

Introduzione alla Blockchain

Bitcoin, crypto-valute e altre applicazioni

Giovanni Ciatto
`giovanni.ciatto@unibo.it`

Dipartimento di Informatica – Scienza e Ingegneria (DISI)
ALMA MATER STUDIORUM – Università di Bologna a Cesena

15 Novembre 2017

Panoramica della presentazione

- 1 Riflessioni sui concetti di “denaro”, “proprietà” e “fiducia” (trust)
- 2 Bitcoin: il contante digitale e P2P di Satoshi Nakamoto
 - Strumenti crittografici
 - Transazioni e blocchi in Bitcoin
 - Proof-of-Work & Mining
- 3 Problemi aperti di Blockchain, Bitcoin e criptovalute
- 4 Applicazioni presenti e future della Blockchain: oltre le criptovalute

Il denaro contante

Denaro *contante* = **gettoni**¹ + valore **convenzionale**

- facili da **riconoscere**, difficili da **riprodurre**
- **intercambiabili** e **additivi**

¹in inglese “tokens”

Il denaro contante

Denaro *contante* = **gettoni**¹ + valore **convenzionale**

- facili da **riconoscere**, difficili da **riprodurre**
- **intercambiabili** e **additivi**

Proprietà del denaro = **Possesso** dei gettoni

¹in inglese “tokens”

Il denaro contante

Denaro *contante* = **gettoni**¹ + valore **convenzionale**

- facili da **riconoscere**, difficili da **riprodurre**
- **intercambiabili** e **additivi**

Proprietà del denaro = **Possesso** dei gettoni

Trasferimento *proprietà* = **cessione** gettoni:

- immediato e istantaneo
- richiede (solo) **fiducia** nel ricevente
- **impossibile** spendere **due volte** lo stesso gettone

¹in inglese “tokens”

Il sistema bancario

Trasferimento denaro su rete → **banca**

- denaro = voce **database**
- **associa** denaro a proprietario

Il sistema bancario

Trasferimento denaro su rete → **banca**

- denaro = voce **database**
- **associa** denaro a proprietario

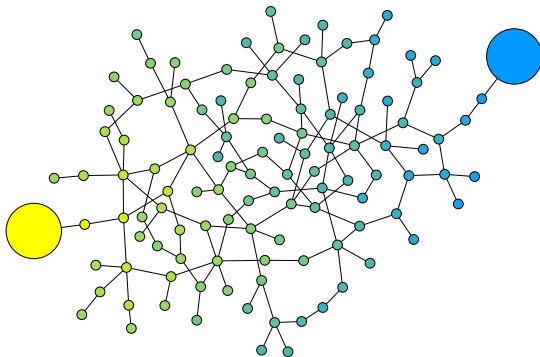
Banca = **intermediario**

- **identifica** mittente e destinatario
- evita conto in rosso
- aggiorna le proprietà

La banca, rispetto al contante

- **Costi** di gestione per trasferimenti
- Suscettibile a leggi locali
- **Autorità**, necessita **fiducia**
- Mantiene **consistente** un sistema distribuito

Il problema della consistenza sulla rete



Consistenza sulla rete

Alice e Bob hanno un conto in comune con 1.050€.

- Alice preleva 100€ dall'Italia
- **contemporaneamente** Bob preleva 1000€ dagli USA

L'idea di Satoshi Nakamoto

Contante digitale: *un meccanismo che permetta di effettuare trasferimenti **consistenti e sicuri di valore** sulla rete, senza intermediari di cui doversi fidare.*

L'idea di Satoshi Nakamoto

Contante digitale: *un meccanismo che permetta di effettuare trasferimenti **consistenti e sicuri di valore** sulla rete, senza intermediari di cui doversi fidare.*

Requisiti gettone:

di valore: permette l'acquisto di beni e servizi

sicuro: **cedibile** solo previo assenso del **proprietario**

L'idea di Satoshi Nakamoto

Contante digitale: *un meccanismo che permetta di effettuare trasferimenti **consistenti e sicuri di valore** sulla rete, senza intermediari di cui doversi fidare.*

Requisiti gettone:

di valore: permette l'acquisto di beni e servizi

sicuro: **cedibile** solo previo assenso del **proprietario**

Requisiti sistema monetario:

consistenza: evitare il **double-spending**

senza intermediari : nessun costo, nessuno può alterare la proprietà dei gettoni, nessuno può impedire transazioni

Problemi del contante digitale

Vari formati di gettone:

```
{"val": 5}
```

Problemi del contante digitale

Vari formati di gettone:

```
{"val": 5}
```

- Valore?
- Proprietario?

Problemi del contante digitale

Vari formati di gettone:

```
{"val": 5}
```

```
{"val": 5,  
  "prop": "Alice"}
```

- Valore?
- Proprietario?
- Produzione illimitata di gettoni?
- Duplicazione illimitata di gettoni?

Problemi del contante digitale

Vari formati di gettone:

```
{"val": 5}
```

```
{"val": 5,  
  "prop": "Alice"}
```

- Valore?
- Proprietario?
- Produzione illimitata di gettoni?
- Duplicazione illimitata di gettoni?
- Double-spending?

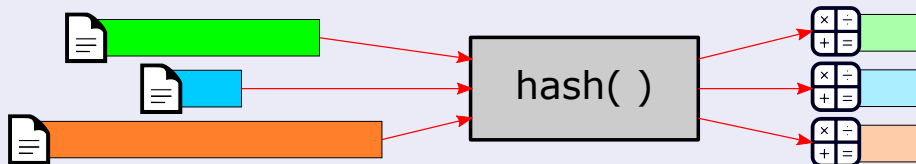
Punto della situazione

- 1 Riflessioni sui concetti di “denaro”, “proprietà” e “fiducia” (trust)
- 2 Bitcoin: il contante digitale e P2P di Satoshi Nakamoto
 - Strumenti crittografici
 - Transazioni e blocchi in Bitcoin
 - Proof-of-Work & Mining
- 3 Problemi aperti di Blockchain, Bitcoin e criptovalute
- 4 Applicazioni presenti e future della Blockchain: oltre le criptovalute

Garantire integrità

Integrità: capire se i dati sono stati alterati

Funzione di hash a-una-via



- ✓ input → hash
- ✗ hash → input
- ✗ 2 input con stesso hash

- piccola variazione input → grande variazione hash

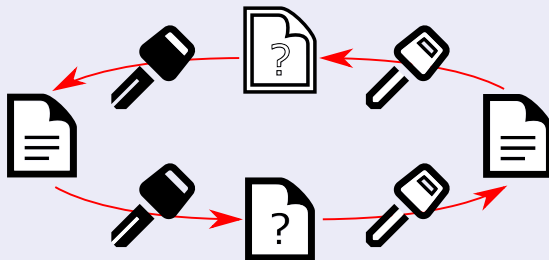
`hash("Alice->Bob: 1 $")`: "47130aff04451de6fbf0765bd059594a"

`hash("Alice->Bob: 2 $")`: "3a876bdfb02a620e183458300569264f"

Garantire autenticità

Autenticità: capire se un dato è stato prodotto da me o no

Crittografia a-simmetrica: chiave pubblica e chiave privata

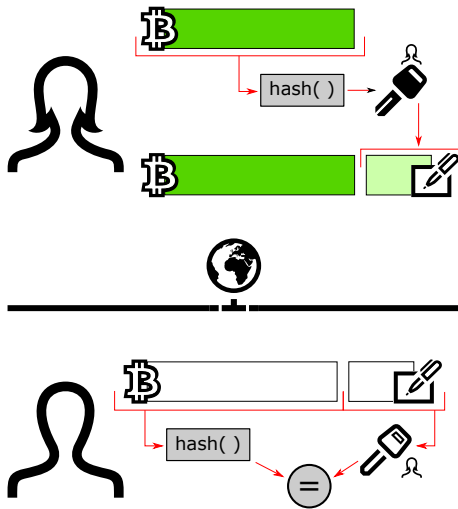


- ✓ creare coppia di chiavi
- ✓ decifrare con chiave giusta
- ✗ decifrare con chiave sbagliata
- ✗ trovare una chiave avendo l'altra

! Cifrare con chiave **privata** → autenticità

! Chiave **pubblica** $\approx$ identificatore utente

Firma digitale = integrità + autenticità



Istantanea VS Flusso

Istantanea:

- Memorizza **stato**
- Operazioni per **variare** stato

Flusso:

- Memorizza **variazioni**
- Operazioni per **calcolare** stato

Istantanea VS Flusso

Istantanea:

- Memorizza **stato**
- Operazioni per **variare** stato

Alice	:	7 €
Bob	:	0 €
Carl	:	0 €

Flusso:

- Memorizza **variazioni**
- Operazioni per **calcolare** stato

Istantanea VS Flusso

Istantanea:

- Memorizza **stato**
- Operazioni per **variare** stato

Alice	:	1 €
Bob	:	3 €
Carl	:	3 €

Flusso:

- Memorizza **variazioni**
- Operazioni per **calcolare** stato

Istantanea VS Flusso

Istantanea:

- Memorizza **stato**
- Operazioni per **variare** stato

Alice	:	1 €
Bob	:	2 €
Carl	:	4 €

Flusso:

- Memorizza **variazioni**
- Operazioni per **calcolare** stato

Istantanea VS Flusso

Istantanea:

- Memorizza **stato**
- Operazioni per **variare** stato

Flusso:

- Memorizza **variazioni**
- Operazioni per **calcolare** stato

$\emptyset \xrightarrow{7}$ Alice

Istantanea VS Flusso

Istantanea:

- Memorizza **stato**
- Operazioni per **variare** stato

Flusso:

- Memorizza **variazioni**
- Operazioni per **calcolare** stato

$\emptyset$	$\xrightarrow{7}$	Alice
Alice	$\xrightarrow{3}$	Bob
Alice	$\xrightarrow{3}$	Carl

Istantanea VS Flusso

Istantanea:

- Memorizza **stato**
- Operazioni per **variare** stato

Flusso:

- Memorizza **variazioni**
- Operazioni per **calcolare** stato

$\emptyset$	$\xrightarrow{7}$	Alice
Alice	$\xrightarrow{3}$	Bob
Alice	$\xrightarrow{3}$	Carl
Bob	$\xrightarrow{1}$	Carl

Istantanea VS Flusso

Istantanea:

- Memorizza **stato**
- Operazioni per **variare** stato

Flusso:

- Memorizza **variazioni**
- Operazioni per **calcolare** stato

`var 0` : $\bigcirc$ $\xrightarrow{7}$ Alice

Istantanea VS Flusso

Istantanea:

- Memorizza **stato**
- Operazioni per **variare** stato

Flusso:

- Memorizza **variazioni**
- Operazioni per **calcolare** stato

<code>var 0</code>	:	$\emptyset$	$\xrightarrow{7}$	Alice
<code>var 1</code>	:	<code>var 0</code>	$\xrightarrow{3}$	Bob
<code>var 2</code>	:	<code>var 0</code>	$\xrightarrow{3}$	Carl

Istantanea VS Flusso

Istantanea:

- Memorizza **stato**
- Operazioni per **variare** stato

Flusso:

- Memorizza **variazioni**
- Operazioni per **calcolare** stato

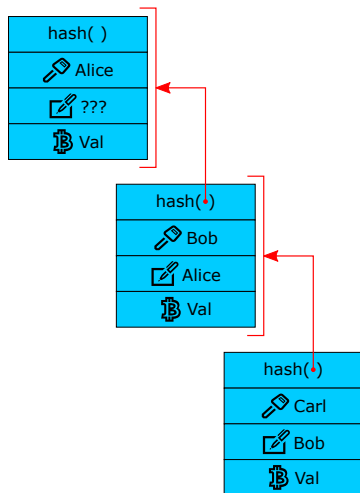
var 0	:	$\emptyset$	$\xrightarrow{7}$	Alice
var 1	:	var 0	$\xrightarrow{3}$	Bob
var 2	:	var 0	$\xrightarrow{3}$	Carl
var 3	:	var 1	$\xrightarrow{1}$	Carl

Le transazioni: i gettoni di Bitcoin

Transazione: trasferimento **firmato** di valore da un proprietario da un altro

Ogni transazione contiene:

1. hash **transazione precedente**
2. chiave *pubblica* **destinatario**
3. firma **mittente**
4. valore da trasferire



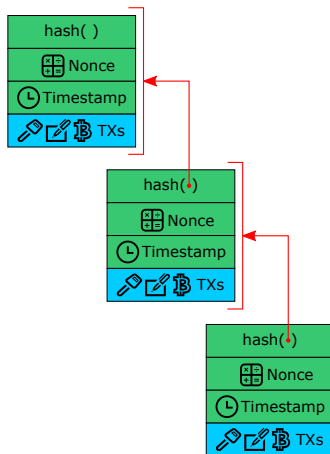
! Solo mittente può firmare la transazione

! Ogni transazione ne **consuma** un'altra

I blocchi: il libro mastro di Bitcoin I

Blocco: sequenza *inviolabile* di transazioni, certamente **valide** a partire da un certo istante

1. hash **blocco precedente**
2. *nonce*
3. **timestamp** blocco corrente
4. lista di transazioni valide



- Alterazione blocco $i \rightarrow$ hash sbagliato nel blocco $i + 1$

$\rightarrow$ Catena di blocchi = storia di tutte le transazioni

Creazione blocco

Pubblicazione nuovo blocco ogni ΔT secondi

Creazione blocco

Pubblicazione nuovo blocco ogni ΔT secondi

Algoritmo: creazione nuovo blocco

- Calcola hash blocco precedente.
- Ascolta pubblicazione transazioni per ΔT secondi
 - ▶ valida ogni transazione
- Alla fine dei ΔT secondi: pubblica blocco.

Creazione blocco

Pubblicazione nuovo blocco ogni ΔT secondi

Algoritmo: creazione nuovo blocco

- Calcola hash blocco precedente.
 - Ascolta pubblicazione transazioni per ΔT secondi
 - ▶ valida ogni transazione
 - Alla fine dei ΔT secondi: pubblica blocco.
-
- *Transazioni rimaste fuori dal blocco non esistono*

Creazione blocco

Pubblicazione nuovo blocco ogni ΔT secondi

Algoritmo: creazione nuovo blocco

- Calcola hash blocco precedente.
 - Ascolta pubblicazione transazioni per ΔT secondi
 - ▶ valida ogni transazione
 - Alla fine dei ΔT secondi: pubblica blocco.
-
- *Transazioni rimaste fuori dal blocco non esistono*
 - Chi crea i blocchi ha un potere immenso
p.e. esclusione chiave pubblica

Chi crea i blocchi?

- × Sempre lo stesso nodo $\implies$ centralizzazione, autorità

Chi crea i blocchi?

- × Sempre lo stesso nodo $\implies$ centralizzazione, autorità
- × Tutti i nodi, **contemporaneamente** $\implies$ problemi di consistenza

Chi crea i blocchi?

- × Sempre lo stesso nodo $\implies$ centralizzazione, autorità
- × Tutti i nodi, **contemporaneamente** $\implies$ problemi di consistenza
- ~ Tutti i nodi, **a turno**
 - ▶ come gestire turni?

Chi crea i blocchi?

- × Sempre lo stesso nodo $\implies$ centralizzazione, autorità
- × Tutti i nodi, **contemporaneamente** $\implies$ problemi di consistenza
- ~ Tutti i nodi, **a turno**
 - ▶ come gestire turni?
- ~ Tutti i nodi, **a caso**
 - ▶ come scegliere un nodo a caso?

Chi crea i blocchi?

- × Sempre lo stesso nodo $\implies$ centralizzazione, autorità
- × Tutti i nodi, **contemporaneamente** $\implies$ problemi di consistenza
- ~ Tutti i nodi, **a turno**
 - ▶ come gestire turni?
- ~ Tutti i nodi, **a caso**
 - ▶ come scegliere un nodo a caso?
- ✓ Alcuni nodi, **in competizione**

Proof-of-Work (PoW)

Proof-of-Work: un blocco è valido se chi l'ha prodotto **dimostra** di aver eseguito un algoritmo **molto oneroso**
! cosa facile da verificare

Proof-of-Work (PoW)

Proof-of-Work: un blocco è valido se chi l'ha prodotto **dimostra** di aver eseguito un algoritmo **molto oneroso**
! cosa facile da verificare

Creazione dei blocchi con PoW

Tutti i nodi provano, *contemporaneamente*:

- a costruire il prossimo blocco
- a risolvere un **puzzle computazionale**
 - ▶ il primo che ci riesce, può pubblicare il blocco e la PoW
- ~ 1 blocco ogni $\Delta T \Rightarrow$ difficoltà puzzle

Proof-of-Work (PoW)

Proof-of-Work: un blocco è valido se chi l'ha prodotto **dimostra** di aver eseguito un algoritmo **molto oneroso**
! cosa facile da verificare

Creazione dei blocchi con PoW

Tutti i nodi provano, *contemporaneamente*:

- a costruire il prossimo blocco
- a risolvere un **puzzle computazionale**
 - ▶ il primo che ci riesce, può pubblicare il blocco e la PoW
- ~ 1 blocco ogni $\Delta T \Leftrightarrow$ difficoltà puzzle

Puzzle computazionale di Bitcoin

Blocco valido $\Leftrightarrow$ hash del blocco comincia con N zeri

- forza bruta sul *nonce*

Consenso sulla catena di blocchi

Consenso: tutti i nodi della rete devono concordare su quale sia la catena di blocchi di riferimento

Consenso sulla catena di blocchi

Consenso: tutti i nodi della rete devono concordare su quale sia la catena di blocchi di riferimento

Strategia per il consenso

1. Pubblicazione blocco = **multi-cast** a vicini
2. Blocco ricevuto valido se
 - ▶ transazioni valide
 - ▶ hash precedente valido
 - ▶ puzzle computazionale risolto
3. Si considera sempre **la catena più lunga**

Consenso sulla catena di blocchi

Consenso: tutti i nodi della rete devono concordare su quale sia la catena di blocchi di riferimento

Strategia per il consenso

1. Pubblicazione blocco = **multi-cast** a vicini
2. Blocco ricevuto valido se
 - ▶ transazioni valide
 - ▶ hash precedente valido
 - ▶ puzzle computazionale risolto
3. Si considera sempre **la catena più lunga**

! Consenso distribuito su storia trasazioni
→ no autorità, no *single-point-of-failure*

PoW e double-spending

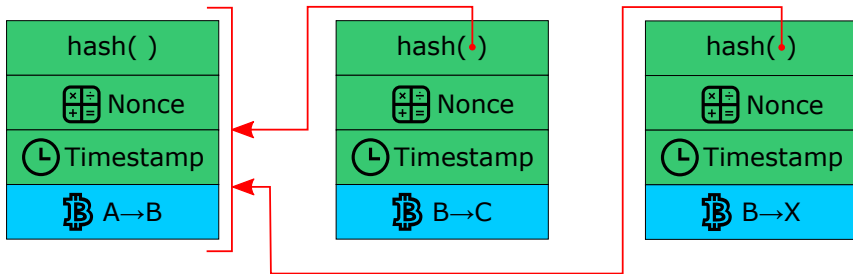
PoW consuma **molte** risorse

- dà **valore** ai gettoni
- impedisce *double-spending*

PoW e double-spending

PoW consuma **molte** risorse

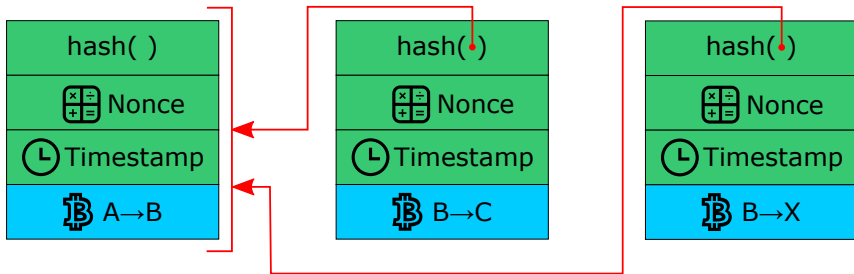
- dà **valore** ai gettoni
- impedisce *double-spending*



PoW e double-spending

PoW consuma **molte** risorse

- dà **valore** ai gettoni
- impedisce *double-spending*



! Che vantaggio hanno i nodi?

PoW, mining e immissione di moneta

Risoluzione *puzzle computazione* → **ricompensa** X **gettoni**

Mining: processo di risoluzione del puzzle computazionale

Miner: nodo che pratica *mining*

- volto al benessere del sistema?
- volto al guadagno?

Ipotesi di fondo: **miner razionali**

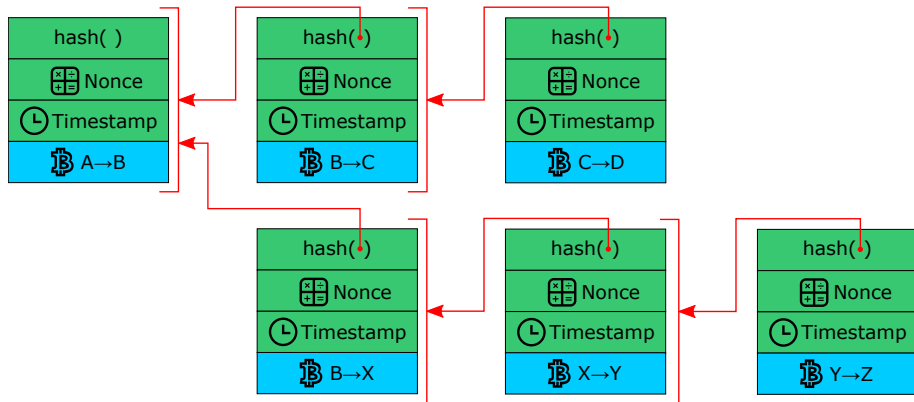
- comportamenti *positivi* **non imposti** ma **incentivati**
- comportamenti *negativi* **non impediti** ma **disincentivati**

Panoramica della presentazione

- 1 Riflessioni sui concetti di “denaro”, “proprietà” e “fiducia” (trust)
- 2 Bitcoin: il contante digitale e P2P di Satoshi Nakamoto
 - Strumenti crittografici
 - Transazioni e blocchi in Bitcoin
 - Proof-of-Work & Mining
- 3 Problemi aperti di Blockchain, Bitcoin e criptovalute
- 4 Applicazioni presenti e future della Blockchain: oltre le criptovalute

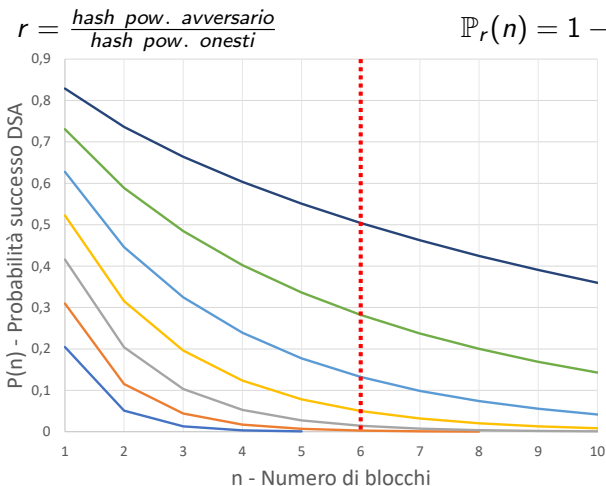
Consenso probabilistico I

Avversario con **molto hashing-power** può tentare DSA



- Quando un blocco è **accettato** dal consenso della rete?
- Quando una transazione si considera valida?

Consenso probabilistico II



In Bitcoin

Transazioni ok se
 $n = 6$

- circa 1h se
 $\Delta T = 10m$

- ok se
hash pow.
avversario <
13%

Lunghezza della Blockchain

Blockchain contiene **tutte le transazioni** avvenute in Bitcoin dalla nascita

Lunghezza della Blockchain

Blockchain contiene **tutte le transazioni** avvenute in Bitcoin dalla nascita
→ impossibile dimenticare il passato (*bug or feature?*)

Lunghezza della Blockchain

Blockchain contiene **tutte le transazioni** avvenute in Bitcoin dalla nascita

→ impossibile dimenticare il passato (*bug or feature?*)

× crescita dimensione continua → full/**light** nodes, *merkle-trees*

Lunghezza della Blockchain

Blockchain contiene **tutte le transazioni** avvenute in Bitcoin dalla nascita

→ impossibile dimenticare il passato (*bug or feature?*)

- × crescita dimensione continua → full/**light** nodes, *merkle-trees*
- × lunga sincronizzazione iniziale $\implies$ barriera d'accesso alla rete

Lunghezza della Blockchain

Blockchain contiene **tutte le transazioni** avvenute in Bitcoin dalla nascita

→ impossibile dimenticare il passato (*bug or feature?*)

- × crescita dimensione continua → full/**light** nodes, *merkle-trees*
- × lunga sincronizzazione iniziale $\implies$ barriera d'accesso alla rete
- × difficile garantire privacy → Zerocoin, **Zerocash**, ...

Distribuzione hash-power

Calcolo parallelo + HW *ad-hoc* $\Rightarrow$ hashing power $\Rightarrow \mathbb{P}(\text{Bitcoin})$
(leggasi *Mining-Farm*)

²in Bitcoin ogni 2016 blocchi (~ 2 sett.)

<https://bitcoinwisdom.com/bitcoin/difficulty>

Distribuzione hash-power

Calcolo parallelo + HW *ad-hoc* $\Rightarrow$ hashing power $\Rightarrow \mathbb{P}(\text{Bitcoin})$
(leggasi *Mining-Farm*)

- $\mathbb{E}(\Delta T) = \text{const} \implies$ auto-organizzazione difficoltà mining²
 - ▶ $\uparrow \text{hash power} \Rightarrow \uparrow \text{difficoltà} \Rightarrow \uparrow \text{hash power} \Rightarrow \dots$

²in Bitcoin ogni 2016 blocchi (~ 2 sett.)

Distribuzione hash-power

Calcolo parallelo + HW *ad-hoc* $\Rightarrow$ hashing power $\Rightarrow \mathbb{P}(\text{Bitcoin})$
(leggasi *Mining-Farm*)

- $\mathbb{E}(\Delta T) = \text{const} \implies$ auto-organizzazione difficoltà mining²
 - ▶ $\uparrow \text{hash power} \Rightarrow \uparrow \text{difficoltà} \Rightarrow \uparrow \text{hash power} \Rightarrow \dots$
- business richiede molti investimenti

²in Bitcoin ogni 2016 blocchi (~ 2 sett.)

Distribuzione hash-power

Calcolo parallelo + HW *ad-hoc* $\Rightarrow$ hashing power $\Rightarrow \mathbb{P}(\text{Bitcoin})$
(leggasi *Mining-Farm*)

- $\mathbb{E}(\Delta T) = \text{const} \implies$ auto-organizzazione difficoltà mining²
 - ▶ $\uparrow \text{hash power} \Rightarrow \uparrow \text{difficoltà} \Rightarrow \uparrow \text{hash power} \Rightarrow \dots$
- business richiede molti investimenti
- barriera d'accesso alla criptovaluta sempre più alta

²in Bitcoin ogni 2016 blocchi (~ 2 sett.)

Distribuzione hash-power

Calcolo parallelo + HW *ad-hoc* $\Rightarrow$ hashing power $\Rightarrow \mathbb{P}(\text{Bitcoin})$
(leggasi *Mining-Farm*)

- $\mathbb{E}(\Delta T) = \text{const} \Rightarrow$ auto-organizzazione difficoltà mining²
 - ▶ $\uparrow \text{hash power} \Rightarrow \uparrow \text{difficoltà} \Rightarrow \uparrow \text{hash power} \Rightarrow \dots$
- business richiede molti investimenti
- barriera d'accesso alla criptovaluta sempre più alta
- × *one-CPU-one-vote*, proposta iniziale di Nakamoto
 - ▶ PoW alternative?

²in Bitcoin ogni 2016 blocchi (~ 2 sett.)

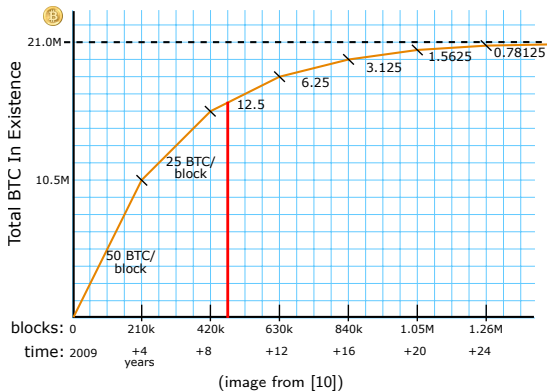
Distribuzione ricchezza

L'immissione di Bitcoin **diminuisce esponenzialmente**: ricompensa X
dimezza ogni 210k blocchi³

³<http://www.bitcoinblockhalf.com/>

Distribuzione ricchezza

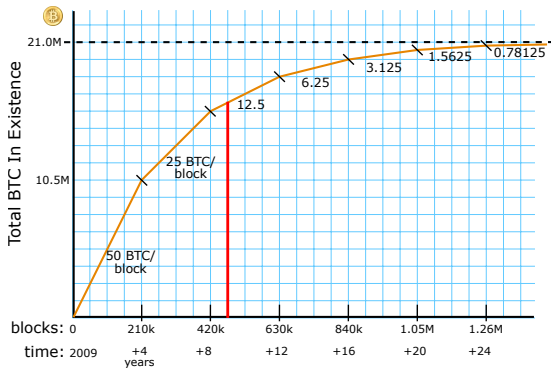
L'immissione di Bitcoin **diminuisce esponenzialmente**: ricompensa $\times$ dimezza ogni 210k blocchi³



³<http://www.bitcoinblockhalf.com/>

Distribuzione ricchezza

L'immissione di Bitcoin **diminuisce esponenzialmente**: ricompensa $\times$ dimezza ogni 210k blocchi³



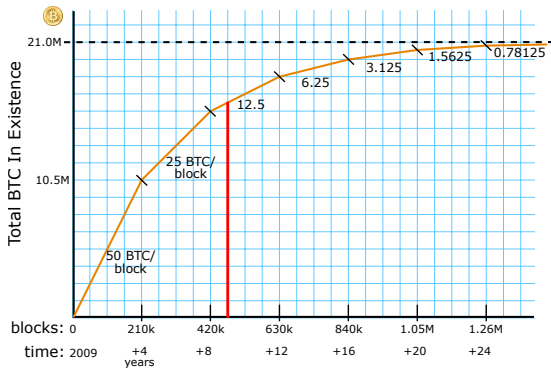
(image from [10])

In futuro i miner guadagneranno solo con **transaction fees**

³<http://www.bitcoinblockhalf.com/>

Distribuzione ricchezza

L'immissione di Bitcoin **diminuisce esponenzialmente**: ricompensa $\times$ dimezza ogni 210k blocchi³



(image from [10])

In futuro i miner guadagneranno solo con **transaction fees**

• Diversi modelli d'inflazione? P.e. **Ethereum**

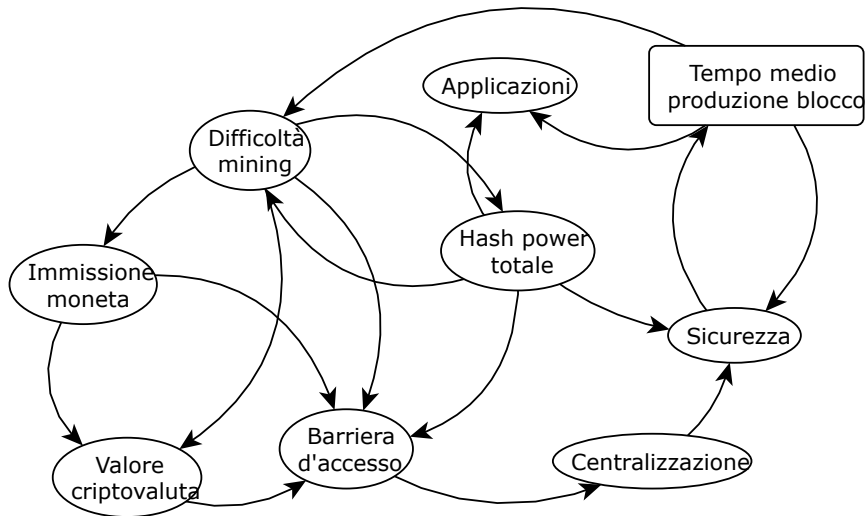
³<http://www.bitcoinblockhalf.com/>

Complessità e auto-organizzazione

Molti aspetti, fortemente interrelati. Difficile modificare parametri.

Complessità e auto-organizzazione

Molti aspetti, fortemente interrelati. Difficile modificare parametri.



Altri problemi

- Tendenza alla centralizzazione

- p.e. Mining-pools

- p.e. Transaction-fees

Altri problemi

- Tendenza alla centralizzazione

- p.e. Mining-pools

- p.e. Transaction-fees

Altri problemi

- Tendenza alla centralizzazione
 - p.e. Mining-pools
 - p.e. Transaction-fees
- Aggiornamento **lento** stato del sistema
 - p.e. proibitivo per vendita al dettaglio

Altri problemi

- Tendenza alla centralizzazione
 - p.e. Mining-pools
 - p.e. Transaction-fees
- Aggiornamento **lento** stato del sistema
 - p.e. proibitivo per vendita al dettaglio
- Attività criminali
 - p.e. vendita beni illegali
 - p.e. evasione fiscale

Altri problemi

- Tendenza alla centralizzazione
 - p.e. Mining-pools
 - p.e. Transaction-fees
- Aggiornamento **lento** stato del sistema
 - p.e. proibitivo per vendita al dettaglio
- Attività criminali
 - p.e. vendita beni illegali
 - p.e. evasione fiscale
- Vuoto legislativo

Altri problemi

- Tendenza alla centralizzazione
 - p.e. Mining-pools
 - p.e. Transaction-fees
- Aggiornamento **lento** stato del sistema
 - p.e. proibitivo per vendita al dettaglio
- Attività criminali
 - p.e. vendita beni illegali
 - p.e. evasione fiscale
- Vuoto legislativo
- Consumo energetico considerevole
 - ▶ alternative a PoW?

Panoramica della presentazione

- 1 Riflessioni sui concetti di “denaro”, “proprietà” e “fiducia” (trust)
- 2 Bitcoin: il contante digitale e P2P di Satoshi Nakamoto
 - Strumenti crittografici
 - Transazioni e blocchi in Bitcoin
 - Proof-of-Work & Mining
- 3 Problemi aperti di Blockchain, Bitcoin e criptovalute
- 4 Applicazioni presenti e future della Blockchain: oltre le criptovalute

Generalizzazione di Bitcoin (parte 1):

- Database **distribuito**, **inviolabile** e **append-only**
- Ad ogni **id-utente** associa **dati** arbitrari
p.e. quantità di denaro
- Manipolazione **consistente** dei dati, previa **autorizzazione**
p.e. trasferimento di denaro
- Un meccanismo efficace per **consenso**
p.e. PoW

Applicazine: registro sicuro

Hp: Memorizzare **dati arbitrari** oltre al denaro

Applicazine: registro sicuro

Hp: Memorizzare **dati arbitrari** oltre al denaro

- ✓ Smart-property al posto di *atti notarili* (auto, casa, terreni, etc)

Applicazine: registro sicuro

Hp: Memorizzare **dati arbitrari** oltre al denaro

- ✓ Smart-property al posto di *atti notarili* (auto, casa, terreni, etc)
- ✓ Identità digitale (carte identità, passaporti, cartelle cliniche, nascite)

Hp: Memorizzare **dati arbitrari** oltre al denaro

- ✓ Smart-property al posto di *atti notarili* (auto, casa, terreni, etc)
- ✓ Identità digitale (carte identità, passaporti, cartelle cliniche, nascite)
- ✓ Bilanci condivisi tra le aziende (e/o il fisco)

Hp: Memorizzare **dati arbitrari** oltre al denaro

- ✓ Smart-property al posto di *atti notarili* (auto, casa, terreni, etc)
- ✓ Identità digitale (carte identità, passaporti, cartelle cliniche, nascite)
- ✓ Bilanci condivisi tra le aziende (e/o il fisco)
- ✓ Gestione brevetti, copyright, proprietà intellettuale

Hp: Memorizzare **dati arbitrari** oltre al denaro

- ✓ Smart-property al posto di *atti notarili* (auto, casa, terreni, etc)
- ✓ Identità digitale (carte identità, passaporti, cartelle cliniche, nascite)
- ✓ Bilanci condivisi tra le aziende (e/o il fisco)
- ✓ Gestione brevetti, copyright, proprietà intellettuale
- ✓ *Domain-Name-Service* (DNS)

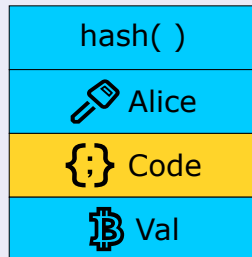
Verso gli smart contract I

Transazioni Bitcoin supportano più **scemi di verifica** delle firme

Linguaggio di programmazione per transazioni

- Firma digitale → **programma** validante
- Computazioni arbitrarie
 - p.e. firma singola, firma multipla, ...
 - ▶ eseguite **da tutti** i miner

! Linguaggio minimale



Verso gli smart contract II

Generalizzazione di Bitcoin (parte 2):

- Database **distribuito**, **inviolabile** e **append-only**
- Ad ogni **id-utente** associa **dati**
 - p.e. quantità di denaro
 - p.e. **programmi**
- Esecuzione di codice **arbitrario** sui dati
 - ▶ in maniera **consistente** e **distribuita** (tra i miner)
 - ▶ previa **autorizzazione** del proprietario
- Un meccanismo efficace per **consenso**

Smart contracts – Intuizione

Smart Contract⁴: programma eseguito **all'unisono** da tutta la rete

⁴Ethereum, Hyperledger

Smart contracts – Intuizione

Smart Contract⁴: programma eseguito **all'unisono** da tutta la rete

- Codice sorgente **pubblico** e **immutabile**

⁴Ethereum, Hyperledger

Smart contracts – Intuizione

Smart Contract⁴: programma eseguito **all'unisono** da tutta la rete

- Codice sorgente **pubblico** e **immutabile**
- **Reattivo** alle transazioni degli utenti

⁴Ethereum, Hyperledger

Smart contracts – Intuizione

Smart Contract⁴: programma eseguito **all'unisono** da tutta la rete

- Codice sorgente **pubblico** e **immutabile**
- **Reattivo** alle transazioni degli utenti
- Dotato di **stato** persistente e **pubblico**

⁴Ethereum, Hyperledger

Smart contracts – Intuizione

Smart Contract⁴: programma eseguito **all'unisono** da tutta la rete

- Codice sorgente **pubblico** e **immutabile**
 - **Reattivo** alle transazioni degli utenti
 - Dotato di **stato** persistente e **pubblico**
- ~ Capaci di gestire **denaro** in modo **trasparente**

⁴Ethereum, Hyperledger

Smart contracts – Applicazioni

Contratto cartaceo → smart-contract **pubblico** e **inviolabile**

Smart contracts – Applicazioni

Contratto cartaceo → smart-contract **pubblico** e **inviolabile**

- Automatizzazione contratti commerciali

Smart contracts – Applicazioni

Contratto cartaceo → smart-contract **pubblico** e **inviolabile**

- Automatizzazione contratti commerciali
- *Distributed Autonomous Organizations* (DAOs)

Smart contracts – Applicazioni

Contratto cartaceo → smart-contract **pubblico** e **inviolabile**

- Automatizzazione contratti commerciali
- *Distributed Autonomous Organizations* (DAOs)
- Votazioni digitali

Contratto cartaceo → smart-contract **pubblico** e **inviolabile**

- Automatizzazione contratti commerciali
- *Distributed Autonomous Organizations* (DAOs)
- Votazioni digitali
- Social-network e/o giornali **non censurabili**

Contratto cartaceo → smart-contract **pubblico** e **inviolabile**

- Automatizzazione contratti commerciali
- *Distributed Autonomous Organizations* (DAOs)
- Votazioni digitali
- Social-network e/o giornali **non censurabili**
- Coordinazione di dispositivi **IoT**

Smart contracts – Problemi aperti

Smart contracts – Problemi aperti

× Privacy e **segretezza** dello stato

Smart contracts – Problemi aperti

- × Privacy e **segretezza** dello stato
- × Esecuzione **posticipata** di codice

Smart contracts – Problemi aperti

- × Privacy e **segretezza** dello stato
- × Esecuzione **posticipata** di codice
- × Computazioni **situate**

Smart contracts – Problemi aperti

- × Privacy e **segretezza** dello stato
- × Esecuzione **posticipata** di codice
- × Computazioni **situate**
- ~ Invio di **eventi** all'esterno della Blockchain

Smart contracts – Problemi aperti

- × Privacy e **segretezza** dello stato
- × Esecuzione **posticipata** di codice
- × Computazioni **situate**
- ~ Invio di **eventi** all'esterno della Blockchain
- ~ Gestione **Turing-completezza**

- Cripto-valute: potenzialità e pericoli
 - ▶ Problemi aperti lungo termine
 - ▶ Non-banalità del sistema

Conclusioni

- Cripto-valute: potenzialità e pericoli
 - ▶ Problemi aperti lungo termine
 - ▶ Non-banalità del sistema
- Bitcoin moneta del futuro?

Conclusioni

- Cripto-valute: potenzialità e pericoli
 - ▶ Problemi aperti lungo termine
 - ▶ Non-banalità del sistema
- Bitcoin moneta del futuro?
- Molte applicazioni per la blockchain
 - ▶ non tutte legate al denaro

- Cripto-valute: potenzialità e pericoli
 - ▶ Problemi aperti lungo termine
 - ▶ Non-banalità del sistema
- Bitcoin moneta del futuro?
- Molte applicazioni per la blockchain
 - ▶ non tutte legate al denaro
- Molte migliorie desiderabili per la tecnologia Blockchain

Riferimenti bibliografici I

- [1] Adam Back.
Hashcash - A Denial of Service Counter-Measure.
[Http://Www.Hashcash.Org/Papers/Hashcash.Pdf](http://www.hashcash.org/papers/hashcash.pdf), (August):1–10, 2002.
- [2] Dave Bayer, Stuart Haber, and W. Scott Stornetta.
Improving the Efficiency and Reliability of Digital Time-Stamping.
Sequences II: Methods in Communication, Security and Computer Science, pages 329–334, 1992.
- [3] J Brito and Andrea Castillo.
Bitcoin: A Primer for Policymakers.
Mercatus Center: George Mason University., 29(4):3–12, 2013.
- [4] Cynthia Dwork and Moni Naor.
Pricing via Processing or Combatting Junk Mail, pages 139–147.
Springer Berlin Heidelberg, Berlin, Heidelberg, 1993.
- [5] Satoshi Nakamoto.
Bitcoin: A Peer-to-Peer Electronic Cash System.
2008.

Riferimenti bibliografici II

- [6] Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller, and Steven Goldfeder.
Bitcoin and Cryptocurrency Technologies.
Network Security, 2016(August), 2016.
- [7] Stuart Haber and Scott Stornetta.
How to Time-Stamp a Digital Document.
Reduced Order Systems, 3(2):1–11, 2010.
- [8] Nick Szabo.
Smart Contracts: Building Blocks for Digital Markets Copyright.
Alamut.Com, (c):16, 1996.
- [9] Andrew S Tanenbaum and Maarten Van Steen.
Distributed Systems: Principles and Paradigms, 2/E.
2007.
- [10] Brian Warner.
Bitcoin: A technical introduction.
<http://www.lothar.com/presentations/bitcoin-brownbag/slides.html>.
- [11] Gavin Wood.
Ethereum: a secure decentralised generalised transaction ledger.
Ethereum Project Yellow Paper, pages 1–32, 2014.