Registros distribuidos

Entendiendo Blockchain

Rodrigo Santamaría

Universidad de Salamanca

April 16, 2024

Índice

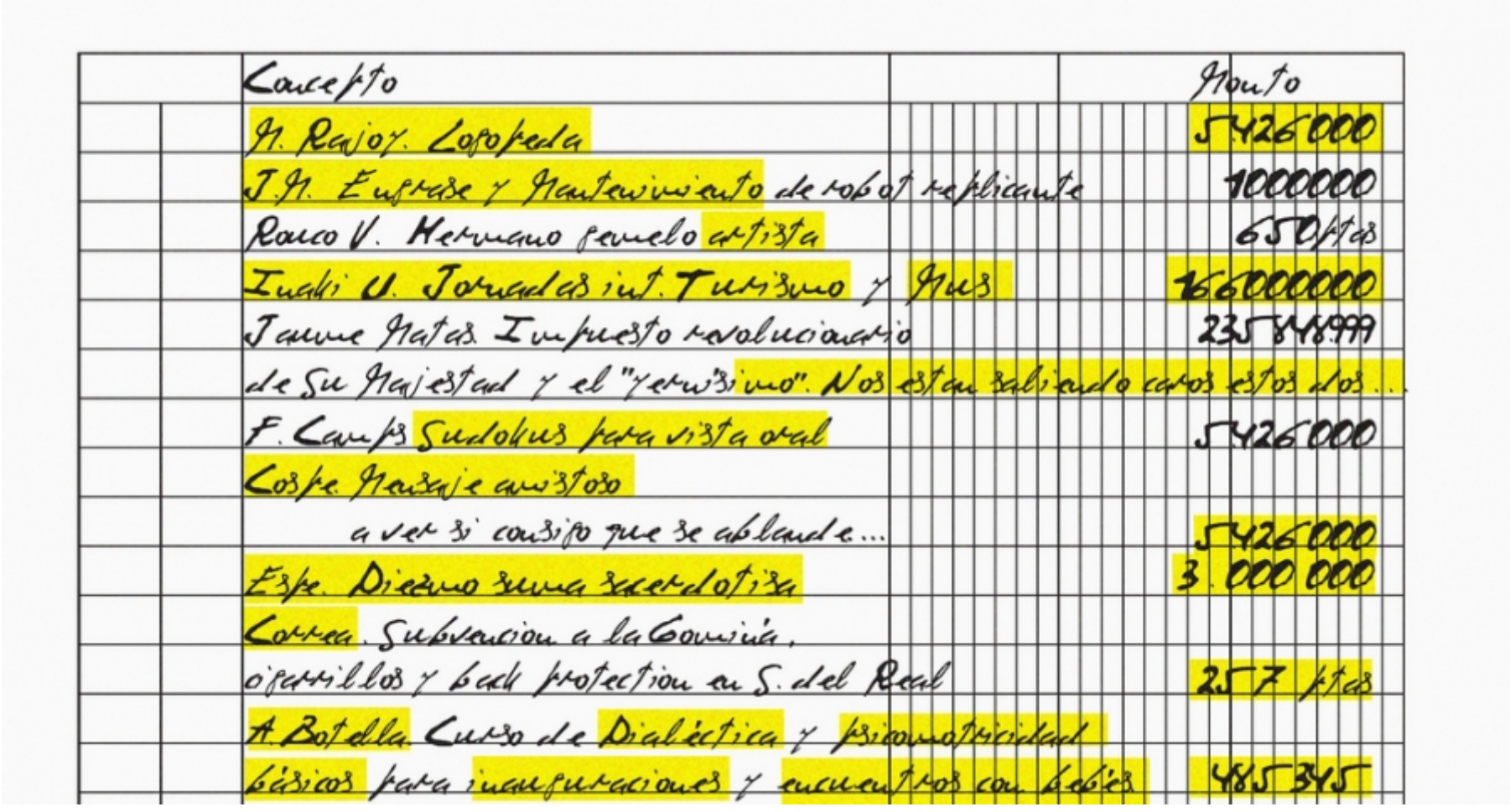
Registros distribuidos

- Un libro de registro distribuido (distributed ledger) es un tipo de replicación
  - Basada en un método de verificación (Hash) y/o un algoritmo de consenso para garantizar la consistencia de las versiones.
  - Basada en una red de pares (P2P) para distribuir transacciones y consolidar la versión del registro consensuada.
- Su aplicación más extendida es la realización de transacciones
  - Banca
  - Criptomoneda
  - Economía 'programable'
- Algunos ejemplos conocidos que implementan un registro distribuido son Bitcoin, Hyperledger o Ethereum.

Registro centralizado

- Un registro centralizado tiene los mismos problemas que otras soluciones centralizadas, ahora aplicado a cuestiones económicas
  - Servidores pesados con multitud de transacciones.
  - Riesgo de pérdida: cachés, servidores replicados, etc.
  - Falta de transparencia: autoridad certificadora.

Problemas del registro centralizado



Concepto	Monto
M. Rajoy. Coso pedal	5'426'000
J.M. Enfrase y Mantenimiento de robot replicante	1000000
Rosco V. Hermano pequeño artista	650 kas
Inaki U. Jornadas int. Turismo y Ocio	166000000
Jaime Matas. Impuesto revolucionario	235'848.999
de Su Majestad y el "yernísimo". Nos están saliendo caros estos dos...	
F. Camps Sudokus para vista oral	5'426'000
Cospe. Masaje amistoso	
a ver si consigo que se ablande...	5'426'000
Esp. Diezmo Santa Sacerdotisa	3.000.000
Correa. Subvención a la Comunia.	
difusibles y bach protection en S. del Real	257 kas
A. Botella. Curso de Dialéctica y psicomotricidad	
básicos para inauguraciones y encuentros con bebés	485'345

Índice

Función Hash

- Una función Hash $H(x) = y$ es una función que asigna a un mensaje de longitud variable x una clave de longitud fija y
- Para que sea útil en criptografía debe cumplir algunas propiedades[1]:
  - **Resistencia a las colisiones**: es impracticable[2] encontrar $x, y | H(x) = H(y)$
  - **Unidireccionalidad**: dado $H(x)$ es computacionalmente imposible obtener x.
  - **Amistad con puzles**: es imposible encontrar en $t << 2^n$ un $x | H(k||x) = y$[3] para todo y de n bits, tomando k al azar.

[1] La función hash más extendida que cumple estas propiedades es SHA-2

[2] No es imposible, pero computacionalmente tardaría, literalmente, octillones de años [Narayanan2015]

[3] $||$ significa 'concatenado a'

Función Hash: puzles

- Un puzle de búsqueda consiste en:
  - Una función hash H
  - Un valor id obtenido al azar.
  - Un conjunto de valores Y
  - La solución es un valor x tal que $H(id||x) \in Y$
- Si H es amistosa a los puzles, garantiza que no hay atajos para resolver el puzle: tenemos que probar con valores x elegidos por fuerza bruta.
- Así, con Y podemos marcar la dificultad del puzle. Si Y es todo el espacio de claves de H, el puzle es trivial. Si Y es un único valor, es lo más complicado posible[4].

[4] O computacionalmente imposible, si H es unidireccional y $|id| = 0$

Puntero Hash

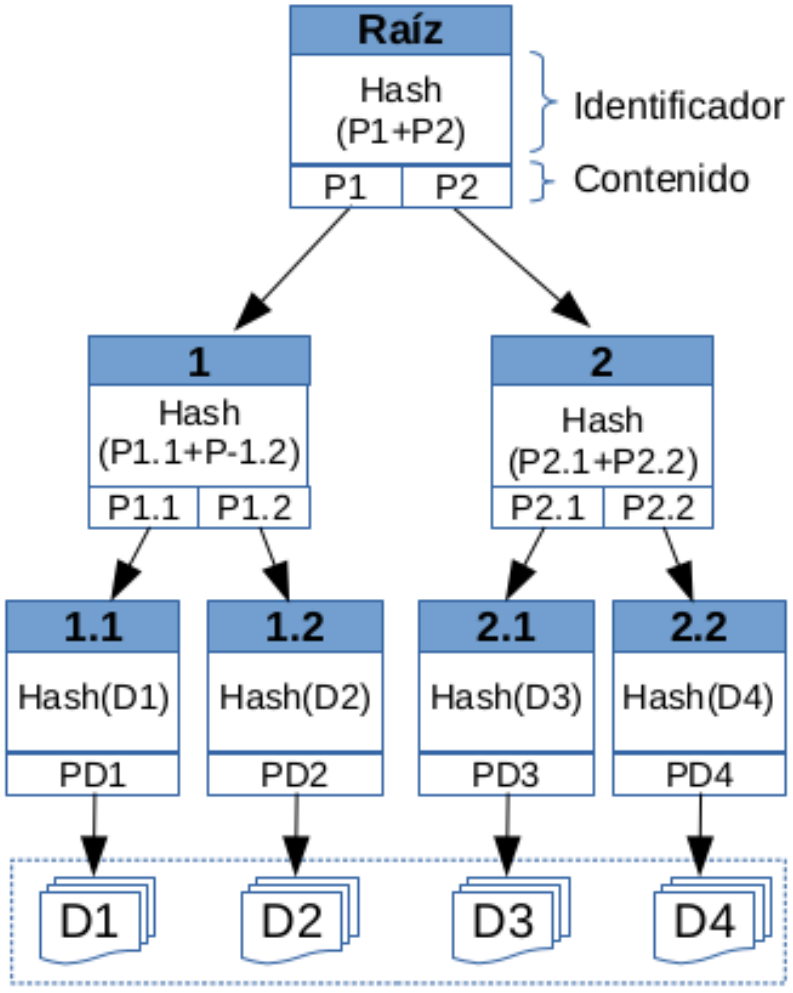
- Un puntero indica dónde se encuentra cierta información.
- Un puntero *hash* contiene además el hash de dicha información.
- Cualquier estructura de datos que use punteros puede usar punteros hash.
  - De esta manera, no sólo sabemos dónde está dicha estructura sino también si su contenido ha sido alterado, chequeando el puntero hash a la estructura.

Árbol Hash[5]

- Es un árbol *binario* en el que:
  - Cada nodo hoja contiene un puntero hash a un bloque de datos, usando dicho hash como su identificador.
  - Cada nodo padre contiene dos punteros hash a sus dos nodos hijo, y se identifica con el hash de la concatenación de los hash de sus hijos.
- Conociendo el puntero hash al nodo raíz podemos acceder a todos los bloques de datos y también saber si han sido alterados.

[5] Propuesto por Ralph Merkle en 1979.

Árbol Hash


Raíz
Hash
(P1+P2)
Identificador
P1
P2
Contenido
1
Hash
(P1.1+P-1.2)
P1.1
P1.2
2
Hash
(P2.1+P2.2)
P2.1
P2.2
1.1
Hash(D1)
PD1
1.2
Hash(D2)
PD2
2.1
Hash(D3)
PD3
2.2
Hash(D4)
PD4
D1
D2
D3
D4

Prueba de inclusión

- Dado un bloque de datos, es posible comprobar si está en un árbol hash chequeando sólo dicho bloque y los nodos internos correspondientes.
  - En un árbol de n nodos, sólo se ven involucrados $log(n)$ nodos
- Si los nodos hoja están ordenados siguiendo algún criterio, también es posible comprobar si un determinado bloque está en el árbol o no en tiempo logarítmico.

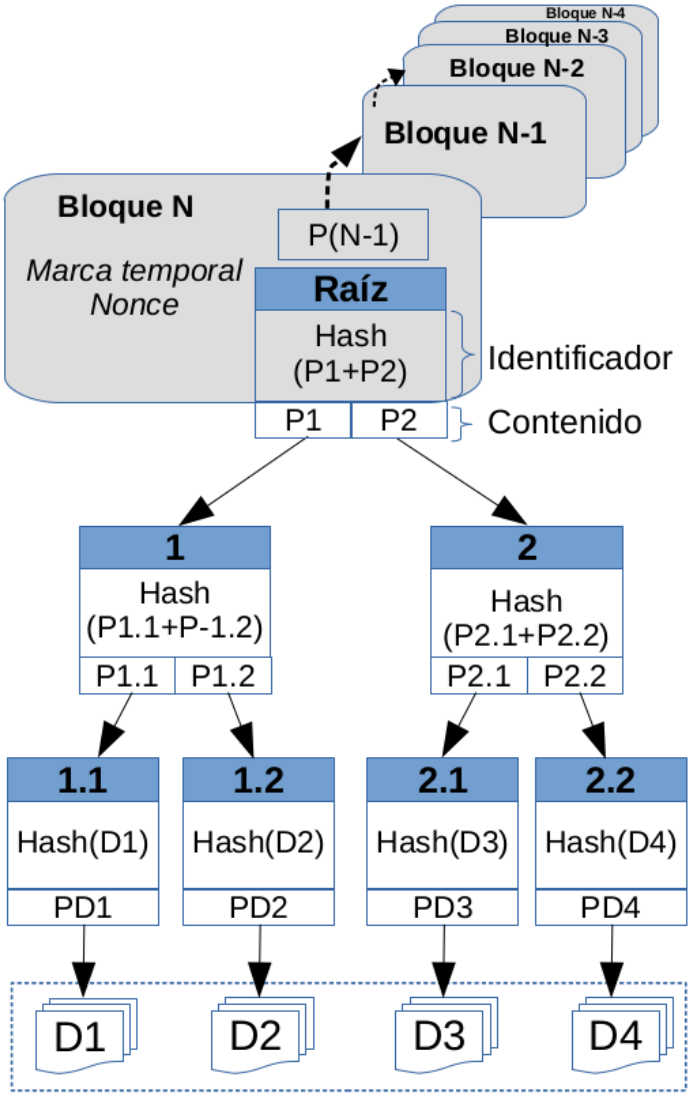
Índice

Blockchain

- Es una **cadena** de árboles hash formados por transacciones (**bloques**)[6].
- Cada bloque contiene:
  - El árbol hash del conjunto de transacciones.
  - Una cabecera con:
    - Nodo raíz de su árbol hash
    - Puntero hash al bloque anterior
    - Marca temporal y nonce (en bitcoin, ver 26)
- Blockchain es una estructura de datos aplicable a distintos protocolos o sistemas (de manera similar a DHT en P2P)

[6] Podrían estar formados por cualquier otro tipo de datos

Blockchain: esquema



Bitcoin

- Es una criptomoneda y sistema de pago descentralizado, donde las transacciones:
  - Se realizan entre usuarios, sin intermediarios.
  - Se almacenan en un registro distribuido mediante blockchain.
  - Se validan mediante el consenso distribuido de una red P2P.
- Desarrollado por Satoshi Nakamoto[7] en 2009, y distribuido como código abierto[8].
- Algunos datos (Wikipedia, enero 2024):
  - Entre 3 y 6 millones de usuarios
  - Unos 19.5M de bitcoins en circulación
  - Un bitcoin = 40.000$

[7] Un pseudónimo que corresponde a una persona o equipo desonocido, más información aquí
[8] Código original

Bitcoin: nodos

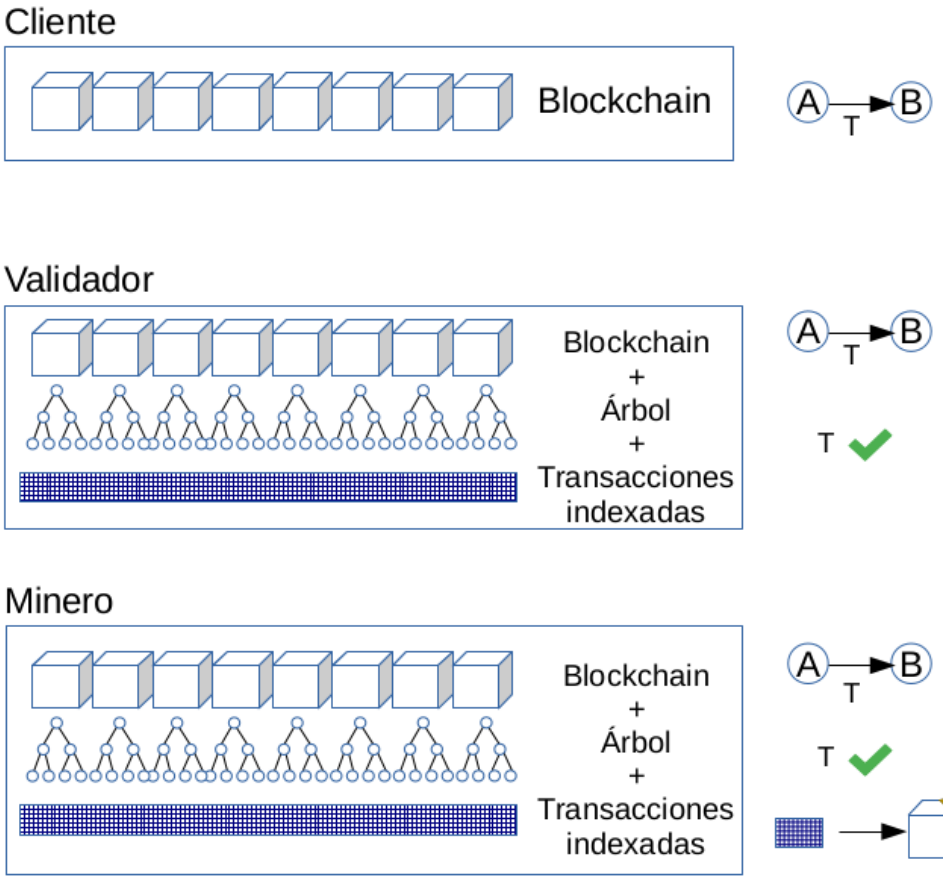
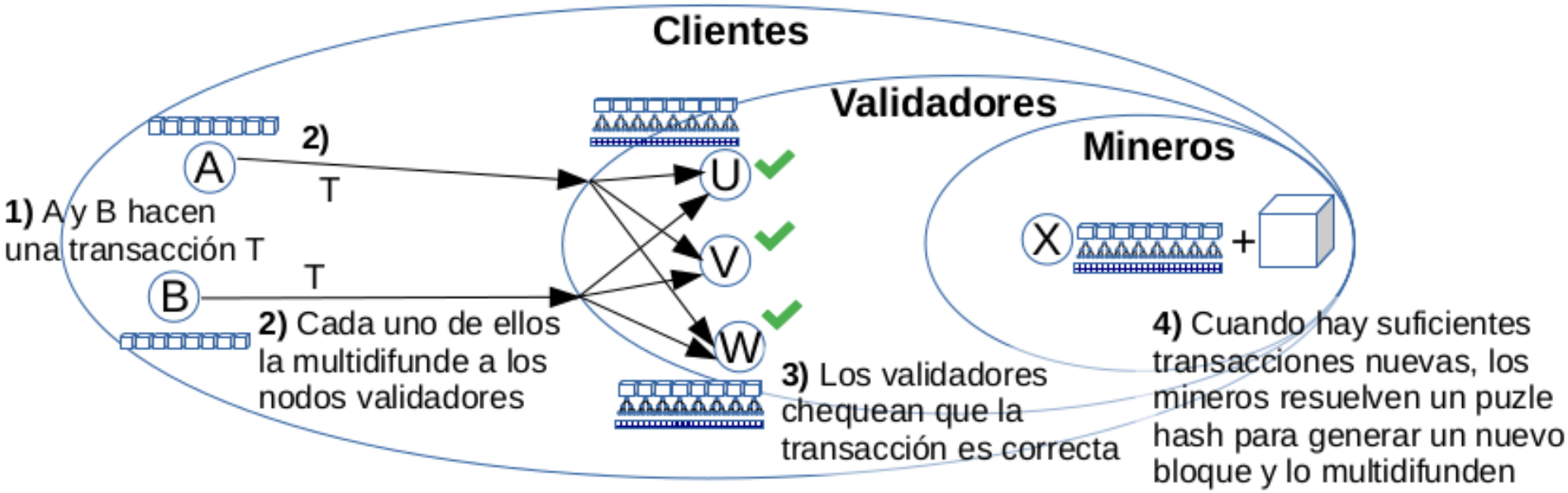


Figure 8.2: Nodos en Bitcoin. El nodo **cliente** necesita poca capacidad en disco o de cálculo, sólo realiza transacciones y confía en la blockchain que emiten y validan los otros nodos. Los nodos **validadores** además contienen todos los árboles hash y bloques de transacciones previas, pueden validar la blockchain y cualquier nueva transacción que reciban. Necesitan un poco más capacidades computacionales, sobre todo de memoria. Los nodos **mineros** necesitan gran capacidad de cálculo para resolver los puzles hash, y son los únicos que pueden proponer nuevo bloques a partir de nuevas transacciones válidas. En teoría, todos los nodos pueden tener los tres roles, pero en la práctica la dificultad de los puzles y la comodidad de no almacenar toda la blockchain estratifican el sistema.

Bitcoin: esquema



Validez y Consenso en Bitcoin

- Cada transacción se multidifunde a los nodos validadores
  - Comprueban la **validez** de la transacción (bien formada, no maliciosa, etc.)
  - Los nodos validadores o full nodes son nodos que implementan toda la funcionalidad de Bitcoin[9].
- Las transacciones llegan en distinto orden a cada nodo, por ello hace falta llegar a un **consenso** para decidir su orden dentro de los bloques:
  - Cada nodo construye los bloques en función de las transacciones que ha visto.
  - Cada peer contendrá una versión *distinta* del blockchain
  - Mediante un **consenso** distribuido, se obtiene la blockchain única.

[9] No hay incentivos para ello más allá de mejorar la seguridad de la red, así que normalmente sólo los nodos de minado son full nodes

Particularidades del consenso en Bitcoin

- Paxos o derivados aseguran consensos consistentes
  - Si hay menos de un tercio de 'traidores'[10].
  - Supone un sistema de 1 máquina = 1 voto.
  - Problemas en entornos sin autenticación y máquinas virtuales.
- Bitcoin relaja en la práctica ciertos requisitos:
  - No hay autoridad central que asigne identificadores[11]
  - No hay un tiempo fijado para alcanzar consensos.
  - El 'consenso' se basa en la comprobación por parte de toda la red de que el bloque propuesto contiene transacciones válidas.
  - Se juega con la probabilidad de que un nodo haya observado cierta transacción en sus bloques: estrategia *perezosa*

[10] Ver Problema de los Generales Bizantinos-tema 6

[11] Esto evita ataques de suplantación de identidad, pero genera otros nuevos.

Algoritmo de consenso de Bitcoin (simplificado)

Consideración: se puede elegir un nodo aleatoriamente sin caer en ataques de suplantación de identidad

1. Según ocurren, se multidifunden las transacciones a los nodos.
2. Cada nodo construye su nuevo bloque con las transacciones recibidas.
3. En cada ronda, un nodo *aleatorio* multidifunde su bloque.
4. Los nodos receptores aceptan el bloque si todas sus transacciones son válidas.
5. La aceptación del bloque se expresa mediante la inclusión de su puntero hash en su blockchain.

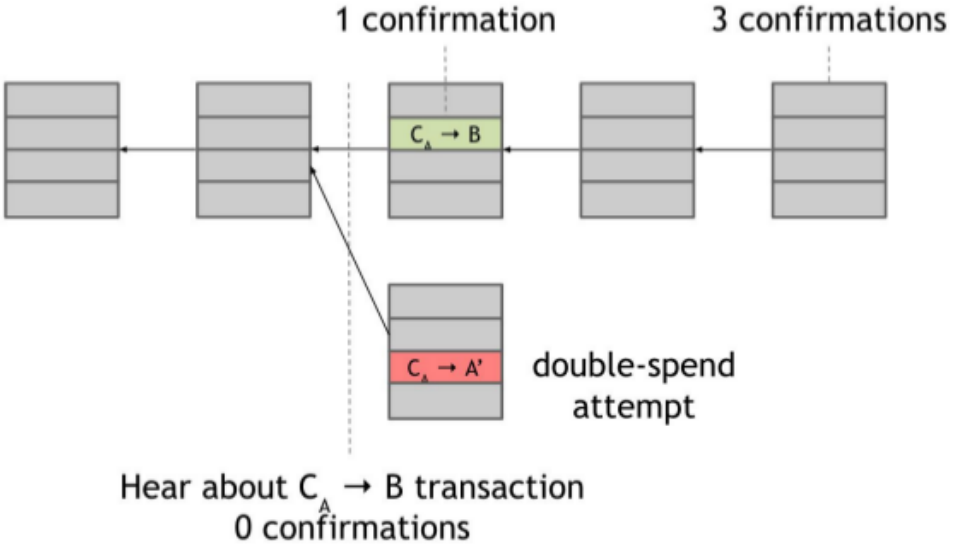
Se trata de un consenso *implícito*: no hay una votación como en Paxos. El consenso lo da la coherencia con los bloques existentes.

Ataque de doble gasto

- Un atacante A con capacidad de generación de bloques crea una transacción *válida* (AA'), por el mismo importe que una transacción *legítima* (AB).
  - B observa que la transacción está en el siguiente bloque y realiza el envío del producto.
  - A busca que prevalezca una nueva cadena de bloques donde donde ha sustituido el bloque con AB por uno con AA'
  - Desde el punto de vista de la estructura de datos, ambos bloques son igualmente válidos.
- El algoritmo de consenso implícito premia los primeros bloques en llegar (en principio, AB) y los que formen blockchains más largos, aunque esto no es una regla estricta.

Ataque de doble gasto (ii)

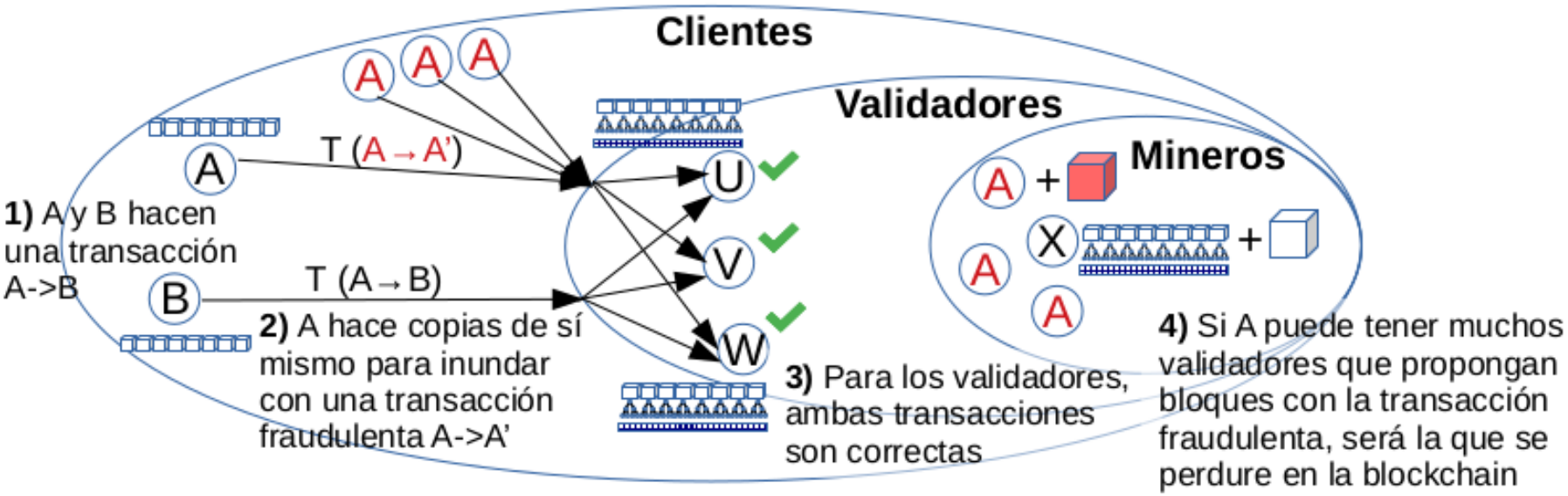
- B debería esperar k (> 6) confirmaciones de que se ha producido AB antes de enviar el producto a A.
  - En ese caso AB seguramente prevalezca en el sistema al haber formado ya una cadena más larga.
  - No hay nada especial en el número 6, es simplemente un buen compromiso entre el tiempo de espera y la garantía de que la transacción prevalecerá.



(Narayanan et al. 2015)

Ataque Sybil

- Nada nos asegura *a priori* que AB vaya a ser la transacción que prevalezca.
- Aún más, si A empieza a generar 'copias' de sí mismo que multidifundan la transacción fraudulenta AA', es muy probable que se imponga.
  - Esto se conoce como ataque Sybil y es un ataque clásico en redes donde los nodos son anónimos.

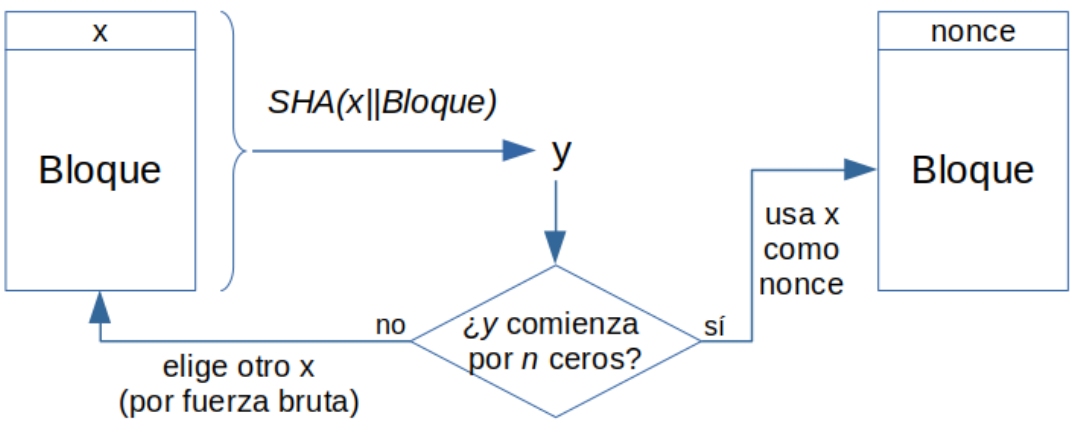


Proof-of-work

- Como vemos nada nos asegura *a priori* que AB vaya a ser la transacción que prevalezca.
- Podría prevalecer AA', en cuyo caso como no hemos obtenido k confirmaciones de que ha ocurrido AB, no enviaríamos nuestro producto.
- Solución: hacer *moderadamente* difícil crear nuevos bloques.
- Otra posible solución:
  - Añadir identificación de nodos para hacer las transacciones (*permissioned* blockchains).
  - En este caso, la coherencia entre bloques se obtiene mediante un algoritmo de tolerancia a fallos bizantinos (p. ej. PBFT en hyperledger)

Proof-of-work: puzles hash y nonces

- **Nonce**: número a proveer por el nodo que propone un bloque:
  - Se concatena el nonce al contenido del bloque, y se calcula el hash de dicha concatenación, que debe estar dentro del espacio de claves hash válidas propuestas[12].
  - Buscar el nonce es equivalente a resolver un puzle hash.
- El nonce es prueba del trabajo invertido en generar el bloque.
  - Penalizando la inyección de transacciones maliciosas tipo AA'.



[12] En el caso de Bitcoin, se proponen como claves válidas aquellas que comiencen por un determinado número de ceros. El número de ceros se puede variar para modular la dificultad del puzle (es decir, variamos $|Y|$).

Minería de bitcoins

- Bitcoin mining: cálculo del nonce de un bloque
  - Premiado con nuevos bitcoins[13]
- En 2024, hacen falta máquinas que puedan calcular unos 550 exahashes ($\sim 10^{18}$ hashes) por segundo, para ser capaces de resolver el puzle en unos 10 minutos.
  - Muy por encima de las capacidades computacionales de un ordenador personal.
  - La minería de bitcoins se centraliza en la práctica en pocos nodos con alto poder de computación.
  - Las posibilidades de que el atacante A 'coloque' en el sistema una transacción fraudulenta AA' pasa a depender de su capacidad de minado.

[13] Bitcoin plantea ir premiando con nuevos bitcoins a los nodos que generen nuevos bloques, hasta llegar a 21M de bitcoins. En teoría se premia sólo la creación del bloque, pero debido al problema de seguridad de doble gasto, en efecto se está pagando el cálculo del nonce. El proceso se alarga artificialmente dividiendo a la mitad la recompensa según aumenta el número total de bitcoins en circulación.

La espiral de la minería

- $|Y|$ (diap. 8) marca la dificultad del puzle.
- Para mantener la tasa de creación de bloques constante (1 bloque cada 10 min.) $|Y|$ se reajusta cada dos semanas.
  - Tener una tasa constante de creación de bloques baja optimiza el sistema al acumular muchas transacciones por bloque.
  - Además penaliza la creación de bloques fraudulentos.
  - En un contexto de incremento de nodos y de capacidades hardware, implica puzles cada vez más complejos.

Minería: resumiendo

- Para evitar ataques de gasto doble y para optimizar el número de transacciones por bloque, se incrementa el coste de creación (minería) de bloques.
  - Se evita el fraude de los nodos pequeños, pero ¿los grandes?[14]
  - Se incrementa el coste energético[15] y computacional.
  - Incentivos a la minería: 6.25 bitcoins por bloque creado[16]

[14] Los cinco grandes centros de minado en 2021 (F2Pool, BTC.com, Poolin, ViaBTC y AntPool) generan más del 60% de los bloques. Si tres se ponen de acuerdo pueden comenzar a agregar bloques fraudulentos con éxito.

[15] En 2021, se estima que equivale al consumo energético de Suecia o de Argentina.

[16] Actualmente. La recompensa se reduce a la mitad cada 210.000 bloques.

Proof-of-Stake y Proof-of-Authority

- **Proof-of-Stake**: sistema de 'lotería' o inversiones
  - Cada nodo "compra" su probabilidad de ser elegido.
  - Aleatoriamente, pero con probabilidades proporcionales a las inversiones, se elige al siguiente nodo validador de bloque[17].
  - La validación del bloque no tiene coste computacional extra.
  - Si el validador hace trampas, pierde parte de su inversión.
- **Proof-of-Authority**: sistema de 'reputación'
  - Sólo los nodos con una buena reputación son elegidos como validadores de bloques.
  - La validación se lleva a cabo mediante un algoritmo de consenso clásico tipo PBFT.
  - La reputación se estima mediante una autoridad externa[18].
- Pueden existir combinaciones de los tres tipos de prueba

[17] Es fácil ver cómo llegaría a un problema de re-centralización similar al de PoW

[18] Volvemos al mismo problema de confianza en una autoridad central.

Proofs: Comparativa

Proof	**of-work**	**of-stake**	**of-authority**
Coste computacional	Alto	Bajo	Bajo
Centralización	Eventual (económica)	Eventual (económica)	Sí (política)
Ejemplos	Bitcoin	Cardano, Ethereum*	Hyperledger

*Etherum era PoW, adoptó PoS en 2022

Índice

Componentes de un sistema de registros distribuidos

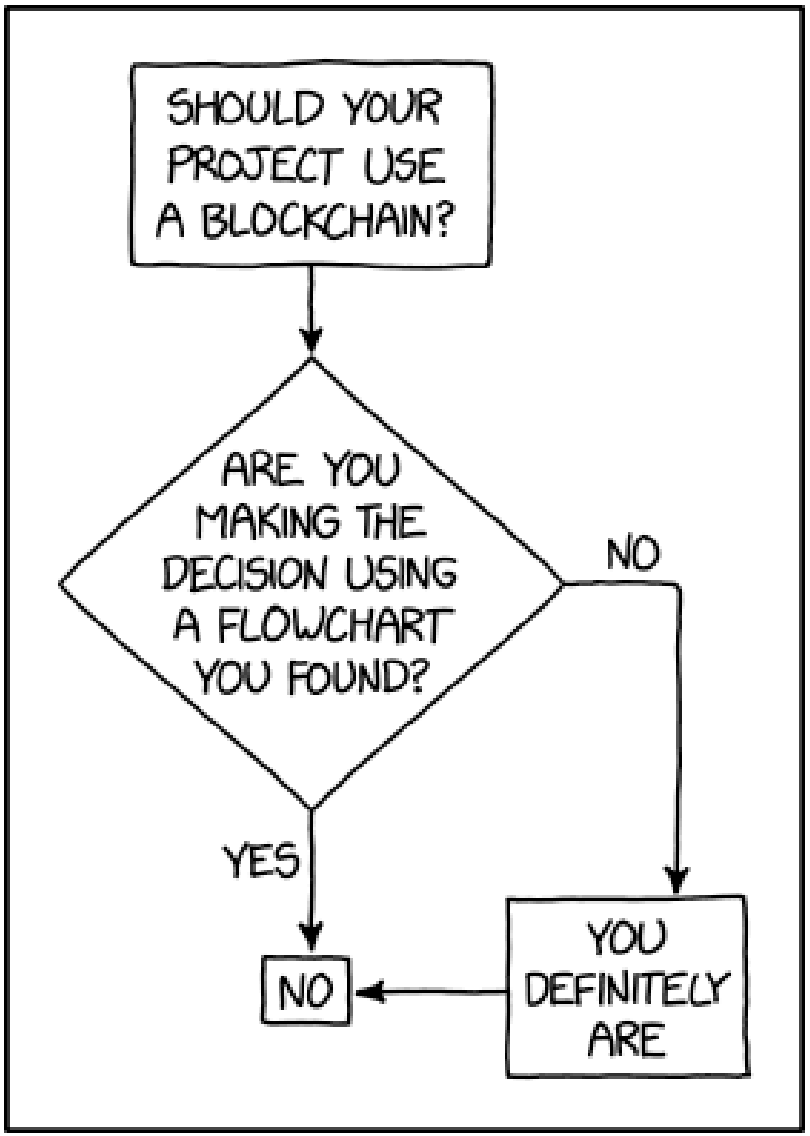
- **Ledger**: libro de registros implementado por blockchain. Mantiene la coherencia hacia atrás mediante encriptación hash
- **Algoritmo de consenso**: decide sobre el siguiente bloque mediante un sistema simple de criptografía y longitud de blockchains, o mediante un algoritmo tolerante a fallos bizantinos (BFT).
- **Smart contract**: definición de las reglas válidas para las transacciones.
- **World state**: base de datos para facilitar la búsqueda en el libro de registros (opcional).
- **Autenticación**: sistema de identificación para evitar el proof-of-work (opcional). Como contrapartida, implica un sistema centralizado de autenticación y la confianza en la autoridad certificadora.

Posibles aplicaciones de blockchain

- Las criptomonedas no son la única aplicación para blockchain.
- La infraestructura es extensible a cualquier sistema que implique:
  - Distintos usuarios en lo que no necesariamente se confía.
  - Un sistema de transacciones consensuado entre ellos.
  - La falta o desconfianza en una autoridad central supervisora.
- Algunos ejemplos[19]:
  - Trazabilidad de comida (un ejemplo en Hyperledger).
  - Registro de matrimonios o identidades (Bitnation, Onename)
  - Servicios de notaría descentralizada (BlockVerify)
  - Almacenamiento distribuido (Filecoin, Storj)
  - Votaciones transparentes y seguras (FollowMyVote)

[19] Más ejemplos en: 21 Blockchain Examples and Applications. Samrat Roy Chowdhuri, 2017.

Posibles aplicaciones: abuso



Fuente: xkcd

No nos dejemos llevar por las modas. Blockchain tiene unos usos muy específicos (entornos de baja confianza/posible fraude).
Si no crees que necesitaras un notario o un árbitro para tu proyecto si fuera 'analógico', probablemente no necesites blockchain.

Sistemas distribuidos y teoría de juegos

- En SSDD, existen límites estrictos sobre cuántos nodos deshonestos admite un algoritmo de consenso.
- En muchos SSDD recientes se empiezan a adoptar límites más flexibles basados en teoría de juegos:
  - No hay comportamientos honestos/deshonestos
  - Hay distintas estrategias con incentivos/penalizaciones.
  - Tit-for-tat en P2P.
  - Minería en Bitcoin.
- En definitiva, los problemas en SSDD muchas veces son problemas de **confianza**[20].

[20] Para reflexionar sobre la confianza mediante la teoría de juegos se recomienda jugar "The Evolution of Trust"

Centralización vs descentralización

- Como en muchos sistemas estudiados, Bitcoin no es un sistema puramente centralizado o descentralizado.
- La red P2P para gestionar blockchains es fundamentalmente descentralizada.
- La minería de bloques, aunque técnicamente descentralizada, requiere gran cantidad de recursos computacionales, haciéndolo en la práctica muy centralizado.
- Otras soluciones que evitan la minería requieren de sistemas de autentificación que también son centralizados (permissioned blockchains).

Tecnología, economía, ecología

- La tecnología de blockchain y bitcoin tienen repercusiones importantes en otros aspectos de la sociedad
- Según algunos autores, Bitcoin puede ser una burbuja ya que su crecimiento en valor no se corresponde con su uso real[21].
  - De hecho en 2019 parece a punto de explotar.
- La tecnología no es neutral y menos en el ámbito económico-político.
  - Bitcoin favorece a especuladores y entidades con gran capacidad computacional.
  - Parte del plan B de Grecia durante la crisis económica implicaba un sistema de pagos basado en blockchain y P2P.
- El minado de Bitcoin consume mucha electricidad.[22]

[21] Yanis Varoufakis: 'Bitcoin is the perfect bubble, but blockchain is a remarkable solution', Wired, 2017

[22] Entre 100MW y 3.4GW, dependiendo de las estimaciones. No obstante el sistema tradicional también tiene sus costes. Más info: How much energy does bitcoin mining really use? It's complicated. Wired, 2017.

Resumen

- **Bitcoin** es un sistema de transacciones ditribuido.
- Confía en **Blockchain** como modelo de datos que empaqueta transacciones mediante un sistema de **punteros hash**.
- Este sistema de **encriptación** garantiza la fiabilidad de las transacciones en casi todos los casos.
- Confía en una **red P2P** y un algoritmo de **consenso distribuido** para garantizar la persistencia y corrección de las transacciones.
- Para optimización y prevención de algunos ataques, implementa un sistema de incentivos y penalizaciones (**proof-of-work**) basado en teoría de juegos y en las cualidades propias de un sistema de transacciones.

Bibliografía

- The great chain of being sure about things, *The Economist*. **2015**.
- S. Nakamoto (publicado por Leo Trottier). Código original de Bitcoin. *GitHub*. **2009**.
- *A. Narayanan et al. Bitcoin and Criptocurrency Technologies. *Univ. de Princeton*. **2015**
- L. S. Who is Shatoshi Nakamoto? *The Economist*, **2015**.
- T. Upchurch. Yanis Varoufakis: 'Bitcoin is the perfect bubble, but blockchain is a remarkable solution'. *Wired*. **2017**.
- Entradas de Bitcoin, Blockchain, Merkle tree, Sybil attack. *Wikipedia en inglés*. **2017** (última consulta)