Lliçó inaugural
del curs acadèmic
2014-2015

Facultat de Matemàtiques

Les matemàtiques al darrere de les criptomonedes

Elitza Maneva

Professora del Departament
de Matemàtica Aplicada i Anàlisi

Universitat de Barcelona

> Estic plenament convençut que no és suficient viure en aquest món tal qual, acceptant el que et presenten i seguint les coses que els adults t'han dit que facis i el que els teus pares t'han dit que facis i el que la societat t'ha dit que facis. Crec que sempre hauries d'estar interrogant. Jo assumeixo l'actitud científica que tot el que has après és provisional, que sempre està obert a la retracció o a la refutació o a la interrogació, i crec que el mateix val per a la societat.
>
> Aaron Swartz (1986-2013)

El fenomen Bitcoin per a molts representa una oportunitat per fer-se ric. Per a d'altres, representa una oportunitat per canviar les regles del joc financer i fer-les més justes. Per a mi, en aquest moment representa una oportunitat excel·lent per explicar com les matemàtiques fan possibles coses que a priori semblen impossibles, com són, per exemple, firmar documents sense ser físicament present en un lloc, regular el funcionament de sistemes distribuïts sense cap autoritat central, o demostrar que saps alguna cosa sense donar-ne més informació que el fet que la saps.

La criptografia ha entrat en gairebé tots els àmbits de la societat moderna, ja que és la branca de les matemàtiques que fa possible l'ús de xarxes públiques per a assumptes privats. Mitjançant internet, aquesta ciència ha canviat la manera com ens relacionem i gestionem el dia a dia. Donades les moltes ocasions en què la seguretat d'internet falla, segurament ja intuïu que aquesta ciència està molt poc desenvolupada. De fet, en bona part es basa en moltes conjectures matemàtiques.

Podem, doncs, considerar que les criptomonedes són una especulació amb peus de fang? Hi ha molts arguments en contra de la seva adopció, però la seguretat de les conjectures matemàtiques és un dels més febles perquè, en realitat, tots els serveis electrònics dels bancs i la seguretat de l'ús de paraules clau secretes es basa en aquestes mateixes conjectures. L'argument més fort en contra de les criptomonedes és que al seu darrere no hi ha cap govern que en reguli el valor. En aquesta exposició no entrarem en qüestions econòmiques ni en comparacions amb les monedes tradicionals, conegudes com *monedes fiduciàries* (*fiat money*). Per entendre millor el protocol i el potencial del sistema és millor deixar de banda les preconcepcions i pensar «outside the box».

1. Bitcoin

L'octubre de 2008 Satoshi Nakamoto (pseudònim d'una o més persones anònimes) va enviar un article titulat «Bitcoin: A Peer-to-Peer Electronic Cash System» a un grup de notícies de recerca en criptografia [5]. Dos mesos més tard en va publicar també la implementació de programari (o codi) obert. La idea va agradar a molts criptògrafs i programadors en general i es va formar una comunitat de gent que va començar a millorar el codi, fer circular les monedes (que no són més que seqüències de nombres) i generar-ne de noves. El protocol inclou una manera de generar monedes noves a mesura que es van efectuant transaccions (el mecanisme per fer-ho és molt interessant i hi tornarem més endavant). Al principi, les transaccions eren intercanvis més o menys simbòlics, com, per exemple, dotacions de premis o a canvi de programari, i, de mitjana, cada 10 minuts es generaven 50 bitcoins més. La primera compra de veritat que es va efectuar amb bitcoins va ser un any i tres mesos més tard. Va ser la compra d'una pizza a canvi de 10.000 bitcoins. Això representava el 0,4% de tots els bitcoins que existien en aquell moment. La cotització de 10.000 bitcoins quatre anys més tard va arribar als quatre milions d'euros.

A mesura que més gent va baixar-se el codi i va començar a efectuar transaccions, el valor dels bitcoins (BTC) va començar a pujar. Es van crear diversos mercats i negocis al voltant de Bitcoin. Avui dia hi ha milers de negocis que accepten pagament en bitcoins.

1.1. Protocol

El protocol és bastant senzill i les eines que fa servir són eines clàssiques de la criptografia. Absolutament totes les transaccions es guarden en un fitxer, anomenat *cadena de blocs* (*block chain*). Aquest fitxer està disponible públicament, es guarda i s'actualitza regularment als ordinadors de tots els usuaris.[1] Consisteix en una seqüència de blocs; cadascun conté una sèrie

1. Aquesta és una petita simplificació. Encara que va ser així al principi, com que la cadena s'ha fet gran (més de 20 GB a setembre de 2014) i com que hi ha usuaris que fan servir dispositius petits, com ara mòbils, avui dia molts clients no guarden tota la informació i

de transaccions de bitcoins. A més de les transaccions, cada bloc conté una quantitat de bitcoins nous en concepte de premi. Per exemple, un bloc podria contenir la informació següent:

- La Marta envia 2 BTC al Jordi.
- L'Elsa envia 1,5675566 BTC al Gerard.
- El Jordi rep 50 BTC nous.

És important subratllar que no hi ha cap autoritat que s'ocupi de mantenir la cadena de blocs. Tots els usuaris —qualsevol persona que s'hagi baixat el programari— col·labora en el manteniment de la informació i tothom la guarda localment al seu ordinador. Si algú intenta manipular la història no podrà fer-ho sol perquè la informació està replicada als ordinadors de tots els usuaris.

Els participants estan organitzats en una xarxa d'igual a igual (*peer to peer*): cada participant es pot comunicar amb un nombre relativament petit d'altres participants, però de tal manera que, si bona part dels usuaris reenvia la informació que rep a totes les seves connexions, aquesta informació arriba a tothom.

Si tu vols enviar una quantitat de bitcoins a algú, tot el que has de fer és anunciar-ho a les teves connexions. Ells comproven si tens els bitcoins que dius que vols gastar mirant a la cadena de blocs totes les transaccions en què has participat en el passat, i, si és així, al seu torn reenvien la transacció a les seves connexions.

En cada moment, qualsevol participant té una sèrie de transaccions que ha rebut però que encara no apareixen a la cadena de blocs. Fent servir una idea anomenada *demostració de feina* (*proof of work*), aquesta sèrie de transaccions es reinterpreta com un trencaclosques, la resolució del qual és, de fet, un bloc que es pot afegir al final de la cadena de blocs. El participant que aconsegueix resoldre el seu trencaclosques abans de rebre una solució d'un altre participant, envia el bloc a totes les seves connexions, que en comproven la validesa, l'afegeixen al final de la seva còpia de la cadena de blocs i el reenvien a les seves connexions.

---

només contribueixen a la connectivitat de la xarxa. La seguretat del protocol depèn dels usuaris amb la versió completa.

Els trencaclosques estan dissenyats perquè, en mitjana, se'n resolgui un cada 10 minuts entre tots els participants de la xarxa. Naturalment, l'autor de la resolució pot destinar els bitcoins nous en concepte de premi a si mateix o a qui vulgui. En tot cas, la quantitat de bitcoins que hi ha a la xarxa creix, en mitjana, cada 10 minuts, quan es genera un bloc nou, però el valor del premi és cada cop més baix, perquè es redueix a la meitat cada 210.000 blocs. L'any 2140 el valor del premi serà menys que un *satoshi*, que és la unitat mínima de bitcoins i equival a 10^{-8} BTC. Arribat aquest punt, en comptes de guanyar monedes noves, el guanyador del premi cobrarà només unes taxes voluntàries que s'especifiquen a cada transacció. Cada participant pot fer servir el seu criteri per incloure o no una transacció en el bloc en què està treballant, segons la taxa voluntària que s'especifiqui en aquesta transacció.

Per garantir que realment és la Marta qui autoritza la transacció «La Marta envia 2 BTC al Jordi» s'aplica una tècnica estàndard de la criptografia anomenada *criptografia de clau pública*. Aquesta tècnica ens proporciona una manera de firmar contractes de manera digital. Només si la frase està firmada per la Marta, la resta d'usuaris acceptaran la transacció com a autèntica. A la secció següent veurem un mètode concret per implementar firmes digitals basat en la teoria de nombres.

Els trencaclosques estan basats en les funcions de hash (també conegudes com funcions resum) que també es fan servir en el context de l'autenticació d'usuaris amb paraules clau, entre moltes altres aplicacions. Explicarem aquesta tècnica a la secció 3.

1.2. Evolució

El primer mercat de bitcoins va ser un web per intercanviar cromos anomenat Mt.Gox (una abreviació de Magic: The Gathering Online Exchange). Va fer fallida a principis de 2014 per problemes tecnològics, però en aquell moment ja s'havien creat moltes alternatives més fiables.

Després de Bitcoin s'han creat desenes d'altres varietats de criptomonedes inspirades en el programari de Bitcoin, o en alguns casos còpies gairebé idèntiques a Bitcoin. Un exemple és Luckycoin, que es diferencia de Bitcoin pel fet que la quantitat del premi en cada bloc és aleatòria. Un altre

és Dogecoin, que va començar a finals de 2013 com una broma (*doge* es refereix a una moda d'internet sobre fotos de gossos en diverses situacions, amb els pensaments del gos escrits en idioma de gos: «Wow. Much funny») i en sis mesos va arribar a tenir un valor total de 24 milions d'euros (comparat amb el valor de 5.000 milions d'euros del mercat de Bitcoin). Dogecoin, a més, és notable perquè se sol fer servir per a projectes altruistes com, per exemple, una campanya per ajudar a finançar l'equip de bob de Jamaica per anar als Jocs Olímpics de Sotxi i un altre per construir un pou a Kenya.

Fins ara només hi ha hagut un problema de seguretat en el programari de Bitcoin, trobat l'any 2010, i es va arreglar ràpidament. El problema més greu de Bitcoin que fa que els criptògrafs busquin alternatives és la falta d'anonimitat en el protocol [8]. Encara que els usuaris facin servir pseudònims —en general fins i tot fan servir pseudònims diferents per rebre diners de diferents fonts—, és molt fàcil fer servir la informació que hi ha a la cadena de blocs per identificar diverses entitats i persones i el flux de diners entre elles. Una alternativa que s'està implementant per abordar aquest problema és el protocol Zerocash [1], que fa servir proves de coneixement nul (*zero-knowledge proofs*). Ho veurem a la secció 4.

2. Firmes digitals

En el món físic verifiquem la identitat de les persones amb imatges: comparem la cara de la persona amb la foto a la targeta d'identitat o les corbes d'una firma amb les de l'original. En canvi, en el món digital tractem amb nombres. La firma no és més que un nombre que envia la persona, tradicionalment anomenada Alice, junt amb el missatge que vol firmar. El receptor, tradicionalment anomenat Bob, hauria de poder comprovar que aquest nombre només el pot haver generat l'Alice. En principi això no sembla possible perquè qualsevol persona, tradicionalment anomenada Òscar, d'«oponent», pot intentar endevinar nombres fins que n'hi surti un que passi el test que fa servir el Bob per comprovar que el missatge ve de l'Alice. En el món físic això correspon a una falsificació de la firma. Però, de fet, falsificar una firma digital és bastant més difícil que falsificar una firma física, perquè si el test del Bob és complicat, pot ser que l'Òscar hagi

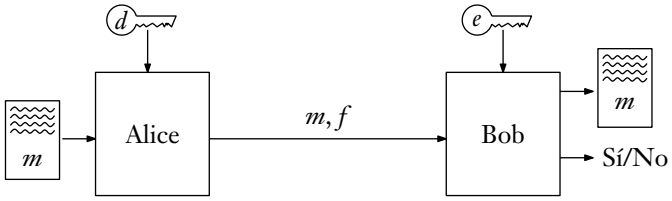

Figura 1. Protocol general de firmes digitals.

de provar essencialment tots els nombres de la mida adequada per trobar el correcte. Si el nombre té una longitud de 100 dígits, per exemple, i l'Òscar triga un nanosegon per comprovar que un nombre passa el test del Bob, en total l'Òscar trigarà $\frac{10^{100}}{10^9 \times 60 \times 60 \times 24 \times 356} > 10^{82}$ anys per provar tots els nombres (l'edat de l'univers és al voltant de 10^{10} anys).

Una possibilitat que no es pot descartar és que, un cop coneguem els detalls del test que fa servir el Bob, se'ns faci molt més fàcil trobar un nombre que passi el test. Ben mirat, actualment les matemàtiques no disposen de tècniques prou fortes per demostrar que això sigui impossible per a algun d'aquests tests (o almenys no hem trobat cap test pel qual puguem demostrar-ho). Per aquesta raó, la seguretat dels sistemes de firmes digitals encara es basa en conjectures sobre la dificultat de certs problemes computacionals. En aquest apartat en veurem dos exemples concrets.

Abans d'entrar en els detalls, és útil definir una mica de notació i introduir el concepte de *clau*. Hi ha dues claus que són dos nombres que fan servir l'Alice i el Bob durant el protocol. Una clau diem que és *privada*, perquè només la sap l'Alice, i la denotarem amb d. L'altra és *pública*, perquè la pot saber qualsevol, i la denotarem amb e. La clau privada es fa servir per crear el nombre que l'Alice envia junt amb el seu missatge m. D'aquest nombre se'n diu *firma* i el denotarem amb f. La clau pública la fa servir el Bob (o qualsevol altra persona) per comprovar que la firma és realment de l'Alice. Un esquema del protocol es pot veure a la figura 1.

2.1. Firmes i claus en Bitcoin

Per tenir bitcoins, l'usuari primer ha de crear una adreça (un nombre o una seqüència de caràcters, segons com vulgueu pensar-hi), en què es guardaran

les monedes. Seria millor dir «amb la qual s'associaran les monedes» perquè en realitat les monedes no tenen una realització física. Aquesta adreça, de fet, és la clau pública e del parell de claus (e, d) del protocol de firmes digitals. La clau privada d es guarda com un secret i es fa servir quan gastem els bitcoins per firmar transaccions. Cada transacció té una o més adreces d'origen i una o més adreces de destinació i s'ha de firmar amb les claus secretes de les adreces d'origen.

Per tant, si algú té la clau privada d'una adreça és com si tingués tots els bitcoins associats a aquesta adreça. Això és important perquè fa que els bitcoins siguin més com claus que com diners. Si un lladre fa una foto de la nostra clau és com si tingués la clau, però si fa una foto d'un bitllet que tenim és clar que no s'apropia d'aquest bitllet.

No fa gaire, uns periodistes de la cadena de televisió Bloomberg dels Estats Units van pagar literalment aquesta lliçó en bitcoins. Durant un reportatge sobre Bitcoin, van tenir l'ocurrència d'ensenyar l'imprès d'un regal de bitcoins que els havien donat. Amb la clau secreta al bell mig de la pantalla de la tele, en pocs segons aquests bitcoins van desaparèixer de la seva adreça.

2.2. Aritmètica modular

Una gran part de la resta d'aquest capítol tracta material que està inclòs en el currículum de l'assignatura d'Aritmètica del primer any del grau de Matemàtiques i de l'assignatura Matemàtica Discreta del grau d'Informàtica. El revisem igualment per a benefici dels nous estudiants.

En la criptografia es fan servir nombres molt grans, per exemple, de centenars de dígits. Els algoritmes demanen fer operacions amb aquests nombres, com ara, elevar un nombre de centenars de dígits a un altre d'una llargada comparable. El resultat és un nombre que no es podria guardar en un disc dur encara que estigués fet de tots els àtoms de l'univers. Llavors, com fem aquests càlculs?

La idea és que, en comptes de fer servir l'aritmètica clàssica dels nombres enters, fem servir aritmètica modular. És a dir, en lloc de pensar en nombres enters, pensem només en els residus que aquests nombres donen en dividir-los per un nombre especial que escollim, N. Per denotar

aquest tipus d'aritmètica fem servir $\equiv$ en comptes de $=$ i posem (mòd. N) al final de l'equació per denotar que les equivalències són *mòdul* N, és a dir, tots els nombres que donen el mateix residu de divisió per N els considerem equivalents. Per exemple, $8 \equiv 23$ (mòd. 5) perquè tots dos nombres donen el mateix residu de divisió per 5, concretament 3. De la mateixa manera, si fem càlculs mòdul 1.000 l'únic que ens interessa són els últims tres dígits dels nombres. La majoria d'operacions aritmètiques es tradueixen de manera natural a l'aritmètica modular. Per exemple, si a i b donen residus r i s en dividir-los per N, llavors la seva suma dóna residu $r+s$ (mòd. N). Per tant no cal modificar l'operació de suma. El mateix val per a la resta i la multiplicació.[2]

L'operació que no és tan fàcil de traspassar a l'aritmètica modular és la divisió. De fet, en l'aritmètica modular no parlem de divisió sinó d'inversos. En lloc d'escriure a/b escrivim $a \times b^{-1}$ i definim l'*invers* de b (denotat b^{-1}) com el nombre c tal que $b \times c \equiv 1$ (mòd. N), si aquest nombre existeix. Per molts casos de b i N, aquest c no existeix. Concretament, l'invers sí que existeix (i és únic) exactament quan b i N no tenen factors comuns no trivials. A més, aquest invers es pot trobar fent servir un algoritme anomenat *algoritme d'Euclides*, que és molt eficient fins i tot per a nombres grans.

Per l'operació d'exponenciació, resulta que tenim una regla que va descobrir Euler, que ens diu que per cada N hi ha un nombre anomenat $\phi(N)$, tal que si $a_1 \equiv a_2$ (mòd. N) i $b_1 \equiv b_2$ (mòd. $\phi(N)$) llavors $a_1^{b_1} \equiv a_2^{b_2}$ (mòd. N). Veurem la demostració del teorema d'Euler a la secció 2.3.2.

Malauradament, el teorema d'Euler no sembla que ens ajudi amb el problema d'elevar nombres grans a altres nombre grans perquè $\phi(N)$ també és un nombre força gran. Per calcular a^b (mòd. N) de la manera òbvia hauríem d'executar b multiplicacions, calculant el residu de divisió per N cada vegada per evitar que els resultats es facin llargs. El truc per fer aquest procés més ràpidament s'anomena *quadratura iterada* (*repeated*

2. Si aquesta és la teva primera introducció a l'aritmètica modular, és un bon exercici demostrar que si $a_1 \equiv a_2$ (mòd. N) i $b_1 \equiv b_2$ (mòd. N), llavors $a_1 - b_1 \equiv a_2 - b_2$ (mòd. N) i $a_1 \times b_1 \equiv a_2 \times b_2$ (mòd. N), i trobar un exemple pel qual no és cert que $a_1/b_1 \equiv a_2/b_2$ (mòd. N) (amb a_1 divisible per b_1 i a_2 divisible per b_2) i un altre pel qual no és cert que $a_1^{b_1} \equiv a_2^{b_2}$ (mòd. N).

squaring). En comptes de calcular $a^2, a^3, a^4, \ldots, a^b$ calculem només $a^2, a^4, a^8, a^{16}, \ldots, a^{2^{\lfloor \log_2 b \rfloor}}$. Cada membre d'aquesta sèrie és el quadrat de l'anterior. El nombre de multiplicacions per generar aquesta sèrie és, doncs, $t = \lfloor \log_2 b \rfloor$, que és comparable al nombre de dígits de b (de l'ordre dels centenars, que són poques multiplicacions per a un ordinador).

Donada aquesta sèrie, podem calcular a^b fent només t multiplicacions més. Per convèncer-vos d'aquest fet, penseu que qualsevol b es pot representar com la suma d'un subconjunt dels nombres $\{1, 2, 4, 8, 16, \ldots, 2^{\lfloor \log_2 b \rfloor}\}$ i, per tant, es pot representar com

$$b = \sum_{i=0}^{t} b_i \times 2^i,$$

per alguns $b_0, b_1, \ldots, b_t \in \{0, 1\}$ (la representació binària de b). Aquest és el mateix truc que fem servir quan paguem una quantitat d'euros amb pocs bitllets i monedes de diferents denominacions, en comptes de fer servir només monedes d'un cèntim.

Les t multiplicacions finals són:

$$a^b = a^{\sum_{i=0}^{t} b_i \times 2^i} = \prod_{i=0}^{t} a^{b_i \times 2^i} = \prod_{i \in \{0, \ldots, t\} : b_i = 1} a^{2^i}.$$

Ara que sabem fer càlculs amb nombres grans en aritmètica modular, ja podem explicar el primer algoritme de firmes digitals.

2.3. Firmes mitjançant RSA

Un dels primers sistemes de firmes digitals va ser el de Ronald Rivest, Adi Shamir i Len Adleman de 1978, conegut com RSA [7]. Per crear les claus, l'Alice escull dos nombres primers p i q de molts dígits (centenars), que són secrets, i en calcula el producte

$$N = p \times q,$$

que serà públic. Una de les conjectures en què es basa el mètode és que si només tenim N, trobar p i q (i.e. factoritzar N) és un problema computacionalment difícil.[3]

Llavors, l'Alice escull dos nombres d i e, el d serà secret, l'e, públic, tals que per qualsevol missatge m el valor de $(m^d)^e$ mòdul N (el qual es pot calcular amb $2\log(d) + 2\log(e)$ multiplicacions de nombres de la mateixa mida que N) és el mateix que m. És a dir,

$$(m^d)^e \equiv m \,(\text{mòd. } N).$$

A la secció 2.3.2 explicarem com l'Alice pot trobar dos nombres amb aquesta propietat.

Un cop l'Alice té un parell de nombres amb aquesta propietat, el protocol és el següent: fent servir la clau secreta d, l'Alice calcula la firma

$$f = m^d \,(\text{mòd. } N)$$

i l'envia junt amb el missatge m. El Bob agafa la firma i l'eleva a e mòdul N. Si el resultat és m, accepta la firma com autèntica; si no, la rebutja.

Si l'Òscar vol firmar un altre missatge m fent veure que és l'Alice, necessitarà la clau d. La seguretat del protocol RSA depèn de la conjectura que trobar d si només saps e i N és computacionalment difícil (es pot comprovar que si pots factoritzar N, llavors sí que pots trobar d).

2.3.1. RSA per xifrar i desxifrar

Abans de continuar amb els detalls tècnics de com escollim les claus pública i privada, cal mencionar que la mateixa idea es fa servir per xifrar i desxifrar missatges. El Bob pot xifrar els missatges que envia a l'Alice elevant-los a la clau pública e. Per tant, el Bob envia $\hat{m} = m^e \,(\text{mòd. } N)$. Només l'Alice pot desxifrar el missatge perquè té la seva clau privada d i pot calcular $\hat{m}^d \equiv \left((m^e)^d\right) \equiv m \,(\text{mòd. } N)$. D'aquí vénen i els noms de les claus e i d: d'*encriptació* i *desencriptació*.

3. Si un dia es construïssin ordinadors quàntics, amb aquests sí que podríem factoritzar nombres grans fent servir l'algoritme de Shor [9].