Prímek, rejtjelzés, komputeralgebra

Előadás

Járai Antal
ELTE IT, Komputeralgebra Tanszék
http://compalg.inf.elte.hu/~ajarai

A Komputeralgebra Tanszék bemutatása:

Mesterünk: Kátai Imre akadémikus.

Kutatott területek: A számelmélet számos területe, fraktálgeometria, általánosított számrendszerek, algebra, sztochasztikus folyamatok, részecskefizika, függvényegyenletek, mértékelmélet, rendszerprogramozás, kódoláselmélet, számítógépes számelmélet, rejtjelzés, komputeralgebra.

Prímek

$697053813 \cdot 2^{16352} \pm 1$, $242206083 \cdot 2^{38880} \pm 1$, $2230907354445 \cdot 2^{48000} \pm 1$, $871892617365 \cdot 2^{48000} \pm 1$, $2409110779845 \cdot 2^{60000} \pm 1$ és $4648619711505 \cdot 2^{60000} \pm 1$ ikerprímek. $p = 157324389 \cdot 2^{16352} - 1$, $p = 470943129 \cdot 2^{16352} - 1$, $p = 2375063906985 \cdot 2^{19380} - 1$ és $p = 37140898895285 \cdot 2^{60000} - 1$ Sophie Germain prímek, azaz p és $2p+1$ is prímek. $p = 4610194180515 \cdot 2^{5056} - 1$-re p, $p+2$ és $2p+1$ is prím. $(235 \cdot 2^{18400})^4 + 1$ prím. A Waring konstans $g(k) = \lfloor 3^k/2^k \rfloor + 2^k - 2$, ha $k \leq 5 \cdot 2^{30}$.

A nagy ikerprímeket és Sophie Germain prímeket Karl-Heinz Indlekofer professzor munkacsoportjában, Universität GH Paderborn, Németország „színeiben” kerestük, ahol 5 évig projektmenedzserként dolgoztam a csoportban, és a programok nagy részét én terveztem, kb. 70%-át pedig én írtam. Más kollégák (Almási Béla, Baligács András, Fazekas Gábor) és számos hallgató is résztvett a munkában.

A híres ikerprím sejtés szerint végtelen sok ikerprím van, és egy erősebb sejtés még azt is megmondja, hogy x-ig $\approx 1.32 \cdot x/(\ln x)^2$ ikerprím van. Egy még erősebb sejtés még a prím s-esek asszimptotikus sűrűségét is megadja egy sokkal általánosabb esetben.

A sejtés által adott becslések nagyon jó egyezésben vannak a számítógépes kísérletek eredményeivel. Az alábbi öt sorozatra végeztünk kereséseket.

sorozat
$(3+30h)2^{38880}\pm 1$
$(5775+30030h)2^{19380+1}\pm 1$
$(5775+30030h)2^{5040+1}\pm 1$
$(5775+30030h)2^{4980+1}\pm 1$
$(21945+30030h)2^{5056+1}\pm 1$

A szitálásokra az alábbi táblázat összeveti az eredményeket a várt értékekkel.

exponens	tartomány	szita határ	szita után	várt
38880	$0\le h<2^{27}$	2^{35}	947 738	949 087
$19380+1$	$0\le h<2^{27}$	$1000\cdot 2^{25}$	223 401	223 641
$5040+1$	$0\le h<2^{28}$	$1000\cdot 2^{25}$	449 119	447 601
$4980+1$	$0\le h<2^{28}$	$1000\cdot 2^{25}$	448 181	447 601
$5056+1$	$0\le h<2^{28}$	$8000\cdot 2^{25}$	349 954	349 641

Az alábbi táblázatba a talált prímek, ikerprímek, stb. számát vetjük össze a várttal.

exponens	tesztelt	prím	várt	iker	várt	S. G.	várt
38880	55440	99	102.6	1	0.1899	–	–
$19380+1$	182 488	598	585.3	0	1.878	1	1.877
$5040+1$	449 119	5452	5510	68	67.6	0	0.829
$4980+1$	448 181	5646	5564	60	69.1	0	0.857
$5056+1$	215 000	2819	2855	31	37.9	1	0.526

Prímtesztek, faktorizálás.

Miller-Rabin féle valószínűségi teszt.

Pontos tesztek: körosztási testek, elliptikus görbék, Agrawal–Kayal–Saxena algoritmus.

Faktorizálás: elliptikus görbékkel, szita módszerekkel.

Egyéb számítógépes számelméleti problémák.

Páros Goldbach-sejtés adott határig.

Ikerprímek száma és reciprokaik összege adott határig.

Prímek ko'zötti rések eloszlása.

Különös prímkombinációk.

Rejtjelzés

RSA-módszer: Két nagy prímet választunk, és modulo a szorzatuk hatványozzuk az üzenetet egy adott, nyilvános kitevőre. Sok egyéb alkalmazás: digitális aláírás, bizonyítványok kiállítása, stb.

Rejtjelzés elliptikus görbékkel.

Komputer algebra

Kifejezésekkel való számolás. A kifejezések egyszerűsítésénél többváltozós, magas fokú és nagy együtthatókkal rendelkező polinomok faktorizálására van szükség.

A kritikus pont: a szorzás

Karacuba algoritmus. A legegyszerűbb módszer, amely gyorsabb, mint a klasszikus algoritmus Karacuba módszere. Ez a rekurzív algoritmus két hosszú előjel nélküli, $n = 2^m$ szavas szám szorzatát számítja ki. Csak 3^m egyszeres pontosságú szorzást használ $2^{2m} = 4^m$ helyett.

Szorzás gyors Fourier transzformációval. A *valós* $2n$ tagú $f_0, f_1, \ldots, f_{2n-1}$ és $g_0, g_1, \ldots, g_{2n-1}$ sorozatok diszkrét Fourier transzformáltját felhasználhatjuk arra, hogy kiszámoljuk a

$$h_k = \sum_{j=0}^{k} f_j g_{k-j}$$

számokat $k = 0, 1, \ldots, 2n-2$-re. Ezek a számok a $\sum_{j=0}^{2n-2} h_j x^j$ polinom együtthatói, amely a $\sum_{j=0}^{n-1} f_j x^j$ és $\sum_{j=0}^{n-1} g_j x^j$ polinomok szorzata. Ezért nagyon hasznosak, ha két hosszú szám szorzatát akarjuk kiszámolni, amelyeket $f_0, f_1, \ldots, f_{n-1}$ illetve $g_0, g_1, \ldots, g_{n-1}$ reprezentál x alapú számrendszerben. Mivel a h_k sorozat az f_k és a g_k sorozatok göngyölt konvolúciója, h_k mint $\hat{h}_k = \hat{f}_k \hat{g}_k$ inverz diszkrét Fourier transzformáltja számolható.

Diszkrét Fourier transzformáltnak és inverzének a számítása $n \log_2 n$ művelettel megoldható a *Fast Fourier Transform* algoritmust használva .

Példa. Osszuk a 2^{19} bites számot 16 bites darabokra. 2^{15} komplex koordinátával rendelkező vektorok Fourier és inverz Fourier transzformáltját kell kiszámítani. Az eredményvektor tagjai legfeljebb 2^{15} darab 32 bites szorzat összegei. Így legalább 47 bit pontosságot kell elérnünk. Elég nagy valószínűséggel rekonstruálni tudjuk a szorzatot.

A Schönhage-Strassen szorzó algoritmus. Ez a nevezetes algoritmus két n-bites szám szorzásához szükséges bit műveletek számát $n \log n \log\log n$ rendig képes redukálni. A fő gondolat FFT-IFFT használata, de minden műveletet moduló egy $2^{2^k}+1$ Fermat szám végzünk egy alkalmas k-val amelyre $2^k \approx \sqrt{n}$. Az egységgyökök kettőhatványok, így az FFT-IFFT alatt csak összeadásra, kivonásra és eltolásra van szükség. A jegyenkénti szorzás rekurzív módon történik. Egy finom trükk lehetővé teszi, hogy az eredményt csak moduló 2^n+1 számoljuk ki.

Példa. Egy 2^{19} bites számot 2^{10} részre oszthatunk, amelyek mindegyike 16 darab 32 bites szóból áll, és az FFT-IFFT 2^{11} taggal történik $\bmod(2^{1024}+1)$.

Programozás

Sok assembly kód, kihasználva a processzor adottságait. A programok többi része C nyelven készült. Az ikerprím kereséshez szükséges tartományban a komplex FFT-t használó program bizonyult a leggyorsabbnak. Masszív parallel alkalmazásokra a Fermat szám transzformációs szorzás alkalmasabb, mert kevesebb adatot mozgat.

A programsorok eloszlása a prímkeresések során használt SuperSPARC verzióra:

Program	C sorok	Assembly sorok
Klasszikus algoritmusok	933	852
Karatsuba szorzás	0	2348
Fermat szám transzformációs szorzás	1471	2168
Komplex FFT szorzás	2078	4520
Rövid és speciális modulus rutinok	0	3138
Általános moduláris aritmetika	4901	0
Szitálás	1403	0
Valószínűségi tesztek	1131	0
Biztos tesztek	512	0

Futásidők. SuperSPARC processzoron a Karacuba módszer ≈ 80 jegytől gyorsabb, mint a klasszikus. $\approx 5\,000$ jegytől Schönhage és Strassen módszere még gyorsabb. A komplex Fourier transzformációs szorzás a leggyorsabb egy széles tartományban, de az eredmény csak egy „valószínűleg helyes" szorzat. A legjobb módszer erősen függ a számítógép architektúrájától.

Más gépeken futó programokkal nehéz az összehasonlítás. Ugyanazon a gépen programjaink kb. ötször gyorsabbak, mint Schönhage munkacsoportjáé.

Az alábbi táblázat másodpercekben mutatja a négyzetreemelés idejét különböző verziókra Fermat szám transzformációval.

	FFT	DIGMUL	IFFT	Total
C, gcc, PC 486/25SX	1.567	20.335	1.747	23.649
C, cc, Sun SS10/40	0.457	4.835	0.500	5.792
C, gcc, Sun SS10/40	0.338	4.397	0.417	5.152
as:wordmul	0.338	2.772	0.420	3.530
as:8wordmul	0.343	0.582	0.422	1.347
as:kara,add,sub,shift	0.087	0.243	0.105	0.435

Komplex FFT szorzás esetén a C program és az assembly közötti különbség nem ilyen nagy: az assembly durván háromszor gyorsabb. Egy 2^{19} bites szám négyzetre emelése 0.305 másodperc 40 MHz-es SuperSPARC processzoron.