Debreceni Egyetem
Természettudományi és Technológiai Kar
Matematikai Intézet

# Diffie-Hellman algoritmus csoportgyűrűkben és moduló 1

készítette:

Ferenczi Fruzsina

témavezető: Dr. Tengely Szabolcs

Debrecen, 2016

# Tartalomjegyzék

# Köszönetnyilvánítás

Köszönettel tartozom családomnak az eddigi egyetemi éveim alatt nyújtott támogatásukért és türelmükért.
Továbbá szeretném megköszönni az oktatóim problémára való nyíltságát, és segítségüket abban, hogy fenntartsam a szenvedélyemet a matematika iránt.

# 1. Bevezető

A matematika hasznos, és szerethető. Azonban sokszor találkozunk az érdeklődés felkeltésének hiányával, vagy a motiváció elvesztésével.
Gyakran a matematika szerethetőségének megtapasztalásában az érdeklődőket a feszes, nagyszámú tételek és fogalmak megállítják. Sokszor hosszú tételsorok után is az elért eredmény kevésbé látványos egy felületesebben a témában szemlélődőnek annál, mint hogy motivációt érezzen a megértésre és a folyatásra.
Szakdolgozatomban a reál érdeklődésű emberek figyelmét és szeretetét szeretném újra felébreszteni a matematika iránt azzal, hogy a tételek és definíciók helyett a metódusokat, megértést segítő és interaktív példákat helyezem előtérbe. Ehhez korunk próbálkozásaival és eredményeivel egy közkedvelt és tapasztalati tudományágat, a kriptográfiát (nagyvonalakban a kódírás, -fejtés, -törés) választottam.

Ebben a fejezetben egy összetettebb matematikai struktúra, a csoportgyűrű kerül bemutatásra, műveleteivel egyetemben. Továbbá a közös kulcsgenerálás lépéseit, hasznosságának tárgyalását is magába foglalja. Ezek az alapok képzik a dolgozat első tárgyalt titkosításának alapját.

## 1.1. A csoportgyűrűkről

A dolgozat első kriptorendszerének mátrixai csoportgyűrűelemekből épülnek fel, így érdemes a csoportgyűrű fogalmának áttekintésével kezdenünk.

**1.1.1. Definíció.** Egy csoport egy olyan $(G, *)$ rendezett pár, ahol $G$ egy halmaz és a $*$ egy olyan bináris művelet $G$-n, amely teljesíti a következő tulajdonságokat:

- Asszociatív. ($x * (y * z) = (x * y) * z$ minden $x, y, z \in G$-re.)
- Invertálható. (Létezik egy olyan $e \in G$ elem, amelyre $e * x = x$ és $x * e = x$ minden $x \in G$ esetén, és minden $x \in G$ esetén létezik egy olyan $y \in G$ elem amelyre $x * y = e$ és $y * x = e$.)

*Megjegyzés.* Abel-csoport, más néven kommutatív csoport az olyan csoport, amelyekben a csoportművelet kommutatív.

*Megjegyzés.* Multiplikatív csoportról beszélünk, ha $*$ szorzás.

*Megjegyzés.* Egy $G$ halmaz a rajta értelmezett $*$ művelettel félcsoport, ha a halmaz zárt a műveletre nézve és a művelet asszociatív.

**1.1.2. Definíció.** A két kétváltozós művelettel rendelkező $(R, +, *)$ struktúrákat gyűrűnek nevezünk, ahol $(R, +)$ Abel csoport,$(R, *)$ félcsoport, tovább a disztributivitás fennáll a műveletekre (minden $x, y, z \in R$-re $x * (y + z) = (x * y) + (x * z)$, és $(x + y) * z = (x * z) + (y * z)$).

**1.1.3. Definíció.** Legyen $G$ egy multiplikatív csoport, és $R$ egy egységelemes asszociatív gyűrű. Jelölje

$$R[G] := \{u | u : G \to R, \#\{g \in G | u(g) \neq 0\} < \infty\}$$

azon függvények halmazát, amelyben az $u$ és $v$ függvények akkor egyenlők, ha $v(g) = u(g)$ teljesül $\forall g \in G$-re.

**1.1.4. Definíció.** A függvények összegét és szorzatát a következőképpen definiálhatjuk $(u + v)(g) := u(g) + v(g)$-t és $(u * v)(g) := \sum_{h \in G} u(h) * v(h^{-1} * g)$.

**1.1.5. Definíció.** Az $R[G]$ a két (jóldefiniált) műveletre gyűrűt alkot, és ezt nevezzük **csoportgyűrűnek**.

Ezt követően a választott $u \in R[G]$ függvénynek egyértelműen megfeleltethető

$$a := \sum_{g \in G} u(g) * g$$

formális összegre térünk át. A formális összegként kezelt függvényekkel a bevezetett összeadást és szorzást is újradefiniáljuk:

**1.1.6. Definíció.** Legyenek

$$a := \sum_{g \in G} u(g) * g \qquad b := \sum_{g \in G} v(g) * g$$

$a, b$ tetszőleges $u, v \in R[G]$ elemeknek megfeleltethetőek. Ekkor:

$$a + b =: \sum_{g \in G} (u(g) + v(g)) * g \qquad a * b =: \sum_{g \in G} (\sum_{h \in g} u(h) * v(h^{-1} * g) * g$$

.

Ezek ismeretében a következő fejezetekben már áttérhetünk a $\mathbb{Z}_m[S_n], n, m \in \mathbb{N}/\{1\}$ csoportgyűrűk tárgyalására, és ezen belül is az $n = 3, m = 7$ könnyen kezelhető esetre.
Végezetül, tekintsük a következő példát:

$$a = 2(1, 2) + 4(1, 2, 3) \qquad b = 6(2, 3)$$

$$a + b = 6(2,3) + 2(1,2) + 4(1,2,3)$$

$$a * b = 2(1,2) + 4(1,2,3) + 5(1,3,2) + 3(1,3) \neq b * a = 5(1,2) + 2(1,2,3)$$

## 1.2. A kriptográfiáról és a közös kulcsgenerálásról

A kriptológia a biztonságos kommunikáció tudománya. Két ága a kriptográfia és a kriptoanalízis. A kriptográfia olyan módszerekkel foglalkozik, melyek biztosítják az üzenetek védettségét és hitelességét. Alapja a betűk átrendezése és helyettesítése. A kriptoanalízis a kódolt szövegek kulcs nélküli megfejtését jelenteti.

A kriptográfiában az I. világháború alatt számos eljárást dolgoztak ki, és törtek fel rövid időn belül. A II. világháború alatt megkezdődött a titkosító gépek tömeges alkalmazása, ami nagy lendületet adott a matematika és az informatika fejlődésének. A rejtjelező eszközökre híres példa a második világháborúban a német Enigma, a japán Purple,a brit Typex, és az amerikai SIGABA, továbbá a navahó indiánok nyelve is.

Modern kriptográfia a számítógép létrejöttével gyorsabb és hatékonyabb kódolásra képes. Emellett a kódolás alapja bináris, különböző kódokkal, például ASCII-vel készül a kódolás. Az Amerikai Szabványügyi Hivatal pályázatot írt ki, és szabványnak a DES-t választotta, egy szimmetrikus kulcsú rejtjelezést. A kulcs szétosztása csak személyesen volt biztonságos. Ezt a problémát a kulcsmegosztás hidalta át [1].

A dolgozat első kriptorendszerének váza az 1970-es években megalkotott Diffie-Hellman közös kulcsgenerálás, ezért kitérünk az algoritmus általánosított felépítésére.

*Megjegyzés.* A rendszerek bemutatása és használata során legyen $A$ fél, azaz a kommunikációt kezdő fél neve Alice, az Alice üzenetét fogadó $B$ fél neve pedig Bob, a kriptográfia hagyományait követve.

A szimmetrikus titkosítás lépései (nagyvonalakban) a következők:

- Alice valamilyen biztonságos csatornán eljuttatja a titkos kulcsát Bobnak.
- Alice a titkos kulccsal kódolja az üzenetet, majd közzéteszi.
- Végül Bob a titkos kulcs segítségével visszakódolja.

A szimmetrikus titkosítók első lépésének támadhatósága motiválta Whitfield Diffie és Martin Hellman közös kulcsgeneráló módszerét:

- Adott egy nyilvános üzenet.

- Mindkét fél titkosítja egy saját, titkos kulccsal az üzenetet, majd közzéteszi.
- Ezután egymás kódolt üzenetét a saját kulcsukkal továbbkódolva, mindkettőjüknek ugyanaz a duplán kódolt üzenet áll rendelkezésére.

Így a kódolt üzenet kezelhető közös kulcsként egy szimmetrikus titkosításnál.

*Megjegyzés.* Ezzel a módszerrel a komplett kommunikáció is lebonyolítható:
Alice az $A$ kulcsával titkosítja az üzenetet. Ekkor Eve nem tudja elolvasni az $A$ kulcs ismerete nélkül.
Bob sem tudja visszafejteni, ezért a saját $B$ kulcsa segítségével tovább alakítja az üzenetet. Eve ekkor a két kulcs ismeretének hiányában nem tud dekódolni.
Ezután Alice visszafejti az $A$ kulcsos titkosítást, de Eve elől még a $B$ kulcs miatt az üzenet tényleges tartalma védve van.
Végül Bob dekódol a $B$ kulcs alapján.

A Diffie-Hellman titkostást használhatjuk moduló egy prím hatványozást véve, így a diszkrét logaritmusok biztosítanak védettséget. Egy másik lehetőség például, hogy az üzenet egy könnyen kezelhető fokszámú polinom, a két külön kulcs pedig, amivel titkosításként szorzunk, magas fokszámú polinomok. Így a gyökkeresés lesz problémás az illetéktelenek számára.

# 2. Közös kulcsgenerálás csoportgyűrűk feletti mátrixokban

## 2.1. A felépítésről

Nagy elemszámú halmazokon érdemes a közös kulcsgenerálást végezni, mivel így a kriptoszöveg titkossága biztosítva van a random elemválasztásos visszafejtés, és a memóriaigényes törések ellen.
$\mathbb{M}_l\mathbb{Z}_m[S_n]$-nek, azaz a $\mathbb{Z}_m[S_n]$ csoportgyűrű felett vett $l \times l$-es mátrixok halmazának igen kis $(l, m, n)$ választással is viszonylag nagy az elemszáma.

- $S_n$ elemei $n$ elem permutációinak összessége, azaz $n!$.
- Minden $S_n$-beli elemhez egy $\mathbb{Z}_m$-beli számot rendelünk, így $\mathbb{Z}_m[S_n]$ elemszáma $m^{n!}$.
- Egy $l \times l$-es mátrix $l^2$ darab eleme $(m^{n!})^{l^2} = m^{n! * l^2}$ féleképpen tölthető fel az adott csoportgyűrű elemeivel.

Így például $(2, 7, 3)$ választással a $\mathbb{M}_2\mathbb{Z}_7[S_3]$-nak $7^{24} \sim 1.9 * 10^{20}$ eleme van. A továbbiakban részletesebben ezen a halmazon reprezentáljuk a protokoll lépéseit.

## 2.2. A közös kulcsgenerálás menetéről

A használt csoportgyűrű egy kitüntetett $g$ eleme, és az eljárás menete nyilvános. A kulcsgenerálás a következőképpen történik:

- Alice választ egy $a$ elemet a csoportgyűrűből, majd nyilvánossá teszi $g^a = g_a$-t.
- Közben Bob szintén választ egy $b$ elemet a csoportgyűrűből, és $g^b = g_b$-t nyilvánossá teszi.
- Alice kiszámolja ${g_b}^a$-t.

- Közben Bob szintén kiszámolja ${g_a}^b$-t. Alice és Bob most ugyanazzal a $g_k = {g_a}^b = {g_b}^a = g^{a*b}$ titkos közös kulccsal rendelkezik [2].

Ezt követően már megkezdődhet a közös kulcs segítségével a tényleges kommunikáció.
A nagy elemszám mellett a protokoll további előnye, hogy a mátrixhatványozás nem túl memória- és műveletigényes [3], így kellően gyorsan végezhető a kulcsgenerálás.

## 2.3. A közös kulcsgenerálás bemutatása választott példán

A könnyebb megértés érdekében érdemes megnéznünk egy konkrét példát is. Az átláthatóság kedvéért a jegyzet a fentebbi megfogalmazást követi. A használt csoportgyűrű, ez esetben $\mathbb{M}_2\mathbb{Z}_7[S_3]$ egy $g$ eleme, és az eljárás menete nyilvános.

$$g = \begin{pmatrix} 3(1,2) + (1,2,3) & 2(1,2) + 5(1,3) \\ 2(2,3) + 2(1,3) & (1,3) \end{pmatrix}$$

A kulcsgenerálás a következőképpen történik:

- Alice választ egy $a \in \mathbb{M}_2\mathbb{Z}_7[S_3]$, majd nyilvánossá teszi $g^a = g_a$-t. Legyen $a = 3$.

$$g_a = \begin{pmatrix} 2 + 4(2,3) + 3(1,2) + 3(1,2,3) + 2(1,3,2) + (1,3) & 4(2,3) + 6(1,2) + 4(1,2,3) + 2(1,3,2) + 4(1,3) \\ 5 + 4(2,3) + 2(1,2) + (1,2,3) + (1,3,2) + (1,3) & 4 + 4(2,3) + 6(1,2) + 3(1,3,2) + 5(1,3) \end{pmatrix}$$

- Közben Bob szintén választ egy $b \in \mathbb{M}_2\mathbb{Z}_7[S_3]$ elemet, és $g^b = g_b$-t nyilvánossá teszi. Legyen $b = 10$.

$$g_b = \begin{pmatrix} 5 + (2,3) + 5(1,2) + 4(1,2,3) + (1,3,2) + 2(1,3) & 3 + 5(2,3) + 2(1,2) + 4(1,3,2) \\ 5(2,3) + 4(1,2,3) + 2(1,3) & 2 + 6(2,3) + 3(1,2) + 4(1,2,3) + 2(1,3,2) + 5(1,3) \end{pmatrix}$$

- Alice kiszámolja ${g_b}^a$-t.

- Közben Bob szintén kiszámolja ${g_a}^b$-t. Alice és Bob most ugyanazzal a $g_k = {g_a}^b = {g_b}^a = g^{a*b}$ titkos közös kulccsal rendelkezik, azaz a következő $\mathbb{M}_2\mathbb{Z}_7[S_3]$-beli elemmel:

$$g_k = g^{3*10} = \begin{pmatrix} 6 + 4(2,3) + 4(1,2) + 4(1,2,3) + 5(1,3,2) + 6(1,3) & 4(2,3) + (1,2) + 5(1,2,3) + (1,3,2) + 2(1,3) \\ 5 + 5(2,3) + 4(1,2) + 2(1,2,3) + 5(1,3) & 2 + 3(2,3) + 4(1,2) + 6(1,2,3) \end{pmatrix}$$

*Megjegyzés.* A csoportgyűrűs mátrixokban való kulcsgenerálás törhető. A probléma visszavezethető véges testek feletti diszkrét logaritmus problémára, ahol már létezik hatékony algoritmus [4].
Legyen $\mathbb{F}_q$ véges test.

- Az $\mathbb{M}_n(\mathbb{F}_q[G])$-beli DLP-t $\mathbb{M}_n(\mathbb{F}_q)$-beli DLP-vé redukáljuk.

- $\mathbb{M}_n(\mathbb{F}_q)$-beli DLP-t $\mathbb{F}_q$ bővítéseinek DLP-jére redukáljuk.

- Erre a DLP-re már létezik hatékony algoritmus.

Ezzel a metódussal a D. Kahrobaei, C. Koupparis és V. Shpilrain eredeti, 2013-as titkosító cikkében [2] kitűzött törési problémát Chris Monico és Mara D. Neusel oldotta meg, kevesebb, mint két évvel később [5].

# 3. Ariffin-Abu kriptorendszer

## 3.1. Az $AA_\beta$ felépítésének menetéről

Az $AA_\beta$ kriptorendszer Ariffin és Abu káosz-alapú nyilvános kulcsú kriptorendszere [6]. Egy Diffie-Hellman közös kulcsgenerálási rendszer, ahol egy szorzás operátorral dolgozunk, a valós számok halmazán moduló 1-ben. Gyakorlatban, mivel az irracionális számok nem reprezentálhatóak számítógépen, valamilyen konzisztens közelítésükkel dolgozhatunk. Ennek a kódolási sémának a lánctörtes törés [7] által hangsúlyozott, diszkrét logaritmus moduló 1-beli probléma az alapja, és ezt előtérbe helyezve a következőképpen építhető fel:

Legyenek $A_0$ és $A_1$ $2 \times 2$-es mátrixok:

$$A_0 = \begin{pmatrix} 1 & \alpha \\ 1 & 0 \end{pmatrix} \qquad A_1 = \begin{pmatrix} \beta & 1 \\ 1 & 0 \end{pmatrix}$$

ahol $\alpha$ és $\beta$ ismert egészek,továbbá $x \in [0, 1[$ ismert valós szám. A fentieket felhasználva Alice és Bob közös kulcs generálása:

- 1. lépés:
  - Alice kiszámít egy $n_A$ privát egészet:
    - Választ egy tetszőleges $k$ bites bináris sztringet, $a_1 \dots a_k$-t.
    - Kiszámítja az $A = A_{a_k} A_{a_{k-1}} \dots A_{a_1}$ egész elemű mátrixot.
    - $n_A$-nak $A[1, 1]$-et választja.
  - $B$ a saját $k$ bites bináris sztringjét megválasztva analóg módon kiszámít egy $n_B \in \mathbb{Z}$-t.
- 2. lépés:
  - $A$ kiszámítja $r_A \equiv n_A * x \pmod 1$ -et, és elküldi $B$-nek.
  - $B$ kiszámítja $r_B \equiv n_B * x \pmod 1$ -et, és elküldi $A$-nak.
- 3. lépés: A megosztott közös kulcs $n_A n_B \equiv n_B n_A \pmod 1$, ezzel a tényleges kommunikáció megkezdhető.

## 3.2. Az $AA_\beta$ bemutatása választott példán

Az $AA_\beta$ kriptorendszer megértését segíti a következő számszerűsített példa:

A kriptorendszer $(\alpha, \beta, x) \in \mathbb{Z}^2 \times [0,1[$ paraméterezését válasszák a felek $(2, 3, 0.5678354675678\ldots)$-nek. Így $A_0$ és $A_1$ $2 \times 2$-es mátrixok

$$A_0 = \begin{pmatrix} 1 & 2 \\ 1 & 0 \end{pmatrix} \qquad A_1 = \begin{pmatrix} 3 & 1 \\ 1 & 0 \end{pmatrix}$$

lesznek, továbbá legyenek a későbbi sztringek $k = 6$ hosszúságúak. Alice és Bob közös kulcs generálása:

- 1. lépés:
  - Alice kiszámít egy $n_A$ privát egészet:
    - $110010 = a_6 \ldots a_1$ bináris sztringet választja.
    - Kiszámítja az $A$ mátrixot a következőképpen:

$$A = A_{a_6} \ldots A_{a_1} = A_0 A_1 A_0 A_0 A_1 A_1 =$$

$$= \begin{pmatrix} 1 & 2 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 3 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 3 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 3 & 1 \\ 1 & 0 \end{pmatrix} =$$

$$= \begin{pmatrix} 196 & 60 \\ 124 & 38 \end{pmatrix}$$

    - $n_A$-nak $A[1,1] = 196$-ot választja.
  - Bob a saját 6 bites 000110 bináris sztringjéből analóg módon előállítja $n_B = 95$-öt.
- 2. lépés:
  - Alice kiszámítja $r_A \equiv 196 * 0.5678354675678\ldots \equiv 0.295751643\ldots \pmod 1$ -et, és elküldi Bobnak.
  - Bob kiszámítja $r_B \equiv 95 * 0.5678354675678\ldots \equiv 0.944369419\ldots \pmod 1$ -et, és elküldi Alice-nak.
- 3. lépés:
  A megosztott közös kulcs $n_A n_B \equiv 0.0964061124\ldots \pmod 1$.

# 4. Az Ariffin-Abu rendszer törése

## 4.1. Röviden a lánctörtekről

Az Ariffin-Abu kriptorendszer töréséhez szükséges ismernünk a lánctörteket. Ezt a témát öleli fel az alfejezet.

A lánctört alak valós számokra a következő reprezentációs formát jelenti:

$$a = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cfrac{1}{a_4 + \dots}}}} := [a_0, a_1, a_2, a_3, a_4, \dots]$$

ahol $a \in \mathbb{R}, a_0 := [a], a_1 := [\frac{1}{\{a\}}], a_i = [\frac{1}{\{a_{i-1}\}}], i \in \mathbb{N}^+/1$ és $[.], \{.\}$ rendre az egészrész, törtrész függvény.

A konstrukció alatt

$$a = a_0 + \frac{b_2}{b_1} = a_0 + \frac{1}{a_1 + \frac{b_3}{b_2}} = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{b_4}{b_3}}} = \dots$$

részekre bomlik $a$, ahol $b_1 \geq b_2 \geq b_3 \geq b_4 \geq \dots$ nemnegatív egész sorozat. Így szigorúan monoton csökkenő $b_i$ sorozat esetén az 1 értékű tag, egyenlőség esetén pedig az egyértelműen következő 0 tag jelenti a sorozat végét. Így minden racionális szám, azaz véges és végtelen szakaszos tizedes tört alakban megjeleníthető szám, felírató véges lánctört alakban. Továbbá minden irracionális szám felírható végtelen lánctört alakban, mindig eggyel több jegyű véges lánctörtet számolva ki.

## 4.2. A diszkrét logaritmus probléma megoldásáról modulo 1-ben

Legyenek $x, y \in [0, 1[$ ismertek. Keressük $n \in \mathbb{Z}$-t, amivel

$$nx \equiv y \pmod 1$$

.

- $y = 0$ esetben: $n = 0$ megoldás, tehát feltehetjük, hogy $y \neq 0$.

- Ezért $x$ nem írható fel $\frac{p}{n}, p \in \mathbb{Z}$ formában, mert

$$n * \frac{p}{n} \equiv y \pmod 1 \Rightarrow p \equiv y \pmod 1 \Rightarrow y = 0$$

  lenne.

- Feltehetjük, hogy $n$ pozitív. Negatív $n$-ekre az $\overline{n}x \equiv \overline{y} \pmod 1$ kongruenciával számolhatunk, ahol $\overline{n} := -n, \overline{y} := 1 - y$.

$x, y$ adottak, $n$ pozitív, $y \neq 0$. Kiszámítunk egy kellően jól approximáló $\frac{a}{b}$ konvergenst $x$-hez. Ekkor a kongruenciából $n$ közelítése kinyerhető:

$$nx \equiv y \pmod 1$$

$$n\frac{a}{b} \cong y \pmod 1$$

$$na \cong yb \pmod b$$

$$n \cong yba^{-1} \pmod b$$

A DLP: $nx \equiv y \pmod 1$ pozitív $n$-ekre a következőképpen oldható meg [7]: $x, y$ valós számok adottak.

- Ha $y = 0$, akkor $n = 0$, és az algoritmus megáll.

- Ha $y \neq 0$, kiszámítjuk a konvergenseket $x$-hez, a lánctörtté alakítást használva. Minden $\frac{a}{b}$ konvergensre:

  - Kiszámítja $a'$-t, a legközelebbi egészet $by$-hoz.
  - Kiszámítja az egyértelmű $n'$ egészet, amivel $n' \equiv a'a^{-1} \pmod b$ és $0 \leq n' \leq b$.
  - Ha $n'x \equiv y \pmod 1$, akkor $n = n'$, és az algoritmus megáll.
  - Egyébként áttér a következő konvergens vizsgálatára.

## 4.3. A megfelelően közeli konvergensválasztásról, és a hibahatárról

Mivel a nevezők bármely valós szám $\frac{a}{b}$ konvergenseiben nőnek, így $x \in \mathbb{R}$-nek van olyan $\frac{a}{b}$ approximációja, amire $b \geq 2n$.

$nx \equiv y \pmod 1$ miatt $\exists m \in \mathbb{Z} : nx = y + m$. Így $x$ minden $\frac{a}{b}$ konvergensével igaz, hogy

$$n\tfrac{a}{b} \text{ megközelítőleg } y + m$$
$$na \text{ megközelítőleg } yb + mb$$
$$na - mb \text{ megközelítőleg } yb.$$

Legyen a hiba $\epsilon := x - \frac{a}{b}$.

$$na - mb = nb\frac{a}{b} - mb = nb(x - \epsilon) - mb = bnx - bn\epsilon - mb = b(y + m) - nb\epsilon$$

Szeretnénk a törésnél a $by$-hoz legközelebbi $a'$ egésszel dolgozni. Tegyük fel, hogy vannak olyan konvergensek, ahol $a'$ az $na - mb$ egésszel egyezik.
Ehhez az kell, hogy $m = 0$ és $|nb\epsilon| < \frac{1}{2}$ teljesüljön, $na - mb = by + bm - nb\epsilon$ miatt.

- Válasszuk maximális hibahatárnak $\frac{1}{b^2}$-et, azaz legyen $|\epsilon| < \frac{1}{2}$. Így $|nb\epsilon| < \frac{nb}{b^2} = \frac{n}{b} \leq \frac{1}{2}$.
- Ha $a' = na - mb$, akkor $a' \equiv na \pmod b$, és mivel $\frac{a}{b}$ konvergenseknél $a$ és $b$ relatív prímek, $n \equiv a'a^{-1} := n' \pmod b$. De mivel $n \leq \frac{1}{2}b < b$, és $n' < b$, így nincs moduláris redukció, azaz $m = 0$, és $n = n'$.

Az $2n \leq b$ küszöböt elérő, és $\frac{1}{2}$ maximális hibahatárt kielégítő konvergensek esetén $a' \approx by$ kerekítéssel hatékonyan és egyértelműen dolgozhatunk.

## 4.4. A diszkrét logaritmus probléma megoldása moduló 1-ben, választott példán

Tekintsük az $y \equiv r_A \equiv n_A * x \pmod 1$ DLP-t az előző fejezetből.

- $y \neq 0$.
- Kiszámítjuk a konvergenseket $x = 0.78217087686061859\ldots$-hez, a lánctörté alakítást használva:

$$x = 0.78217087686061859\ldots = 0 + \cfrac{1}{1 + \cfrac{1}{3 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{2 + \ldots}}}}}$$

Az $x$-hez tartozó kovergenssorozat tehát

$$0 + 0 = 1$$

$$0 + \frac{1}{1 + 0} = 1$$

$$0 + \frac{1}{1 + \frac{1}{3 + 0}} = \frac{3}{4}$$

$$0 + \frac{1}{1 + \frac{1}{3 + \frac{1}{1 + 0}}} = \frac{4}{5}$$

$$\ldots \frac{7}{9}; \frac{18}{23}; \frac{61}{78}; \frac{79}{101}; \frac{1009}{1290}; \frac{1088}{1391}; \frac{2097}{2681} \ldots$$

Minden konvergensre, így a megfelelő $\frac{2097}{2681}$-re is:

- Kiszámítja $a'$-t, a legközelebbi egészet $by$-hoz.
- Kiszámítja az egyértelmű $2415 := n' \equiv a'a^{-1} = 2527{*}2097^{-1} \pmod{2681}$, $0 \leq n' \leq b$ egészet.
- $n'x \equiv y \pmod 1$, azaz $2415{*}0.78217087686061859\ldots \equiv 0.94266761839389801\ldots$, akkor $n = n'$, és az algoritmus megáll.
- Visszakaphatjuk $n = n_A$-t $b * y = 2681 * 0.94266761839389801\ldots$ kerekítésével a legközelebbi egészhez, mi megtaláljuk az $n' = 2415 = n_A$-t, amit kerestünk.(Hasonlóképpen visszakaphatjuk $n_B = 3725$-öt, használva az $\frac{5282}{6753}$ konvergenst $x$-re, az $y = r_B$ értéket, és $a' = 3961$ számítását.

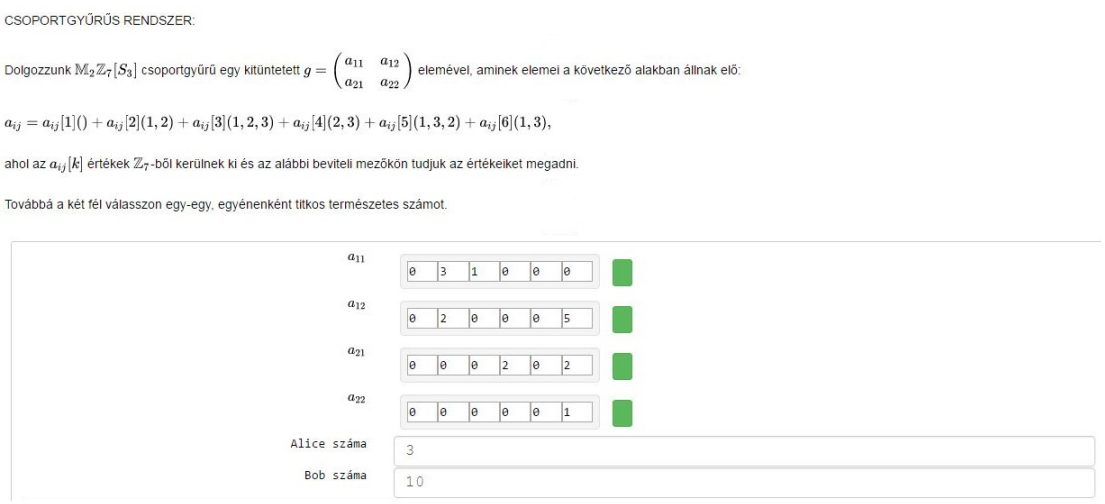
# 5. Sage kódok

Ebben a fejezetben a dolgozat algoritmusait implementáltam a SageMath [8] programcsomagban.

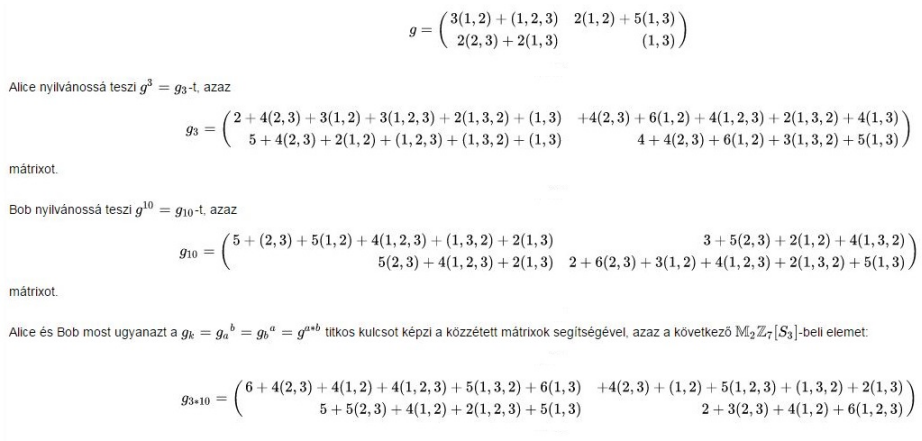
```
pretty_print(html(r'<p>CSOPORTGYŰRŰS RENDSZER:</p>'))

h1=SymmetricGroup(3).gen(0)
h2=SymmetricGroup(3).gen(1)

G3S7=GroupAlgebra(SymmetricGroup(3),Integers(7))

pretty_print(html(r'<p>Dolgozzunk
$\mathbb{M} _2 \mathbb{Z} _7[S_3]$ csoportgyűrű egy kitüntetett
$g=\left(\begin{array}{cc}a_{11}&a_{12}\\
a_{21}&a_{22}\end{array}\right)$
elemével, aminek elemei a következő alakban állnak elő:</p>'))

pretty_print(html(r'<p>$a_{ij}=a_{ij}[1]()+a_{ij}[2](1,2)
+a_{ij}[3](1,2,3)+a_{ij}[4](2,3)+a_{ij}[5](1,3,2)
+a_{ij}[6](1,3),$</p>'))

pretty_print(html(r'<p>ahol az $a_{ij}[k]$ értékek
$\mathbb{Z}_7$-ből kerülnek ki és az alábbi beviteli mezőkön
tudjuk az értékeiket megadni.</p>'))

pretty_print(html(r'<p>Továbbá a két fél válasszon egy-egy,
egyénenként titkos természetes számot.</p>'))

@interact
def
_(a11=input_grid(1,6, default = [0,3,1,0,0,0], label='$a_{11}$'),
a12=input_grid(1,6, default = [0,2,0,0,0,5], label='$a_{12}$'),
a21=input_grid(1,6, default = [0,0,0,2,0,2], label='$a_{21}$'),
a22=input_grid(1,6, default = [0,0,0,0,0,1], label='$a_{22}$'),
```

```
x=('Alice száma',3),y=('Bob száma',10)):

    A=[0,0,0,0]
    B=[a11,a12,a21,a22]
    k=0

    for n in range(4):
        for i in range(3):
            for j in range(2):
                A[n]=A[n]+G3S7(B[n][0][k])*h1^i*h2^j
                k=k+1
        k=0

    g=matrix([[A[0],A[1]],[A[2],A[3]]])

    pretty_print(html('$$g=%s$$'%latex(g)))

    pretty_print(html(r'<p>Alice nyilvánossá teszi
    $g^{%s}=g_{%s}$-t, azaz $$g_{%s}=%s$$ mátrixot.</p>'
    %(x,x,x,latex(g^x))))

    pretty_print(html(r'<p>Bob nyilvánossá teszi
    $g^{%s}=g_{%s}$-t, azaz $$g_{%s}=%s$$ mátrixot.</p>'
    %(y,y,y,latex(g^y))))

    pretty_print(html(r'<p>Alice és Bob most ugyanazt a
    $g_{k}={g_a}^b={g_b}^a=g^{a*b}$ titkos kulcsot képzi
    a közzétett mátrixok segítségével, azaz a következő
    $\mathbb{M} _2\mathbb{Z} _7[S_3]$-beli elemet:</p>'))

    pretty_print(html(r'<p>$$g_{%s*%s}=%s$$</p>'
    %(x,y,latex((g^x)^y))))
```

5.1. ábra. Csoportgyűrűs rendszer




5.2. ábra. Csoportgyűrűs rendszer




```
pretty_print(html(r'<p>ARIFFIN-ABU RENDSZER:</p>'))

pretty_print(html(r'<p>A kriptorendszer
$\alpha ,\beta ,x \in \mathbb{Z} ^2 \times [0,1[$
paraméterezését válasszák a felek meg a következőképpen,
továbbá $A$ és $B$ felek válasszanak egy-egy tetszőleges
$6$ bites bináris sztringet, rendre
$a_1\ldots a_6$-t és $b_1\ldots b_6$-t.</p>'))

A0=MatrixSpace(ZZ,2,2)
A1=MatrixSpace(ZZ,2,2)
```

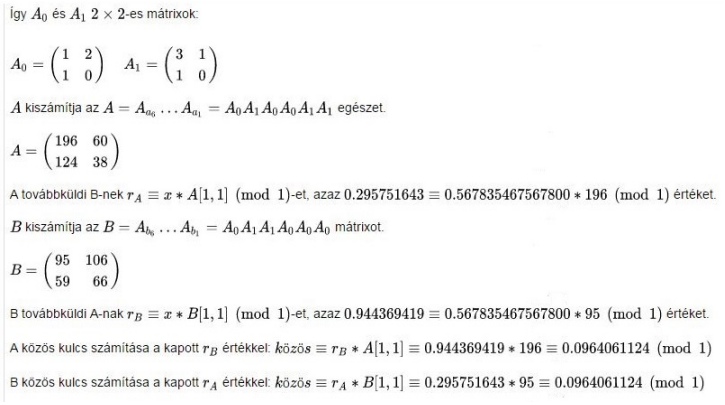
```
@interact
def _(valt=input_grid(1,2, default = [2,3],
label=r'$ \alpha , \beta $'), x=('x',0.5678354675678),
a=input_grid(1,6, label='$a_1 \ldots a_6$',
default=[1,1,0,0,1,0]),
b=input_grid(1,6, label='$b_1 \ldots b_6$',
default=[0,0,0,1,1,0])):

    ii=0
    for i in range(6):
        if a[0][i]==1 or a[0][i]==0:
            ii=ii+1
    jj=0
    for j in range(6):
        if a[0][j]==1 or a[0][j]==0:
            jj=jj+1

    if ii!=6:
        pretty_print(html(r'<p>A félnek bináris sztringet
        kell választania.</p>'))

    elif jj!=6:
        pretty_print(html(r'<p>B félnek bináris sztringet
        kell választania.</p>'))

    else:
        pretty_print(html(r'<p>Így $A_0$ és $A_1$
        $2\times 2$-es mátrixok:</p>'))

        A0=matrix([[1,valt[0][0]],[1,0]])
        A1=matrix([[valt[0][1],1],[1,0]])

        pretty_print(html('$A_0=%s \quad A_1=%s$'
        %(latex(A0),latex(A1))))

        pretty_print(html('$A$ kiszámítja az
        $A=A_{a_6} \ldots A_{a_1}=A_%sA_%sA_%sA_%sA_%sA_%s$
        egészet.'
        %(a[0][5],a[0][4],a[0][3],a[0][2],a[0][1],a[0][0])))
        A=matrix([[1,0],[0,1]])
```

```
while i>=0:
    if a[0][i]==1:
        A=A*A1
    else:
        A=A*A0
    i=i-1

pretty_print(html('$A=%s$'%latex(A)))

rA=(x*A[0,0]) % 1
if rA<0:
    rA=1+rA
pretty_print(html('A továbbküldi B-nek
$r_A \equiv x*A[1,1] \pmod 1$-et,
azaz $%s \equiv %s*%s \pmod 1$ értéket.'
%(rA.n(digits=9),x,A[0][0])))

pretty_print(html('$B$ kiszámítja az
$B=A_{b_6} \ldots A_{b_1}=A_%sA_%sA_%sA_%sA_%sA_%s$
mátrixot.'
%(b[0][5],b[0][4],b[0][3],b[0][2],b[0][1],b[0][0])))
B=matrix([[1,0],[0,1]])
while j>=0:
    if b[0][j]==1:
        B=B*A1
    else:
        B=B*A0
    j=j-1

pretty_print(html('$B=%s$'%latex(B)))
rB=(x*B[0,0]) % 1
if rB<0:
    rB=1+rB

pretty_print(html('B továbbküldi A-nak
$r_B \equiv x*B[1,1] \pmod 1$-et,
azaz $%s \equiv %s*%s \pmod 1$ értéket.'
%(rB.n(digits=9),x,B[0][0])))

koz1=(rB*A[0][0]) % 1
```

```
if koz1<0:
    koz1=1+koz1

pretty_print(html('A közös kulcs számítása
a kapott $r_B$ értékkel:
$közös \equiv r_B*A[1,1] \equiv %s*%s \equiv %s \pmod 1$'
%(rB.n(digits=9),A[0][0],koz1.n(digits=9))))

koz2=(rA*B[0][0]) % 1

if koz2<0:
    koz2=1+koz2

pretty_print(html('B közös kulcs számítása a kapott
$r_A$ értékkel:
$közös \equiv r_A*B[1,1] \equiv %s*%s \equiv %s \pmod 1$'
%(rA.n(digits=9),B[0][0],koz2.n(digits=9))))
```

5.3. ábra. Ariffin-Abu rendszer


5.4. ábra. Ariffin-Abu rendszer

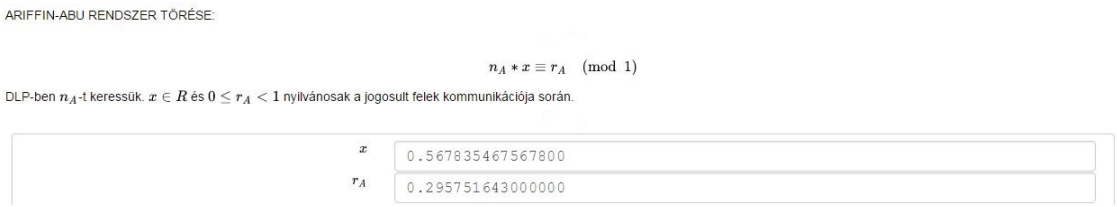



```
pretty_print(html(r'<p>ARIFFIN-ABU RENDSZER TÖRÉSE:</p>'))

pretty_print(html(r'<p>$$n_A*x \equiv r_A \pmod 1$$ DLP-ben
$n_A$-t keressük. $x \in R$ és $0 \leq r_A < 1$ nyilvánosak
a jogosult felek kommunikációja során.</p>'))

@interact
def _(x=('$x$',0.5678354675678),rA=('$r_A$',0.295751643)):

    if x==0:
        if rA==0:
            pretty_print(html(r'<p>$n_A \in R$ minden esetben
            kielégíti a kongruenciát, ez az eset triviális.</p>'))

        else:
            pretty_print(html(r'<p>$n_A$ nem létezik,
            ez az eset triviális.</p>'))
```

```
else:
    if rA==0:
        pretty_print(html(r'<p>$n_A=0$,
        ez az eset triviális.</p>'))

    elif rA<0 or 1<=rA:
        pretty_print(html(r'<p>A $0 \leq r_A < 1$
        kritérium nem teljesül.</p>'))

    else:
        pontos=len(x.str())-1
        Rp=RealField(pontos+100)

        ct=QQ(x).continued_fraction()

        i=2
        hiba=1

        convs=Sequence([])

        while abs(hiba)>0 and i<30:
            a=ct.numerator(i)
            b=ct.denominator(i)
            convs.extend([a/b])
            brA=round(b*rA)
            nA=Integers(b)(brA)*(Integers(b)(a^(-1)))
            hiba=Rp(RR(nA)*x).frac().n(digits=17)-rA
            i=i+1

        pretty_print(html(r'<p>Az $x$-hez tartozó
        lánctörtekből képzett konvergenssorozat tagjai:</p>'))

        pretty_print(html(r'<p>$\ldots%s\ldots$</p>'
        %str(convs)[1:-1]))

        pretty_print(html(r'<p>$\frac{a}{b}=%s$
        kellően közeli konvergens x-hez.</p>'
        %latex(convs[i-3])))

        pretty_print(html(r'<p>Kiszámítjuk $\overline{a}$-t,
```

```
a legközelebbi egészet $b*r_A$-hoz:</p>'))

pretty_print(html
(r'<p>$\overline{a}\approx b*r_A=%s*%s\approx %s$</p>'
%(b,rA,brA)))

pretty_print(html(r'<p>A keresett $n_A$ így
$\overline{a}$ és $a^{-1} \pmod b$-beli
szorzata lesz:</p>'))

pretty_print(html
(r'<p>$n_A\equiv \overline{a}*a^{-1}\equiv
%s*%s\equiv %s \pmod {%s}$</p>'
%(brA,Integers(b)(a^(-1)),nA,b)))

pretty_print(html(r'<p>Az $n_A$ kulcs tehát:
%s</p>'%nA))

pretty_print(html(r'<p>Az $n_B$ kulcs $r_B$-vel
és $x$-el analóg módon számolható.</p>'))
```

5.5. ábra. Ariffin-Abu rendszer törése



5.6. ábra. Ariffin-Abu rendszer törése

Az $x$-hez tartozó lánctörtekből képzett konvergenssorozat tagjai:

$\ldots 1/2, 4/7, 21/37, 46/81, 67/118, 113/199, 745/1312, 3838/6759, 4583/8071, 8421/14830, 29846/52561, 38267/67391, 106380/187343, 995687/1753478, 3093441/5447777, 4089128/7201255,$
$11271697/19850287, 26632522/46901829, 26632522/46901829, 26632522/46901829, 26632522/46901829, 26632522/46901829, 26632522/46901829, 26632522/46901829, 26632522/46901829$
$26632522/46901829, 26632522/46901829, 26632522/46901829 \ldots$

$\frac{a}{b} = \frac{26632522}{46901829}$ kellően közeli konvergens x-hez.

Kiszámítjuk $\overline{a}$-t, a legközelebbi egészet $b * r_A$-hoz:

$\overline{a} \approx b * r_A = 46901829 * 0.295751643000000 \approx 13871293$

A keresett $n_A$ így $\overline{a}$ és $a^{-1} \pmod{b}$-beli szorzata lesz:

$n_A \equiv \overline{a} * a^{-1} \equiv 13871293 * 19850287 \equiv 196 \pmod{46901829}$

Az $n_A$ kulcs tehát:196

Az $n_B$ kulcs $r_B$-vel és $x$-el analóg módon számolható.

# Irodalomjegyzék


[1] *A kriptográfia története*, Wikipedia, 2015.
https://hu.wikipedia.org/wiki/A_kriptogr%C3%A1fia_t%C3%B6rt%C3%A9nete.

[2] Delaram Kahrobaei, Charalambos Koupparis and Vladimir Shpilrain: *Public Key Exchange Using Matrices Over Group Rings*, Cornell University Library, Computer Science, Cryptography and Security, 21 pages, 2013.
https://eprint.iacr.org/2013/114.pdf.

[3] Dr. Tengely Szabolcs: *Mátrixok n-edik hatványának zárt alakja*, Debreceni Egyetem, Oktatási segédletek, 2014.
http://shrek.unideb.hu/~tengely/Magyary/oktatas.html.

[4] Alex Myasnikov: *Discrete logarithm problem for matrices over finite group rings*, Stevens Institute of Technology, 18 pages.
http://web.stevens.edu/algebraic/GAGTA/Slides/Myasnikov.pdf.

[5] Chris Monico and Mara D. Neusel: *Cryptanalysis of a system using matrices over group rings*, Groups Complexity Cryptology, volume 7, 2015.

[6] M.R.K. Ariffin and N.A. Abu: *AAB-Cryptosystem: A Chaos Based Public Key Cryptosystem*, International Journal of Cryptology Research, 149-163, 2009.
http://www.mscr.org.my/V1(2)/PP%20149-163.pdf.

[7] Simon R. Blackburn: *The discrete logarithm problem modulo one: cryptanalysing the Ariffin-Abu cryptosystem*, Journal of Mathematical Cryptology, volume 4, pages 193-198 ,2010.
https://eprint.iacr.org/2010/114.pdf.

[8] W. A. Stein et al.: *Sage Mathematics Software (Version 7.1)*, The Sage Development Team, 2016.
http://www.sagemath.org.