Number Theory 1 - Modulo

Error detecting codes

Tại sao lại cần cái này?

Mã phát hiện lỗi là một thuật toán hướng tới việc tận dụng chữ số kiểm tra (check digit) để phát hiện những lỗi nhập liệu của người dùng (chứ không dùng để sửa lỗi) trong quá trình nhập những mã như ISBN (mã số sách - International Standard Book Number) hay UPC (mã số hàng hóa - Universal Product Code), nhãn vận chuyển (Shipping label), thẻ thư viện, ...

Những lỗi thường gặp

Sai số do nhập liệu là một việc rất hay xảy ra, trong tất cả những lỗi nhập liệu, có hai loại dễ bị sai nhất (chiếm gần 90% tỉ lệ những lỗi sai) đó là:

- Sai một chữ số (123**4**5 $\rightarrow$ 123**9**5)
- Hoán vị hai số sát nhau (12**34**5 $\rightarrow$ 12**43**5)

Ngoài hai lỗi trên thì còn rất nhiều lỗi khác nhưng ít phổ biến hơn

ISBN - International Standard Book Number

- Sử dụng một số gồm 10 chữ số (và 13 chữ số kể từ 1-1-2007) để đánh số cho hầu hết những quyển sách được xuất bản ở các nước công nghiệp
- Được dùng hơn 49 năm (1970)
- Có thể phát hiện tất cả các trường hợp sai một chữ số và đa số trường hợp hai số sát nhau bị hoán vị
- Dùng phép $\mod 11$ ở ISBN-10 hoặc $\mod 10$ ở ISBN -13 để kiểm lỗi với chữ số phải cùng là check digit
- **ISBN-10:** Xem rằng số ISBN-10 được đánh số theo dạng $\overline{x_{10}x_9x_8\ldots x_2x_1}$, công thức kiểm lỗi của ta là: $10x_{10} + 9x_9 + 8x_8 + \ldots + 2x_2 + x_1 \equiv 0 \pmod{11}$
- **ISBN-13:** Xem rằng số ISBN-13 được đánh số dưới dạng $\overline{x_{13}x_{12}x_{11}\ldots x_2x_1}$, công thức kiểm lỗi của ta là: $x_{13} + 3x_{12} + x_{11} + 3x_{10} + \ldots + 3x_2 + x_1 \equiv 0 \pmod{10}$

Ví dụ:

Xét cuốn sách *Turtles All the Way Down* của John Green, ta có:

- **ISBN-10:** 0525555374 (theo công thức của ta, ta có $198 \equiv 0 \pmod{11}$
- **ISBN-13:** 978-0525555377 (theo công thức của ta, ta có $120 \equiv 0 \pmod{10}$

Giả sử số ISBN-10 của sách bị sai một chữ số 052555**7**374, ta sẽ có $206 \equiv 8 \pmod{11}$, hoặc bị đảo ngược hai chữ số kề nhau 0525555**47**, ta có $195 \equiv 8 \pmod{11}$

Tương tự nếu số ISBN-13 của sách bị sai một chữ số **8**78-0525555377, ta sẽ có $119 \equiv 9 \pmod{10}$, hay bị đảo ngược hai chữ số kề nhau 97805**52**555377 sẽ cho ta $126 \equiv 6 \pmod{10}$

Ứng dụng thực tế

Khi một mã bị nhập sai định dạng, tùy từng hệ thống có thể xử lý khác nhau. Có những hệ thống vẫn tiếp tục quá trình tìm kiếm, nhưng có những hệ thống sẽ lập tức từ chối việc tìm kiếm nếu mã sai, làm máy chủ không bị quá tải nếu quá nhiều người dùng tìm kiếm cùng lúc

Symmetric encryption

Tư tưởng cơ bản

Như mọi người đã biết, nếu ta có số a bất kỳ, ta nghịch đảo a thì sẽ thành a^{-1}, và nếu ta nghịch đảo thêm a^{-1} nữa thì sẽ thành $(a^{-1})^{-1} = a$. Như vậy, nếu ta lấy nghịch đảo của nghịch đảo của một số thì ta lại có số ban đầu.

Phép nghịch đảo modulo cũng mang tính chất tương tự, nếu ta lấy nghịch đảo modulo của nghịch đảo modulo với cùng modulo, ta sẽ quay lại được số ban đầu.

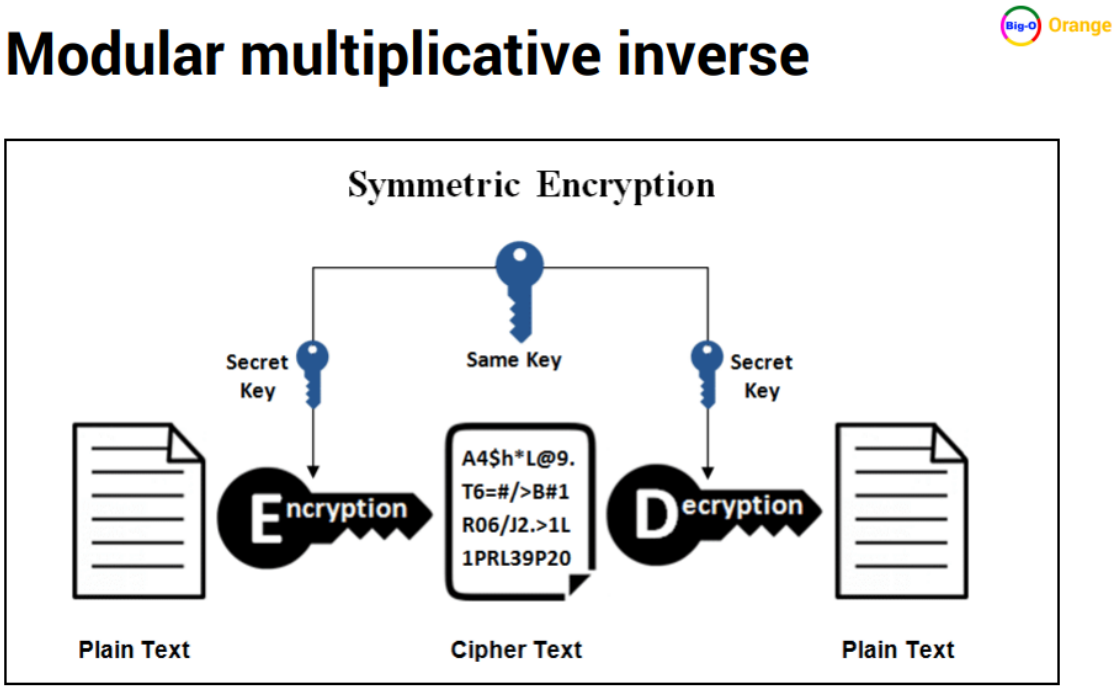
Ví dụ:

Nghịch đảo modulo của 9 với modulo 11 là 5: $9^{-1} \mod 11 = 5$

Nghịch đảo modulo của 5 với modulo 11 là 9: $5^{-1} \mod 11 = 9$

Áp dụng tính chất trên, **Mã hóa đối xứng** được sinh ra.

Sơ lược về Symmetric encryption



Theo như sơ đồ trên, ta sẽ dùng chung một khóa cho cả việc mã hóa (Encryption) và giải mã (Decryption), do đó ta gọi đây là một thuật toán mã hóa đối xứng (vì quá trình làm việc sẽ đối xứng thông qua chung một key)

Một số phép toán đối xứng của ta có thể kể ra như là

Phép toán	Phép đối xứng
$+$	$-$
$\times$	$\div$
$\wedge$	$\log$

Ở đây, việc mã hóa của ta với một key là việc ta áp dụng một phép toán theo sở thích và giải mã là việc ta áp dụng phép toán đối xứng (hay là "ngược lại") với phép toán ta đã chọn để mã hóa và dùng chung một key.

Ở đây demo phía dưới, code của mình sẽ dùng phép mod nghịch đảo để làm vấn đề này.

Demo

Source code demo: https://ideone.com/dPepMq

Output:

```
Original text: Big-O Coding
Original text in numbers (Ascii): 66 105 103 45 79 32 67 111 100 105 110
103

With modulo: 311
    Modulo inverse of  66 (B):  33 (!)
    Modulo inverse of 105 (i): 234 (ê)
    Modulo inverse of 103 (g): 154 (•)
    Modulo inverse of  45 (-): 235 (ë)
    Modulo inverse of  79 (O):  63 (?)
    Modulo inverse of  32 ( ): 243 (ó)
    Modulo inverse of  67 (C):  65 (A)
    Modulo inverse of 111 (o): 297 (ĩ)
    Modulo inverse of 100 (d):  28 (•)
    Modulo inverse of 105 (i): 234 (ê)
    Modulo inverse of 110 (n):  82 (R)
    Modulo inverse of 103 (g): 154 (•)

Cipher text in numbers: 33 234 154 235 63 243 65 297 28 234 82 154
Cipher text: !ê•ë?óAĩ•êR•

With modulo: 311
    Modulo inverse of  33 (!):  66 (B)
    Modulo inverse of 234 (ê): 105 (i)
    Modulo inverse of 154 (•): 103 (g)
    Modulo inverse of 235 (ë):  45 (-)
    Modulo inverse of  63 (?):  79 (O)
    Modulo inverse of 243 (ó):  32 ( )
    Modulo inverse of  65 (A):  67 (C)
    Modulo inverse of 297 (ĩ): 111 (o)
    Modulo inverse of  28 (•): 100 (d)
    Modulo inverse of 234 (ê): 105 (i)
    Modulo inverse of  82 (R): 110 (n)
    Modulo inverse of 154 (•): 103 (g)

Decrypted text in numbers: 66 105 103 45 79 32 67 111 100 105 110 103
Decrypted text: Big-O Coding
```

Tài liệu tham khảo

[1] https://www.uu.edu/dept/math/SeniorPapers/01-02/Oldham.pdf

[2] https://en.wikipedia.org/wiki/International_Standard_Book_Number

[3] http://mathworld.wolfram.com/ISBN.html

Number theory 2 - Primes & Euler's Totient Function

RSA (Encryption)

RSA (Rivest–Shamir–Adleman) là một trong những thuật toán mã hóa công khai đầu tiên và được dùng rất rộng rãi để đảm bảo tính bảo mật trong việc truyền dẫn dữ liệu. Và đây là một ứng dụng nổi tiếng nhất của Hàm Phi Euler.

Các bước làm của RSA

Tư tưởng cơ bản của RSA là tìm ba số nguyên cực kỳ lớn $e,\ d$ và n sao cho mô-đun lũy thừa của mọi số nguyên $m\ (0 \leq m < n)$ thỏa mãn

$(m^e)^d \equiv m \pmod n$

từ đó cho dù bạn có biết được e và n (công khai) thì vẫn rất khó để tìm được d (bí mật).

Tạo khóa

RSA có hai loại khóa: *khóa công khai* (public key) và *khóa bí mật* (secret key). Khóa trong RSA được tạo như sau:

1. Chọn hai **số nguyên tố** khác nhau p và q
    - Trên thực tế, hai số p và q là hai số rất rất lớn và chúng được chọn một cách ngẫu nhiên và độc lập. Có ý kiến bảo là độ dài của hai số này nên chênh lệch với nhau ít nhất là vài ký tự để việc truy lại hai số khó hơn
2. Tính $n = p \cdot q$
3. Tính $\phi(n) = (p-1)(q-1)$
4. Chọn một số nguyên e sao cho $1 < e < \phi(n)$ và nó là số nguyên tố cùng nhau với $\phi(n)$ (hay nói cách khác là $gcd(e, \phi(n)) = 1$)
5. Tính d sao cho $d \cdot e \equiv 1 \pmod{\phi(n)}$ (hay nói cách khác d là **Inverse Modulo** của e với modulo $\phi(n)$)
    - Bước này có thể dùng thuật toán Euclid mở rộng để giải

Lúc này, **khóa công khai** của ta là cặp giá trị (e, n) và **khóa bí mật** sẽ là cặp giá trị (d, n).

Theo lý thuyết, mọi người đều có thể biết khóa công khai và dùng nó để mã hóa thông điệp. Tuy nhiên thông điệp đã được mã hóa bởi khóa công khai chỉ có thể được giải mã bởi khóa bí mật.

Mã hóa

Giả sử bên B muốn chuyển cho bên A một đoạn văn bản M. Bên B đầu tiên phải chuyển M thành một số nguyên không âm $m < n$ bằng một hàm có thể đảo ngược và hàm này phải được biết bởi hai bên.

Sau khi có được m, n và e, bên B sẽ tính và gửi cho bên A bản mã hóa của m được tính theo công thức

$c = m^e \mod n$

Giải mã

Sau khi A nhận được đoạn mã hóa c từ B, A có thể sử dụng d để tìm được m từ c bằng công thức

$$m = c^d \mod n$$

Biết được m, A lại tiếp tục chuyển đổi từ m sang M theo hàm chuyển đổi đã được thỏa thuận từ trước.

Chứng minh

Để đọc thêm về việc chứng minh tính đúng đắn của thuật toán RSA, mọi người có thể đọc thêm ở link sau: https://aip.scitation.org/doi/pdf/10.1063/1.3526259

Demo

Source code demo: https://ideone.com/doL3dB

Output:

```
p = 503 | q = 541 | n = p * q = 272123
phi(n) = 271080
e = 139
gcd(e, phi(n)) = 1
d = e^(-1) mod phi(n) = 171619

Original text: Big-O Coding
Original text in numbers (Ascii): 66 105 103 45 79 32 67 111 100 105 110 103

Encrypting (c = m^e mod n):
     66 ->  66^139 mod 272123 = 249103
    105 -> 105^139 mod 272123 = 237131
    103 -> 103^139 mod 272123 = 170574
     45 ->  45^139 mod 272123 = 215163
     79 ->  79^139 mod 272123 = 67870
     32 ->  32^139 mod 272123 = 159780
     67 ->  67^139 mod 272123 = 260501
    111 -> 111^139 mod 272123 = 122469
    100 -> 100^139 mod 272123 = 129362
    105 -> 105^139 mod 272123 = 237131
    110 -> 110^139 mod 272123 = 163634
    103 -> 103^139 mod 272123 = 170574

Cipher text: 249103 237131 170574 215163 67870 159780 260501 122469 129362 237131 163634 170574

Decrypting (m = c^d mod n):
    249103 -> 249103^171619 mod 272123 = 66
    237131 -> 237131^171619 mod 272123 = 105
    170574 -> 170574^171619 mod 272123 = 103
    215163 -> 215163^171619 mod 272123 = 45
     67870 ->  67870^171619 mod 272123 = 79
    159780 -> 159780^171619 mod 272123 = 32
    260501 -> 260501^171619 mod 272123 = 67
    122469 -> 122469^171619 mod 272123 = 111
    129362 -> 129362^171619 mod 272123 = 100
    237131 -> 237131^171619 mod 272123 = 105
    163634 -> 163634^171619 mod 272123 = 110
    170574 -> 170574^171619 mod 272123 = 103
```

```
39
40 Decrypted text in numbers: 66 105 103 45 79 32 67 111 100 105 110 103
41 Decrypted text: Big-O Coding
42 ========================================================================
```

Tài liệu tham khảo

[1] https://simple.wikipedia.org/wiki/RSA_algorithm

[2] https://en.wikipedia.org/wiki/RSA_(cryptosystem)

[3] https://www.youtube.com/watch?v=4zahvcJ9glg

[4] https://www.youtube.com/watch?v=oOcTVTpUsPQ

[5] https://www.youtube.com/watch?v=9sY57iwNDJw

[6] https://aip.scitation.org/doi/pdf/10.1063/1.3526259