RSA 暗号に関するもの

BV19002　須藤 啓太

2020 年 6 月 12 日

目次

1　研究背景

一年生の情報処理の授業で習った RSA 暗号というものがどういうものなのかを知りたくこの研究を始めました.

2　RSA(暗号) とは

RSA について軽く説明すると, 大きな二つの異なる素数 p, q を用意し, それらの積 ($n = pq$ とする) を公開鍵としてメッセージを暗号化するシステムである. 暗号なのだから当然簡単に破られてはいけない. RSA における「暗号を破る」とは n(公開されている) を素因数分解して p と q を見つけることである. n をとても大きい数にすると n を分解するのにとても多大な時間と資源を要し, 破るのは難しいことが言える. 今回は RSA についての説明をするための準備を研究としている.

3　ユークリッド

3.1　ユークリッドアルゴリズム

ユークリッドアルゴリズムは 2 つの整数の最大公約数を計算するために使われる. ユークリッドアルゴリズムの仕組みは以下のようである. a と b は正の整数で, $a \geqq b$ と仮定しよう. このとき a と b の最大公約数を求めるには, まず a を b で割る. この除算の剰余を r_1 と呼ぶ. もし $r_1 \neq 0$ ならば b を r_1 で割る. r_2 をこの除算の剰余とする. 同様に $r_2 \neq 0$ ならば r_1 を r_2 で割り, 剰余 r_3 を得る. このように続けていき, $r_i = 0 (i = 2, 3, \cdots)$ となったときの $r_{i-1} (r_1 = 0$ のとき, $b)$ が求める最大公約数となる. このアルゴリズムを記述する.

ユークリッドアルゴリズム

入力:正の整数 $a \geqq b$.

出力:a と b の最大公約数

STEP1:$A = a$ と $R = B = b$ とおいて始める.

STEP2:R の値を A の B による除算の剰余で置き換え, STEP3 へ行く.

STEP3:もし $R = 0$ なら"a と b の最大公約数は B である" と書いて停止する. そうでないなら STEP4 へ行く.

STEP4:A の値を B で置き換え, B の値を R の値で置き換え, STEP2 へ戻る.

以下にユークリッドアルゴリズムの証明を記述する.

証明

3.1 の冒頭の記述より

$$b > r_1 > r_2 > \cdots \geqq 0$$

という関係がわかるが, b と 0 の間には有限個の整数しかないから, この列はどこまでも続くことはない. これはアルゴリズムが常に停止することを意味する. 次に, r_{i-1} が a と b の最大公約数であることを示す. ここで補題を用意する.

補題 3.1. a と b を正の整数とする. $a = bg + s$ である整数 g と s が存在するとしよう. すると

$gcd(a,b) = gcd(b,s)$ である.

補題 3.1 の証明 $d_1 = gcd(a,b), d_2 = gcd(b,s)$ とおく. 示したいことは $d_1 = d_2$ である. まず $d_1 \leqq d_2$ を示す. $d_1 = gcd(a,b)$ であることから, $a = d_1u, b = d_1v$ を満たす整数 u, v が存在する. これらを $a = bg + s$ に代入して, $s = d_1(u - vg)$ を得る. この式から d_1 は s を割り切ることがわかり d_1 は b と s の公約数であることがわかる. 今, d_2 は b と s の最大公約数であるから $d_1 \leqq d_2$ が示された. $d_2 \leqq d_1$ も同様に示せる. よって, $d_1 = d_2$ が示された.

この補題を繰り返し用いることによって $gcd(a,b) = gcd(b,r_1) = \cdots = gcd(r_{i-1},0) = r_{i-1}$ となり, 確かに a と b の最大公約数は r_{i-1} であることが示された.

3.2 拡張ユークリッドアルゴリズム

ユークリッドアルゴリズムでは出力しるものが a と b の最大公約数 (d とおく) だけであったが, 拡張ユークリッドアルゴリズムはそれに加えて以下の式を満たす α, β も出力する.

$$\alpha a + \beta b = d$$

以下, d, α, β を求めるのに必要な除算の列を書き下す

$$a = ax_{-1} + by_{-1} \cdots (-1 \text{ 行})$$
$$b = ax_0 + by_0 \cdots (0 \text{ 行})$$
$$a = bq_1 + r_1 \text{かつ } r_1 = ax_1 + by_1 \cdots (1 \text{ 行})$$
$$b = r_1q_2 + r_2 \text{かつ } r_2 = ax_2 + by_2 \cdots (2 \text{ 行})$$
$$r_1 = r_2q_3 + r_3 \text{かつ } r_3 = ax_3 + by_3 \cdots (3 \text{ 行})$$
$$r_2 = r_3q_4 + r_4 \text{かつ } r_4 = ax_4 + by_4 \cdots (4 \text{ 行})$$
$$\vdots$$
$$r_{n-3} = r_{n-2}q_{n-1} + r_{n-1} \text{かつ } r_{n-1} = ax_{n-1} + by_{n-1} \cdots (n-1 \text{ 行})$$
$$r_{n-2} = r_{n-1}q_n \text{かつ } r_n = 0 \cdots (n \text{ 行})$$

上の除算の列の左の列はユークリッドアルゴリズムと同じもので, 右の列が新たに考えている式である. ここで, 上の除算の列から

$$r_j = r_{j-2} - r_{j-1}q_j, r_{j-2} = ax_{j-2} + by_{j-2}, r_{j-1} = ax_{j-1} + by_{j-1}$$

をとってこれる, これらより

$$r_j = a(x_{j-2} - q_jx_{j-1}) + b(y_{j-2} - q_jy_{j-1})$$

が得られ, ゆえに,

$$x_j = x_{j-2} - q_jx_{j-1}, y_j = y_{j-2} - q_jy_{j-1}$$

である. この式から x_j と y_j を計算するために商 q_j と行 j の直前の 2 つの行からのデータを使うことがわかり, -1 行と 0 行が用意してある理由であることもわかるだろう.

$$a = ax_{-1} + by_{-1}, b = ax_0 + by_0$$

この式から,

$$x_{-1} = 1, y_{-1} = 0, x_0 = 0, y_0 = 1$$

のように選ばれ, これはアルゴリズムを開始するのに十分のものである. 除算をし終えるとユークリッドアルゴリズム同様 $(gcd(a,b) =)d = r_{n-1}$ がわかり次のような x_{n-1}, y_{n-1} を計算したことになる,

$$d = r_{n-1} = ax_{n-1} + by_{n-1}$$

ゆえに $\alpha = x_{n-1}$ かつ $\beta = y_{n-1}$ である. α と $d = r_{n-1}$ がわかれば, β は次の式

$$\beta = \frac{d - a\alpha}{b}$$

で見出すことができる. このアルゴリズムはユークリッドアルゴリズムに x と y の計算のための指示をいくつか足しただけなので停止し α を x_{n-1}, β を y_{n-1} として選ぶと $\alpha a + \beta b = d$ が成り立つ. 拡張ユークリッドアルゴリズムから次の定理が導かれる.

定理 3.2. a と b を正の整数とし, d を a と b の最大公約数とする. 次のような整数 α と β が存在する.

$$\alpha a + \beta b = d$$

注意:数 α と β は一意でない. k を整数として $\alpha a + \beta b = d \Leftrightarrow (\alpha + kb)a + (\beta - ka)b = d$ とできるからである.

4 一意素因数分解

ここでいう” 素数” という用語は” 正の素数” を略したものとする.

4.1 素因数分解の存在

4.1 では整数 $n \geqq 2$ が与えられるとき, それが素数の積で表せることを示す. これをするため, 入力を整数 $n \geqq 2$ とし出力が n の素因数とその重複度であるようなアルゴリズムを記述する. このアルゴリズムの準備段階として n の素因数をひとつだけ出力するアルゴリズムを考える. このようなアルゴリズムで最も単純なものは, 入力を $n \geqq 2$ とし, n を 2 から $n-1$ までの整数で試し割する. これらの整数の一つが n を割り切れれば, n は合成数でその最小の因数を見つけたことになる. そうでないなら, n は素数である. そして, n が合成数のとき見つけた因数は素数でなければならない. この最後の言明についての理由は, f を $2 \leqq f \leqq n-1$ である整数とする f を n の最小の因数とし, $f' > 1$ を f の因数とする. 整除性の定義により, 次のような整数 a と b が存在する

$$n = f * a \text{ かつ } f = f' * b$$

ゆえに $n = f' * a * b$ となり f' は n の因数でもある. f は n の最小の因数だから $f \leqq f'$ であり, f' は f の因数であるから $f' \leqq f$ である. これらの不等式から $f = f'$ である. したがって, $f' \neq 1$ が f を割り切るなら $f = f'$ であることを証明した. つまり, f が素数である. このアルゴリズムを詳細に調べる前にもう一つ注意すべきことがある. このアルゴリズムのなすことが因数を正整数の中に見つけることであったが, 一体どこまでこの探索を実行しなければならないのかというと $n-1$ を超える

ことがないのは自明である．整数はそれ自身より大きな因数を持つことはないからである．実際, $\sqrt{n}$ より大きな因数を探す必要はない．これもアルゴリズムが n の 1 より大きい最小の因数を探すことに依拠している．よって, 示すことは n が合成数で $f > 1$ が n の最小の因数なら, $f \leqq \sqrt{n}$ であることのみである．これは, a を n における f の余因子とすると, $n = fa$ である．$f > 1$ は n の最小の因数なので, $f \leqq a$ である．今, $a = \frac{n}{f}$ であるから $f^2 \leqq n$ となりこれは $f \leqq \sqrt{n}$ である．よって示された．アルゴリズムの議題に戻りこれまでをまとめると, アルゴリズムは 2 から始まり $\sqrt{n}$ までの整数を動いていく因数の探索からなる．n が合成数なら 2 以上の n の最小の因数が見出され, それは素数である．この探索で因数が見つからない場合 n 自身が素数である．実際, このアルゴリズムでは $\sqrt{n}$ の代わりに [] をガウス記号として $[\sqrt{n}]$ が分かっていればよい．4.1 で記述したかったものは前述のアルゴリズムを繰り返し用いて重複したものを数えるというアルゴリズムである．これを試行除算アルゴリズムという．このアルゴリズムは記述しないが, それについての考察を行う．もし素因数分解したい n が 100 桁の素数であったとすると, n が素数であることから $[\sqrt{n}]$ 回ループすることになるが今 $n \geqq 10^100$ であるので少なくとも 10^50 回ループすることになる．ここで, 計算機が 1 秒間に 10^10 回除算を行えるとすると, 計算機がこの n が素数であることを証明するのに 10^40 秒かかるということである．これは約 10^32 年に等しいとされている．大きな正の整数 n を素因数分解したくてさらにそれが素数だった場合には試行除算アルゴリズムは適していない．といえる．

4.2　フェルマーの素因数分解

4.1 で扱ったアルゴリズムと同じく, 与えられた数を素因数分解するためのアルゴリズムを 4.2 で扱う．まず n を奇数として, $n = x^2 - y^2$ となるような非負整数 x, y を見出すことがこのアルゴリズムの背景である．これらの数がみつかったと仮定すると,

$$n = x^2 - y^2 = (x + y)(x - y)$$

である．よって, $x + y, x - y$ は n の因数である．$y = 0$ のときがこのアルゴリズムの易しい場合である．このときある整数 r に対して $n = r^2$ であり $x = r, y = 0$ となる．一方, $y > 0$ の場合

$$x = \sqrt{n + y^2} > \sqrt{n}$$

となる．これらは x と y を見出す次の戦略を暗示する．

フェルマーの素因数分解アルゴリズム

入力:正の整数 n.

出力:n の因数あるいは n は素数である旨のメッセージ.

STEP1:$x = [\sqrt{n}]$ で始める．$n - x^2$ なら x は n の因数であり, 停止できる．そうでなければ x を 1 増やし STEP2 へ行く.

STEP2:$x = \frac{n+1}{2}$ なら n は素数であり, 停止できる．そうでなければ $y = \sqrt{x^2 - n}$ を計算する.

STEP3:y が整数なら (すなわち $[y]^2 = x^2 - n$ なら)n は因数 $x + y$ と $x - y$ をもち, 停止できる．そうでなければ x を 1 増やし STEP2 へ行く.

5　今後の課題

本資料は参考とした本の RSA 暗号の理解に必要な準備の部分的なものを記したものに過ぎない. なのでこの研究をもっと進めたものを近いうちに発表したいと思う.

参考文献

S.C. コウチーニョ, 暗号の数学的基礎, シュプリンガー・ジャパン,2001.