Sinh số nguyên tố mạnh dùng trong mật mã

Nguyễn Đức Thắng

Sinh viên TI26

TLU

Yêu cầu

1. Sinh số nguyên tố ngẫu nhiên độ dài 3072 bit.

Yêu cầu

1. Sinh số nguyên tố ngẫu nhiên độ dài 3072 bit.
2. Thời gian để sinh mỗi số ngắn. Thời gian trung bình nên nhỏ hơn 5 giây/1 số trên các máy tính thông thường.

Định nghĩa

- ***Số nguyên tố*** là số nguyên lớn hơn 1, không chia hết cho số nguyên dương nào ngoài 1 và chính nó.

Định nghĩa

- ***Số nguyên tố*** là số nguyên lớn hơn 1, không chia hết cho số nguyên dương nào ngoài 1 và chính nó.
- Số nguyên lớn hơn 1 không phải số nguyên tố được gọi là ***hợp số***.

100 số nguyên tố đầu tiên

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	103	107	109	113
127	131	137	139	149	151	157	163	167	173
179	181	191	193	197	199	211	223	227	229
233	239	241	251	257	263	269	271	277	281
283	293	307	311	313	317	331	337	347	349
353	359	367	373	379	383	389	397	401	409
419	421	431	433	439	443	449	457	461	463
467	479	487	491	499	503	509	521	523	541

825154968537631912768298749170739297854398716362661816275503116079
481244776851676369506181060016953565332332544104243302099970453768
573552526611491950457625797020175677070075982187142166723199505455
790940666545512029823115363737022514932296310461843759943426637450
457978700698013163674793031149391910570974516004314717775131667097
126098742572843325657395082030947465507253903496576780657530224681
101778363387269042871961460357674470219471788386403599087114004172
62229372209083170147380517906145636406605403890844426358143534126
759193985336262167688110337743953182116252070586929061201355598602
898620963724847193471057235471237921703384265217076231273197853548
551705842298080177124144337311712909603561659196347298251868740384
298808360296925212361003693208144299852907112109370230400750357870
997794622389754516766792389806834086164756749904260739674386541960
302825628178473942663151037197890442447506340719073733827930075184
694843829985334125752222656014432200550207371044791552615829608023
729568421318530916220310811657584027558305745037362938267941655342
687150141185981390338210844918719377398248258106753462919520441851
026786481875648951265795109384767346853898863411878053416947898154
479434332215017695703979981056520395957368803

Số lượng số nguyên tố có "nhiều" không?

Với mỗi số nguyên dương n cho trước, ta ký hiệu $\pi(n)$ là số các số nguyên tố không vượt quá n.

Số lượng số nguyên tố có “nhiều” không?

Với mỗi số nguyên dương n cho trước, ta ký hiệu $\pi(n)$ là số các số nguyên tố không vượt quá n.

Định lý

$$\lim_{n\to\infty} \frac{\pi(n)}{n/\ln n} = 1.$$

Số lượng số nguyên tố có "nhiều" không?

Với mỗi số nguyên dương n cho trước, ta ký hiệu $\pi(n)$ là số các số nguyên tố không vượt quá n.

Định lý

$$\lim_{n\to\infty} \frac{\pi(n)}{n/\ln n} = 1.$$

Nói cách khác, giá trị $\pi(n)$ xấp xỉ bằng với $n/\ln n$ khi n lớn.

Sinh ngẫu nhiên số nguyên tố lớn

Số nguyên tố ngẫu nhiên

- Nếu ta lấy ngẫu nhiên một số nguyên dương k bit, xác suất để số này là số nguyên tố bằng $1/\ln 2^k$.

Số nguyên tố ngẫu nhiên

- Nếu ta lấy ngẫu nhiên một số nguyên dương k bit, xác suất để số này là số nguyên tố bằng $1/\ln 2^k$.
- Về trung bình, ta cần $\ln 2^k$ lần thử để lấy được một số nguyên tố k bit.

Ví dụ

Chọn ngẫu nhiên khoảng

$$\ln 2^{3072} \approx 2130$$

số nguyên dương 3072 bit, ta sẽ được một số nguyên tố 3072 bit.

Thuật toán sinh số nguyên tố ngẫu nhiên lớn

1. Chọn ngẫu nhiên một số nhị phân p độ dài 3072 bit.

Thuật toán sinh số nguyên tố ngẫu nhiên lớn

1. Chọn ngẫu nhiên một số nhị phân p độ dài 3072 bit.
2. Đặt cả hai bit cao nhất và bit thấp nhất của p lên 1.

Thuật toán sinh số nguyên tố ngẫu nhiên lớn

1. Chọn ngẫu nhiên một số nhị phân p độ dài 3072 bit.
2. Đặt cả hai bit cao nhất và bit thấp nhất của p lên 1.
3. Kiểm tra xem p có là số nguyên tố.

Thuật toán sinh số nguyên tố ngẫu nhiên lớn

1. Chọn ngẫu nhiên một số nhị phân p độ dài 3072 bit.
2. Đặt cả hai bit cao nhất và bit thấp nhất của p lên 1.
3. Kiểm tra xem p có là số nguyên tố.
4. Nếu có thì trả ra số nguyên tố p. Còn nếu không thì quay lại Bước 1.

Sinh số ngẫu nhiên trên HĐH GNU/Linux

- Sử dụng dãy bit ngẫu nhiên từ nguồn ngẫu nhiên của hệ thống (các thao tác chuột, bàn phím, chương trình chạy...).

Sinh số ngẫu nhiên trên HĐH GNU/Linux

- Sử dụng dãy bit ngẫu nhiên từ nguồn ngẫu nhiên của hệ thống (các thao tác chuột, bàn phím, chương trình chạy...).
- Các dãy bit này có thể lấy từ tệp tin `/dev/urandom`.

Sinh số ngẫu nhiên trên HĐH GNU/Linux

- Sử dụng dãy bit ngẫu nhiên từ nguồn ngẫu nhiên của hệ thống (các thao tác chuột, bàn phím, chương trình chạy...).
- Các dãy bit này có thể lấy từ tệp tin `/dev/urandom`.
- Trong trường hợp các dãy bit sinh ra chưa đủ ngẫu nhiên do entropy nhỏ, hệ thống sẽ sử dụng một hàm giả ngẫu nhiên an toàn.

Kiểm tra số nguyên tố

- Thuật toán Rabin-Miller cho phép kiểm tra một số nguyên n có là nguyên tố hay không với một xác suất sai có thể làm nhỏ tuỳ ý.

Kiểm tra số nguyên tố

- Thuật toán Rabin-Miller cho phép kiểm tra một số nguyên n có là nguyên tố hay không với một xác suất sai có thể làm nhỏ tuỳ ý.
- Chúng tôi chọn xác suất sai bằng $1/2^{128}$.

Kiểm tra số nguyên tố

- Thuật toán Rabin-Miller cho phép kiểm tra một số nguyên n có là nguyên tố hay không với một xác suất sai có thể làm nhỏ tuỳ ý.
- Chúng tôi chọn xác suất sai bằng $1/2^{128}$.
- Cụ thể, nếu thuật toán thông báo n là hợp số, vậy n chắc chắn là hợp số.

Kiểm tra số nguyên tố

- Thuật toán Rabin-Miller cho phép kiểm tra một số nguyên n có là nguyên tố hay không với một xác suất sai có thể làm nhỏ tuỳ ý.
- Chúng tôi chọn xác suất sai bằng $1/2^{128}$.
- Cụ thể, nếu thuật toán thông báo n là hợp số, vậy n chắc chắn là hợp số.
- Còn nếu thuật toán thông báo n là nguyên tố thì xác suất n là hợp số chỉ là $1/2^{128}$.

Kiểm tra Fermat

Định lý

Nếu n là số nguyên tố thì, với mọi số nguyên $n > b > 1$,

$$b^{n-1} = 1 \mod n.$$

Thuật toán *TestFermat(n)*:

1. **if** $(2^{n-1} \neq 1 \mod n)$ **return** "n là hợp số";
   else return "n có khả năng là số nguyên tố".

Thuật toán kiểm tra số nguyên tố

1. Tính sẵn 1000 số nguyên tố đầu tiên: $q_1, q_2, \ldots, q_{1000}$;

Thuật toán kiểm tra số nguyên tố

1. Tính sẵn 1000 số nguyên tố đầu tiên: $q_1, q_2, \ldots, q_{1000}$;
2. **if** (n chia hết cho q_i nào đó) **return** "hợp số";

Thuật toán kiểm tra số nguyên tố

1. Tính sẵn 1000 số nguyên tố đầu tiên: $q_1, q_2, \ldots, q_{1000}$;
2. **if** (n chia hết cho q_i nào đó) **return** "hợp số";
3. **if**(*TestFermat*(n) = "hợp số") **return** "hợp số";

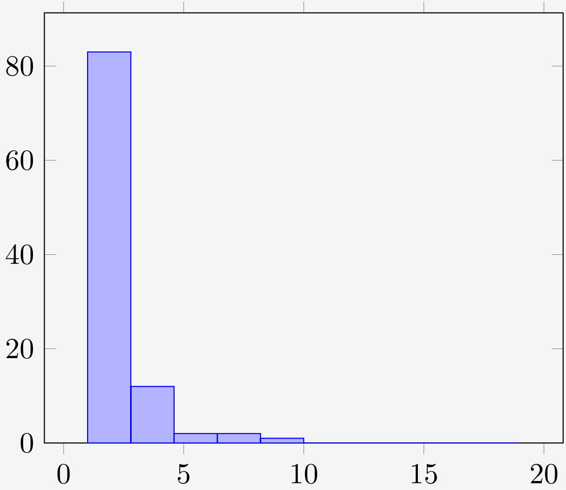
Thuật toán kiểm tra số nguyên tố

1. Tính sẵn 1000 số nguyên tố đầu tiên: $q_1, q_2, \ldots, q_{1000}$;
2. **if** (n chia hết cho q_i nào đó) **return** "hợp số";
3. **if**(*TestFermat*(n) $=$ "hợp số") **return** "hợp số";
4. **if**(*Rabin-Miller*(n, $128/2$) $=$ "hợp số") **return** "hợp số";
   else return "nguyên tố";

Định lý

Với mỗi số nguyên lẻ $n > 1$ *và số nguyên dương* s*, xác suất sai của thuật toán Rabin-Miller*(n, s) *không vượt quá* $1/2^{2s}$*.*

Thử nghiệm : Sinh 100 số nguyên tố 3072 bit



Sinh số nguyên tố mạnh

Số nguyên tố mạnh

- Thuật toán $p-1$ của Pollard là một trong những thuật toán hiệu quả để phân tích thừa số nguyên tố $n = pq$ khi p và q thỏa mãn một số tính chất đặc biệt.

Số nguyên tố mạnh

- Thuật toán $p-1$ của Pollard là một trong những thuật toán hiệu quả để phân tích thừa số nguyên tố $n = pq$ khi p và q thỏa mãn một số tính chất đặc biệt.
- Để tránh phương pháp tấn công này, các số p và q nên là các *số nguyên tố mạnh*.

Số nguyên tố mạnh

- Thuật toán $p-1$ của Pollard là một trong những thuật toán hiệu quả để phân tích thừa số nguyên tố $n = pq$ khi p và q thỏa mãn một số tính chất đặc biệt.
- Để tránh phương pháp tấn công này, các số p và q nên là các *số nguyên tố mạnh*.
- Đây là lý do mà chuẩn `ANSI X9.31` yêu cầu sử dụng số nguyên tố mạnh để làm khóa cho các hệ chữ ký điện tử dựa trên RSA.

Định nghĩa

Số nguyên tố lớn p được gọi là số nguyên tố mạnh nếu nó thỏa mãn cả ba điều kiện sau:

1. $p-1$ có thừa số nguyên tố u đủ lớn;

Định nghĩa

Số nguyên tố lớn p được gọi là số nguyên tố mạnh nếu nó thỏa mãn cả ba điều kiện sau:

1. $p-1$ có thừa số nguyên tố u đủ lớn;
2. $p+1$ có thừa số nguyên tố s đủ lớn; và

Định nghĩa

Số nguyên tố lớn p được gọi là số nguyên tố mạnh nếu nó thỏa mãn cả ba điều kiện sau:

1. $p - 1$ có thừa số nguyên tố u đủ lớn;
2. $p + 1$ có thừa số nguyên tố s đủ lớn; và
3. $u - 1$ có thừa số nguyên tố t đủ lớn.

Thuật toán Gordon

- Năm 1984 John Gordon đã đề xuất một thuật toán hiệu quả để sinh số nguyên tố mạnh.

Thuật toán Gordon

- Năm 1984 John Gordon đã đề xuất một thuật toán hiệu quả để sinh số nguyên tố mạnh.
- Thuật toán Gordon chỉ mất thêm 19% thời gian tính toán so với thời gian tìm một số nguyên tố cùng kích thước bằng thuật toán Rabin-Miller.

Thuật toán Gordon

1. Tìm t và s là hai số nguyên tố lớn, $|t| = |s| \approx \dfrac{|p|}{2}$

Thuật toán Gordon

1. Tìm t và s là hai số nguyên tố lớn, $|t| = |s| \approx \frac{|p|}{2}$
2. Tính u là số nguyên tố nhỏ nhất theo công thức sau:

$$u = at + 1 \quad \text{với } a = 2, 4, 6, 8, \ldots$$

Thuật toán Gordon

1. Tìm t và s là hai số nguyên tố lớn, $|t| = |s| \approx \dfrac{|p|}{2}$
2. Tính u là số nguyên tố nhỏ nhất theo công thức sau:

$$u = at + 1 \quad \text{với } a = 2, 4, 6, 8, \ldots$$

3. Tính $p_0 = (s^{u-1} - u^{s-1}) \mod us.$

Thuật toán Gordon

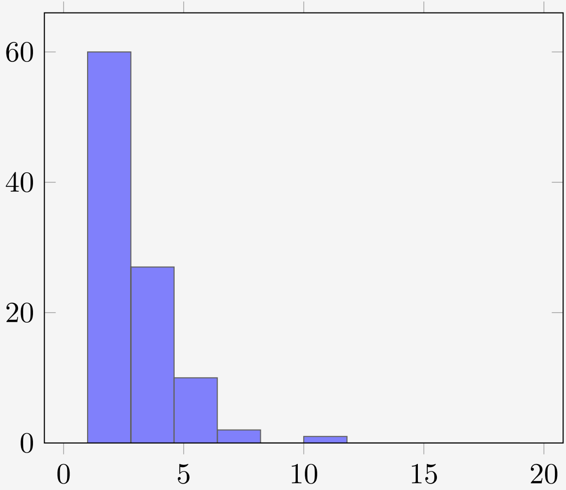
1. Tìm t và s là hai số nguyên tố lớn, $|t| = |s| \approx \dfrac{|p|}{2}$
2. Tính u là số nguyên tố nhỏ nhất theo công thức sau:

$$u = at + 1 \quad \text{với } a = 2, 4, 6, 8, \dots$$

3. Tính $p_0 = (s^{u-1} - u^{s-1}) \mod us$.
4. Tính p là số nguyên tố nhỏ nhất theo công thức sau:

$$p = p_0 + aus. \quad \text{với } a = 1, 2, 3, 4, \dots$$

Thử nghiệm: sinh 100 số nguyên tố mạnh 3072 bit



Cảm ơn Thầy Cô đã lắng nghe !