Eötvös Loránd Tudományegyetem
Természettudományi Kar
Algebra és Számelmélet Tanszék

Modern prímfaktorizáció

Készítette:
Varga Zsolt
Alkalmazott Matematikus szak

Témavezető:
Gyarmati Katalin
egyetemi adjunktus
Algebra és Számelmélet Tanszék
Eötvös Loránd Tudományegyetem, Természettudományi Kar

Budapest, 2012

Tartalomjegyzék

1. Bevezetés

Whit Diffie és Martin Hellman ötlete alapján 1975-ben Ron Rivest, Adi Shamir és Len Adleman kifejlesztették az első széles körben is elterjedt nyílt kulcsú titkosítási eljárást. A módszer lényege, hogy az eljárást használó felhasználók egy-egy kulcspárral rendelkeznek, melyek közül az egyiket szabadon terjeszthetik, a másikat pedig titokban kell tartaniuk. A nyilvános kulccsal kódolt üzenetet csak a titkos kulccsal lehet dekódolni és bár a kódolás algoritmusa mindenki számára ismert, a titkos kulcsot - és így az üzenetet - mégsem lehet könnyedén meghatározni.

Az RSA és más titkosítási rendszerek is matematikai alapon garantálják a dekódolás nehézségét, olyan un. egyirányú függvényeket használnak, melyek gyorsan kiértékelhetők, de az inverzük meghatározása a gyakorlatban szinte lehetetlen. Több módszer is számelméleti problémákra alapozza erejét, ebben a dolgozatban az RSA működését és a hozzá kapcsolódó prímfaktorizáció kérdését vizsgáljuk.

Az első fejezetben megismerkedünk a prímfaktorizáció történetével, majd a dolgozatban használt alapvető tételeket és definíciókat idézzük fel a második fejezetben. A matematikai alapok tárgyalása után megvizsgáljuk az RSA séma működését, és tárgyalunk néhány módszert, mely a sémát nem körültekintően használó felhasználók ellen bevethető. Az ötödik fejezetben sorra vesszük a prímfaktorizációra alkalmazható módszereket, különös tekintettel a GNFS algoritmusra, mely jelenlegi tudásunk szerint a legbonyolultabb és egyben leghatékonyabb algoritmus.

2. A prímfaktorizáció története

A prímfaktorizáció problémájának megszületése i.e. 300 környékére tehető, ekkor született ugyanis a görög Euklidész. Bár munkássága főként a geometria révén ismert, ő definiálta először a prímszámokat és kimondott a számelmélet alaptételével ekvivalens állításokat is. A számelmélet alaptétele kimondja, hogy minden egész szám felírható prímszámok szorzataként, lényegében egyértelműen. A prímfaktorizáció feladata ennek a felírásnak - vagyis egy adott egészt osztó prímeknek - a megkeresése. Mivel a prímfaktorizációnak az 1970-es évekig nem volt nagy gazdasági és elméleti jelentősége, ezért a kor matematikusai kevés figyelmet fordítottak a problémára. Egészen Fermatig nem is született a próbaosztásnál gyorsabb algoritmus a feladat megoldására, az ő gondolatai viszont visszaköszönnek még a mai modern algoritmusokban is, így tőle érdemes számítani a probléma történelmét.

Fermat (~ 1640) algoritmusának alapja, hogy minden páratlan N szám felírható két négyzetszám különbségeként, mivel ha $N = ab$, akkor $N = ((a+b)/2)^2 - ((a-b)/2)^2$. Ekkor a jól ismert azonosság miatt $N = x^2 - y^2 = (x+y)(x-y)$, azaz ha valamilyen módon találunk olyan x,y párt amire $N = x^2 - y^2$, akkor osztót is találtunk. Fermat módszere különböző x-ekre ellenőrzi, hogy $x^2 - N$ négyzetszám-e, mert ha igen, akkor meg is találtuk a keresett $x, y = \sqrt{x^2 - N}$ párt. Ez az algoritmus nagyon gyors, ha a keresett osztók közel vannak egymáshoz, egyébként viszont a próbaosztásnál (5.1.1) is lassabb. Az algoritmus ötletén alapszik a kvadratikus szita és a GNFS algoritmus is, melyeket később tárgyalunk az 5.2.2 és a 5.2.3 szakaszokban.

Euler (~ 1750), minden idők legproduktívabb matematikusa kivette a részét a számelmélet fejlődéséből is, nevéhez fűződik a relatív prímeket számláló Euler-függvény, illetve kidolgozott egy speciális alakú számokat faktorizáló algoritmust is. Fermat módszeréhez hasonlóan ő is négyzetszámokkal dolgozik, de az ő algoritmusában a faktorizálandó egészt négyzetszámok összegeként kell felírni, kétféleképpen. (Nem minden egész írható fel így, ebből adódik a speciális alak.)

Nem sokkal Euler után Legendre (~ 1800) munkássága lendítette előre a faktorizáció problémáját. Kétszáz évvel később a Legendre-szimbólum használata lesz a kvadratikus szita alapja, a kongruens négyzetek ötletére pedig több algoritmus is születik (a modern algoritmusok nagy része ezt használja).

1801-ben Gauss kiadta élete fő művét, a Disquisitiones Arithmeticae-t, mely több módszert és ötletet is tartalmaz az egészek prímfelbontásáról. Gauss munkája és főleg az eliminációs eljárása nélkül a 5.2 szakaszban tárgyalt algoritmusok nem működhetnének.

Az 1800-as évek végén többen mechanikus, illetve elektromos gépekkel igyekeztek kiváltani a hosszadalmas kézi számolásokat, melyekre érthető okokból nem pazarolhatták az idejüket: kézzel számolva több száz évig is eltarthat egy-egy nehezebb szám prímekre bontása, így mindenképpen egy gyorsabb megoldásra volt szükség.

1896-ban Lawrence mutatta be terveit egy szitáló gépről, mely egy mozgó papírt lyukasztott ki azokon a pontokon, ahol az algoritmus megengedett maradékhoz ért. Lawrence gépe végül soha nem épült meg, de ötlete hasonló elvekkel működő gépek születését vonta maga után.

1910-ben Maurice Kraitchik, G'erardin és a Carissan testvérek is építettek ilyen gépeket. Közülük a Carissan testvérek munkája érte el a legjobb eredményt gyorsaságban, de az ő gépüket is kézzel kellett még hajtani.

Az első automatikus szitáló gépet D. H. Lehmer építette 1926-ban, majd - követve a technika fejlődését - újabbakat készített. Az első még egy bicikli láncot hajtó villanymotorból állt, később napelemeket (1932), 16 mm-es mozi filmet (1936), végül analóg késleltetőket (1965) használt gépeihez.

Az 1900-as évek közepén a számítógépek fejlődése elérte azt a pontot, hogy legyőzhették a mechanikus gépek gyorsaságát, bár ekkor még nem léteztek jó implementációk a meglévő algoritmusokra.

1970-ben Daniel Shank változtatott ezen, algoritmusa (SQUFOF) megtette az első lépést a számítógépek győzelme felé[1], ezután évekig az ő módszerét használták jó eredményekkel.

1974-ben Pollard a $p-1$, 1975-ben pedig a ϱ-módszer bemutatásával szerzett hírnevet, algoritmusai speciális célúak, azaz nem alkalmazhatóak minden számra. Az első általánosan alkalmazható eljárást Morrison és Brillhart mutatta be szintén 1975-ben, az algoritmusuk (CFRAC) néhány évig a leggyorsabbnak számított.

1983-ben C. Pomerance QR-algoritmusával sikeresen faktorizált 70-jegyű számokat, a CFRAC osztásait elkerülő (általános célú) algoritmus ezzel átvette a leggyorsabbnak járó helyet.

[1] Valójában a szitáló gépek nagyon gyorsnak számítanak még ma is, Williams egy faktorizáló gépe $2 \cdot 10^{12}$ próbát végzett másodpercenként, ami egy átlagos számítógépnél kb. 1 milliószor gyorsabb számolást jelent! Az áttörést valójában a programok párhuzamosíthatósága és a bonyolultabb algoritmusok jelentették, amiket már igen nehéz volna mechanikus gépeken "futtatni".

1987-ben Lenstra keze által megszületett a máig leggyorsabb speciális célú algoritmus (ECM), melynek érdekessége, hogy módszere teljesen eltér az addig alkalmazottaktól.

1988-ban Pollard körlevélben tájékoztatta kutató társait új ötletéről (SNFS), az ebből születő algoritmus fejlesztett változata (GNFS) az általános célú algoritmusok között a leggyorsabbá vált. Az elmúlt 30 évben az GNFS sebessége különböző matematikai ill. implementációs trükkökkel nőtt, de más nagy eredmény nem született a témában.

1994-ben Peter Shore bemutatta az első kvantumszámítógépekre írt algoritmusát, mellyel polinom időben faktorizálható bármely összetett szám. Az algoritmus erősen épít kvantum-jelenségekre, így amíg várnunk kell a kvantumszámítógépek igazi megszületésére és elterjedésére, addig ez az algoritmus főként elméleti jelentőséggel bír.

3. Matematikai alapok

Ebben a fejezetben a dolgozat során használt alapvető definíciókon és tételeken haladunk végig, bizonyítások nélkül. A tételek bizonyítását megtalálhatjuk [3]-ban, [4]-ben, illetve [1]-ben.

3.1. Számelméleti alapfogalmak

3.1.1. Definíció (Osztó). *A b egész számot az a egész szám osztójának nevezzük, ha létezik olyan q egész szám, melyre $a = qb$. Ezt $b \mid a$-val jelöljük.*

3.1.2. Definíció (Egység). *Ha egy szám minden számnak osztója, akkor egységnek nevezzük.*

3.1.3. Definíció (Legnagyobb közös osztó (lnko)). *Az $a, b \in \mathbb{Z}$ számok legnagyobb közös osztójának azt a $d \in \mathbb{Z}$ számot nevezzük, melyre $d \mid a$, $d \mid b$ és ha egy c-re $c \mid a$ és $c \mid b$ teljesül, akkor $|c| \leq |d|$.*

3.1.4. Definíció (Felbonthatatlan szám). *A p egységtől és nullától különböző számot felbonthatatlan számnak nevezzük, ha*

$$p = ab \quad \Rightarrow \quad a \text{ vagy } b \text{ egység} \quad \forall a, b \in \mathbb{Z}.$$

3.1.5. Definíció (Prímszám). *A p egységtől és nullától különböző számot prímszámnak nevezzük, ha*

$$p \mid ab \quad \Rightarrow \quad p \mid a \text{ vagy } p \mid b \quad \forall a, b \in \mathbb{Z}$$

3.1.1. Tétel. *Az egész számok körében p akkor és csak akkor prím, ha felbonthatatlan.*

3.1.6. Definíció (Relatív prím). *Az $a, b \in \mathbb{Z}$ számokat relatív prímeknek nevezzük, ha $lnko(a, b) = 1$.*

3.1.2. Tétel (A számelmélet alaptétele). *Minden $1 < n \in \mathbb{Z}$ szám felírható $n = p_1^{k_1} p_2^{k_2} \cdots p_r^{k_r}$ alakban, ahol $p_1, p_2, \ldots, p_r$ különböző prímek és $k_1, k_2, \ldots, k_r$ pozitív egészek. Az $n = p_1^{k_1} p_2^{k_2} \cdots p_r^{k_r}$ felírást a szám kanonikus alakjának nevezzük és ez a $p_i^{k_i}$ tényezők sorrendjétől eltekintve egyértelmű.*

3.1.7. Definíció (B-sima). *Legyen $B_0 = \{p_1, \ldots, p_k\}$ prímek egy halmaza, és $B = B_0 \cup \{-1\}$. Ekkor $k \in \mathbb{Z}$-t akkor nevezzük B-simának, ha k kanonikus alakjában csak B-beli prímek szerepelnek. Szokás a B-sima számokat úgy is definiálni, hogy k akkor B-sima, ha az összes prím osztója kisebb mint B. Általában ha egy számra csak azt mondjuk, hogy sima, akkor ez utóbbi definícióra gondolunk, valamilyen kis B-vel.*

3.1.8. Definíció (Euler féle φ függvény). *Legyen $n \in \mathbb{N}$, ekkor*

$$\varphi(n) = |\{k \in \mathbb{Z} \,|\, 0 < k \leq n : lnko(n,k) = 1\}|$$

azaz $\varphi(n)$ az n-nél nem nagyobb, n-hez relatív prímek száma.

3.1.3. Tétel. *Ha n kanonikus alakja $n = p_1^{k_1} \cdot p_2^{k_2} \cdots p_r^{k_r}$, akkor*

$$\varphi(n) = (p_1^{k_1} - p_1^{k_1-1})(p_2^{k_2} - p_2^{k_2-1}) \cdots (p_r^{k_r} - p_r^{k_r-1}).$$

3.1.4. Tétel (Euler-tétel). *Ha $n \in \mathbb{N}$ és $lnko(a,n) = 1$, akkor $a^{\varphi(n)} \equiv 1 \pmod{n}$.*

3.1.5. Tétel (A "kis" Fermat-tétel). *Ha $a \in \mathbb{Z}$, p prím és $lnko(a,p) = 1$, akkor $a^{p-1} \equiv 1 \pmod{p}$.*

3.1.9. Definíció (Kvadratikus maradék). *Legyen $p > 1$ prímszám, valamint $lnko(n,p) = 1$. Ekkor n-et kvadratikus maradéknak nevezzük modulo p, ha az $x^2 \equiv n \pmod{p}$ kongruencia megoldható, ellenkező esetben pedig kvadratikus nemmaradéknak hívjuk. (Az $n \equiv 0 \pmod{p}$ számokat nem hívjuk egyiknek sem.)*

3.1.10. Definíció (Legendre-szimbólum). *Legyen p prím. Ekkor az $\left(\frac{a}{p}\right)$ Legendre-szimbólumot a következőképpen értelmezzük:*

$$\left(\frac{a}{p}\right) = \begin{cases} 1, & \text{ha } a \text{ kvadratikus maradék} \pmod{p} \\ -1, & \text{ha } a \text{ kvadratikus nemmaradék} \pmod{p} \\ 0, & \text{ha } a \equiv 0 \pmod{p} \end{cases}$$

3.1.6. Tétel. *A Legendre-szimbólum tulajdonságai:*

1. $a \equiv b \pmod{p} \quad \Rightarrow \quad \left(\frac{a}{p}\right) = \left(\frac{b}{p}\right)$
2. $\left(\frac{ab}{p}\right) = \left(\frac{a}{p}\right)\left(\frac{b}{p}\right)$
3. $\left(\frac{-1}{p}\right) = \begin{cases} 1, & \text{ha } p \equiv 1 \pmod{4} \\ -1, & \text{ha } p \equiv -1 \pmod{4} \end{cases}$
4. $\left(\frac{2}{p}\right) = \begin{cases} 1, & \text{ha } p \equiv \pm 1 \pmod{8} \\ -1, & \text{ha } p \equiv \pm 3 \pmod{8} \end{cases}$

3.1.7. Tétel (Kvadratikus reciprocitás). *Ha $p > 2$ és $q > 2$ különböző prímszámok, akkor*

$$\left(\frac{q}{p}\right) = \begin{cases} -\left(\frac{p}{q}\right), & \text{ha } p \equiv q \equiv -1 \pmod 4 \\ \left(\frac{p}{q}\right), & \text{egyébként} \end{cases}$$

3.1.11. Definíció (Lánctört). *Legyen x valós szám. Ekkor x lánctörtjegyein a következő $c_0, c_1, \ldots$ egész számokat értjük:*

$$\begin{array}{rclrcl} & & & x_0 & = & x \\ c_0 & = & \lfloor x_0 \rfloor & x_1 & = & \frac{1}{x_0 - c_0} \\ c_1 & = & \lfloor x_1 \rfloor & x_2 & = & \frac{1}{x_1 - c_1} \\ & \vdots & & & \vdots & \\ c_i & = & \lfloor x_i \rfloor & x_{i+1} & = & \frac{1}{x_i - c_i} \\ & \vdots & & & \vdots & \end{array}$$

A definícióból következik, hogy minden valós számnak létezik egyértelmű lánctört alakja, valamint $c_i > 0$, ha $i \geq 1$.

3.1.1. Lemma. *Ha $c_0, c_1, \ldots$ az x valós szám lánctörtjegyei, akkor*

$$x = c_0 + \cfrac{1}{c_1 + \cfrac{1}{c_2 + \cfrac{1}{c_3 + \cfrac{1}{\ddots}}}}$$

x lánctört alakjára a következő jelölést használjuk: $x = [c_0, c_1, c_2, c_3 \ldots]$

3.1.8. Tétel. *Minden $x \in \mathbb{R} \backslash \mathbb{Q}$ esetén az x-et a megfelelő értelemben legjobban közelítő racionális szám megkapható x lánctört alakjának véges kiértékelésével. A megfelelő értelemben vett legjobb közelítés itt azt jelenti, hogy csak a számláló növelésével lehet ennél közelebbi racionális számot megadni. A kiértékelést a következőképpen végezhetjük rekurzívan:*

$$x \approx [c_0, c_1, c_2, \ldots, c_k] = \frac{p_k}{q_k} \quad \text{, ahol}$$

$$\frac{p_0}{q_0} = c_0$$

$$\frac{p_1}{q_1} = \frac{c_0 c_1 + 1}{c_1}$$

$$\vdots$$

$$\frac{p_i}{q_i} = \frac{c_i p_{i-1} + p_{i-2}}{c_i q_{i-1} + c_{i-2}}.$$

Ekkor a közelítés hibája a nevező négyzetével arányos, azaz:

$$|x - \frac{p_k}{q_k}| \leq \frac{1}{q_k^2}.$$

3.1.9. Tétel. *Minden nem négyzetszám $r \in \mathbb{Q}$ esetén*

$$\sqrt{r} = [a_0, \overline{a_1, a_2, \ldots, a_2, a_1, 2a_0}].$$

3.2. Algebrai alapfogalmak

3.2.1. Definíció (Csoport). *A $(G, \circ)$ algebrai struktúrát csoportnak nevezzük, ha G tetszőleges nem üres halmaz és $\circ : G \times G \to G$ olyan függvény, melyre teljesülnek a következő tulajdonságok (csoportaxiómák):*

1. *$a, b, c \in G \Rightarrow (a \circ b) \circ c = a \circ (b \circ c)$ (asszociativitás)*
2. *$\exists\, e \in G : e \circ a = a \circ e = a \quad \forall\, a \in G$-re (egységelem létezése)*
3. *$a \in G \Rightarrow \exists\, a^{-1} : a \circ a^{-1} = a^{-1} \circ a = e$, ahol e-re teljesül a 2. axióma. (inverz elemek létezése)*

3.2.2. Definíció (Abel-csoport). *Egy G csoportot Abel-csoportnak nevezünk, ha $\forall a, b \in G : a \circ b = b \circ a$*

3.2.3. Definíció (Ciklikus csoport). *Egy G csoportot ciklikusnak nevezünk, ha létezik olyan $g \in G$ elem, melyre $\forall a \in G\, \exists\, k : g^k = a$.*

3.2.4. Definíció (Csoport rendje). *Egy G csoport elemszámát G rendjének nevezzük és $||G||$-vel jelöljük.*

3.2.5. Definíció (Elem rendje). *Egy $a \in G$ elem rendje az a legkisebb k egész, melyre $a^k = e$. Ha ilyen k nem létezik, a rendjét végtelennek mondjuk.*

3.2.6. Definíció (Gyűrű). *Az $(R, \oplus, \otimes)$ algebrai struktúrát gyűrűnek nevezzük, ha R nemüres halmaz,*

1. *$\oplus$ egy R-en értelmezett művelet, melyet összeadásnak nevezünk. $\oplus$ asszociatív és kommutatív, létezik nullelem, valamint minden elemnek létezik ellentettje*

2. $\otimes$ *egy* R*-en értelmezett művelet, melyet szorzásnak nevezünk.* $\otimes$ *asszociatív*

3. *Minden* $a, b, c \in R$*-re* $a \otimes (b \oplus c) = a \otimes b + a \otimes c$ *és* $(b \oplus c) \otimes a = b \otimes a + c \otimes a$ *teljesül*

3.2.7. Definíció (Test). *Egy* $(T, \oplus, \otimes)$ *gyűrűt testnek nevezünk, ha* $\otimes$ *kommutatív, létezik nullelem és minden elemnek létezik ellentettje.*

3.2.8. Definíció (Test karakterisztikája). *Egy* T *test karakterisztikája az a* k *egész szám, ahányszor össze kell adnunk a test (multiplikatív) egységelemét, hogy megkapjuk a nullelemet. Ha ez sosem történik meg, a karakterisztika nulla.*

3.2.9. Definíció (Ideál). *Egy* R *gyűrű nemüres* $I \subseteq R$ *részhalmazát* R *ideáljának nevezzük, ha*

1. $i, j \in I \quad \Rightarrow \quad i + j \in I$

2. $-i \in I$

3. $i \in I,\ r \in R \quad \Rightarrow \quad ri \in I,\ ir \in I$

3.2.10. Definíció (Főideál). *Legyen* a *az* R *(kommutatív, egységelemes, nullosztómentes) gyűrű tetszőleges eleme. Ekkor az* $\{ra \,|\, r \in R\}$ *halmazt az* a *elem által generált főideálnak nevezzük és* $< a >$*-val jelöljük.*

3.2.11. Definíció (Ideál szerinti maradékosztály). *Legyen* $< a >$ *az* I *ideál egy főideálja, ekkor az*

$$a + I = \{a + i \,|\, i \in I\}$$

halmazt az a *által reprezentált maradékosztálynak nevezzük.*

3.2.12. Definíció (Faktorgyűrű). *Legyen* I *ideál az* R *gyűrűben. Ekkor az* I *szerinti* $a + I = \{a + i | i \in I\}$ *maradékosztályok az* R *gyűrű diszjunkt részhalmazai, melyek egyesítése* R *és ezek a*

- $[a + I] + [b + I] = [a + b] + I$

- $[a + I][b + I] = ab + I$

műveletekre nézve gyűrűt alkotnak. Ezt az R *gyűrű* I *szerinti maradékosztálygyűrűjének vagy faktorgyűrűjének nevezzük és* R/I*-vel jelöljük.*

3.2.13. Definíció (Ideál osztója). *A B ideál osztója az A ideálnak, ha létezik olyan C ideál, amellyel $BC = A$. Ezt $B \mid A$-val jelöljük.*

3.2.14. Definíció (Felbonthatatlan ideál). *Az R gyűrű egy nemtriviális (vagyis $<0>$-tól és $<1>$-től különböző) F ideálja felbonthatatlan, ha*

$$F = AB \quad \Rightarrow \quad A =<1> \text{ vagy } B =<1>.$$

3.2.15. Definíció (Prímideál). *Az R gyűrű egy nemtriviális P ideálja prímideál, ha*

$$P \mid AB \quad \Rightarrow \quad P \mid A \text{ vagy } P \mid B.$$

3.2.1. Tétel. *Ha egy R gyűrűben igaz az egyszerűsítési szabály, azaz*

$$AB = AC,\ A \neq <0> \quad \Rightarrow \quad B = C$$

és

$$B \mid A \Leftrightarrow A \subseteq B$$

akkor egy P ideál akkor és csak akkor prímideál, ha felbonthatatlan ideál.

3.2.16. Definíció (Dedekind-gyűrű). *Ha egy R gyűrű ideáljaira teljesül a "számelmélet alaptétele", azaz bármely $<0>$-tól és $<1>$-től különböző ideál prímideálok szorzatára bontható és ez a felbontás a tényezők sorrendjétől eltekintve egyértelmű, akkor R-et Dedekind-gyűrűnek nevezzük.*

3.2.17. Definíció (Homomorfizmus). *Ha G, H csoportok és $\phi : G \to H$ olyan leképezés, amire $\forall g, h \in G :\ \phi(gh) = \phi(g)\phi(h)$, akkor ϕ-t homomorfizmusnak nevezzük.*

3.2.18. Definíció (Izomorfizmus). *A bijektív homomorfizmusokat izomorfizmusoknak nevezzük. Ha két csoport között létezik izomorfizmus, akkor a két csoportot izomorfnak nevezzük.*

3.2.19. Definíció (Automorfizmus). *A ϕ izomorfizmust automorfizmusnak nevezzük, ha $\phi : G \to G$.*

3.2.20. Definíció (Gyűrű-homomorfizmus). *Legyenek $(R, +, \cdot)$, $(S, \oplus, \otimes)$ gyűrűk. Ekkor egy $\phi : R \to S$ leképezést gyűrű-homomorfizmusnak nevezünk, ha*

1. $\phi(a + b) = \phi(a) \oplus \phi(b)$

2. $\phi(a \cdot b) = \phi(a) \otimes \phi(b)$

3.2.21. Definíció (Testbővítés). *Az M testet az L test bővítésének nevezzük, ha L részteste M-nek, azaz $L \subseteq M$ és az L testben a műveletek éppen az M-beli műveletek megszorításai.*

3.2.22. Definíció (Algebrai szám). *Legyen L részteste M-nek. A $\theta \in M$ elem algebrai az L test felett, ha létezik olyan nemnulla $f \in L[x]$ polinom, amelynek θ gyöke, azaz $f(\theta) = 0$. Ha ez az f polinom egész együtthatós, akkor θ-t algebrai egésznek nevezzük.*

3.2.23. Definíció (Minimálpolinom). *Legyen L részteste M-nek. Az L felett algebrai $\theta \in M$ elem minimálpolinomja a(z egyik) legalacsonyabb fokú $f \in L[x]$ polinom, amelynek θ gyöke. A θ foka a minimálpolinomjának a foka.*

3.2.24. Definíció (Egyszerű bővítés). *Tetszőleges $\theta \in \mathbb{C}$ esetén $\mathbb{Q}$ test θ-val történő egyszerű bővítésének nevezzük és $\mathbb{Q}(\theta)$-val jelöljük az alábbi alakú komplex számok halmazát:*

$$\frac{g(\theta)}{h(\theta)}, \qquad \text{ahol } g, h \in \mathbb{Q}[x], \quad h(\theta) \neq 0.$$

Ha θ algebrai szám, akkor egyszerű algebrai bővítésről beszélünk.

3.2.2. Tétel. *Ha θ n-edfokú algebrai szám, akkor $\mathbb{Q}(\theta)$ elemei egyértelműen felírhatók a következő alakban:*

$$a_0 + a_1\theta + \ldots + a_{n-1}\theta^{n-1} \qquad (a_i \in \mathbb{Q}).$$

3.2.3. Tétel. *Ha $f \in \mathbb{Q}[x]$ egy normált (azaz 1 főegyütthatós), irreducibilis polinom $\theta \in \mathbb{C}$ gyökkel, akkor a $\mathbb{Q}(\theta)$-ban lévő algebrai egészek halmaza részgyűrűt alkot $\mathbb{Q}(\theta)$-ban. Jelöljük ezt a részgyűrűt $\mathfrak{D}$-vel.*

3.2.4. Tétel. *Ha $f \in \mathbb{Z}[x]$ egy normált, irreducibilis, d fokú polinom $\theta \in \mathbb{C}$ gyökkel, akkor $\{1, \theta, \theta^2, \ldots, \theta^{d-1}\}$ elemeinek $\mathbb{Z}$-lineáris kombinációi részgyűrűt alkotnak $\mathfrak{D}$-ben. Jelöljük ezt a részgyűrűt $\mathbb{Z}[\theta]$-val.*

3.2.5. Tétel. *Legyen $f \in \mathbb{Z}[x]$ egy normált, irreducibilis polinom $\theta \in \mathbb{C}$ gyökkel. Ekkor $\mathfrak{D}$ Dedekind-gyűrű és igazak a következők:*

1. *$\mathfrak{D}$ Noether-féle, azaz az ideálok minden nem üres halmazában van tartalmazásra maximális*

2. *$\mathfrak{D}$ prímideáljai maximális ideálok és fordítva*

3. *Az ideálszorzásra nézve $\mathfrak{D}$ ideáljai egyértelműen felbonthatóak $\mathfrak{D}$ prímideáljainak szorzatára*

3.2.25. Definíció (Algebrai szám konjugáltja). *Egy θ algebrai szám minimálpolinomjának (komplex) gyökeit a θ $\mathbb{Q}$ feletti konjugáltjainak nevezzük.*

3.2.26. Definíció (Algebrai szám relatív konjugáltja). *Legyenek a θ algebrai szám konjugáltjai*

$$\theta_{(1)} = \theta, \theta_{(2)}, \ldots, \theta_{(n)}$$

és $\alpha \in \mathbb{Q}(\theta)$. Ekkor legyen $f \in \mathbb{Q}[x]$ az az (egyértelműen meghatározott) polinom, melyre

$$\alpha = f(\theta) \quad \text{és} \quad deg\, f \leq n-1 \text{ vagy } f = 0.$$

Ekkor az

$$f(\theta_{(j)}), \quad j = 1, 2, \ldots, n$$

számokat az α-nak a $\mathbb{Q}(\theta)$-ra vonatkozó relatív konjugáltjainak nevezzük.

3.2.27. Definíció (Norma). *Egy $\alpha \in \mathbb{Q}(\theta)$ elem normáján a relatív konjugáltjainak szorzatát értjük és $N(\alpha)$-val jelöljük:*

$$N(\alpha) = \prod_{j=1}^{n} f(\theta_{(j)}),$$

ahol θ relatív konjugáltjai $\theta_{(1)} = \theta, \theta_{(2)}, \ldots, \theta_{(n)}$ és $\alpha = f(\theta)$.

3.2.28. Definíció (Ideál normája). *Legyen az R gyűrű egy ideálja I. Ekkor az I normáján az I szerinti maradékosztályok számát értjük, azaz $N(I) = |R/I|$.*

3.2.6. Tétel. *Legyen $f \in \mathbb{Q}[x]$ egy normált, irreducibilis polinom $\theta \in \mathbb{C}$ gyökkel. Ekkor a norma egy multiplikatív függvény, ami $\mathfrak{D}$ ideáljait $\mathbb{Z}^+$-ba képezi. Sőt, ha $\alpha \in \mathfrak{D}$, akkor $N(<\alpha>) = |N(\alpha)|$.*

3.2.7. Tétel. *Legyen D Dedekind-gyűrű. Ekkor ha egy $\mathfrak{p} \in D$ ideálra $N(\mathfrak{p}) = p$ valamilyen $p \in \mathbb{Z}$ prímre, akkor $\mathfrak{p}$ prímideálja D-nek. Fordítva, ha $\mathfrak{p}$ egy prímideálja D-nek, akkor $N(\mathfrak{p}) = p^e$ valamilyen $p \in \mathbb{Z}$ prímre és $e \in \mathbb{Z}^+$-ra.*

3.2.29. Definíció (Első rendű prímideál). *Első rendű prímideálnak azt a D Dedekind-gyűrűben lévő $\mathfrak{p}$ prímideált nevezzük melyre $N(\mathfrak{p}) = p$ valamilyen $p \in \mathbb{Z}$ prímre.*

3.2.8. Tétel. *Legyen $f \in \mathbb{Z}[x]$ egy normált, irreducibilis polinom, $\theta \in \mathbb{C}$ gyökkel. Ekkor az $\{(r,p) \mid r \in \mathbb{Z}/p\mathbb{Z}, f(r) \equiv 0 \pmod{p}\}$ halmaz és a $\mathbb{Z}[\theta]$-beli elsőrendű prímideálok között bijektív kapcsolat van.*

3.2.9. Tétel. *Minden $\mathfrak{p}_i \in \mathbb{Z}[\theta]$ prímideálhoz létezik egy $l_{\mathfrak{p}_i}$ csoport-homomorfizmus, melyre igazak a következők:*

1. $l_{\mathfrak{p}_i}(\beta) \geq 0 \quad \forall \beta \in \mathbb{Q}(\theta)^*$, *ahol $\mathbb{Q}(\theta)^*$ a $\mathbb{Q}(\theta)$ nemnulla elemeinek multiplikatív csoportja*
2. $l_{\mathfrak{p}_i}(\beta) > 0 \quad \Leftrightarrow \quad \mathfrak{p}_i$ *osztója a* $<\beta>$ *főideálnak*
3. $l_{\mathfrak{p}_i}(\beta) = 0$ *véges sok* $\mathfrak{p}_i \in \mathbb{Z}[\theta]$ *prímideál kivételével és* $|N(\beta)| = \prod N(\mathfrak{p}_i)^{l_{\mathfrak{p}_i}}$ *minden* $\mathfrak{p}_i \in \mathbb{Z}[\theta]$ *prímideálra*

3.2.10. Tétel. *Legyen $\beta = a + b\theta$ alakú $\mathbb{Z}[\theta]$-beli szám, ahol a, b relatív prím, valamint legyen $\mathfrak{p} \in \mathbb{Z}[\theta]$ prímideál. Ekkor a 3.2.9 tételbeli $l_{\mathfrak{p}}(\beta) = 0$, ha $\mathfrak{p}$ nem első rendű prímideál, valamint ha $\mathfrak{p}$ első rendű prímideál, amelyhez az (r,p) 3.2.8 tételbeli számpár tartozik, akkor*

$$l_{\mathfrak{p}}(\beta) = \begin{cases} ord_p(N(\beta)), & ha\ a \equiv -br \pmod{p} \\ \\ 0, & egyébként \end{cases}$$

ahol $ord_p(N(\beta))$ jelöli az $N(\beta)$-ban lévő p kitevőjét.

3.2.11. Tétel. *Legyen $f \in \mathbb{Z}[x]$ egy normált, irreducibilis polinom $\theta \in \mathbb{C}$ gyökkel és legyen $m \in \mathbb{Z}/n\mathbb{Z}$ olyan, melyre $f(m) \equiv 0 \pmod{n}$. Ekkor a $\phi : \mathbb{Z}[\theta] \to \mathbb{Z}/n\mathbb{Z}$ leképezés amire $\phi(1) \equiv 1 \pmod{n}$ és ami θ-t m-be viszi, egy szürjektív gyűrű-homomorfizmus.*

3.3. Elliptikus görbék

3.3.1. Definíció (Elliptikus görbe). *Az $E(T)$ elliptikus görbének az*

$$y^2 + a_1xy + a_2y = x^3 + a_3x^2 + a_4x + a_5$$

egyenletet kielégítő $(x,y) \in T$ pontpárok halmazát értjük, ahol T test és $a_1, \ldots, a_5 \in T$.

3.3.2. Definíció (Szingularitás). *Egy $p \in T$ pontot az $E(T)$ görbe szinguláris pontjának nevezünk, ha a görbe összes parciális deriváltja p-ben nulla. Szinguláris pont például a görbének az önmagával vett metszéspontja, vagy egy olyan "csúcs", ahol a görbe deriváltja előjelet vált.*

3.3.3. Definíció (Szinguláris görbe). *Egy görbét szinguláris nak nevezünk, ha van egy vagy több szinguláris pontja. Egyéb esetben a görbét nemszinguláris nak nevezzük.*

3.3.4. Definíció (Elliptikus görbe $\mathbb{R}$ felett). *Egy elliptikus görbe $\mathbb{R}$ felett azon $(x, y) \in \mathbb{R}$ pontpárok halmaza, melyek kielégítik az*

$$y^2 = x^3 + ax + b$$

egyenletet, ahol $a, b \in \mathbb{R}$ és $4a^3 + 23b^2 \neq 0$. (Ez utóbbi feltétel azt garantálja, hogy a görbe ne legyen szinguláris.)
A fenti egyenletet Weierstrass egyenletnek, az általános definícióban szereplőt pedig általánosított Weierstrass egyenletnek nevezzük.

Szeretnénk a valós számok feletti elliptikus görbék pontjai között műveleteket definiálni. Legyen $p, q \in E(T)$, ekkor meghatározhatjuk a p-re és q-ra illesztett egyenes és az elliptikus görbe metszéspontját. Ha az egyenes nem függőleges és $p \neq q$, akkor a harmadik metszéspont egyértelműen létezik, ha pedig $p = q$, akkor a görbe p-beli érintője által kimetszett pontot tekintjük harmadik metszéspontnak.
Belátható, hogy ha $p = (x_1, y_1)$, $q = (x_2, y_2)$, akkor az egyenes meredeksége

$$m = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1}, & \text{ha } x_1 \neq x_2 \\ \\ \frac{3x_1^2 + a}{2y_1}, & \text{egyébként} \end{cases}$$

Ekkor a harmadik metszéspont koordinátái

$$x_3 = m^2 - x_1 - x_2 \qquad y_3 = m(x_3 - x_1) + y_1$$

Ezzel már definiálhatjuk a kívánt műveletet a görbe pontjai között:

$$(x_1, y_1) + (x_2, y_2) = \begin{cases} \infty, & \text{ha } x_1 = x_2 \text{ és } y_1 = -y_2 \\ \\ (x_3, -y_3), & \text{egyébként} \end{cases}$$

Mivel a három metszéspont csak akkor létezik, ha nem függőleges az egyenes, így erre az esetre külön ki kellett térnünk az összeadás definiálásakor. A ∞ szimbólumot erre az esetre, mint nullelemet vezettük be.

Megmutatható, hogy a görbe pontjai ezzel az összeadással Abel-csoportot alkotnak. Általánosan az is belátható, hogy egy T test feletti elliptikus görbével is Abel-csoportot kapunk, ha T karakterisztikája nem 2 vagy 3.
A műveletet a görbén $\mathrm{mod} n$ is definiálhatjuk, persze ekkor az osztás nem biztos, hogy elvégezhető, így ekkor nem kapunk csoportot. A gyakorlatban ezt fogjuk kihasználni a 5.1.5 szakaszban.

3.3.1. Tétel (Hasse tétele). *Ha $p > 3$ prímszám, akkor egy $\mathbb{Z}/p\mathbb{Z}$ felett vett elliptikus görbe rendje $p + 1 - 2\sqrt{p}$ és $p + 1 + 2\sqrt{p}$ közé esik.*

3.4. Lucas sorozatok

Legyenek λ_1 és λ_2 az egész együtthatós

$$x^2 - Px + Q = 0$$

karakterisztikus egyenlet gyökei. $\lambda_1 \neq \lambda_2$ esetén az U_n, V_n un. Lucas sorozatokat a következőképpen definiáljuk:

$$U_n = \frac{a^n - b^n}{a - b}, \quad V_n = a^n + b^n \qquad n \geq 0$$

A Lucas sorozatok gyors számolására az alábbi rekurziót használhatjuk:

$$U_{2n} = U_n V_n$$

$$V_{2n} = V_n^2 - 2Q^n$$

$$U_{2n+1} = U_{n+1} V_n - Q^n$$

$$V_{2n+1} = V_{n+1} V_n - PQ^n$$

A Lucas sorozat néhány P és Q értékre ismert számsorozatokhoz vezet:
$U_n(1, -1)$: Fibonacci számok ($F_0 = 0$, $F_1 = 1$, $F_n = F_{n-1} + F_{n-2}$)
$V_n(1, -1)$: Lucas számok ($L_n = F_{n-1} + F_{n+1}$)
$U_n(3, 2)$: Mersenne számok ($2^n - 1$)

4. Nyílt kulcsú titkosítás

Titkosírással kapcsolatos feljegyzések már az ókortól kezdve megtalálhatóak a világ több részén, természetes közegeként először a hadviselésben jelent meg, s azóta is megmaradt legfontosabb alkalmazási területeként. Az újkorig nagyrészt a monoalfabetikus[2] helyettesítéssel működő rejtjelezés volt elterjedt, 1466-ban készült el az első általunk ismert polialfabetikus[3] rejtjelezőgép Leon Battista Alberti keze által.

Az újkor végén a gyors technikai fejlődés magával vonta a kommunikáció gyorsulását is, ez pedig a kriptográfia természetes fejlődésével járt. A világháborúkban nagy szerepe volt a titkosírásnak, a háborúk a kódok szintjén is zajlottak. A 20. század második felében a számítógépek megjelenésével új szintre lépett a kommunikáció is, az addig megfejthetetlennek hitt rejtjelek gyorsan elavulttá váltak.

1976-ban jelent meg Whitfield Diffie és Martin Hellman *Új direktívák a kriptográfiában* című könyve, ők vezették be a kulcsmegosztáson alapuló kriptográfia fogalmát. Egy évvel később Ron Rivest, Adi Shamir és Len Adleman munkájának köszönhetően elkészült az első nyílt kulcsú titkosító eljárás.

4.1. Az RSA-séma

Az RSA-séma használatához szükségünk van a publikus és titkos kulcsok előállítására:

– Inicializálás

1. Válasszunk két prímet, p-t és q-t
2. Számoljuk ki $n = pq$-t, és $\varphi(n) = (p-1)(q-1)$-et
3. Válasszunk d-t, amire teljesül, hogy $lnko(d, \varphi(n)) = 1$
4. Válasszunk e-t, amire teljesül, hogy $ed \equiv 1 \pmod{\varphi(n)}$
5. Az (e, n) párt publikus-, a (d, n) párt pedig a privát kulcsnak nevezzük

– Az $\mathcal{M}$ üzenet titkosítása

1. Reprezentáljuk $\mathcal{M}$-et k db számmal, legyenek ezek $\mathcal{M}_1, \ldots, \mathcal{M}_k$
2. Minden $\mathcal{M}_i \in \mathcal{M}_0, \ldots, \mathcal{M}_k$-re számoljuk ki $\mathcal{C}_i = \mathcal{M}_i^e \pmod{n}$-et

[2] A titkosítandó szöveg egészében ugyanazt a helyettesítést használó módszer.

[3] A titkosítandó szövegen többféle helyettesítést használó módszer, ilyen lehet például ha a gyakran ismétlődő részeken nem egy helyettesítést használunk.

3. A titkosított üzenet $\mathcal{C} = \mathcal{C}_0, \ldots, \mathcal{C}_k$

– A $\mathcal{C} = \mathcal{C}_0, \ldots, \mathcal{C}_k$ titkosított üzenet dekódolása

1. Minden $\mathcal{C}_i \in \mathcal{C}_0, \ldots, \mathcal{C}_k$-ra számoljuk ki $\mathcal{M}_i = \mathcal{C}_i^d \pmod{n}$-et
2. A dekódolt üzenet $\mathcal{M} = \mathcal{M}_0, \ldots, \mathcal{M}_k$

Példa az RSA-séma alkalmazására

Tegyük fel, hogy Alice titkosított üzenetet szeretne küldeni Bobnak. A következőképp kell eljárnia:

– Inicializálás

1. Legyen $p = 11$ és $q = 29$
2. Ekkor $n = pq = 319$, és $\varphi(n) = (p-1)(q-1) = 280$
3. Legyen $d = 17$, mivel $lnko(17, 280) = 1$, ezért ez megfelelő választás
4. Legyen $e =$ 33-t, mivel $33 \cdot 17 = 561 \equiv 1 \pmod{280}$, ezért ez megfelelő választás
5. A $(33, 319)$ pár a publikus-, a $(17, 319)$ pár pedig a privát kulcs

– Az $\mathcal{M}$="nem túl biztonságos..." üzenet titkosítása

1. Reprezentáljuk az üzenetet a betűk ASCII kódjának megfelelő számokkal: $\mathcal{M}$="nem túl biztonságos..." = 110, 101, 109, 032, 116, 250, 108, 032, 098, 105, 122, 116, 111, 110, 115, 225, 103, 111, 115, 046, 046, 046
2. Minden $\mathcal{M}_i \in \mathcal{M}_0, \ldots, \mathcal{M}_k$-ra számoljuk ki $\mathcal{C}_i = \mathcal{M}_i^e \pmod{n}$-et:
   $110^{33} \pmod{319} = 286$
   $101^{33} \pmod{319} = 19$
   $\vdots$
   $046^{33} \pmod{319} = 162$
3. A titkosított üzenet $\mathcal{C}$=286, 19, 274, 98, 29, 160, 234, 98, 43, 73, 265, 29, 210, 286, 202, 158, 284, 210, 202, 162, 162, 162

– A $\mathcal{C}$=286, 19, 274, 98, 29, 160, 234, 98, 43, 73, 265, 29, 210, 286, 202, 158, 284, 210, 202, 162, 162, 162 üzenet dekódolása

1. Minden $\mathcal{C}_i \in \mathcal{C}_0, \ldots, \mathcal{C}_k$-ra számoljuk ki $\mathcal{M}_i = \mathcal{C}_i^d \pmod{n}$-et:
   $286^{17} \pmod{319} = 110$
   $019^{17} \pmod{319} = 101$
   $\vdots$
   $162^{17} \pmod{319} = 46$
2. A dekódolt üzenet $\mathcal{M}$ = 110, 101, 109, 032, 116, 250, 108, 032, 098, 105, 122, 116, 111, 110, 115, 225, 103, 111, 115, 046, 046, 046 = "nem túl biztonságos..."

4.2. Helyesség

Ebben a részben az RSA-séma helyességét bizonyítjuk. Tegyük fel, hogy kiválasztottuk az összes szükséges számot a módszer használatához. Ekkor azt kell belátnunk, hogy az $\mathcal{M}$ üzenetet kódolva, majd dekódolva visszakapjuk $\mathcal{M}$-et, azaz

$$(\mathcal{M}^e)^d = \mathcal{M} \pmod{n}.$$

Az e és d megválasztásakor feltétel volt, hogy e a d multiplikatív inverze legyen, azaz

$$ed \equiv 1 \pmod{\varphi(n)} \quad \Rightarrow \quad ed = k \cdot \varphi(n) + 1 \qquad (k \in \mathbb{N}).$$

Ezt felhasználva a következőt kapjuk:

$$\begin{aligned} (\mathcal{M}^e)^d &= \mathcal{M}^{ed} && \pmod{n} \\ &= \mathcal{M}^{k\cdot\varphi(n)+1} && \pmod{n} \end{aligned}$$

A "kis" Fermat tételből könnyen látható, hogy minden p-re és $\mathcal{M}$-re ahol $p \nmid \mathcal{M}$:

$$\begin{aligned} \mathcal{M}^{p-1} \equiv 1 \pmod{p} \quad &\Rightarrow \quad \mathcal{M}^{k\cdot(p-1)} \equiv 1 && \pmod{p} \\ &\Rightarrow \quad \mathcal{M}^{k\cdot(p-1)+1} \equiv \mathcal{M} && \pmod{p} \quad (k \in \mathbb{N}) \end{aligned}$$

Alkalmazva ezt p-re és q-ra is, azt kapjuk, hogy

$$\begin{aligned} \mathcal{M}^{k\cdot(p-1)+1} &\equiv \mathcal{M} \pmod{p} \\ \mathcal{M}^{k\cdot(q-1)+1} &\equiv \mathcal{M} \pmod{q} \end{aligned}$$

és ezt felhasználva

$$\begin{aligned} \mathcal{M}^{k\cdot(p-1)(q-1)+1} &\equiv \mathcal{M} \pmod{pq} \\ \mathcal{M}^{k\cdot\varphi(n)+1} &\equiv \mathcal{M} \pmod{n} \\ \mathcal{M}^{ed} &\equiv \mathcal{M} \pmod{n} \end{aligned}$$

ami pont a bizonyítandó állítás.

4.3. Biztonság

Az előző pontban láttuk, hogy az RSA-séma helyesen működik, de arról még nem bizonyosodtunk meg, hogy tényleg biztonságban van az üzenetünk, ha ezt a módszert használjuk. Könnyen látható, hogy n osztóinak ismeretében megtalálhatjuk d-t, mivel csak meg kell oldanunk az $ed \equiv 1 \pmod{(p-1)(q-1)}$ lineáris kongruenciát, amit hatékonyan meg is tudunk tenni.
Felmerül tehát a kérdés, hogy a prímfaktorizáció nehézsége elegendő védelem-e. Valójában azonban az nem bizonyított még, hogy egy RSA-val kódolt üzenet dekódolása éppen olyan nehézségű feladat mint a prímfaktorizáció, de azt látjuk hogy a visszafelé irány teljesül. De ha tudnánk, hogy a két probléma ekvivalens, akkor sem nyugodhatnánk még meg teljesen, mivel a prímfaktorizációról nem tudjuk, hogy valóban annyira nehéz probléma mint amilyennek reméljük.
Matematikailag a problémák nehézségét a bonyolultság-osztályokkal jellemezhetjük jól, az RSA-val kapcsolatban az volna a jó, ha kiderülne, hogy az egészek prímfaktorizációja valamilyen nehéz bonyolultság-osztályba tartozik. Bár bizonyítva nincs ilyesmi, de annak alapján hogy milyen régóta megoldatlan a probléma, megalapozottnak gondolhatnánk a sejtést, hogy ez a feladat nem oldható meg polinom időben.
Most, hogy kiderült hogy gyakorlatilag semmilyen kézzel fogható bizonyítékunk nincs arra, hogy a módszer valóban biztonságos, fel kell tennünk a kérdést, hogy miért is használnánk akkor ezt a rendszert? A válasz egyszerűen annyi, hogy a kifejlesztése óta eltelt 36 évben semmilyen olyan módszer nem került napvilágra, amellyel hatékonyan megfejthető lenne egy RSA-val titkosított üzenet. Minden eddigi próbálkozás valamilyen a felhasználó által elkövetett - és így könnyen orvosolható - hibára vezethető vissza. Az ilyen támadások 4 kategóriába sorolhatóak:

1. Implementációs problémákat kihasználó támadások
2. A homomorf struktúrát kihasználó támadások
3. Rosszul választott paramétereket kihasználó támadások
4. Könnyű faktorizációt kihasználó támadások

4.3.1. Implementációs támadások

Az ebbe a kategóriába tartozó támadások az implementációhoz kapcsolódó hibákat vagy gyengeségeket kihasználva szereznek információt az üzenetről vagy a paraméterekről.

Idő alapú támadások

Az idő alapú támadások a titkosítás folyamatának az idejét mérik, az ilyen támadásokhoz szükséges ismerni a titkosítást végző hardware-t is, ami általában egy különálló modul. Kocher megmutatta, hogy csupán az idő mérésével meghatározható a titkos kulcs. Az ilyen támadások ellen az egyik módszer amit használhatunk, hogy úgy alakítjuk ki az titkosítást végző algoritmust, hogy minden kódolást és dekódolást ugyanannyi idő alatt végezzen el. Ezt a gyakorlatban nem használják, mivel egyrészt nehéz kivitelezni, másrészt pedig rendkívül csökken az adott hardware hatékonysága, ha arra kényszerül folyton, hogy a gyors műveleteket is lassan végezze el.
A másik módszert 1982-ben Chaum mutatta be, technikáját vakításnak nevezte el. A módszer alapja, hogy a titkosított $\mathcal{C}$ üzenet helyett egy $\mathcal{C}' = c \cdot r^e \pmod{n}$ üzenetet dekódoljon a rendszer, ahol r egy véletlenszerűen választott $\mathbb{Z}/n\mathbb{Z}$-beli szám. Ekkor a dekódolás után megkapott $\mathcal{M}'$-ből megkaphatjuk az eredeti üzenetet, ha leosztunk r-el: $\mathcal{M} = \mathcal{M}'/r$. Ezzel a módszerrel kiküszöbölhetőek az idő alapú támadások, mivel a dekódolás közben egy a támadó számára ismeretlen szöveget dekódol a rendszer, így a támadással kinyerhető adatok sem lesznek a valóságnak megfelelőek.

Teljesítmény analízis

A teljesítmény analízis az előző támadási formához nagyon hasonló, de itt nem az időt, hanem a titkosítást végző eszköz által felvett energia mennyiségét méri a támadó. Természetesen itt is szükséges ismernie a támadónak az adott hardware tulajdonságait és csak akkor használhatja ezt a módszert, ha a titkosítást végző modul energiafelvételéről pontos adatai vannak, vagyis ha nem egy különálló modul végzi a titkosítást - mint mondjuk egy otthoni PC-ben - akkor lehetetlen megállapítani, hogy pontosan mekkora energiát igényelt a kódolás/dekódolás. A legkönnyebb védekezés az ilyen támadás ellen, ha a támadót soha nem engedjük közel a hardwarehez, de megtehetjük azt is, hogy az áramfelvételt egy zajos jellel terheljük, így elkerülve az információ kinyerését. A teljesítmény analízishez hasonló támadások léteznek a titkosító modul által kibocsátott elektromágneses sugárzást mérve is, de az ilyen módszerek mind megakadályozhatóak azzal, ha a támadót fizikailag elkülönítjük a titkosítást végző géptől.

4.3.2. Homomorf struktúra elleni támadások

Az RSA homomorf struktúrája azt jelenti, hogy a módszer által használt kódoló eljárás homomorf, azaz

$$(\mathcal{M}_1\mathcal{M}_2)^e \quad \equiv \quad \mathcal{M}_1^e\mathcal{M}_2^e \quad \equiv \quad \mathcal{C}_1\mathcal{C}_2$$

Tegyük fel, hogy Trudy (a támadó) dekódolni akar egy $\mathcal{C}$ üzenetet. Ehhez felhasználja a naív Alice-t, akit rávesz, hogy dekódoljon egy $\mathcal{C}' = \mathcal{C} \cdot r^e \pmod{n}$ üzenetet, ahol $r \in \mathbb{Z}_n$ véletlenszerű és (e, n) Alice nyilvános kulcsa.
Ekkor azzal, hogy Alice dekódolja $\mathcal{C}'$-t, dekódolja $\mathcal{C}$-t is, de erről ő mit sem tud. Ha a dekódolt $\mathcal{M}'$ üzenetet Trudy megszerzi, akkor a vakítás technikájához hasonlóan r-el leosztva megkaphatja az eredeti $\mathcal{M}$ üzenetet.
Ez a módszer természetesen csak a nagyon ügyetlen Alice ellen vethető be, a gyakorlatban kevésbé hatékonyan alkalmazható.

4.3.3. Rossz paramétereket kihasználó támadások

Abban az esetben, ha az e vagy d paramétert nem kellő körültekintéssel választjuk meg, néhány ezt kihasználó támadásnak tehetjük ki magunkat.

Kis d választása

1989-ben Martin Wiener publikált egy lánctörteket használó módszert, mellyel kis d esetén felfedhető annak értéke. Megmutatta, hogy ha $d < n^{1/4}$ akkor nem biztonságos a titkosítás. 10 évvel később Boneh és Durfee módosította ezt a korlátot $n^{0.292}$-re, de javaslatuk szerint ne válasszunk $n^{1/2}$-nél kisebb d-t.

Kis e választása

Bár az előző pont miatt azt gondolhatnánk, hogy kis a e választása hasonlóan veszélyes, valójában itt nem annyira rossz a helyzet. Kis e választása esetén csak akkor határozható meg d értéke, ha a támadónak rendelkezésére áll néhány lineárisan összefüggő kódolt üzenet. Ilyen lehet például egy ismétlődő elköszönés az üzenetek végén.
A gyakorlatban használt RSA algoritmusok a kódolás előtt módosítanak az üzeneten amiatt, hogy ezt elkerüljék, de Coppersmith megmutatta, hogy elég nagy módosításra van szükség ahhoz, hogy a támadás teljesen kivédhető legyen. A jó védekezés természetesen a nagy e választása.

Közös paraméterek használata

Abban az esetben, ha valaki közös modulust használ két különböző felhasználóval való kommunikációja során, dekódolhatóak azon üzenetei, melyeket mindkét felhasználónak elküldött. Legyen $\mathcal{C}_1$ és $\mathcal{C}_2$ az $\mathcal{M}$ üzenet kódolt változatai, melyeket az e_1 illetve e_2 exponensekkel kódoltak.

Ekkor mivel e_1 és e_2 relatív prímek, ezért a támadó kereshet olyan x, y párt, amire $xe_1 + ye_2 \equiv 1 \pmod{n}$. Ekkor $\mathcal{M}$-et előállíthatja a következő módon:

$$\mathcal{C}_1^x \mathcal{C}_2^y \equiv \mathcal{M}^{xe_1+ye_2} \equiv \mathcal{M} \pmod{n}$$

Érdemes megjegyezni, hogy ennél a támadásnál bár az üzenetet elolvasta a támadó, d-t nem tudta meg.

4.3.4. Könnyű faktorizációt kihasználó támadások

Láttuk, hogy n faktorizálásával megszerezhető d, így ügyelnünk kell arra, hogy ezt megnehezítsük. A legkevesebb amit tehetünk, hogy p-t és q-t nagynak választjuk, de ezen kívül Rivest és Silverman sok javaslatot tesz arra, hogy milyen prímeket ne válasszunk. Erős prímeknek nevezik azokat a p prímeket, melyek esetében $p + 1$-nek és $p - 1$-nek van nagy prímfaktora. Erős prímeket használva elkerülhetjük, hogy Pollard faktorizáló algoritmusai hatékonyan használhatóak legyenek. Hasonló feltételekkel választható olyan n, mellyel az összes ismert faktorizáló algoritmus nehezen boldogul.

Most, hogy tudjuk mennyire körültekintően kell eljárnunk a paraméterek megválasztásánál, megértjük Alice üzenetét a 4.1 pontban, ahol szinte az összes itt felsorolt hibát elkövettük.

5. Faktorizációs algoritmusok

A faktorizációs algoritmusok két csoportra oszthatóak, a speciális és az általános célú algoritmusok csoportjára. A speciális célú algoritmusok az egész számok egy halmazára alkalmazhatóak sikeresen, ilyen például azon számok halmaza, melyeknek csak kis osztóik vannak vagy az osztók szomszédai ilyenek. A speciális célú algoritmusok ezeken a számokon gyorsak, de a halmazon kívül vagy lassúak vagy teljesen kudarcot vallanak.

Az általános célú algoritmusok bármely egész számra alkalmazhatóak, de ugyanannyi idő alatt faktorizálják a sima egészeket, mint például az RSA modulusokat, így egy véletlenszerűen választott számra csak akkor érdemes alkalmazni őket, ha a speciális célú algoritmusokkal már kudarcot vallottunk.

Minden a dolgozatban tárgyalt algoritmus felteszi, hogy a faktorizálandó szám összetett, így minden esetben tesztelnünk kell először annak prímségét. Ezt hatékonyan megtehetjük valamelyik valószínűségi prímtesztel vagy ha biztosra akarunk menni használhatjuk a polinomidejű AKS-algoritmust is.

5.1. Speciális célú algoritmusok

A speciális célú algoritmusok a nehezen faktorizálható számokon általában elbuknak, de egy véletlenszerűen választott számnak nagy valószínűséggel vannak kis osztói, ezeket pedig a leggyorsabban a speciális célú algoritmusokkal találhatjuk meg.

Legtöbbször nem tudjuk, hogy a faktorizálandó szám melyik algoritmus speciális halmazába tartozik, ezért azt sem, hogy melyiket érdemes használnunk. A gyakorlatban, ha kis számot kell faktorizálnunk, akkor a próbaosztással és a ϱ módszerrel próbálkozzunk, egyébként az ECM algoritmussal választhatjuk le a kis osztókat, s ha még így sem jártunk teljes sikerrel, akkor érdemes áttérni az általános célú algoritmusokra és bevetni a QR algoritmust vagy a GNFS-t.
A legegyszerűbb speciális célú algoritmus a próbaosztás.

5.1.1. Próbaosztás (trial division)

A próbaosztás a leglassabb, viszont a legkönnyebben megérthető algoritmus mellyel találkozhatunk a témában. Alapgondolata, hogy ha n a faktorizálandó egész, akkor teszteljünk le minden 1-nél nagyobb, de n-nél kisebb egész számot, hogy osztója-e n-nek. Ha osztót találtunk írjuk le, majd folytassuk a tesztelést az $n/\{\text{a talált osztó}\}$ számmal. Nyilván n minden osztója beleesik az $1, \ldots, n$ intervallumba, így előbb vagy utóbb megtaláljuk a teljes felbontást. Érdemes meggondolni, hogy az egészek 80%-ának van 100-nál kisebb osztója, 92%-ának pedig 1000-nél kisebb, így bár ez az algoritmus

egy olyan számot aminek csak nagy osztói vannak nagyon lassan faktorizál, de egy "átlagos" számnak gyorsan talál osztót.

Mivel a prímeken kívül minden n egésznek van osztója a $\{2, 3, 4, \ldots, \lceil\sqrt{n}\rceil\}$ halmazban, így valójában a tesztelést nem szükséges folytatnunk $\lceil\sqrt{n}\rceil$ után (hiszen ekkor már megtaláltuk az összes prímosztót).

Könnyen látható az is, hogy a próbaosztás (az osztások sorrendje miatt) minden esetben prím osztókat talál. Kihasználhatjuk ezt úgy, hogy a tesztelés során minden összetett számot kihagyunk. A $\{2, 3, 4, \ldots, k\}$ halmazban nagyjából $\frac{k}{\log k}$ prím található a prímszámtétel szerint, így jelentősen csökken a tesztelések száma ezzel a módszerrel.

Az összetett számok kihagyása felveti a kérdést, hogy hogyan menjünk végig a prímeken. Ősi módszer Eratoszthenész szitája, mellyel kiszitálhatjuk n-ig az összetett számokat $\mathcal{O}(n(\log n)(\log(\log(n))))$ bitművelettel. Ekkor tárolhatjuk egy listában a megmaradt prímeket, így elkerülve, hogy minden használatkor elő kelljen állítani azokat. A lista tárolása viszont tárhelyigényes, ami nem mindig áll rendelkezésre, például számológépek esetén sem. Prímlista helyett generálhatunk olyan pszeudo-prím sorozatot, ami lefedi a prímek halmazát, ilyen például a következő:

$$2, 3, 6k \pm 1 \qquad k \in \mathbb{Z}, k > 0$$

Könnyen igazolható, hogy a sorozat tényleg jó, hiszen ha $l \in \{0, 2, 3, 4\}$, akkor minden $6k+l$ alakú szám 2 vagy 3 többszöröse. Természetesen kapunk néhány összetett számot is a sorozatba, de az a kevés osztás amit ezekre kell fordítanunk, nem nagy ár a lista elkerüléséért.

Több módot is választhatunk a sorozat előállítására, például ha 5-el kezdünk, majd felváltva 2-t és 4-et adunk az aktuális számhoz, megkapjuk a sorozatot és ráadásul elkerüljük a szorzásokat is. Ezt a megoldást vázoljuk a következőkben:

1. Ha $n \equiv 0 \pmod 2$
   legyen $p = 2$ és álljunk le, az eredmény p

2. Ha $n \equiv 0 \pmod 3$
   legyen $p = 3$ és álljunk le, az eredmény p

3. Legyen $p = 3$
   Legyen $b = 2$

4. Amíg $p < \sqrt{n}$

   – Legyen $p = p + b$

– Ha $n \equiv 0 \pmod{p}$
álljunk le, az eredmény p

– Legyen $b = 6 - b$

A próbaosztás idő komplexitása

$$\mathcal{O}(\sqrt{n})$$

Látni fogunk ennél sokkal gyorsabb algoritmusokat is a következőkben, de ezért bonyolultsággal és így a nehezebb implementálhatósággal fogunk (örömmel) fizetni.

5.1.2. A ϱ módszer (The rho method, Pollard's rho)

1975-ben Pollard bemutatta az új Monte Carlo módszerét, aminek egy változatával 5 évvel később sikeresen faktorizálták a 8. Fermat-számot[4].

A ϱ módszer[5] alapja, hogy ha elő tudunk állítani olyan a_1, a_2 egészeket melyekre $a_1 \equiv a_2 \pmod{p}$, ahol p az n egy nemtriviális osztója, akkor a különbségüket véve p egy többszörösét kapjuk, amiből *lnko* művelettel jó eséllyel kinyerhető p (ha éppen n többszörösét kapjuk, akkor nem tudjuk kinyerni).
Mivel éppen p-t keressük, ezért nem tudjuk eldönteni az a_1, a_2 egészekről hogy kongruensek-e, viszont kiszámolhatjuk rögtön $lnko(n, a_2 - a_1)$-et és ha szerencsések vagyunk, akkor egy osztót kapunk.

A gyakorlatban a_1 és a_2 helyett egy $a_1, a_2, \ldots, a_k$ sorozatot használunk, melyet rekurzívan, egy f egész együtthatós polinommal állítunk elő:

$$a_{m+1} = f(a_m) \pmod{n}$$

Ez a sorozat nyilván ciklikus és $\bmod p$ sokkal hamarabb válik azzá, mint $\bmod n$. Ha ennek a ciklusnak a hossza t, akkor $a_m \equiv a_{m+t} \pmod{p}$, de nagy valószínűséggel $a_m \not\equiv a_{m+t} \pmod{n}$ és így kinyerhetjük a keresett osztót a fent leírt módon. Általában $f(a) = a^2 + c$ alakú polinomokat használunk, valamely $c \geq 1$ egésszel és $a_0 = 1$ kezdőértékkel. Pollard algoritmusa Floyd módszerét használja a ciklus megkeresésére:

1. Legyen $a = b = 2, \quad c = 1$

2. Legyen $f_c(x) = x^2 + c \pmod{n}$

[4] Az n-edik Fermat-számot a következőképpen definiáljuk: $F_n = 2^{2^n} + 1$ $(n \in \mathbb{N} \cup \{0\})$.

[5] A nevét az algoritmusban használt sorozatról kapta, mely egy idő után ciklikussá válik: ezt elképzelhetjük, mint ahogy a görög ϱ betű szára visszakanyarodik önmagába.

3. Legyen $a = f_c(a), \quad b = f_c(f_c(b))$

4. Legyen $d = lnko(a - b, n)$

5. Ha $1 < d < n$, az eredmény $p = d$

6. Ha $d = 1$, menjünk 3-ra

7. Ha $d = n$, legyen $c = c + 1$ és menjünk 2-re

Az említett 8. Fermat szám faktorizációját Richard P. Brent találta meg 1980-ban. A faktorizáció 2 óráig tartott egy UNIVAC[6] számítógépen. Brent módosított algoritmusa körülbelül 25%-al gyorsabb, mint az eredeti Pollard féle, ezért általában az ő verzióját használják a gyakorlatban. A különbség csak a ciklus megkeresésében van, Brent a saját módszerét használta erre.

Pollard ϱ algoritmusának idő komplexitására csak sejtésünk van: ha p a faktorizálandó n egy osztója és $p = \mathcal{O}(\sqrt{n})$, akkor a várható futási idő

$$\mathcal{O}(\sqrt{p}) = \mathcal{O}(n^{1/4})$$

5.1.3. A $p-1$ módszer (Pollard's $p-1$ method)

Pollard másik faktorizációs módszere a $p-1$ módszer, melyet szintén 1975-ben mutatott be. A módszer a Kis Fermat-tételt használva hatékonyan bontja prímtényezőkre azokat az egészeket, melyeknek valamelyik p prímosztója esetén $p-1$ B-sima, ahol B viszonylag kicsi.

Az algoritmus ötlete, hogy ha $p - 1$ B-sima, akkor egy n-hez relatív prím a-ra $a^{B!} - 1$ osztható p-vel a Kis Fermat-tétel szerint, de reményeink szerint n-el nem, mert ekkor az osztó kinyerhető $lnko$ számolással. A gyakorlatban $B!$-nál alacsonyabb kitevőt is használhatunk, mivel az algoritmus trükkje abban rejlik, hogy $a^c - 1$-nek rengeteg osztója van, ha c sima.

Az algoritmus:

1. Válasszuk ki a B simasági korlátot

2. Válasszunk a-t

3. Minden $q \leq B$ prímre:

[6] A **UNIVAC** a **UNIV**ersal **A**utomatic **C**omputer kifejezés rövidítése, ez volt 1951-ben az első kereskedelmi számítógép amerikában, alkotói J. Presper Eckert és John Mauchly, az ENIAC feltalálói.

- Legyen $s = \lfloor \frac{\ln(n)}{\ln(q)} \rfloor$
- Legyen $a = a^{q^s} \pmod{n}$

4. Legyen $p = lnko(a-1, n)$
5. Ha $1 < p < n$, az eredmény p
   egyébként válasszunk új a-t és menjünk 3-ra

Pollard $p-1$ algoritmusának várható futásideje:

$$\mathcal{O}\left(B\frac{\ln(n)}{\ln(B)}\right)$$

5.1.4. A $p+1$ módszer (Williams's $p+1$ method)

Hugh C. Williams 1982-ben mutatta be módszerét, mely Pollard $p-1$ módszerének egy variánsa Lucas sorozatokkal. A módszer alapja V_M kiszámolása különböző M-ekre, ugyanis ha a faktorizálandó n egy p prímfaktorára $\left(\frac{P^2-4Q}{p}\right) = -1$ és $p+1|M$, akkor $V_M - 2$ osztható p-vel. Jó eséllyel $n \nmid V_M - 2$ és így p kinyerhető $lnko$ számolással.

Természetesen nem tudjuk előre $\left(\frac{P^2-4Q}{p}\right)$ értékét, így érdemes több P-vel próbálkoznunk. Ha $\left(\frac{P^2-4Q}{p}\right) = 1$, akkor Williams $p+1$ módszere azonos Pollard $p-1$ módszerének egy lassabb változatával.

M-re gyakran az $1, 2!, 3!, 4!, \ldots$ sorozatot használjuk, ekkor $Q = 1$ és $P \neq \pm 2$ esetén érdemes $V_{k!}$-t $V_{(k-1)!}$-ból számolni a következő rekurzióval (melynek bizonyítását megtalálhatjuk [2]-ben):

$$U_{mk} = U_k(P)U_m(V_k(P))$$

$$V_{mk} = V_m(V_k(P))$$

Pollard és Williams módszereit Eric Bach és Jeffrey Shallit általánosította körosztási polinomokra[7]. Módszerükkel hatékonyan faktorizálható n, ha $\phi_k(p)$ sima. Sajnos $k = 2$ fölött egyre kisebb az esély rá, hogy $\phi_k(p)$ sima legyen, így ezek a módszerek a gyakorlatban nem olyan hatékonyak.

5.1.5. Faktorizálás elliptikus görbékkel (Elliptic Curve Method,ECM)

Három évvel a $p+1$ módszer bemutatása után, 1985-ben Lenstra az elliptikus görbék használatát javasolta faktorizáláshoz, ötletét Brent csiszolta tovább. Ezzel megszületett a leggyorsabb speciális célú algoritmus. Az ECM alapjaiban hasonlít Pollard $p-1$

[7] A körosztási polinomok a primitív egységgyökök minimálpolinomjai. Az első néhány körosztási polinom: $\phi_1(p) = p-1$, $\phi_2(p) = p+1$, $\phi_3(p) = p^2+p+1$, $\phi_4(p) = p^2+1$

módszeréhez, a különbség az algebrai struktúra amiben dolgoznak. Pollard a $\mathbf{Z}/n\mathbf{Z}$ multiplikatív csoportot használja, Lenstra pedig az elliptikus görbékkel definiált csoportját (szintén $\bmod n$).

Az algoritmus:

1. Válasszunk véletlenszerűen egy elliptikus görbét $\mathbf{Z}/n\mathbf{Z}$ felett. Ezt megtehetjük úgy, hogy véletlenszerűen választunk egy $P = (x, y) \neq (0, 0)$ számpárt $x, y \pmod n$ értékekkel, valamint egy szintén véletlenszerű a-t, majd az $y^2 = x^3 + ax + b \pmod n$ egyenletet átrendezve meghatározzuk b-t.

2. Válasszunk egy B simasági korlátot.
   Legyen e a B-nél kisebb prímek szorzata.

3. Számoljuk ki a görbén $eP \pmod n$-et. A szorzást számoljuk ismételt összeadással, a 3.3.4 szakaszban tárgyalt formulát használva. Ha a formulában szereplő hányados (a meredekség) u/v és $lnko(u, n) = 1$, akkor $v = 0 \pmod n$ esetén a görbén definiált ∞ elemet kaptuk. Ha $lnko(v, n)$ nem 1 vagy n, akkor az összeadás egy a görbén nem értelmezett pontot ad, ez mutatja, hogy az elliptikus görbénk nem csoport $\bmod n$. (De fontosabb, hogy $lnko(v, n)$ nemtriviális osztó!)

4. – Ha ki tudtuk számolni eP-t vagyis nem ütköztünk nem invertálható elembe, akkor kezdjük előről az algoritmust új görbével és kezdőértékkel.
   – Ha a számítás során $kP = \infty$-be ütközünk, kezdjük előről az algoritmust új görbével és kezdőértékkel. (Hiszen ezen a ponton nem tudnánk túljutni, mivel $\infty + \ldots + \infty = \infty$)
   – Ha a számítás során valahol $lnko(v, n)$ nem 1 vagy n, megtaláltunk egy nemtriviális osztót.

Az algoritmus működése azon alapszik, hogy ha p és q két különböző prímosztója n-nek és $y^2 = x^3 + ax + b \pmod n$ fennáll, akkor az egyenlet igaz $\bmod p$ és $\bmod q$ is. Ezek a görbék már biztosan csoportot alkotnak, és Lagrange tétele szerint ha az eredeti görbén egy P pontra $kP = \infty \pmod p$, akkor k osztója a csoport rendjének. Ez természetesen q-ra is igaz, és ha az elliptikus görbét véletlenszerűen választottuk, akkor a két csoport rendje $p + 1$-hez, illetve $q + 1$-hez esik közel a 3.3.1 tétel szerint.

Mivel annak az esélye nagyon kicsi, hogy a két csoport rendjének faktorai megegyeznek, ezért amikor az algoritmus során $kP = \infty \pmod p$, akkor valószínűleg $\bmod q$ ez nem igaz. Ekkor kP nincs rajta az eredeti görbén és a számolás során talált v-re $lnko(v, p) = p$ vagy $lnko(v, q) = q$, de nem egyszerre, így $lnko(v, n)$ egy nemtriviális

osztót ad.

Ha p az n legkisebb prímosztója, akkor az ECM algoritmus várható futásideje:

$$\mathcal{O}\left(e^{(\sqrt{2}+o(1))\,(\ln(p)\,\ln(\ln(p)))}\right)$$

5.2. Általános célú algoritmusok

Az általános célú algoritmusok bármely összetett szám ellen bevethetőek, egyetlen hátrányuk a speciális célú algoritmusokkal szemben, hogy ugyanannyi idő alatt faktorizálják a feladat szempontjából egyszerű számokat, mint a legnehezebbeket. Ez azért okoz gondot, mert ezen algoritmusok számítógépes implementációi általában nagy memória- és számításigénnyel rendelkeznek, így előfordulhat, hogy sokkal tovább tart a könnyen felbontható számokat faktorizálni általános célú algoritmussal. Érdemes úgy gondolnunk rájuk mint az utolsó alkalmazandó módszerekre, hacsak nem tudjuk előre, hogy mással nem érhetünk el eredményt. Minden itt tárgyalt algoritmusról elmondható, hogy Legendre kongruens négyzetekről szóló ötletén alapul:

5.2.1. Definíció (Legendre kongruenciája). *Legyen $x, y \in \mathbb{Z}$, $0 \leq x < y \leq n$, $x + y \neq n$. Ekkor Legendre kongruenciájának a következő egyenletet nevezzük:*

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

Tegyük fel, hogy valamilyen módon találtunk egy x, y párt, mely kielégíti Legendre kongruenciáját. Ekkor jó eséllyel $lnko(n, x + y)$ vagy $lnko(n, x - y)$ egy nem triviális osztója n-nek, mivel

$$x^2 \equiv y^2 \pmod{n} \quad \Leftrightarrow \quad n|x^2 - y^2 \quad \Leftrightarrow \quad n|(x + y)(x - y)$$

Könnyen igazolható (a lehetőségek felsorolásával), hogy 2/3 eséllyel kapunk nemtriviális osztót. A trükk az egészben a feltételek enyhítése. Fermat módszerénél olyan x, y párt kerestünk, melyre $x^2 - y^2 = n$, mert ekkor biztosan fel tudjuk bontani n-et. Sajnos ilyen x, y párból elég kevés van, gyakorlatilag az osztók számával arányos a mennyiségük. Legendre módszerénél pedig n egy többszörösét faktorizáljuk és reménykedünk, hogy a $lnko$ művelettel jó helyen vágtuk el a számot. Természetesen az algoritmusok jól kezelik a 2/3-os valószínűséget, minden módszer törekszik arra, hogy ha egyszer eljutunk idáig, akkor sikertelenség esetén kevés plusz számolással újra próbálkozhassunk.

5.2.1. Faktorizálás lánctörtekkel (Continued Fraction Method, CFRAC)

1931-ben D. H. Lehmer és R. E. Powers mutatták be az első modern lánctörtekkel faktorizáló algoritmust. A módszert Michael A. Morrison és John Brillhart fejlesztették tovább számítógépekre 1975-ben. Az algoritmusban ötvöződik a gyökök lánctörtekkel való jó közelítésének ereje Legendre módszerével, ezzel hosszú időre (a QR megjelenéséig) ez volt a leggyorsabb faktorizáló algoritmus nagy számokra. Lánctörtekről és az azokkal való faktorizálásról bővebben olvashatunk [8]-ban és [9]-ben.

Az algoritmus $\sqrt{n}$ közelítéseiből állít elő olyan y_i-ket, amikre $x_i^2 \equiv y_i \pmod{n}$, ahol x_i $\sqrt{n}$ egy közelítése. Ha $\sqrt{n}$-et lánctörtekkel közelítjük, akkor mivel $\frac{P_i^2}{Q_i^2} \approx n$, ezért $P_i^2 - Q_i^2 n \approx 0$. Az itt fellépő eltérést jelöljük y_i-vel, s mivel ez láthatóan kicsi, így nagy valószínűséggel felbontható kis prímszámok szorzatára. Ha össze tudunk gyűjteni olyan y_i-ket, melyeknek a kanonikus alakjában lévő prímek azonosak, akkor egy részük szorzatából előállíthatjuk a keresendő $y_{i1}\, y_{i2} \ldots y_{ik} = y^2$ négyzetszámot. Mivel a $x_i^2 \equiv y_i \pmod{n}$ kongruencia bal oldalán négyzetszámok vannak, így a szorzatuk is négyzetszám marad, vagyis megkapjuk a Legendre kongruencia egy megoldását:

$$x^2 = x_{i1}^2\, x_{i2}^2 \ldots x_{ik}^2 \equiv y_{i1}\, y_{i2} \ldots y_{ik} = y^2 \pmod{n}$$

Az algoritmus vázlata:

1. Válasszunk egy B korlátot, majd az ennél kisebb prímek halmazát jelöljük fb-vel, ez lesz az un. faktorbázis.

2. Számítsuk ki $\sqrt{n}$ lánctört alakját, majd minden P_i/Q_i közelítésre számoljuk ki $y_i = P_i^2 - Q_i^2 n$-et, és ha ez fb-sima, akkor tároljuk el a felbontását egy vektorban. A vektor koordinátái a faktorbázisban lévő prímekhez tartozó kitevőket jelentsék a szám kanonikus alakjából.
   Ismételjük addig ezt a lépést, amíg nem találunk $|fb|$-nél több fb-sima számot. Ha a lánctört hossza nem elég hosszú ehhez, $\sqrt{n}$ helyett használjuk $\sqrt{kn}$-et valamilyen $k \in \mathbb{N}$-el.

3. Az összegyűjtött vektorokból készítsünk mátrixot, melyre alkalmazzuk a Gauss eliminációt mod2, így megkeresve a Legendre kongruencia megoldását.

4. Számítsuk ki $lnko(n, x \pm y)$-t, ami 2/3 eséllyel nemtriviális osztó. Ellenkező esetben ugorjunk 2-re és cseréljünk le néhány vektort.

A CFRAC algoritmus várható futási ideje:

$$\mathcal{O}\left(n^{\sqrt{1.5\frac{\ln\ln(n)}{\ln(n)}}}\right)$$

5.2.2. Kvadratikus szita (Quadratic Sieve, QS)

A kvadratikus szita Carl Pomerance által kifejlesztett módszer, melyet 1981-ben mutatott be. Több mint 10 évig - a GNFS megjelenéséig - ez volt a leggyorsabb faktorizáló algoritmus, sőt még ma is annak mondható a maximum 110 jegyű számok között. Az algoritmussal sikeresen faktorizáltak több RSA modulust is, a legjobb eredmények jelenleg a 130 jegyű számok környékén vannak.
A kvadratikus szita is Legendre kongruenciájára épít, vagyis kongruens négyzetszámokat kell keresnünk $\bmod n$, mert ekkor 2/3 eséllyel nemtriviális osztót is találunk egyben. Legyen

$$Q(x) = (x + \lfloor\sqrt{n}\rfloor)^2 - n = \tilde{x}^2 - n \quad x \in \mathbb{Z}$$

ekkor célunk találni olyan $\{x_1, x_2, \ldots, x_k\}$ halmazt, amire $Q(x_1)Q(x_2)\ldots Q(x_k)$ négyzetszám, jelöljük ezt y^2-el. Mivel minden x-re $Q(x) \equiv \tilde{x}^2 \pmod n$, ezért

$$y^2 = Q(x_1)Q(x_2)\ldots Q(x_k) \equiv (\tilde{x_1}\tilde{x_2}\ldots\tilde{x_k})^2 \pmod n$$

vagyis megvannak a keresett kongruens négyzetek.
Az, hogy a $Q(x_i)$-k szorzata négyzetszám, azt jelenti hogy a szorzat kanonikus alakjában minden kitevő páros, ennek eldöntéséhez faktorizálnunk kell a $Q(x_i)$-ket. Ahhoz, hogy ez könnyű legyen, $Q(x_i)$-t minél kisebbnek kell választanunk, vagyis x-nek 0 közelinek kell lennie. Legyen tehát M egy általunk választott korlát, és $x \in [-M, M]$.

A $Q(x_i)$-kből akkor könnyű négyzetszámot összerakni, ha ugyanazoknak a prímeknek a különböző hatványai szerepelnek a kanonikus alakjukban. Ilyen számokat úgy érdemes keresni, hogy választunk egy $\{p_1, p_2, \ldots, p_B\}$ faktorbázist és megnézzük, hogy $Q(x)$ $(x = 1, 2, \ldots)$ felbomlik-e a faktorbázison. Ha egy p prím osztója $Q(x)$-nek, akkor

$$(x + \lfloor\sqrt{n}\rfloor)^2 \equiv n \pmod p,$$

vagyis n kvadratikus maradék $\bmod p$. Ez azt jelenti, hogy a faktorbázisban olyan p prímeknek kell lenniük, amire

$$\left(\frac{n}{p}\right) = 1.$$

Mivel $Q(x)$ lehet negatív is, így a -1-et is be kell vennünk a faktorbázisba, ezen kívül

még arra kell ügyelnünk, hogy se a faktorbázis mérete, se M ne legyen túl nagy.
Az optimális futási idő eléréséhez a faktorbázis mérete legyen körülbelül

$$B = \left(e^{\sqrt{\ln(n)\,\ln(\ln(n))}}\right)^{\sqrt{2}/4},$$

az intervallum korlátja pedig ennek köbe, azaz

$$M = \left(e^{\sqrt{\ln(n)\,\ln(\ln(n))}}\right)^{3\sqrt{2}/4}.$$

Ha találunk B db olyan $Q(x_i)$-t amelyik teljesen felbomlik a faktorbázison, akkor a kitevőkből mátrixot készítve Gauss eliminációval $\bmod 2$ megtalálhatunk egy olyan $\{x_1, x_2, \ldots, x_k\}$ részhalmazt, amire $Q(x_1)Q(x_2)\ldots Q(x_k)$ négyzetszám. Tekintsük tehát a sikeresen faktorizált $Q(x_i)$-k kanonikus alakját, s a kitevőket tegyük be egy-egy vektorba. Az ilyen x_i és kitevő-vektor párokat relációnak hívjuk. A $Q(x_i)$-k összeszorzása ekkor ezen vektorok összeadására egyszerűsödik, az pedig, hogy megtaláljuk azt a részhalmazt aminek a szorzata négyzetszám, az azt jelenti, hogy azt a részhalmazt keressük, ahol a vektorok összege éppen $0\,(\bmod 2\,)$, ez pedig egy egyenletrendszer megoldásával megtalálható. Véges test feletti Gauss eliminációra Wiedemann és Lánczos algoritmusa is rendelkezésre áll, összehasonlításukról [10]-ben olvashatunk.

Az egyetlen megválaszolatlan kérdés az, hogy hogyan faktorizáljunk hatékonyan sok számot. Megtehetjük, hogy sorra vesszük a $Q(1), Q(2), \ldots$ sorozat elemeit, leellenőrizzük egyenként próbaosztással, hogy felbomlanak-e a faktorbázison és ha igen akkor megtartjuk őket, egyébként megyünk tovább amíg nem találunk B db olyat ami felbomlik. Ez persze túl lassú volna így, helyette szerencsére megtehetjük, hogy az egész $[-M, M]$ intervallumon egyszerre faktorizálunk.
A módszer azon alapul, hogy ha $x \equiv y \pmod p$, akkor $Q(x) \equiv Q(y) \pmod p$. Oldjuk meg a

$$Q(x) = s^2 \equiv 0 \pmod p$$

kongruenciát, ezt megtehetjük a Shanks-Tonelli algoritmussal. Legyen a kapott két megoldás $s_{p,1}$ és $s_{p,2} = -s_{p,1}$, ekkor $x_i = s_{p,1},\ s_{p,2} + kp\ \ (k \in \mathbb{Z})$ esetén $Q(x_i)$ osztható lesz p-vel.
Tegyük a $Q(x_i)$ értékeket egy vektorba, majd a faktorbázison végighaladva minden $s_{p,1},\ s_{p,2} + kp$-edik elemet osszunk le p-vel annyiszor ahányszor tudjuk és tároljuk el, hogy hányszor sikerült $\bmod 2$. Ha ezt megcsináljuk a faktorbázis összes elemével, akkor a végén azon $Q(x_i)$-k helyén lesz 1, amik teljesen felbonhatóak a faktorbázison. Az ezekhez tartozó kitevő-vektorokat rakjuk egy mátrixba és oldjuk meg az így adódó

egyenletrendszert. Mivel ekkor csak 2/3 eséllyel találunk nemtriviális osztót, ezért jó ha B-nél több vektorunk van, mert ha elsőre nem járnánk sikerrel, akkor egy vektort lecserélve a mátrixban, újra próbálkozhatunk.
Az algoritmus vázlata:

1. Válasszunk egy B korlátot, majd készítsük el a B méretű faktorbázist olyan prímekből, melyekre $(\frac{n}{p}) = 1$.

2. Válasszunk M-et, majd számoljuk ki a $Q(x_i)$ értékeket minden $x \in [-M, M]$-re. Szitálással faktorizáljunk legalább annyi $Q(x_i)$-t, mint amennyi B.

3. A kapott relációkra alkalmazzunk Gauss eliminációt mod 2, ezzel megoldva a Legendre kongruenciát.

4. A kapott x, y értékekkel számoljuk ki $p = lnko(n, x \pm y)$-t ami 2/3 eséllyel nemtriviális osztó. Ha $p = 1$, menjünk az előző pontra és cseréljünk le a mátrixban egy vektort.

A QS algoritmus várható futási ideje:

$$\mathcal{O}\left(e^{\sqrt{\ln(n)\ \ln(\ln(n))}}\right)$$

5.2.3. GNFS (General Number Field Sieve, GNFS)

1988-ban John Pollard ötlete alapján elkészült a legmodernebb speciális célú algoritmus, a Special Number Field Sieve. A módszerrel azon $r^e \pm s$ alakú számok faktorizálhatók hatényokan, ahol r és s kicsi. Az SNFS általánosításaként létrejövő algoritmus a General Number Field Sieve, mely bármilyen alakú számra alkalmazható, de egy kicsivel lassabb az SNFS-nél. A GNFS a jelenlegi tudásunk szerint a leggyorsabb algoritmus a több mint 100 jegyű számok körében.

A GNFS megértéséhez érdemes látni a folyamatot, ahogy az algoritmus kifejlődött. A Legendre-kongruenciás módszerek Dixon algoritmusával kezdődtek, mely hasonló módon működik mint a kvadratikus szita, azzal a különbséggel, hogy a sima számokat nem szitálással, hanem véletlenszerűen választva próbálta összeszedni. A kvadratikus szita azzal javította ezt a módszert, hogy a sima számok keresését egy sokkal hatékonyabb algoritmussal helyettesítette. Mindkét algoritmus azon alapul, hogy két különböző gyűrűben, konkrétan $\mathbb{Z}$-ben és $\mathbb{Z}/n\mathbb{Z}$-ben keresünk négyzetszámokat és egy megfelelő leképezéssel a $\mathbb{Z}$-beli négyzetszámot egy $\mathbb{Z}/n\mathbb{Z}$-beli négyzetszámba visszük. A két gyűrű közötti leképezést a QS-ben a $\phi(k) = k \pmod n$ függvény végzi, ami $k = x^2 - n$ esetén láthatóan megtartja a négyzetszám tulajdonságot $\mathbb{Z}$ és $\mathbb{Z}/n\mathbb{Z}$ között.

A GNFS algoritmus két meggondolással javít ezen a módszeren, az egyik hogy a szitálásnál használt sima számokat előállító $x^2 - n$ polinomot lecserélhetjük magasabb fokúra is anélkül, hogy kevésbé sima számokat kapnánk, a másik pedig, hogy $\mathbb{Z}$ helyett használhatunk más gyűrűt is, ha a négyzetszámokat ugyanúgy le tudjuk képezni $\mathbb{Z}/n\mathbb{Z}$-beli négyzetszámokra.

Tegyük fel, hogy adott az R gyűrűnk és a $\phi : R \to \mathbb{Z}/n\mathbb{Z}$ gyűrű-homomorfizmusunk. Ekkor ha találunk egy $\beta \in R$-et, melyre $\phi(\beta^2) = y^2 \pmod{n}$ és $x = \phi(\beta) \pmod{n}$, akkor

$$x^2 \equiv \phi(\beta)^2 \equiv \phi(\beta^2) \equiv y^2 \pmod{n},$$

amivel megvan a Legendre-kongruencia egy megoldása. Látni fogjuk, hogy a kérdéses gyűrű és a hozzá tartozó homomorfizmus is viszonylag természetes módon konstruálható.

Polinom kiválasztás

Elsőként foglalkozzunk a megfelelő polinom kiválasztásával. Célunk olyan $f(x)$ polinomot választani, mely sok sima számot állít elő, azaz *hasznos*. Két tulajdonsággal fogható meg hasznosság, az egyik, hogy $f(x)$ értéke kicsi legyen, ezt a polinom *méret* tulajdonságának nevezzük. Minél kisebb a mérete egy polinomnak, annál jobb. A másik fontos tulajdonság, hogy a polinom által előállított számok lehetőleg minél több kis osztóval rendelkezzenek. Ezt úgy fogalmazhatjuk meg matematikailag jól, hogy a $f(x)$ polinomnak sok gyöke legyen $\bmod p$ kis p-re. Ezt a tulajdonságot *gyök* tulajdonságnak nevezzük.

A GNFS-ben olyan polinomokat használunk, melyek irreducibilisek $\mathbb{Z}[x]$-ben és van egy gyökük $\mathbb{Z}/n\mathbb{Z}$-ben. Az m gyökkel rendelkező $f(x)$ polinom megválasztása általában irreducibilis is egyben, mivel $f(m) = n$ mellett, ha $f(x)$ nem lenne irreducibilis, azaz például $f(x) = g(x)h(x)$ igaz lenne, akkor $f(m) = g(m)h(m) = n$ alakban meg is találnánk n egy osztóját. A polinom megválasztásához használjuk az n egész m-alapú reprezentációját, ami ebben az alakban áll elő:

$$n = \sum_{i=0}^{d} a_i m^i,$$

ahol $0 < a_i < m$. Ekkor a d fokú, a_i együtthatós polinom automatikusan teljesíti is az

elvárásainkat, így legyen tehát

$$\begin{aligned} f(x) &= a_d x^d + a_{d-1}x^{d-1} + \ldots + a_0 \\ f(m) &= 0 \pmod{n} \end{aligned}$$

A polinom méret tulajdonságát úgy tudjuk csökkenteni, ha az a_i együtthatókat minél kisebbnek választjuk, különös tekintettel a_d és a_{d-1} nagyságára (d-t általában 3 és 6 közé választjuk, így ezek lesznek a dominánsak). Egy trükk amivel ez elérhető amellett, hogy az m gyök megmaradjon, ha az együtthatókat így módosítjuk:

$$\begin{aligned} a_i &= a_i - m \\ a_{i+1} &= a_{i+1} + 1 \end{aligned}$$

A megfelelő hasznosság eléréséhez még az kell, hogy a polinom gyök tulajdonsága is megfelelő legyen. Nincs jó módszerünk arra, hogy könnyedén előállítsunk olyan polinomokat, melyeknek jó a gyök tulajdonsága is, a gyakorlatban általában sok lehetséges jelölt közül választjuk ki a legjobbat. Egy kis intervallumon szitálással megmérjük az összes jelölt gyök tulajdonságát és az így kapott legjobbat használjuk.
A hasznosság mérésére létezik egy heurisztika is, mellyel a legrosszabb polinomokat még a szitálás előtt eldobhatjuk. Ez a Murphy által kidolgozott $\alpha(P)$ függvény a következőképpen definiálható:

$$\alpha(P) = \sum_{p \le B} \left(1 - q_p \frac{p}{p+1}\right) \frac{\log p}{p-1}$$

ahol B egy simasági korlát, p-k prímek és q_p a P polinom gyökeinek száma mod p. Így végül a polinom kiválasztása a következő módon történik:

1. Választunk d-t és m_0-t, melyre $\lfloor n^{\frac{1}{d+1}} \rfloor \le \frac{|a_d|}{m_0} \le \lfloor n^{\frac{1}{d}} \rfloor$.

2. Válasszunk egy (x, y) intervallumot, melyben a legjobb a_i együtthatókat keressük. Az intervallum olyan legyen, amire $0 < x < \frac{|a_d|}{m_0} < y < 0.5$ igaz.

3. Válasszunk egy intervallumot, melyben a legjobb m-et keressük. Válasszuk az intervallumot olyanra, hogy az a_{d-1} a lehető legkisebb legyen, vagyis $m \in [m_0, \lfloor (\frac{n}{a_d})^{\frac{1}{d}} \rfloor]$.

4. Minden olyan a_d és m kombinációra, ahol a_d megfelelően sima, számoljuk ki az m-alapú reprezentációból $f_m(x)$-et, majd $\alpha(f_m(x))$-et. Ha $\alpha(f_m(x))$ kicsi, dobjuk el $f_m(x)$-et.

5. A megmaradó polinomokból szitálással válasszuk ki a legjobbat.

Kongruens négyzetek $\mathbb{Z}[\theta]$-ban

Legyen $f \in \mathbb{Z}[x]$ polinom, melynek θ egy gyöke $\mathbb{C}$-ben és m gyöke $\mathbb{Z}/n\mathbb{Z}$-ben, azaz $f(m) \equiv 0 \pmod{n}$. Ekkor használjuk a $\mathbb{Z}[\theta]$ és $\mathbb{Z}$ gyűrűket és tekintsük a 3.2.11 tételben szereplő homomorfizmust. Ekkor ha találunk olyan $S \subset \mathbb{Z}^2$ halmazt, melyre

$$\prod_{(a,b)\in S}(a+b\theta) = \beta^2 \qquad \text{és} \qquad \prod_{(a,b)\in S}(a+bm) = y^2$$

teljesül, ahol $\beta \in \mathbb{Z}[\theta]$, $y \in \mathbb{Z}$, akkor az $x := \phi(\beta) \in \mathbb{Z}/n\mathbb{Z}$ jelölés mellett alkalmazzuk ϕ-t:

$$x^2 \equiv \phi(\beta)^2 \equiv \phi(\beta^2) \equiv \phi\Big(\prod_{(a,b)\in S}(a+b\theta)\Big) \equiv$$

$$\equiv \prod_{(a,b)\in S}\phi(a+b\theta) \equiv \prod_{(a,b)\in S}(a+bm) \equiv y^2 \pmod{n}.$$

Fontos megjegyeznünk, hogy az $a+b\theta$ alakú számok szorzataként előálló négyzetszámnak $\mathbb{Z}[\theta]$-ban kell lennie, mert ϕ csak ott van értelmezve. A gyakorlatban ezt a feltételt enyhíthetjük annyival, hogy a négyzetszám $\mathbb{Q}(\theta)$-ban is lehet, mert abból könnyű $\mathbb{Z}[\theta]$-belit csinálni a következő módon. Legyen $\alpha \in \mathbb{Q}(\theta)$ és $z \in \mathbb{Z}$ úgy, hogy

$$\prod_{(a,b)\in S}(a+b\theta) = \alpha^2 \qquad \text{és} \qquad \prod_{(a,b)\in S}(a+bm) = z^2 \tag{1}$$

teljesüljön. Ekkor $\alpha \in \mathfrak{D}$ ($\mathfrak{D}$ a $\mathbb{Q}(\theta)$-beli algebrai számok részgyűrűje (3.2.3)-ból) és $f'(\theta)\cdot\alpha \in \mathbb{Z}[\theta]$.
Legyen $\beta = f'(\theta)\cdot\alpha$, $y = f'(\theta)\cdot z$ és $x = \phi(\beta) \in \mathbb{Z}/n\mathbb{Z}$, ekkor

$$x^2 \equiv \phi(\beta)^2 \equiv \phi(\beta^2) \equiv \phi\Big(f'(\theta)^2 \cdot \prod_{(a,b)\in S}(a+b\theta)\Big) \equiv$$

$$\equiv \phi(f'(\theta))^2 \cdot \prod_{(a,b)\in S}\phi(a+b\theta) \equiv f'(m)^2 \cdot \prod_{(a,b)\in S}(a+bm) \equiv y^2 \pmod{n},$$

vagyis a keresett kongruens négyzetek ilyenkor is előállíthatóak.
Keressünk tehát egy olyan S halmazt, melyre igaz (1). Ezt úgy tehetjük meg, hogy olyan $(a,b) \in \mathbb{Z}^2$ párokat keresünk, melyre $a+b\theta$ sima egy "algebrai" faktorbázis felett és $a+bm$ sima egy "racionális" faktorbázis felett, majd a QS-ben megismert módon lineáris algebra segítségével megkeressük az S halmazt. Az algebrai faktorbázis $\mathbb{Z}[\theta]$-hoz, a racionális pedig $\mathbb{Z}$-hez tartozik. Természetes, hogy a racionális faktorbázis a megszokott módon prímszámokat tartalmaz, az viszont nem világos elsőre, hogy $\mathbb{Z}[\theta]$-ban mi szerint

faktorizálunk. Valójában $\mathbb{Z}[\theta]$-ban általában még a számelmélet alaptétele sem érvényes, így a kérdés valóban okoz némi fejtörést. A GNFS megszületésekor (SNFS) azzal a feltevéssel éltek, hogy igaz az alaptétel és $\mathbb{Z}[\theta] = \mathfrak{D}$. Általánosan persze egyik sem igaz, ennek ellenére az SNFS algoritmus nagyon hatékony a mai napig.

Simaság $\mathbb{Z}[\theta]$-ban

A GNFS-ben használt megoldás az, hogy a $\mathbb{Z}[\theta]$-ban lévő simaságot a gyűrű prímideáljai szerint vesszük, vagyis az $a + b\theta$ elem akkor sima az algebrai faktorbázis felett, ha az $< a + b\theta >$ főideál felbomlik a faktorbázisban lévő prímideálok szorzatára. Ahhoz, hogy a $\mathbb{Z}[\theta]$-beli faktorizálást könnyedén elvégezhessük, át kell fogalmaznunk az ideálok oszthatóságával kapcsolatos kérdéseket kezelhetőbb formára.

Az derült ki, hogy ha elsőrendű prímideálokat használnunk, akkor ezeket a 3.2.8 tétel szerint reprezentálhatjuk (r, p) számpárok halmazaként, ahol $p \in \mathbb{Z}$ prím, $r \in \mathbb{Z}/p\mathbb{Z}$ és $f(r) \equiv 0 \pmod{p}$. Ezzel a formával számítógépen is jól kezelhetővé válnak az ideálok, ha a szükséges műveleteket is el tudjuk végezni ebben az ábrázolásban. A 3.2.10 tétel szerint létezik olyan homomorfizmus, mellyel az oszthatósági kérdéseket könnyedén vizsgálhatjuk. Ez a tétel egyrészt azt mondja ki, hogy az $< a + b\theta >$ alakú ideálok (ahol a és b relatív prímek) elsőrendű prímideálok szorzatára bomlanak, vagyis a faktorbázist van értelme ilyen prímideálok halmazának választani, másrészt jó feltételt ad arra, hogy egy elsőrendű prímideál osztója-e egy $< a + b\theta >$ alakú ideálnak. A tétel szerint egy elsőrendű prímideál pontosan akkor osztója az $< a + b\theta >$ ideálnak, ha a hozzá tartozó (r, p) párra $a \equiv -br \pmod{p}$ igaz. A 3.2.9 tétel szerint végül az (a, b) elem akkor lesz sima az algebrai faktorbázis felett, ha a hozzá tartozó ideál normája az. Így pedig $\mathbb{Z}[\theta]$-ban faktorizálni egy faktorbázis felett pont olyan egyszerű, mint a $\mathbb{Z}$-ben.

Ahhoz, hogy az algebrai faktorbázist összeállítsuk, olyan (r, p) párokat kell keresnünk, melyekre $p \in \mathbb{Z}$ prím, $r \in \mathbb{Z}/p\mathbb{Z}$ és $f(r) \equiv 0 \pmod{p}$ igaz. Másként fogalmazva ez éppen az $f(x)$ polinom gyökeinek megkeresése $\bmod p$. Ezt a feladatot általában olyan egyszerűen oldjuk meg, hogy leellenőrizzük sorban az összes $r \in \mathbb{Z}/p\mathbb{Z}$ elemet, hogy kielégíti-e az egyenletet, de ez a módszer csak viszonylag kis p esetén hatékony. A GNFS-ben jellemzően nagy prímek is előfordulnak, így itt egy gyorsabb módszert érdemes használunk, melyre találunk példát [1]-ben.

Négyzetszámok $\mathbb{Z}[\theta]$-ban

Most, hogy tisztáztuk a $\mathbb{Z}[\theta]$-beli faktorizálás mikéntjét, térjünk vissza az eredeti célunkhoz, miszerint kongruens négyzetszámokat akarunk előállítani $\bmod p$. A módszerünk az, hogy a $\mathbb{Z}$ és $\mathbb{Z}[\theta]$ gyűrűkben egyszerre keresünk sima számokat, melyekből Gauss-eliminációval előállítjuk a kívánt négyzeteket. A probléma az, hogy a Gauss-eliminációval előállított $\mathbb{Z}[\theta]$-beli négyzetszámról jelenleg nem tudjuk garantálni, hogy az tényleg az, mivel $\mathbb{Z}[\theta]$-ban nem biztos, hogy igaz a számelmélet alaptétele. A GNFS-ben úgy érjük el, hogy négyzetszámot kapjunk, hogy nem csak azt a feltételt szabjuk, hogy a megfelelő kitevők párosak legyenek, hanem azt is hogy a megfelelő Legendre-szimbólumok is 1-et vegyenek fel. $\mathbb{Z}$-ben ha x négyzetszám, akkor $\bmod p$ is az minden p prímre, ezért ha egy x-re és valamilyen p prímre az $\left(\frac{x}{p}\right)$ Legendre-szimbólum értéke -1, akkor x biztosan nem négyzetszám. $\mathbb{Z}[\theta]$-ban ugyanilyen feltételt kvadratikus karakterek segítségével tehetünk.

5.2.1. Tétel. *Legyen $S \subseteq \mathbb{Z}^2$ olyan (a, b) számpárok halmaza, melyre*

$$\prod_{(a,b)\in S} (a + b\theta) = \alpha^2$$

valamilyen $\alpha \in \mathbb{Q}(\theta)$-val. Vegyünk egy olyan elsőrendű $\mathfrak{p}$ prímideált, amihez az (r, p) pár tartozik és amelyik nem osztja az $< a + b\theta >$ ideált semmilyen (a, b)-re, valamint $f'(r) \not\equiv 0$. Ekkor

$$\chi_{\mathfrak{p}}(\alpha^2) := \prod_{(a,b)\in S} \left(\frac{a + br}{p}\right) = 1.$$

Ahhoz tehát, hogy jó eséllyel négyzetszámot kapjunk, arra van szükség, hogy minél több prímideállal igaz legyen ez a feltétel. Természetesen ezzel a módszerrel nem lehet a gyakorlatban biztosra menni, de nagyon kis eséllyel fogunk hibázni. Szükségünk van így még egy faktorbázisra, mely azokat a prímideálokhoz tartozó (r, p) párokat fogja tartalmazni, melyekkel ez utóbbi feltételt írjuk elő. Ezt a faktorbázist kvadratikus faktorbázisnak nevezzük. Ebben a halmazban természetesen ugyanolyan tulajdonságú ideálok vannak mint az algebrai faktorbázisban, de nem azonosak velük.

Van még egy fontos különbség a GNFS-ben a kvadratikus szitához képest. Itt a lineáris algebrai lépésben egy $\mathbb{Q}(\theta)$-beli négyzetszámot kapunk, amiből $f'(\theta)^2$-val való szorzással állítunk elő $\mathbb{Z}[\theta]$-belit. Ebből az is következik, hogy nem is ismerjük a négyzetszám gyökét $\mathbb{Z}[\theta]$-ban, azt külön ki kell számolnunk. Mivel a $\mathbb{Z}$-beli négyzetszámot is beszorozzuk, ezért annak ugyanúgy nem ismerjük a gyökét. A QS-ben ezeket automati-

kusan megkaptuk a kanonikus alakokból. Bár a gyökvonás elvégzése nem tűnik komoly nehézségnek, mégis egy fontos és a futási időben sem elhanyagolható részét képezi ez az algoritmusnak. A $\mathbb{Z}$-beli gyökvonásra számtalan módszer ismert, a $\mathbb{Z}[\theta]$-beli gyökvonás viszont egy kevésbé vizsgált terület. Egy Newton iterációra épülő algoritmust találhatunk [5]-ben, illetve egy komplexebb módszerről [7]-ben olvashatunk.

Szitálás

A szitálás algoritmusa egy kicsit eltér a kvadratikus szitában használt módszertől, mivel itt más feltételek alapján keresünk sima számokat. A keresett elemek itt az

- $lnko(a, b) = 1$
- $a + bm$ sima a racionális faktorbázis felett
- $N(a, b) = b^{deg(f)} f(a/b)$ sima az algebrai faktorbázis felett

tulajdonságokkal rendelkeznek. Mivel (a, b) számpárokat keresünk, ezért 2-dimenzióban kellene szitálni, de ehelyett általában az egyik változót lerögzítve szokás a keresést végezni. Legyen tehát b rögzített érték mellett $a \in [-C, C]$. Ekkor $a + bm$ akkor osztható p-vel, ha $a = -bm + kp$ $(k \in \mathbb{Z})$ alakú. Hasonlóan $a + b\theta$ akkor osztható az (r, p)-nek megfelelő prímideállal, ha $a = -br + kp$ $(k \in \mathbb{Z})$ alakú. Innentől kezdve a szitálást teljesen hasonlóan végezhetjük mint a QS-ben, annyi különbséggel, hogy két vektorban kell egyszerre szitálnunk, s végül azokat az elemeket tartjuk meg, melyek mindkét vektorban 1-re csökkentek. Ha az $a \in [-C, C]$ intervallumon elvégeztük a szitálást, de még nincs elég relációnk, növeljük meg b-t és szitáljunk újra.

Lineáris algebra

A szitálással olyan (a, b) párokat kaptunk, amire $a+bm$ és $a+b\theta$ sima a racionális illetve az algebrai faktorbázis felett. Ahhoz, hogy megkeressük azt a részhalmazt, melynek elemeit összeszorozva négyzetszámokat kapunk, meg kell oldanunk egy egyenletrendszert. Az eredményül kapott halmaztól azt várjuk el, hogy a szorzatok kanonikus alakja $\mathbb{Z}$-ben és $\mathbb{Z}[\theta]$-ban is nullvektor legyen $\bmod 2$. A kvadratikus bázissal hasonlóan járunk el, az előzőek mellett azt is elvárjuk, hogy a szorzatok kvadratikus karakterei 1-ek legyenek. Mivel a kvadratikus karakter értékkészlete $\{-1, 0, 1\}$, de az eddigi műveleteket mod 2 végeztük, ezért itt egy kis módosításra van szükség. Írjunk a mátrixba 1-et, ha a megfelelő kvadratikus karakter nem 1, egyébként nullát. Így egy-egy relációhoz tartozó sorvektor álljon a következő elemekből:

- Az első elem legyen 0, ha $(a + bm) > 0$, egyébként 1
- A második elemtől kezdve írjuk le $(a + bm)$ kanonikus alakját a racionális faktorbázis felett. Írjunk 1-et, ha az adott prím páratlanszor, illetve 0-t ha párosszor osztja $(a + bm)$-t.
- Folytassuk a vektort $(a+b\theta)$ elem algebrai faktorbázison vett kanonikus alakjával. Hasonlóan csak a paritást írjuk le.
- Folytassuk a vektort a kvadratikus bázissal, írjunk 1-et, ha $\left(\frac{a+br_i}{p_i}\right) \neq 1$ egyébként 0-t.

Végül a vektorokból mátrixot képezve kapunk egy (relációk száma) · (racionális faktorbázis elemszáma + algebrai faktorbázis elemszáma + kvadratikus faktorbázis elemszáma +1) méretű mátrixot. A keresett részhalmazt megkaphatjuk az $Ax = 0 \pmod 2$ egyenletrendszer megoldásával, ahol A az előbbi mátrix.

Az algoritmus vázlata

Az eddigieket összefoglalva, a GNFS algoritmus a következő lépésekből áll:

1. Válasszunk egy $\mathbb{Z}[x]$-beli irreducibilis polinomot, melyre $f(m) \equiv 0 \pmod n$
2. Válasszunk meg a racionális, algebrai és kvadratikus faktorbázisok elemeit
3. Szitálással keressünk olyan $(a, b) \in \mathbb{Z}^2$ elemeket, melyekre
   - $lnko(a, b) = 1$
   - $a + bm$ sima a racionális faktorbázis felett
   - $N(a, b) = b^{deg(f)} f(a/b)$ sima az algebrai faktorbázis felett

   Az ilyen elemeket relációknak nevezzük. Gyűjtsünk össze annyi relációt, amennyit csak tudunk, de legalább annyit, mint ahány elem a faktorbázisokban van összesen.
4. A relációkból készítsünk mátrixot, majd Gauss eliminációval keressük egy-egy négyzetszámot $\mathbb{Z}$-ben és $\mathbb{Z}[\theta]$-ban.
5. Számoljuk ki az előző pontban kapott négyzetszámok gyökeit

$$y^2 = \prod_{(a,b)\in S} (a - bm) \qquad \text{és} \qquad x^2 = \prod_{(a,b)\in S} (a - b\theta)$$

segítségével.

6. n egy osztóját jó eséllyel megkapjuk $lnko(n, x+y)$ és $lnko(n, x-y)$ kiszámításával.

A GNFS algoritmus várható futásideje (sejtés):

$$\mathcal{O}\left(e^{(\sqrt[3]{\frac{64}{9}}+\mathcal{O}(1))\sqrt[3]{\log n}\sqrt[3]{(\log\log n)^2}}\right)$$

6. Összefoglalás

Dolgozatom célja a prímfaktorizáció problémájának összegzése volt. Mivel a témával oldalak ezreit meg lehetne tölteni, így itt csak egy betekintést kaphattunk a legismertebb módszerekről. A téma iránt érdeklődőknek érdemes lehet az egyes algoritmusokat egyenként megvizsgálni, hiszen szinte mindegyik más módon optimalizálható, más implementációs trükköket kell bevetni. Érdekes oldala a területnek, hogy a gyakorlatban nem feltétlenül a leggyorsabb algoritmusra van szükségünk, sokszor más módon kell mérnünk a jóságot, például a pénzbeli költséget kellhet minimalizálni.

A dolgozatban először a matematikai alapokat vettük át, azután megismerkedtünk a legfontosabb alkalmazással, az RSA algoritmussal. Miután felmértük e titkosítás gyengéit, a legkézenfekvőbb és a gyakorlatban is használt támadási lehetőségel - a prímfaktorizációval - foglalkoztunk. Az 5. fejezetben sorra vettük az ismert módszereket, míg végül a GNFS-el elérkeztünk napjaink legjobb algoritmusához. A GNFS megfelelő számítási kapacitások mellett alkalmas az RSA-modulusok faktorizálására is, de mivel az algoritmus futásideje csak szubexponenciális, ezért elég nagy prímek választásával még ez a támadás is kivitelezhetetlenné tehető.

Nem foglalkoztunk más tikosítási rendszerekkel, pedig - szerencsére - vannak az RSA mellett más alternatívák is, idővel valószínűleg váltanunk is kell valamelyikre. Shor algoritmusa bizonyítottan polinomiális futásidejű kvantumszámítógépeken, így amikor az első gyakorlatban is használható kvantumszámítógépek megjelennek, onnantól kezdve az RSA-t többé nem tekinthetjük bizonságosnak.

Köszönettel tartozom témavezetőmnek Gyarmati Katalinnak, amiért lehetőséget biztosított a dolgozatom elkészítéséhez, valamint a körültekintő ellenőrzéseiért, amikkel dolgozatom hibáira és hiányosságaira mutatott rá.

Hivatkozások

[1] Matthew E. Briggs: *An Introduction to the General Number Field Sieve*, MSc thesis, Blacksburg, Virginia, 1998
http://www.math.vt.edu/people/brown/doc/briggs_gnfs_thesis.pdf

[2] Járai Antal: *Számítógépes számelmélet*, ELTE, Budapest, 2009
http://compalg.inf.elte.hu/~ajarai/cnt.pdf

[3] Freud Róbert, Gyarmati Edit: *Számelmélet*, Nemzeti tankönyvkiadó, Budapest, 2000 **ISBN 963 19 0784 8**

[4] Freud Róbert: *Lineáris Algebra* ELTE Eötvös Kiadó, Budapest, 1996
ISBN 963 463 471 0

[5] Per Leslie Jensen: *Integer Factorization*, Master Thesis, Department of Comupter Science, University of Copenhagen, 2005
http://www.pgnfs.org/DOCS/thesis.pdf

[6] Johannes Buchmann, Volker Müller: *Algorithms for Factoring Integers*, Technische Hochschule Darmstadt, Institut für theoretische Informatik, 2005
http://www.cdc.informatik.tu-darmstadt.de/~buchmann/Lecture%20Notes/Algorithms%20for%20factoring%20integers.pdf

[7] Peter L. Montgomery: *Square roots of products of algebraic numbers*, Mathematics of Computation 1943-1993: a half-century of computational mathematics, American Mathematical Society 1994

[8] Jörn Steuding, Rasa Šleževičienė: *Factoring with continued franctions, the pell equation, and weighted median*, Johann Wolfgang Goethe-Universität Frankfurt, 2003

[9] Niels Lauritzen: *Continued fractions and factoring*, Department of Mathematical Sciences, University of Aarus, Denmark
http://www.dm.unito.it/~cerruti/ac/cfracfact.pdf

[10] Alexander Kruppa: *Comparison of Block-Lanczos and Block-Wiedemann for Solving Linear Systems in Large Factorizations*, Centrum Wiskunde & Informatica, Amsterdam
http://event.cwi.nl/wcnt2011/slides/kruppa.pdf