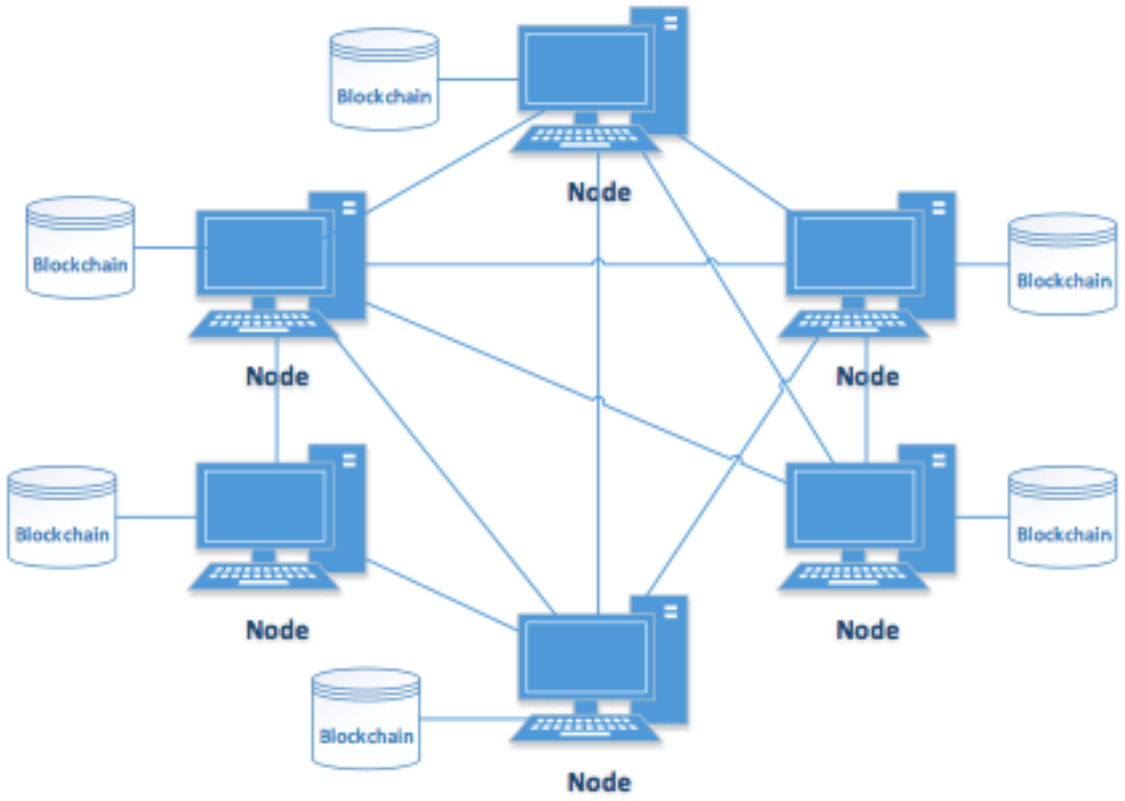
Introduzione alle Blockchain

P. Rullo
Corso di Basi di Dati
Corso di Laurea in Informatica
Unical
a.a. 2022-23

Blockchain e criptovalute

- La Blockchain (BC) nasce come la tecnologia sottostante il Bitcoin, come *registro* (libro mastro - *ledger*) su cui sono memorizzate tutti i pagamenti e le transazioni finanziarie
- Il Bitcoin è una criptovaluta: valuta ‘nascosta', nel senso che è visibile/utilizzabile solo conoscendo le ‘chiavi di accesso' pubblica e privata
- La criptovaluta non esiste in forma fisica (anche per questo viene definita ‘virtuale'), ma si genera e si scambia esclusivamente per via telematica. Non è pertanto possibile trovare in circolazione dei bitcoin in formato cartaceo o metallico

Blockcahin e criptovalute



- La Blockchain supporta transazioni *peer-to-peer*, eliminando la necessità di un intermediario di fiducia che verifica le operazioni - un ruolo che è necessario quando i partecipanti non si conoscono o non si fidano
- Quindi non c'è nessuna banca o nessun soggetto terzo che verifica le transazioni
- È invece la rete, nel suo complesso, che verifica le transazioni attraverso un "meccanismo di consenso" decentralizzato

Definizioni preliminari
Funzione hash crittografica

- Una funzione hash associa ad una stringa di lunghezza m arbitraria, detta *messaggio*, una stringa di lunghezza *n* fissa, detta *digest*
- **Esempio**. Usando il calcolatore hash (SHA-256)

https://tools.keycdn.com/sha256-online-generator

il messaggio “Una funzione hash associa ad una stringa di lunghezza m arbitraria, una stringa di lunghezza n fissa, detta digest.” viene convertito nel seguente digest di 64 caratteri esadecimali (256 bit):

32a152b9923a6c9f3fa034687f785dbfd6a19929e71932d9551497f4e0be74c3

Definizioni preliminari
Funzione hash crittografica

- La funzione non è iniettiva – più messaggi associati allo stesso digest
- Messaggi di p bit e digest di q bit, con p > q
  - M = 2^p possibili messaggi
  - D = 2^q possibili digest
  - MPD = M/D = 2^{p-q} messaggi per digest - assumendo una distribuzione uniforme
- Ad esempio, se consideriamo messaggi di 1000 caratteri (piccoli), quindi p=8.000, e digest di 256, quindi q=256
  - M =$2^{8.000}$
  - D = 2^{256}
  - MPD = $2^{7.746}$

Definizioni preliminari
Funzione hash crittografica

- La probabilità che due messaggi collidano (cioè, siano sinonimi) è

$$s \cong 1/2^q = 1/2^{256}$$

- Quello delle collisioni non è un fenomeno desiderato, in quanto il digest si usa come una impronta digitale, quindi univoca, di ogni dato documento

Definizioni preliminari
Funzione hash crittografica

Proprietà

- Deve essere facile calcolare il digest D di un messaggio M
- Dato il digest D, deve essere computazionalmente difficile trovare un messaggio M tale che D=hash(M) -- *unidirezionalità* o *non invertibilità* delle funzioni di hash – unico metodo forza bruta
- Deve essere computazionalmente difficile trovare due messaggi M e M’ che abbiano lo stesso digest D, cioè, D = hash(M) = hash(M’) – unico metodo forza bruta

Definizioni preliminari
Funzione hash crittografica

- Per la stringa S
  - “Una funzione hash associa ad una stringa di lunghezza m arbitraria, il messaggio, una stringa di lunghezza n fissa, il digest.”
- digest(S) = 32a152b9923a6c9f3fa034687f785dbfd6a19929e71932d9551497f4e0be74c3
- Ci sono algoritmi efficienti per generare il digest
- Risalire dalla stringa digest(S) alla stringa S è invece praticamente impossibile – l’unico metodo noto è procedere per tentativi

Definizioni preliminari
Funzione hash crittografica

- **Effetto valanga**: Si consideri la seguente stringa
  - S1: “Una funzione hash associa ad una stringa di lunghezza m arbitraria, il messaggio, una stringa di lunghezza n fissa, il digest.”
- digest(S1) = 32a152b9923a6c9f3fa034687f785dbfd6a19929e71932d9551497f4e0be74c3
- Modifichiamo S1 eliminando solo il simbolo finale “.”
  - S2: “Una funzione hash associa ad una stringa di lunghezza m arbitraria, il messaggio, una stringa di lunghezza n fissa, il digest”
- digest(S2) = 2136c12eeab60010f45084a82e98fff3d6b49d6c46c4d3da3e94ccf92c3043ff
- I due digest sono completamente diversi!!

Definizioni preliminari
Funzione hash crittografica

Un algoritmo che trasforma un messaggio *M* di lunghezza L arbitraria in un digest di *n* bit, con *n* < *L*

- Suddividi *M* in $k = \lceil L/n \rceil$ blocchi di *n* bit

$$M = [M_1, \ldots, M_i, \ldots, M_k]$$

- dove

$$M_i = [b_{i1} \ \ldots \ b_{ij} \ \ldots b_{in}]$$

- con l'ultimo blocco M_k eventualmente riempito con zeri, in modo da avere *n* bit

Definizioni preliminari
Funzione hash crittografica

- Si consideri la matrice di bit

$$\begin{bmatrix} b_{11} \dots b_{1j} \dots b_{1n} \\ \dots \\ b_{i1} \dots b_{ij} \dots b_{in} \\ \dots \\ b_{k1} \dots b_{kj} \dots b_{kn} \end{bmatrix}$$

- Sommando per colonna modulo 2 si ottiene il vettore riga

$$digest(M) = [c_1, \dots c_j, \dots, c_n]$$

Definizioni preliminari
Funzione hash crittografica

- **Esempio**: generare un digest di n=5 bit per la stringa binaria M= 1001100010110111 di L=16 bit

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

- Sommando per colonna modulo 2 si ottiene il vettore riga

$$\text{digest}(\text{M}) = [1\ 1\ 0\ 1\ 0]$$

La Blockchain – struttura dati

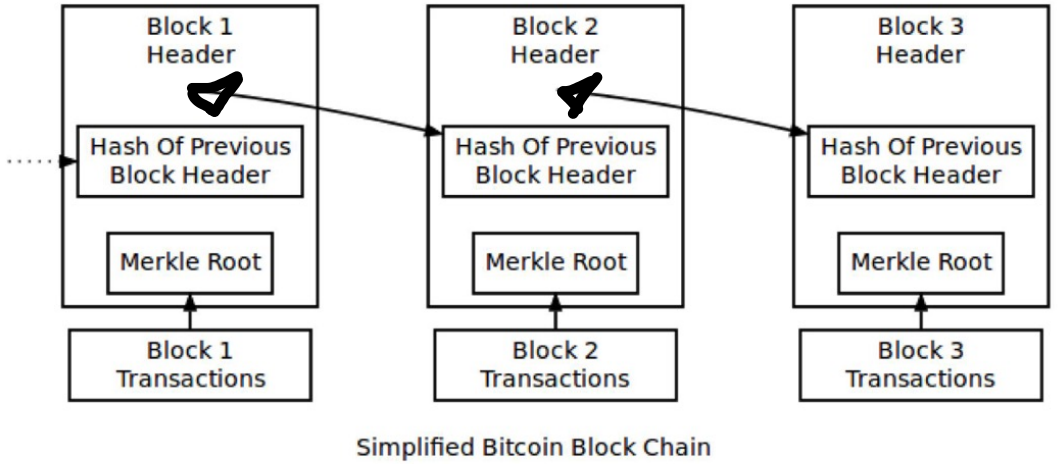
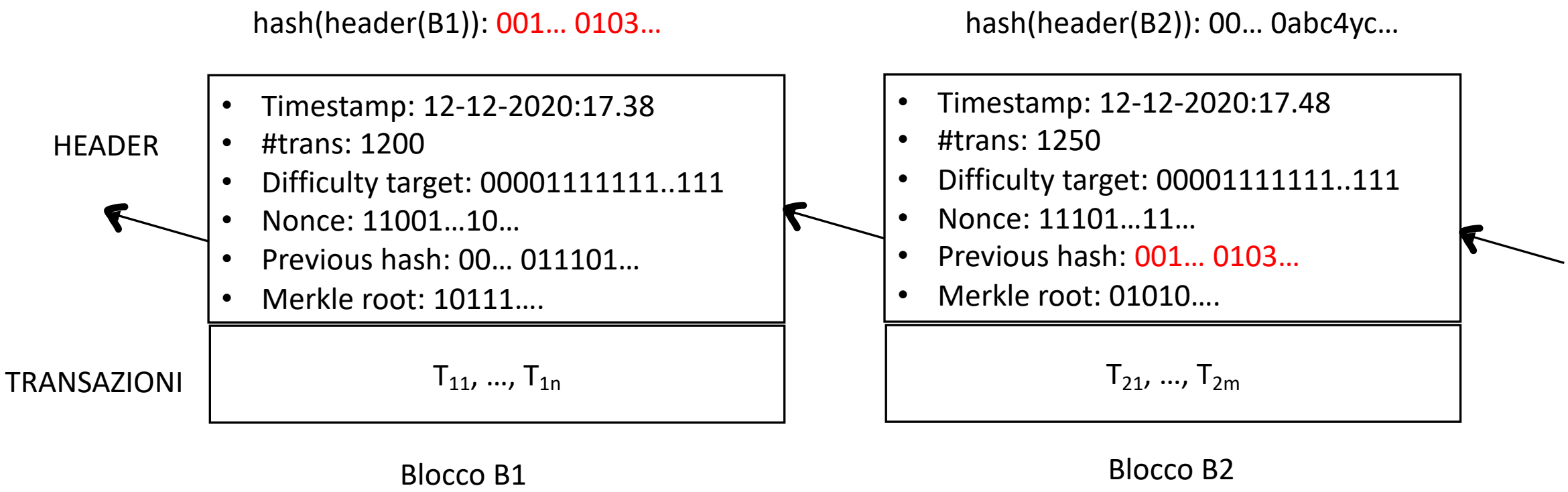


Fig. 6. Schematizzazione semplificata della Blockchain dei Bitcoin.

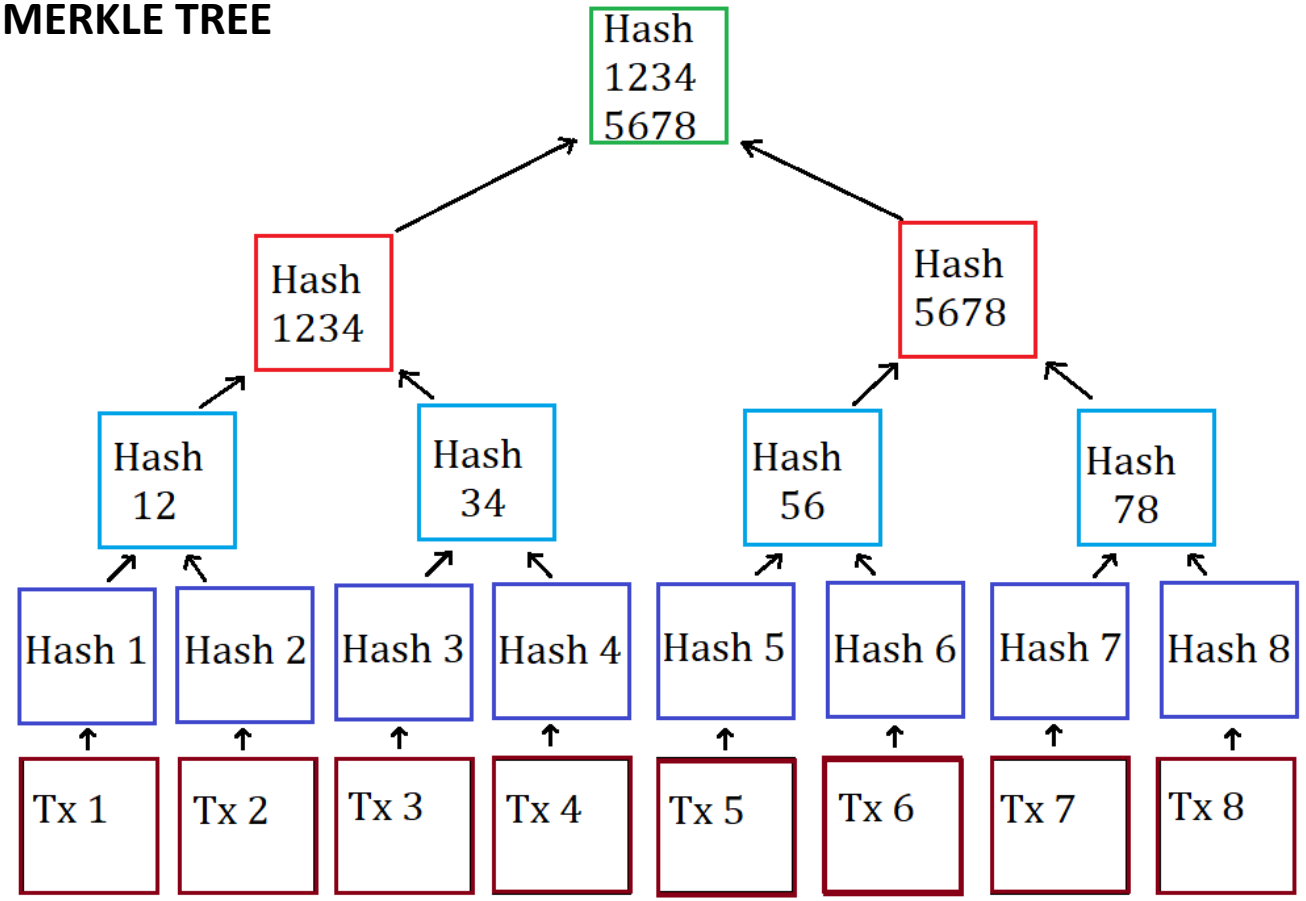
- E’ una catena di blocchi
- Ogni blocco è formato da
  - un header
  - un blocco di transazioni
- Header
  - Data e ora (timestamp)
  - Nr. di transazioni contenute nel blocco
  - Difficulty target
  - Nonce
  - Hash (digest) del block header precedente
  - Merkle root del blocco delle transazioni

La Blockchain – struttura dati



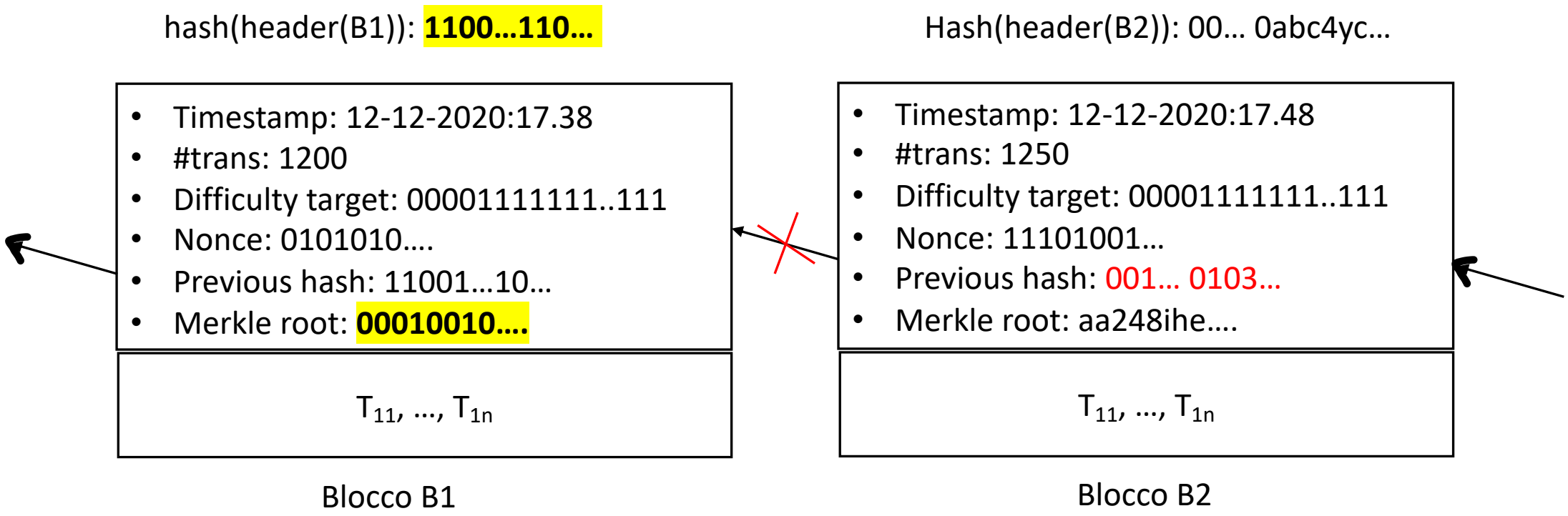
La Blockchain – struttura dati

MERKLE TREE



- Tx_1, …, Tx_n sono stringhe che rappresentano le transazioni del blocco
- $h_1 = hash(Tx_1)$, …, $h_n = hash(Tx_n)$ rappresentano le foglie del MT
- $h_{1,2} = hash(h_1,h_2)$, …, $h_{n-1,n} = hash(h_{n-1}, h_n)$ rappresentano i nodi padri dei nodi foglia
- … e così via fino alla radice

L’integrità della catena



Se viene modificata qualche transazione in B1

- Cambia il Markle root
- Cambia l’header
- Cambia hash(header)

- Il valore del Previous hash in B2 non è corretto
- Si rompe la catena!

L’integrità della catena

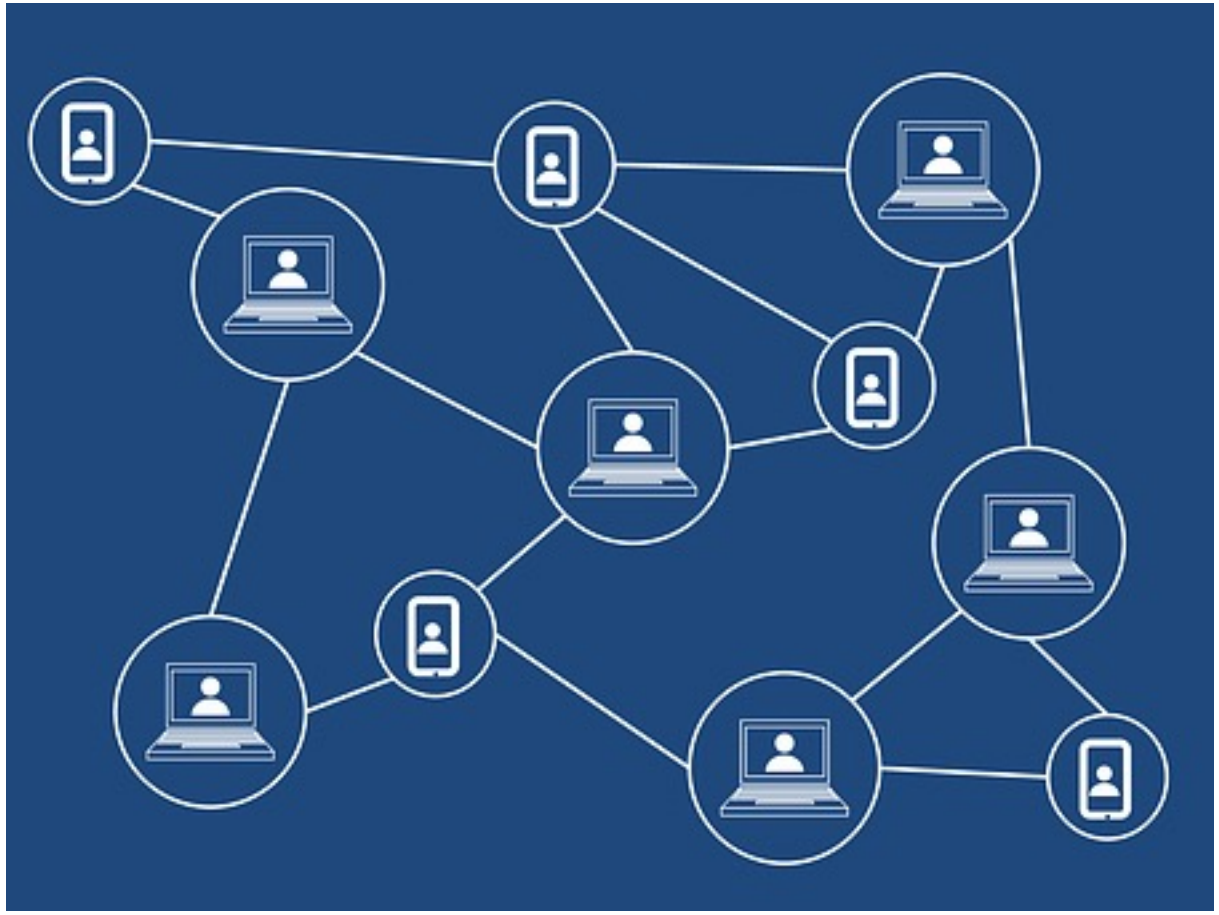
- Rimedio: al fine di ripristinare il collegamento tra B1 e B2
  - aggiorno il valore del *previous hash* di B2 al nuovo valore hash(header(B1))= 1100…110…
  - Modificando il *previous hash* di B2, cambia l’header di B2 e, quindi, il suo hash
  - si rompe il link tra B2 e il blocco successivo B3
  - il problema si reitera per tutti i blocchi a seguire!!

L’integrità della catena

- Verificare se due BC sono uguali è facile: basta verificare se gli hash header dell’ultimo blocco sono uguali
- L’hashing rende ogni manomissione della BC evidente

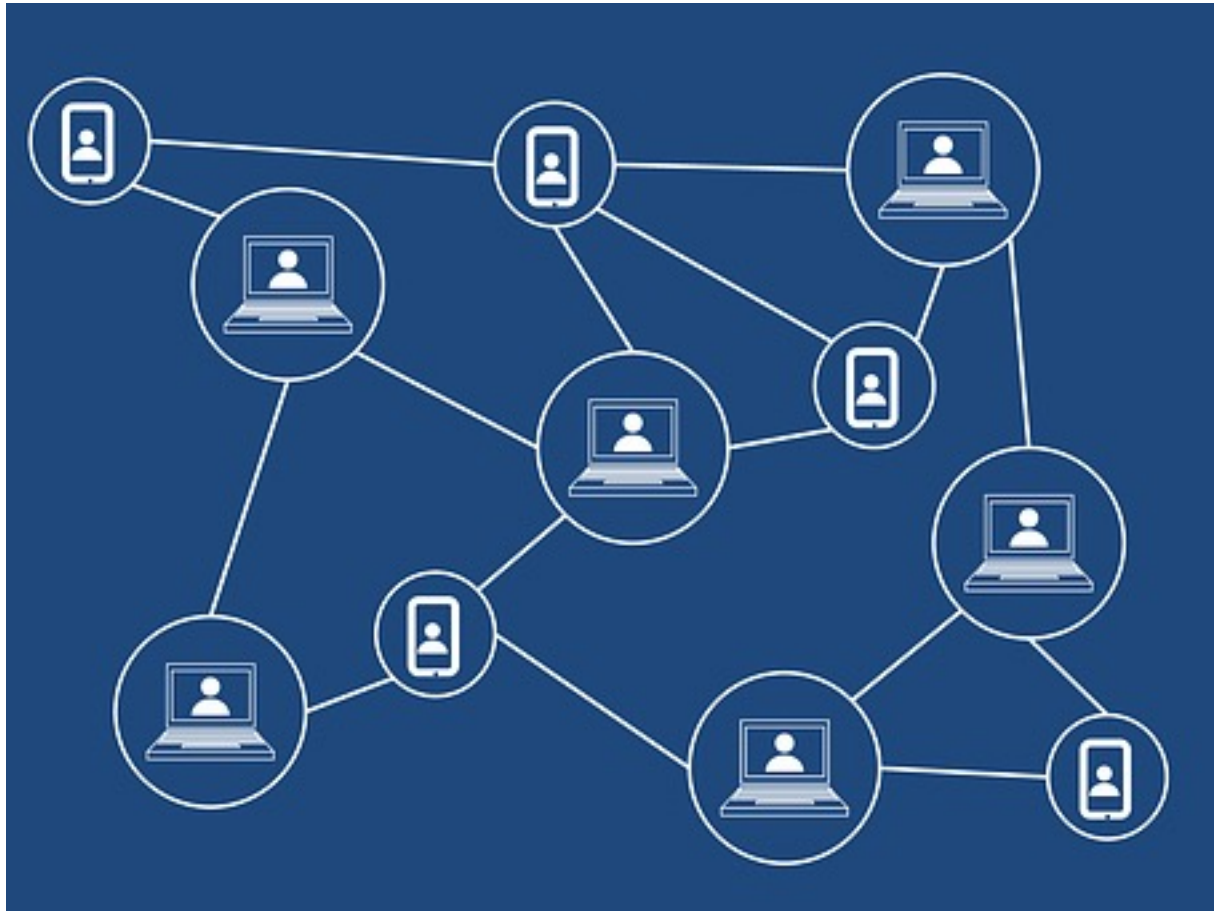
Il network della blockchain

- La blockchain è replicata su ogni nodo della rete - architettura *distribuita*
- La rete è *decentralizzata:* non esiste un server centrale e tutti i nodi sono considerati uguali – svolgono le stesse funzioni (*peer-to-peer*)
- La rete è caratterizzata dall'assenza di una autorità centrale



Il network della blockchain

- Due tipi di nodi: completi (*full node*) e leggeri (*light node*)
- Nodo completo ospita una istanza della BC
- Nodo leggero viene solo usato da utente finale per eseguire transazioni



Transazioni

- Una *transazione* è un record che registra il trasferimento di un asset digitale (ad esempio, Bitcoin) da un mittente ad un destinatario
- dati di una transazione
  - indirizzo del mittente (pagante)
  - indirizzo del destinatario (ricevente)
  - ammontare da trasferire
  - firma digitale del mittente
  - chiave pubblica del ricevente

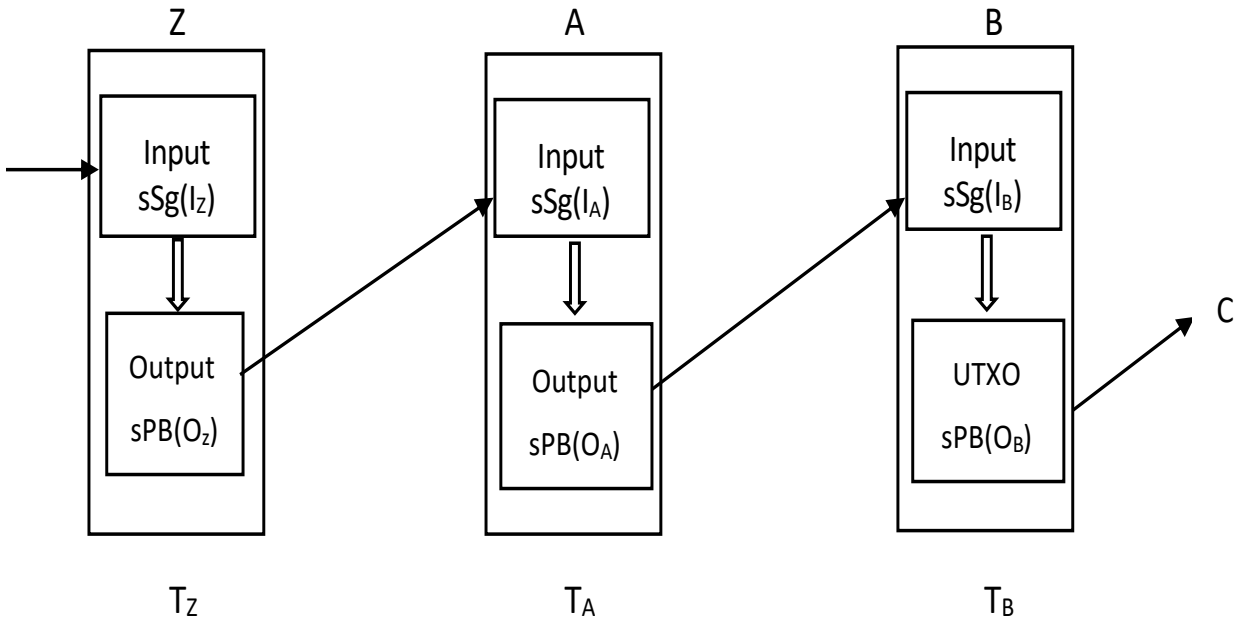
Transazioni

Il pagante A vuole trasferire bitcoin al ricevente B:

- Il ricevente B comanda alla sua applicazione su PC o smartphone di creare un **indirizzo (**equivale all'IBAN)
- Il ricevente B invia l'indirizzo al mittente A tramite qualunque mezzo: mail, messaggio, QR code, ecc.
- Il pagante A
  - inserisce nel suo software l'indirizzo di B e l'ammontare da inviare
  - specifica l'ammontare della commissione da pagare al *miner*
  - conferma la transazione

Transazioni

Il pagante A vuole trasferire bitcoin al ricevente B



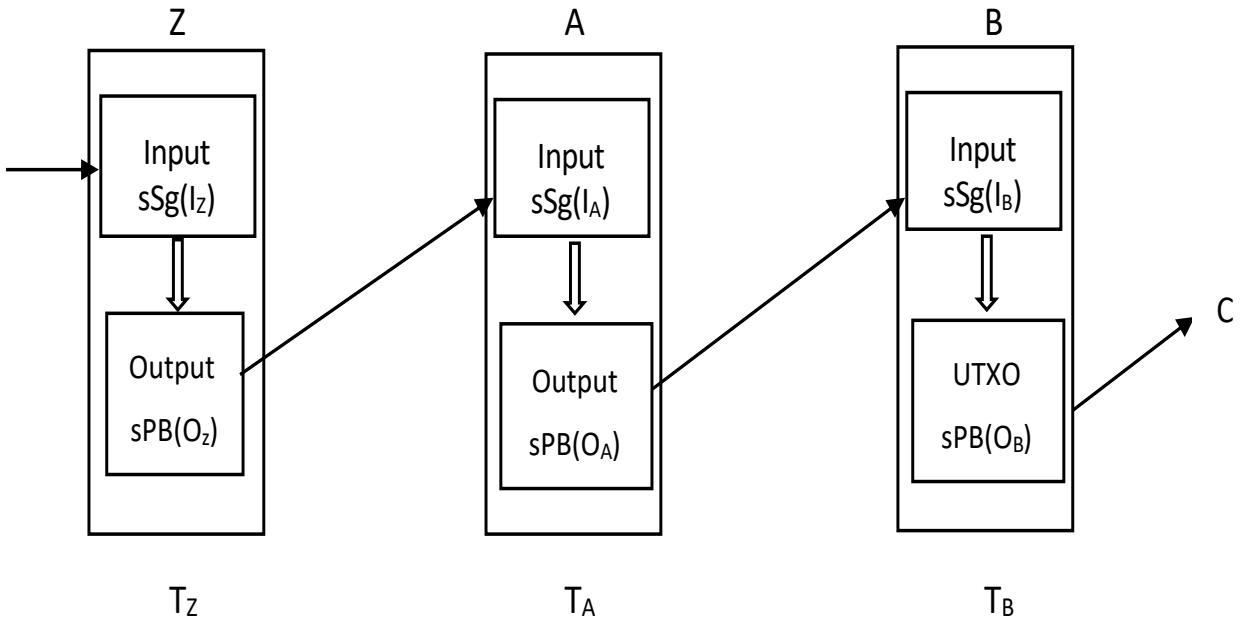
La transazione T_Z costituisce l'input per il trasferimento da A a B; infatti, T_Z ha prodotto un output non speso (UTXO) a favore di A. Quindi, A possiede l'importo specificato.

Affinché A possa utilizzare tale importo per fare un trasferimento a B, deve

1. creare la transazione T_A, con output O_A a favore di B, ed input I_A che fa riferimento a T_Z
2. dimostrare che l'UTXO di Tz è suo.

Transazioni

Il pagante A vuole trasferire bitcoin al ricevente B



- In un certo senso, l'UTXO di Tz è come una cassetta di sicurezza cui solo A può accedere
- A tal fine, lo script *sSg(I_A)* funge da chiave per aprire la serratura *sPK(O_Z)* che custodisce l'UTXO e, quindi, *sSg(I_A)* e *sPK(O_Z)* devono combaciare
- I due script sono formulati utilizzando chiave pubblica e privata di S

Validazione delle transazioni

- Una volta confermata dal pagante, una transazione T viene inviata ai nodi (full node) della rete per la validazione
- Ogni nodo applica il protocollo della blockchain indipendentemente dagli altri

La filosofia di base della BC è che la validazione delle transazioni e dei blocchi debba essere fatta da tutti i partecipanti alla rete, senza una autorità centrale

Validazione delle transazioni

- Quando un nodo riceve una transazione, il protocollo prevede che vengano fatte alcune verifiche, come:
  - L’ammontare da trasferire deve essere nella disponibilità del pagante
  - Il pagante deve essere chi dice di essere
  - ….

Validazione delle transazioni

- Dopo che un nodo (full node) ha localmente verificato la transazione T, procede al suo invio agli altri nodi della rete (T viene propagata), che a loro volta validano T
- Se T non risulta essere valida, allora la sua propagazione viene bloccata
- Le transazioni che risultano valide vengono quindi propagate a tutti i nodi della rete

Consenso e Mining

- Possiamo fidarci della semplice validazione delle transazioni da parte dei nodi della rete?
- **Problema di fiducia**: è possibile, in assenza di una autorità centrale, fidarsi di tutti i partecipanti, cioè, che ogni singolo nodo applichi correttamente il *protocollo* di validazione e, di conseguenza, tutti i nodi posseggono la stessa versione della blockchain?
- Risposta: NO!!!
- *Sybil Attack*: un partecipante disonesto (attaccante) potrebbe creare molti nodi anonimi (anche migliaia o milioni), apparentemente indipendenti, e prendere il controllo della rete per validare proprie transazioni non valide!

Consenso e Mining

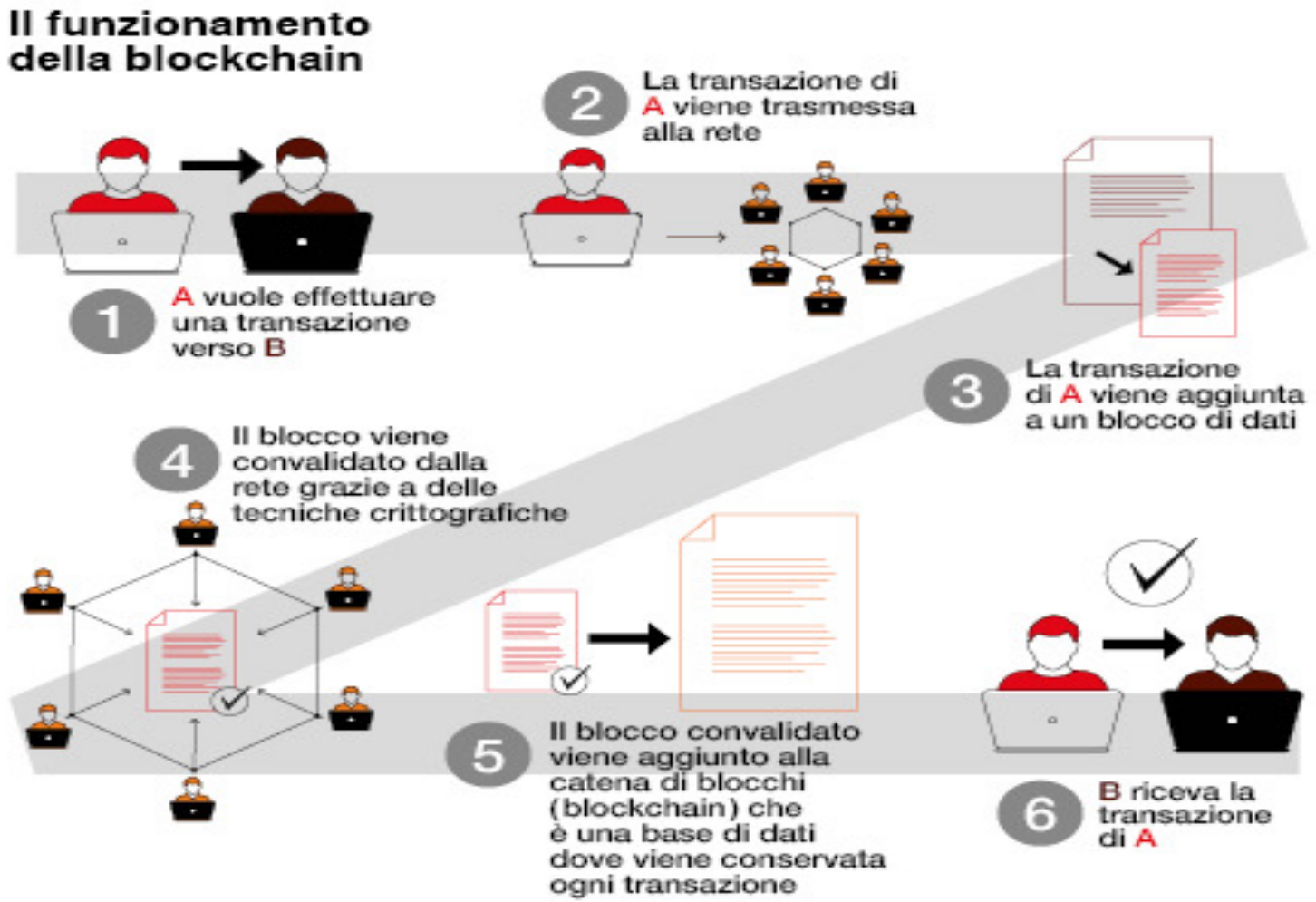
- Bisogna creare meccanismi robusti di *consenso* distribuito, che “forzano” la rete alla applicazione delle regole previste dal protocollo, al riparo da attacchi Sybil
- **Proof of Work** svolto dai **miner**

Consenso e Mining

Nella rete ci sono dei nodi speciali, detti **miner**, che hanno il compito di

1. collezionare le transazioni che ricevono, dopo averle validate, nel cosiddetto *mempool*
2. Formare un nuovo blocco con queste transazioni
3. Effettuare il **mining** sul blocco per poi eventualmente trasmetterlo alla rete affinchè sia inserito in ogni singola istanza della BC

Validazione delle transazioni



Proof of Work (PoW)

- I miner sono utenti della rete disposti ad investire denaro per partecipare al processo di creazione dei blocchi, in cambio di un ricompensa
- Tale processo di basa sul PoW, una competizione che richiede la risoluzione di un *problema matematico*
- A tal fine, i miner devono disporre di un hardware molto veloce e costoso (anche in termini di consumi energetici)
- Il “vincitore” propone alla rete il *suo* blocco che, una volta verificato dai singoli nodi, viene accettato definitivamente

La Competizione Matematica

- **Problema matematico** del PoW: trovare un hash (SHA2-256) dell'header del blocco da creare che inizi con un numero *k* prefissato di zeri
- Problema computazionalmente difficile da risolvere per tentativi
- Ad ogni tentativo, l'argomento dell'hash (header) è variato incrementando il *nonce*

La Competizione Matematica

HEADER

- Timestamp: 12-12-2020:17.38
- #trans: 1200
- Difficulty target: 00001111111..111 (256 bit)
- Nonce: 11001…10… (32 bit)
- Previous hash: 00… 01…
- Merkle root: pq998ier….

TRANSAZIONI

$T_1, \ldots, T_n$

- *digest* = hash(header) è una stringa di 256 bit
- *digest* è una funzione del nonce
- Il difficulty target (DT) impone che *digest* incominci con 4 zeri per essere valido
- Se *digest* non rispetta questo vincolo, allora si ricalcola modificando il nonce
- Il processo viene reiterato fino a soddisfare il DT

- Cambia il nonce per modificare l'header alla ricerca di un digest che soddisfa il difficulty target

La conferma delle transazioni

- Il primo miner M che risolve il problema di ricerca dell'hash vince la competizione, e invia il blocco B a tutti i nodi della rete
- Quando un nodo riceve B esegue le seguenti operazioni:
  - calcola l'hash dell'header di B e verifica che rispetti il difficulty target – questo è un calcolo veloce
  - se la verifica ha successo, il nodo valida le transazioni in B
  - se anche questa verifica ha successo, il nodo accetta il blocco e lo inserisce nella propria istanza della BC. Ogni transazione in B è a questo punto *confermata*

La conferma delle transazioni

- Se il miner M ha invece “imbrogliato” (cioè, ha inviato alla rete il blocco B come valido nonostante contenga qualche transazione non valida, oppure l’hash generato non rispetta il *difficulty target*), il blocco viene semplicemente *ignorato* dai nodi che lo hanno ricevuto
- Il miner perde la ricompensa

Un po’ di numeri

Quanti tentativi sono in media necessari per trovare una soluzione?

- Supponiamo che il digest dell’header sia di n bit, e che il difficulty target richieda hash che iniziano con k zeri. Il numero di configurazioni binarie di *n* bit con k zeri iniziali è pari a 2^{n-k}. Pertanto, la probabilità che una configurazione soddisfi il target è

$$p = \frac{2^{n-k}}{2^n} = \frac{1}{2^k}$$

- Pertanto, il valore atteso del numero di hash da calcolare per trovare un digest ’giusto’, è pari a 2^k
- Più alto è il valore di k, cioè, più basso il valore del digest da generare, maggiore è il numero di tentativi da fare, maggiore è il tempo necessario per risolvere il problema

Un po’ di numeri

Qual è la probabilità p(M) che un miner M vinca la competizione PoW?

$$p(M) = \frac{hashrate(M)}{hashrate\ totale} = \frac{hashrate(M)}{\sum_{\forall\ miner\ X} hashrate(X)}$$

- L’hashrate totale nel 2021 era pari a circa 2^{67} hash/sec
- L’hashrate di un processore ASIC (progettato per il mining) è di circa 14 Th/s = $14*2^{40}$ hash/sec
- Quindi se M possiede un unico ASIC

$$p(M) \approx 0{,}000000014$$

Un po’ di numeri

Quanto costa acquisire la maggioranza dei nodi*?*

- L’hashrate totale rete Bitcoin (2021) ≅ 2^{67} hash/sec ≅ 150 EH/s
- Affinché un miner (o un gruppo di miner) abbia una potenza di calcolo superiore al 50% dell’hashrate totale, deve quindi dotarsi di hardware per immettere in rete una potenza pari a 150 EH/s
- Il prezzo di un comune dispositivo per il mining è di circa 16€ per TH/s
- Il costo complessivo sarebbe quindi di circa $2*10^9$ € (2 miliardi di euro)
- Ai costi dell’investimento in dispositivi hw, andrebbero poi aggiunti gli enormi costi dell’energia necessaria per farli funzionare

Perché il PoW protegge dal Sybil Attack

- Grazie al PoW, acquisire la maggioranza dei nodi costa troppo
- Il valore dell'hash rate è quindi una misura della sicurezza della rete di mining di una criptovaluta
- La regola è semplice: maggiore è l'hash rate più sicura è la rete, in quanto la potenza richiesta per poter eseguire un sybil attack è maggiore

La velocità di creazione dei blocchi

- La velocità (throughput) con cui i nuovi blocchi vengono creati (o le transazioni confermate) dipende da
  - la potenza di calcolo messa in campo dai miner, cioè, l'*hashrate* totale
  - il difficulty target che definisce il numero atteso di hash che devono essere calcolati per confermare un blocco
- La velocità della rete Bitcoin è pari a 1 blocco/10 min

Regolazione del difficulty target

- Il difficulty target viene regolato in maniera da mantenere la rete la velocità costante
- Se la potenza di calcolo (hashrate) aumenta, il difficulty target viene regolato verso il basso per aumentare la difficoltà e richiedere più calcoli per trovare un hash valido
- Al contrario, se la potenza di calcolo diminuisce, il difficulty target verrà regolato verso l'alto per ridurre la difficoltà e rendere più facile il processo di mining.

La velocità di creazione dei blocchi

- Assumendo un hashrate totale HPS = 2^{67} hash/sec, per mantenere la velocità di 1 blocco/10 min, pari a

$$BPS = 0{,}0016 \text{ blocchi/sec}$$

- è necessario che il valore atteso HPB degli hash necessari per convalidare un blocco sia pari a

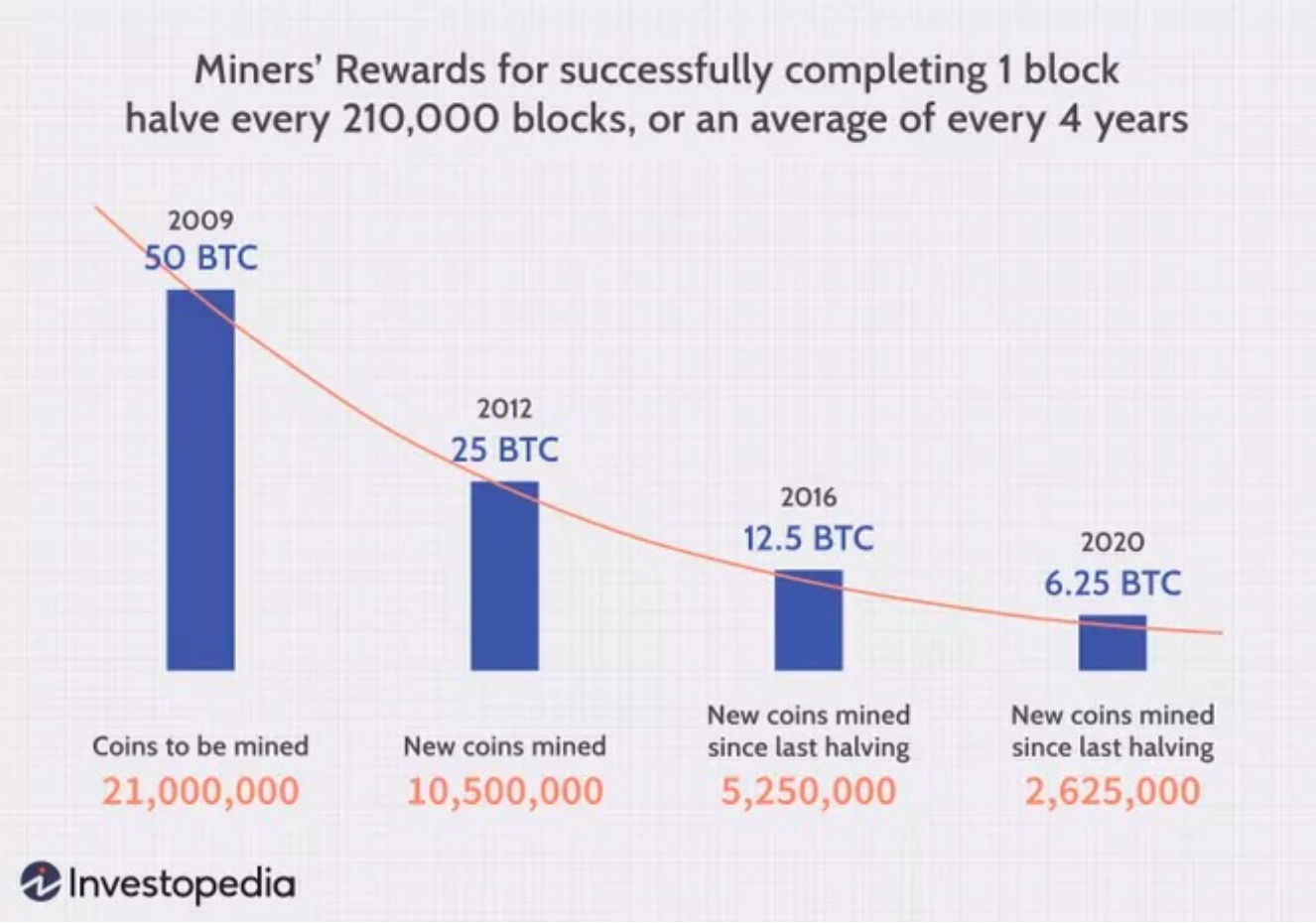
$$HPB = HPS/BPS = 10^{70} \text{ hash/blocco}$$

- Pertanto il numero k di zeri del difficulty target è pari a 70
- Il difficulty target viene regolato ogni 2016 blocchi nuovi (2 settimane)

Ricompensa dei miner

- Il mining non è una attività gratuita, coloro i quali hanno investito risorse economiche per acquistare i calcolatori dedicati ricevono dalla rete la ricompensa prevista
- I miner, attualmente, vengono remunerati con un premio di 6,25 bitcoin per ogni blocco confermato
- Il mining quindi produce una remunerazione di circa 37,5 bitcoin l'ora (6 blocchi l'ora), circa 330.000 bitcoin l'anno
- Con il bitcoin che oggi è quotato a circa 25.000 euro, ciò significa che la ricompensa complessiva è di circa 8 miliardi di euro all'anno

Dimezzamento della ricompensa



- *halving:* le ricompense per l'estrazione di Bitcoin si riducono di circa la metà ogni quattro anni
- Aumenta il valore del Bitcoin

La sicurezza della rete

- Tre fattori
  - tecniche crittografiche
  - architettura distribuita con ridondanza dei dati
  - protocollo del consenso - PoW
- Conviene essere onesti: se la rete percepisce la mancanza di onestà, gli utenti lasciano, ed il sistema crolla. I miner perdono i loro investimenti. Essere onesti, paga!

La sicurezza della rete - Crittografia

- Un nodo, non possedendo le chiavi private relative agli indirizzi di altri utenti, non può sottrarre fondi altrui e accreditarli nel proprio indirizzo
- Ciò perché la crittografia a protezione del sistema è praticamente ineludibile, considerando che è impossibile ricavare la chiave privata a partire da quella pubblica o dall'indirizzo

La sicurezza della rete - Ridondanza

- La BC è un database *resiliente* grazie alla sua struttura ridondante
- I dati memorizzati sulla Blockchain non possono infatti andare persi perché sono replicati su tutti i nodi della Blockchain
- Uno o più nodi possono guastarsi, essere sottoposti ad attacchi di haker, ecc., ma è praticamente impossibile che tutti i server della rete siano colpiti contemporaneamente da eventi negativi

La sicurezza della rete - Consenso

- L'*immutabilità* dei dati della BC deriva dal protocollo del consenso (PoW), dal quale consegue questo principio generale:

modificare in locale la propria BC privata può essere più o meno facile, ma fare accettare le modifiche al resto della rete, se non conformi al protocollo, è molto difficile

- Solo disponendo di almeno il 50% dell'hashrate totale è possibile un double-spending attack

Double Spending

Alice tenta di spendere lo stesso BTC per acquistare due beni diversi, ognuno del valore di 1 BTC

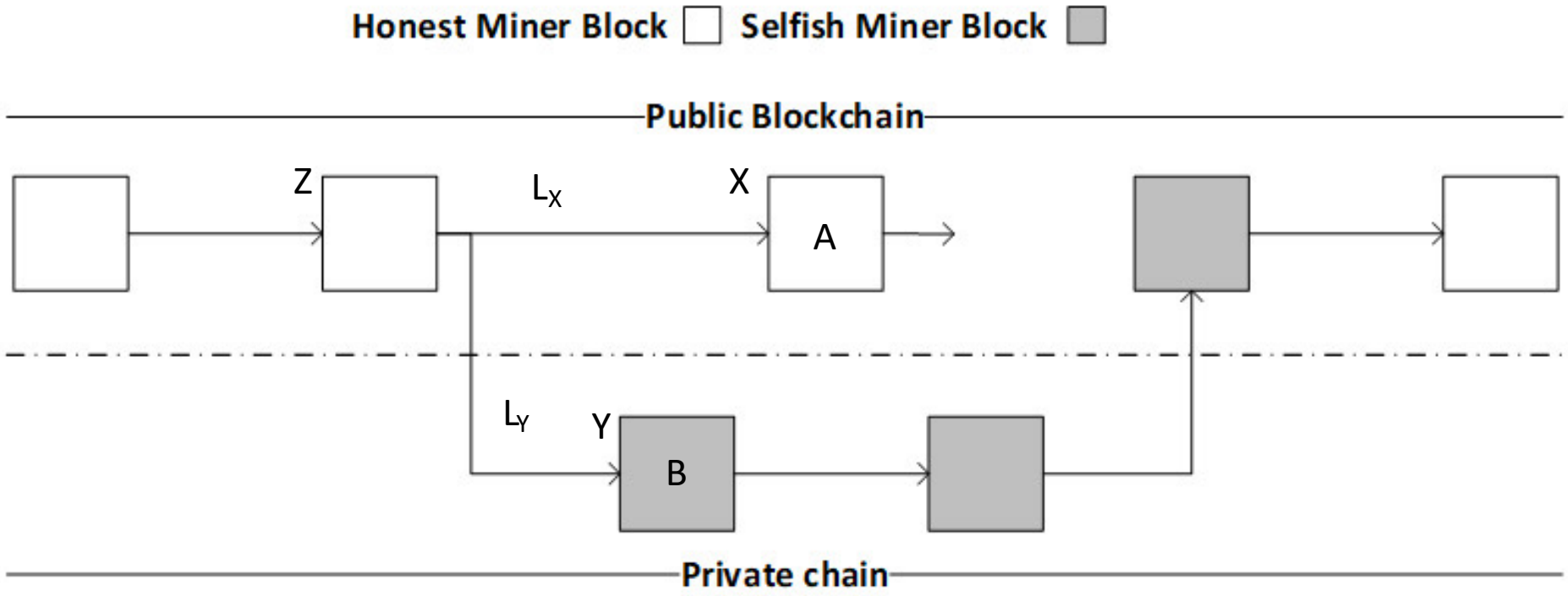
Double Spending

- A tal fine, crea due transazioni separate, A e B, per acquistare due beni ognuno del valore di 1 BTC
- Se A viene inserita per prima in un blocco e confermata, la seconda transazione B verrà rigettata dalla rete
- Pertanto, un miner M disonesto che vuole utilizzare più volte lo stesso ammontare di denaro, dovrà ideare una strategia di attacco che consenta di eludere il controllo sulla compatibilità di A e B

Double Spending

1. M invia alla rete la transazione A con cui acquista un bene online presso il commerciante C1 – sia X il blocco che include A
2. M crea segretamente un altro blocco Y (*selfish mining*) che contiene la transazione B, in conflitto con A, con cui spende, a favore del commerciante C2, lo stesso denaro già speso con C1
3. Il blocco Y NON viene comunicato alla rete, e viene agganciato alla BC locale di M allo stesso blocco Z cui è agganciato X, creando una biforcazione
4. La catena Lx, che contiene X, è quella nota alla rete e include le transazioni oneste, A compresa; la catena Ly non è invece visibile alla rete, ma è solo locale al nodo M

Double Spending



Double Spending

5. Quando il commerciante C1 vede confermata la transazione A (dopo una decina di minuti), procede alla spedizione del bene
6. Mentre la catena Lx continua a crescere per il lavoro dei miner onesti, M continua a validare nuovi blocchi aggiungendoli alla catena privata Ly
7. se M ha sufficiente hashrate, prima o dopo Ly diventa più lunga di Lx. Quando ciò avviene, M rilascia Ly alla rete (un blocco alla volta, incominciando da Y)

Double Spending

8. A questo punto i nodi onesti passano, in accordo con il protocollo della BC, a lavorare sulla catena più lunga Ly in cui è presente B; M fa quindi proprie tutte le ricompense per la risoluzione dei nuovi blocchi di Ly
9. I blocchi della catena onesta Lx diventano orfani, cosicché le rispettive transazioni dovranno essere ri-confermate; ma, poiché A è inconsistente con B (in quanto ha speso il denaro che utilizza A), essa verrà rigettata
10. Il risultato netto è che solo il commerciante C2 riceverà il pagamento, mentre il miner M è entrato in possesso sia del bene acquistato presso C1 che di quello acquistato presso C2

Double Spending

- La possibilità di successo del *double spending* è chiaramente legata alla hashrate di M. In particolare, vale quanto segue:
  - se M controlla più del 50% dell'hashrate totale, *certamente* riuscirà nel tentativo fraudolento – in altri termini, se riesce a minare blocchi più rapidamente di tutti gli altri nodi della rete messi assieme, riuscirà nel tentativo di creare una catena più lunga
  - se invece l'hashrate di M è inferiore alla soglia del 50%, la probabilità di successo decresce esponenzialmente
- Per rendere più difficile l'attacco, la conferma di un blocco viene data solo dopo che altri n blocchi sono stati inseriti nella catena

Controindicazioni del PoW

- Eccessivo consumo di energia: una elevata hashrate richiede elevati consumi di energia, e relativi danni ambientali. Il Bitcoin, che è basato sul PoW, consuma circa lo 0,3% della produzione mondiale di energia. Questo è il costo che la rete deve pagare per garantire sicurezza e immutabilità
- Concentrazione del mining: creazione di grandi *mining farm* e *mining pool* per mettere in campo un potere computazionale elevatissimo. Con la conseguenza che l'hashrate totale aumenta, e diminuisce la probabilità di vincere la gara da parte dei nodi poco attrezzati