$ der Summand $\frac{1}{2}\log(\log u + 1)$ nicht mehr ins Gewicht fällt. Aus (14) und (15) erhalten wir:

$$u \overset{(14)}{=} \sqrt{\frac{\log x}{\log u + 1}} \overset{(15)}{\sim} \sqrt{\frac{\log x}{\frac{1}{2}\log\log x + 1}} = \sqrt{\frac{2\log x}{\log\log x + 2}} \sim \sqrt{\frac{2\log x}{\log\log x}}. \tag{16}$$

Die letzte Abschätzung folgt wieder, da für $x \to \infty$ die 2 nicht mehr ins Gewicht fällt. Somit erhalten wir für unsere „Glätte-Schranke“ y:

$$y = x^{\frac{1}{u}} = \exp\Big(\tfrac{1}{u}\log x\Big) \overset{(16)}{\sim} \exp\Big(\Big(\tfrac{2\log x}{\log\log x}\Big)^{-\frac{1}{2}} \cdot \log x\Big) = \exp\Big(2^{-\frac{1}{2}} \cdot (\log x \cdot \log\log x)^{\frac{1}{2}}\Big). \quad (17)$$

So erhalten wir aber nicht nur die Schranke, sondern können auch die Rechenzeit abschätzen. Mit (15)-(17) haben wir:

$$\begin{aligned}\varphi(x,u) &\overset{(13)}{\approx} x^{\frac{1}{u}} \cdot u^u = \exp\Big(\tfrac{1}{u}\log x + u\log u\Big)\\ &\overset{(17),(16),(15)}{\sim} \exp\Big(2^{-\frac{1}{2}} \cdot (\log x \cdot \log\log x)^{\frac{1}{2}}) + 2^{\frac{1}{2}}(\log x)^{\frac{1}{2}}(\log\log x)^{-\frac{1}{2}} \cdot 2^{-1}\log\log x\Big)\\ &= \exp\Big(2 \cdot 2^{-\frac{1}{2}}(\log x \cdot \log\log x)^{\frac{1}{2}}\Big) = \exp\Big(2^{\frac{1}{2}}(\log x \cdot \log\log x)^{\frac{1}{2}}\Big).\end{aligned}$$

Für $x = \sqrt{n}$, wie beim quadratischen Sieb, gilt dann:

$$y \sim \exp\Big(\tfrac{1}{2}(\log n \cdot \log\log n)^{\frac{1}{2}}\Big),$$

$$\varphi(x,y) \sim \exp\Big((\log n \cdot \log\log n)^{\frac{1}{2}}\Big).$$

Somit liegt die Laufzeit zum Faktorisieren einer Zahl n, bei gegebenen Annahmen, in der Größenordnung von $\exp(\sqrt{\log n \cdot \log\log n})$.

4 Ausblick

Wir haben einige Beispiele für Anwendungen von glatten Zahlen gesehen. Dabei konnten wir aber schon feststellen, dass mit ihrer Hilfe die Rechenzeit eines Algorithmus stark verkürzt werden kann. Und konnten so die Bedeutung von glatten Zahlen in Faktorisierungsverfahren erahnen.
So verwendet man sie zum Beispiel auch bei anderen Faktorisierungsverfahren wie dem Zahlkörpersieb und bei elliptischen Kurven. Doch bleibt die Wahl der Schranke für Laien etwas vage und schlecht fassbar.

5 Quellen

Die beiden Quellen [1] und [2] sind Artikel aus *Surveys in algorithmic number theory*, edited by J. P. Buhler and P. Stevenhagen Math. Sci. Res. Inst. Publ. 44, Cambridge University Press, New York, 2008.
[1] C. Pomerance, „Smooth numbers and the quadratic sieve“, pp. 69-81.
[2] A. Granville, „Smooth numbers: computational number theory and beyond“, pp. 267-323.
[3] Persönliche E-Mail von C.Pomerance vom 30.3.2011, siehe Anhang.

roundcube
Free webmail for the masses

E-Mail Adressbuch Einstellungen Abmelden

Verschieben nach...

Ordner
- **Posteingang**
- Entwürfe
- Gesendet
- Trash
- Gesendet
- Papierkorb
- sent-mail

Betreff **Re: Question about "Smooth numbers and the quadratic sieve"**
Absender carlp@gauss.dartmouth.edu
Empfänger sriedel
Datum 30.03.2011 17:43

Dear Sonja,

It comes down to how rough the approximation is. By the prime number theorem, pi(B) is of order B/log B, which is of the general shape B^{1+o(1)}, where "o(1)" tends to 0 as the variables go to infinity (and in this case is negative). Multiplying by the small function loglog B doesn't change this. So if B is defined as X^{1/u}, then we have pi(B)loglog B = X^{(1+o(1))/u}. So, the symbol \approx in this case involves a small error in the exponent. The two quantities have the property that the ratio of their logarithms tends to 1 as the variables go to infinity.

I hope this helps, and good luck with the rest of the paper.

Best wishes,
Carl Pomerance

> Dear Mr Pomerance,
>
> I'm working with your paper in one of my courses at the university at the moment. And now a problem has appeared, which I couldn't solve in many hours of thinking about it. But maybe you're so kind to help me. On page 77 (in Algorithmic number theory) in the passage "The choice of the smoothness bound" you are making rough approximations. And until now I was't able to understand how you do the first one, \pi(B)loglogB\approx X^{\frac{1}{u}}. The only thing comes in mind to me is the prime number thm, but I can just use loglogB instead of logB.
>
> I would be really thankful, if you could help me.
>
> Yours sincerely,
> Sonja Riedel

Nachricht 5 von 15