Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE

Corso di Laurea in Informatica per il Management

Oltre i blocchi:

Una Analisi della Rete Lightning Network di Bitcoin

Relatore:
Chiar.mo Prof.
STEFANO FERRETTI

Presentata da:
SIMONE SAMOGGIA

Sessione IV
Anno Accademico 2023/2024

Running bitcoin

Hal Finney

Abstract

Bitcoin è un sistema di pagamento **peer-to-peer** sicuro e decentralizzato, ma presenta limiti di scalabilità dovuto al tempo di conferma delle transazioni e alla dimensione limitata dei blocchi. Per affrontare queste problematiche è stato sviluppato **Lightning Network (LN)**, un protocollo peer-to-peer di secondo livello che consente transazioni istantanee e commissioni molto basse. Affinché Lightning Network possa essere considerato una soluzione sostenibile, è fondamentale che preservi il principale punto di forza di Bitcoin, la decentralizzazione.

La tesi analizza la **struttura topologica** di Lightning Network per valutare la presenza di **hub** e il loro impatto sul livello di **centralizzazione** della rete, ipotizzando che segua le proprietà delle **reti Scale-Free** e **Small-World**.

Per realizzare l'analisi, sono stati utilizzati strumenti di **teoria dei grafi**, con metriche come **Clustering Coefficient**, **Average Path Length** e **Network Centralization**, studiando anche la **Degree Distribution** e la **Betweenness by Degree Distribution**. I dati sono stati elaborati tramite **Cytoscape** e **Gephi**, oltre a script personalizzati in **Python**.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

Nel 2009 è stato creato Bitcoin, *un sistema di pagamento peer-to-peer* [1], altamente sicuro e distribuito, che, però, presenta delle limitazioni in termini di velocità. Ogni transazione richiede mediamente 10 minuti per essere confermata, poiché questo è il tempo richiesto ai *miners* per risolvere il problema del consenso. Inoltre, lo spazio disponibile nei blocchi è una risorsa limitata, in quanto i dati vengono memorizzati sugli hard disk dei nodi. Per garantire un elevato livello di **decentralizzazione**, gli sviluppi mirano a ridurre il carico sui nodi, facilitandone la gestione, in modo da permettere a un numero sempre maggiore di utenti di partecipare alla rete.

Poiché lo spazio è limitato, le commissioni per inviare una transazione dipendono dalla quantità di spazio che l'utente desidera occupare nel blocco, ovvero dal *peso* della sua transazione e dalla domanda per tale spazio.

Sostanzialmente, lo spazio nei blocchi è simile a una risorsa messa all'asta, per esempio come i metri quadri delle abitazioni nel marcato immobiliare: quando c'è grande richiesta, le commissioni aumentano, poiché gli utenti devono competere per ottenere una porzione di **spazio limitato**. Questo meccanismo è perfetto per ottimizzare lo spazio del blocco, ma porta ad un aumento delle commissioni in caso di transazioni pesanti oppure in caso di alta domanda per il *block space*.

Per risolvere questi problemi, nel 2018 è stato creato **Lightning**

Network, un layer superiore a Bitcoin, che consente di scambiare fondi tra utenti con commissioni molto basse, riducendo l'uso dello spazio dei blocchi e, di conseguenza, abbassando il costo delle commissioni, rendendo i pagamenti virtualmente istantanei. Lightning Network, però, **non** deve portare ad una *centralizzazione*, poiché verrebbe meno uno degli obiettivi principali per cui è nato Bitcoin, ovvero la **decentralizzazione**.

Questa tesi si propone di analizzare la reale decentralizzazione di Lightning Network, sfruttando un nodo Bitcoin e due nodi Lightning. L'analisi viene condotta tramite strumenti della *teoria dei grafi*, utilizzando software come *Cytoscape* e *Gephi*, insieme al linguaggio di programmazione **Python** per l'elaborazione dei dati e la gestione dei grafi.

Capitolo 2

Bitcoin

2.1 Le basi di Bitcoin

Bitcoin nasce il 3 gennaio 2009, dall'idea di una figura anonima con lo pseudonimo *Satoshi Nakamoto*, con l'obiettivo di creare un sistema di pagamento *Peer-to-Peer* sicuro, verificabile e privo di intermediari, basato interamente sulla **crittografia**. L'idea iniziale della struttura di Bitcoin viene descritto nel whitepaper *"Bitcoin: A Peer-to-Peer Electronic Cash System"*, pubblicato da Satoshi Nakamoto il 31 ottobre 2008 [1].

All'interno del network Bitcoin operano diverse figure con ruoli specifici:

- **Nodi**
- **Miners**
- **Utenti**

Nodi

I **nodi** sono dispositivi che mantengono una copia della blockchain e verificano la validità delle transazioni e dei blocchi.

Miners

I **miners** – che utilizzano le informazioni gestite dai *nodi* – competono per aggiungere nuovi blocchi alla blockchain attraverso il meccanismo della **Proof-of-Work (PoW)**, cioè l'**algoritmo di consenso** di Bitcoin. Questo processo richiede infrastrutture specifiche, dato l'elevato consumo di energia legato all'attività che i miners devono svolgere, ovvero un enorme numero di calcoli (*hashing*) per validare i blocchi. Come *ricompensa* per il loro *lavoro*, ricevono nuovi bitcoin e le commissioni associate alle transazioni incluse nel blocco.

Utenti

Gli **utenti** sono coloro che utilizzano il network per inviare e ricevere bitcoin tramite i propri wallet. Interagiscono con la rete senza necessariamente partecipare al processo del mining o alla validazione diretta delle transazioni. Per garantire che le loro operazioni vengano confermate nel minor tempo possibile, gli *utenti* pagano ai miners le commissioni (**fee**). Queste commissioni incentivano i miners ad inserire le transazioni nei blocchi, privilegiando quelle con *fee più alte* in caso di congestione della rete. [1]

2.1.1 Le transazioni

Il processo di una transazione avviene nel seguente modo: **Alice** decide di inviare dei fondi a **Bob** e genera una transazione dal proprio wallet. Questa transazione viene trasmessa a un nodo della rete Bitcoin, che ne verifica la correttezza, controllando aspetti come la firma digitale, la disponibilità dei fondi e il rispetto delle regole del protocollo Bitcoin. Se la transazione è valida, il nodo la propaga ad altri nodi della rete, che, a loro volta, svolgeranno

lo stesso processo. In caso contrario, la transazione viene rigettata e non viene aggiunta alla blockchain. Se la transazione supera la verifica, entra in uno *stato di attesa* e viene inserita nella **mempool** del nodo, una sorta di "coda temporanea" che raccoglie tutte le transazioni non ancora *confermate*, ovvero non ancora incluse in un blocco. Ogni nodo possiede la propria mempool, che contiene tutte le transazioni che il nodo ha ricevuto e validato, ma che non sono ancora state confermate dai miners. Per confermare una transazione, i miners selezionano un insieme di transazioni dalla mempool e le *aggregano* in un blocco, che rappresenta lo spazio massimo disponibile per essere aggiunto alla blockchain di Bitcoin.
Ogni blocco viene aggiunto alla blockchain ogni 10 minuti circa, a seguito della risoluzione del problema di consenso tramite la **PoW**.

2.2 Consenso distribuito e PoW

La **Proof-of-Work** è l'**algoritmo di consenso** di Bitcoin e viene applicato dai **miners**. Questo algoritmo è una delle componenti principali che caratterizza la sicurezza e decentralizzazione di Bitcoin, poichè permette di non avere un coordinatore e di gestire il consenso sullo stato della rete in modo distribuito, utilizzando un *consenso emergente* [2]. Il consenso viene definito "emergente" perché non si raggiunge attraverso un'elezione o in un momento preciso, ma si forma gradualmente in modo spontaneo. Il consenso viene creato dall'interazione asincrona di tutti i nodi e i miners che rispettano il protocollo.
Il consenso viene raggiunto in 4 fasi principali:

Ecco una versione più concisa e scorrevole:

1. I nodi validano in modo indipendente ogni transazione ricevuta.
2. Ogni miner aggrega autonomamente le transazioni, le include in un nuovo blocco e certifica il lavoro svolto tramite la *Proof-of-Work*.
3. I nodi verificano i blocchi e li concatenano per formare la blockchain.
4. Ciascun nodo seleziona la catena che presenta il maggior calcolo cumulativo, attestato dalla *Proof-of-Work*.

Alla fine di questo processo, i nodi concorderanno sulla catena da seguire. [2]

Reorg

Durante questo processo potrebbe verificarsi il caso in cui più miners trovino, simultaneamente, una soluzione all'algoritmo di consenso: in questo caso entrambi i blocchi verranno considerati validi da alcuni nodi. Questo crea una *biforcazione* temporanea della blockchain, nota come **fork**. In questo caso, alcuni nodi seguiranno un blocco, altri l'altro, formando più catene parallele. Nel frattempo, i miners continueranno a lavorare sulla propria catena, finché una di esse non ottiene un nuovo blocco, diventando la più lunga e quindi quella con più *lavoro*. Di conseguenza, l'altra catena verrà abbandonata e il blocco, da cui è nata la biforcazione, non sarà più considerato valido, diventando un *blocco orfano* [3].
Questo evento viene chiamato comunemente **reorg** (*chain reorganization*) e non è molto frequente: ad esempio tra il 1 gennaio 2022 e il 28 febbraio 2025 si è verificato solo 26 volte, pari allo 0.016% dei blocchi minati. Rappresenta un caso particolare che aiuta a comprendere il meccanismo di consenso di Bitcoin. Questo sistema garantisce che la catena scelta dai nodi come unica catena valida sia sempre quella con il **maggior accumulo di lavoro** (PoW), assicurando l'integrità della rete, senza la necessità di intermediari o coordinatori. [4]
L'immagine 2.1 illustra i reorg della blockchain dal momento dell'installazione del nodo per la tesi, ovvero dal 27 dicembre 2024.

```
samoggino@serverino:~$ bitcoin-cli getchaintips
[
  {
    "height": 885744,
    "hash": "00000000000000000000fa606cf45c897218b42d06faa7e0e01d36dacdf51a7e",
    "branchlen": 0,
    "status": "active"
  },
  {
    "height": 883183,
    "hash": "00000000000000000002585e763064d66b3f2554ef9dfb8603a2251ae16439e2",
    "branchlen": 1,
    "status": "valid-fork"
  },
  {
    "height": 881953,
    "hash": "00000000000000000007faee36bc09b2ab33e348e5991858f1ad938fb88a570",
    "branchlen": 1,
    "status": "valid-headers"
  },
  {
    "height": 881780,
    "hash": "00000000000000000002745046f0bb6fab9fd306b5df8afea25e5a4cc80f9282",
    "branchlen": 1,
    "status": "valid-fork"
  },
  {
    "height": 877770,
    "hash": "00000000000000000000451bdfb9fbd0b836ee62da952e68bbb7c5a3ceb191df",
    "branchlen": 1,
    "status": "valid-fork"
  }
]
```

Figura 2.1: Reorg della blockchain dal 27 dicembre 2024 al 28 febbraio 2025, visibile dal nodo installato per la tesi

2.3 Blocchi

Block time

Il tempo di generazione di un blocco in Bitcoin è stato fissato a 10 minuti, per consentire a miners e nodi di raggiungere un consenso in modo efficiente, riducendo il rischio di conflitti tra blocchi concorrenti e limitando la formazione di *fork* indesiderati nella blockchain, legati alla propagazione dei blocchi tra i nodi. Un block time più breve, infatti, aumenterebbe la probabilità di *reorg*, poichè la propagazione dei blocchi tra i nodi richiede del tempo, e in quel tempo potrebbero essere minati altri blocchi creando, appunto, dei *fork*.

Block space

La dimensione massima di 4 MB per blocco è stata introdotta per evitare una crescita eccessiva dello spazio di archiviazione necessario per mantenere una copia completa della blockchain. Limitarne la dimensione permette a un numero maggiore di utenti di eseguire un nodo completo (*full node*), rendendo più accessibile la partecipazione alla rete e contribuendo a preservare la decentralizzazione, evitando che solo attori con risorse elevate possano gestire nodi completi.

Ottimizzazione delle risorse dei nodi

Se il block space fosse maggiore o il tempo di blocco fosse ridotto, la quantità di dati da trasmettere e validare aumenterebbe significativamente, comportando requisiti tecnici più elevati per gestire un nodo. Ciò renderebbe più costoso, per gli utenti comuni, partecipare alla rete favorendo la centralizzazione e concentrando il controllo nelle mani di pochi attori con risorse più avanzate, diminuendo così il livello di decentralizzazione della rete.

2.4 UTXO - Unspent Transaction Output

Nel sistema Bitcoin non si possiedono bitcoin nel senso tradizionale, ma si detiene il diritto di *spenderli*. In altre parole, le transazioni non comportano lo spostamento fisico di bitcoin, bensì ne aggiornano il titolare di una determinata quantità. Quando **Alice** "invia" 1 bitcoin a **Bob**, sta comunicando alla rete che, d'ora in poi, solo **Bob** potrà *spendere* quel bitcoin. Questo accade perchè, a differenza della moneta tradizionale che si conserva in un portafoglio, Bitcoin si basa su UTXO (*Unspent Transaction Output*), ovvero delle quantità di bitcoin che cambiano proprietà ad ogni transazione.

Diversamente dall'euro, che ha tagli fissi (come 1 o 2 euro e le relative frazioni), Bitcoin non ha tagli predefiniti e può essere suddiviso in 100 milioni di **satoshi**, ovvero l'unità più piccola di Bitcoin (mentre l'euro è divisibile solo 100 volte, ovvero fino ad ottenere un centesimo). Questo consente agli utenti di inviare importi esatti senza vincoli. L'insieme delle UTXO che un utente può spendere rappresenta il suo saldo, che può essere speso in transazioni future. [5]

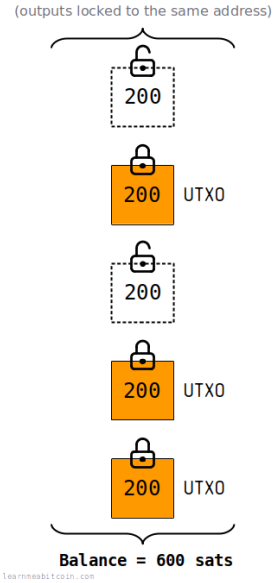

Figura 2.2: In questo caso, l'utente possiede tre UTXO da 200 sats

Esempio di transazione con modello UTXO

Alice possiede 1 bitcoin e decide di trasferirne 0.7 a **Bob** la transazione genererà due output (*UTXO*) 2.3:

- Uno da 0.7 bitcoin per **Bob**.
- Uno da 0.2 bitcoin come **resto** per **Alice**.

La differenza, pari a 0.1 bitcoin, verrà trattenuta come commissione dal miner che confermerà la transazione (**mining fee**) .

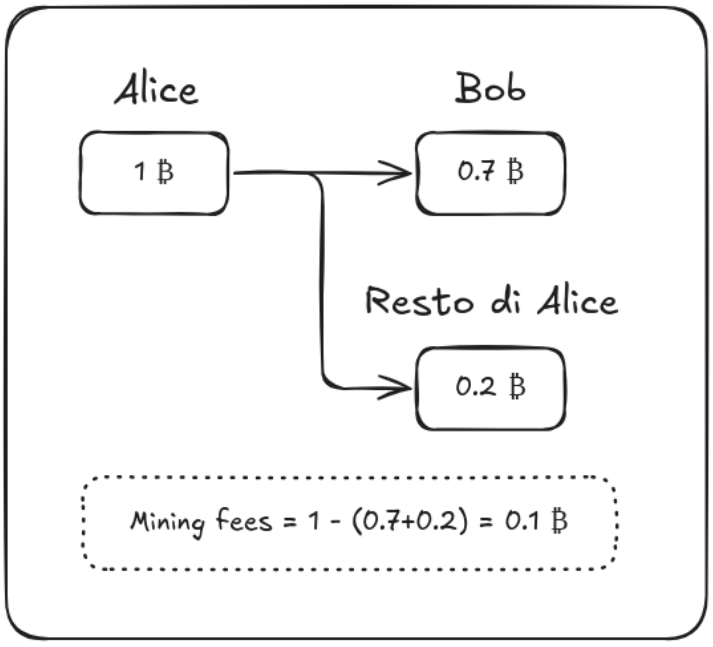

Figura 2.3: Schema del modello UTXO

Per la stessa transazione possono essere utilizzate più di una UTXO in input. Ad esempio, se **Alice** ha due UTXO da 1 e 0.5 bitcoin, può combinarle per inviare 1.2 bitcoin a **Bob**, che riceverà un'unica UTXO di quel valore.

Il saldo di un utente è quindi la somma di tutte le sue UTXO e viene definito **UTXO set**. In pratica, una transazione è costituita dagli input (UTXO di precedenti transazioni) e dagli output (nuove UTXO generate), che potranno essere usati per future transazioni [6].

2.4.1 Vantaggio per i nodi

Il modello **stateless** di Bitcoin riduce significativamente il carico computazionale sui nodi, consentendo loro di mantenere un database aggiornato di tutti gli UTXO esistenti in un dato momento e dei relativi proprietari [6], senza la necessità di modificare l'intero stato della blockchain, bensì solo le UTXO che cambiano proprietario [7].

Grazie a questo database, i nodi possono verificare rapidamente la validità delle transazioni, assicurando che gli input utilizzati possano essere spesi. Questo meccanismo previene il **double-spending** [8] e garantisce l'integrità del sistema. I nodi, infatti, verificano la validità di una transazione controllando che la quantità in input sia pari o superiore alla somma degli output, così da impedire agli utenti di spendere due volte gli stessi UTXO, un'azione illecita chiamata, appunto, **double-spending** [6]. Un esempio di transazione non valida è mostrato in figura 2.4.

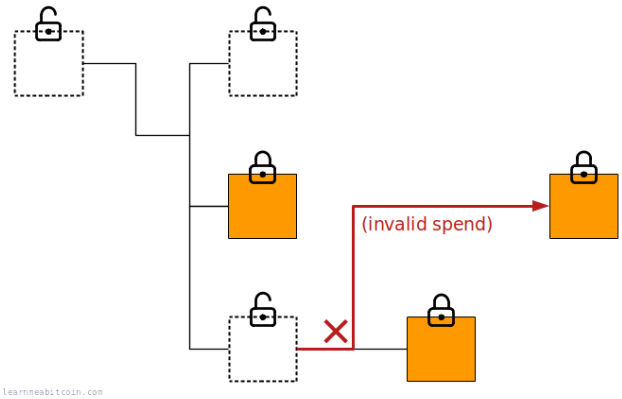

Figura 2.4: Esempio di transazione non valida

Poichè ogni UTXO viene consumato integralmente e ricreato come output, non è necessario gestire saldi complessi, rendendo il processo di validazione più efficiente. Questo approccio migliora la **scalabilità** della rete, riducendo i tempi di elaborazione delle transazioni. Inoltre, le UTXO di transazioni differenti sono indipendenti tra loro, il che permette di processare più transazioni in parallelo, aumentando ulteriormente l'efficienza del sistema.

2.5 Fork e altre blockchain

Per affrontare questi problemi si potrebbe considerare l'adozione di diverse soluzioni, come l'aumento della dimensione del blocco oppure la riduzione del tempo di conferma. Di conseguenza, sono nati progetti come **Litecoin** (*LTC*), **Bitcoin Cash** (*BCH*) e **Bitcoin Satoshi Vision** (*BSV*). Litecoin, ad esempio, è stato concepito a partire dal codice di Bitcoin, modificandone il tempo di blocco da 10 minuti a 2.5 minuti [9], mentre Bitcoin Cash (*hard-fork* di Bitcoin) e Bitcoin Satoshi Vision (*hard-fork* di Bitcoin Cash), hanno puntato sull'incremento della dimensione dei blocchi. BSH ha aumentato la dimensione a 8 MB (in seguito a 32 MB [10]), mentre BSV la portata a 128 MB, per poi rimuovere il limite massimo nel 2020 [11], creando il blocco più grande nel 2021 con una dimensione di 2 GB [12].

Centralizzazione

Le soluzioni adottate da BCH e BSV hanno incrementato notevolmente i costi per mantenere un nodo, in quanto blocchi di dimensioni maggiori richiedono risorse più elevate per la validazione e accelerano la crescita complessiva della blockchain. Tale dinamica ha favorito una progressiva centralizzazione, con il numero di nodi ridotto a circa 50 per BSV [13] e 500 per BCH [14]. Una situazione analoga si riscontra con Litecoin: la maggiore rapidità nella generazione dei blocchi porta a un'espansione più veloce della blockchain, incrementando di conseguenza i costi per l'esecuzione di un nodo, il cui conteggio si aggira intorno ai 1100 [15].

A titolo comparativo, Ethereum – la seconda blockchain per dimensione e diffusione – conta circa 4000 nodi [16], mentre Solana ne registra circa 4400 [17]. È importante sottolineare che, per Ethereum e Solana, il discorso è diverso, poiché i nodi non solo validano i blocchi ma possono anche partecipare attivamente alla scrittura della blockchain, ricevendo in cambio la block-reward [18]. Tutto questo non si riscontra in Bitcoin e nelle sue chain derivate, dove l'inclusione delle transazioni nei blocchi è affidata ai miners,

pertanto non c'è incentivo economico per l'esecuzione di un nodo. Per quanto riguarda Bitcoin, le stime risultano meno precise, dato che la maggior parte dei nodi opera tramite **TOR** [19]; le valutazioni variano tra 60mila [20] e 93mila [21], di cui circa 22mila [22] sono nodi pubblici.

2.5.1 SegWit, Taproot e Lightning Network

Il *soft-fork* **SegWit** [23], adottato nell'agosto 2017, ha modificato la dimensione dei blocchi di Bitcoin, aumentandone il limite massimo del peso da 1 MB a 4 MB, senza modificarne la dimensione effettiva, ma ottimizzandone la gestione interna. SegWit ha anche ottimizzato la sincronizzazione dei nodi [24] risolvendo il problema del **quadratic hashing**, che rallentava la validazione delle transazioni [25]. Questo aggiornamento ha aperto la strada allo sviluppo di soluzioni di scaling di secondo livello, come **Lightning Network** [26], che spostano le transazioni **off-chain**, riducendo così tempi e commissioni e alleggerendo il carico computazionale sui nodi. Inoltre, grazie al successivo *soft-fork* **Taproot** [27] che introduce le **firme Schnorr** [28] e altre ottimizzazioni, è possibile aggregare più firme in un singolo output, migliorando l'efficienza nell'uso dello spazio all'interno dei blocchi e contribuendo così a una maggiore scalabilità della rete.

2.6 Problematiche di Bitcoin

Nonostante Bitcoin sia un network molto solido e distribuito, presenta alcuni problemi. I principali sono:

- Tempo di conferma del blocco
- Scalabilità limitata
- Commissioni (Fees)

La block space limita il numero di transazioni che possono essere verificate **ogni 10 minuti**. Pur essendo questa scelta una caratteristica intenzionale e non un difetto, essa comporta un limite alla **scalabilità della rete**. Di conseguenza, in periodi di alta richiesta, gli utenti si trovano a competere per far includere le loro transazioni nei blocchi successivi, determinando un aumento delle fee che vengono pagate ai miners per ottenere conferme più rapide. [14]

Capitolo 3

Lightning Network

3.1 Le basi di Lightning Network

Mentre Bitcoin si basa su transazioni pubbliche e conferme sulla blockchain che richiedono tempo, Lightning Network si affida a canali di pagamento *off-chain* tra nodi Lightnig. Questi canali offrono velocità virtualmente istantanea e costi molto bassi, ma con una logica di funzionamento differente, che implica il monitoraggio dei canali e la loro gestione attiva. I nodi Lightning comunicano tra loro tramite un **gossip protocol**, ovvero un sistema che permette loro di aggiornare la loro conoscenza del network.

The entirety of all payment channels forms the Lightning Network.

[29]

3.2 Attori

3.2.1 Nodo Lightning

Il nodo Lightning è lo strumento che permette all'utente la gestione di canali e della loro liquidità su Lightning Network. Per eseguire un nodo Lightning è **necessario** l'utilizzo di un nodo Bitcoin [30] *sincronizzato* [31], perchè il nodo Lightning deve interagire con la blockchain per la gestione dei canali. Come vedremo successivamente, i canali Lightning sono strettamente collegati alla blockchain e, di conseguenza, devono basarsi sullo stato più aggiornato della stessa: in caso contrario il nodo Lightning non potrebbe confermare lo stato dei canali e dei pagamenti, compromettendone il funzionamento.

3.2.2 Canali di pagamento

Un canale di pagamento *è una relazione finanziaria tra due nodi su Lightning Network* [32] che permette a due *peer* di effettuare illimitati pagamenti, senza dover scrivere ogni operazione sulla blockchain. Solo l'apertura (**funding**) e la chiusura del canale richiedono una scrittura *on-chain*, mentre le transazioni intermedie no. Questo approccio riduce il carico su Bitcoin, diminuendo significativamente i costi delle transazioni *on-chain*, poiché vengono effettuate solo due scritture effettive.
Un canale di pagamento si basa su uno **smart-contract multi-signature** *2-di-2* [33] tra due parti che detengono bitcoin. Sebbene i fondi siano detenuti in modo cooperativo dalle due parti sul piano della blockchain, ogni parte ne possiede una porzione e ne tiene traccia localmente [34]. Ad esempio, **Alice** chiede a **Bob** di aprire un canale, nel quale **Alice** impegna 0.9 bitcoin e **Bob** 0.1 bitcoin. Il canale avrà una capacità totale di 1 bitcoin e, quando **Alice** avrà pagato le commissioni *on-chain* per l'apertura del canale, le due parti potranno scambiarsi illimitatamente questa somma fino alla chiusura del canale, che può avvenire consensualmente o in modo unilaterale (force-close), con una transazione che aggiorna i saldi sulla blockchain di Bitcoin.

Commitment transactions

Le **commitment transaction** sono transazioni **off-chain** che si scambiano i nodi Lightning e rappresentano la corretta allocazione corrente dei fondi all'interno del canale, rappresentati da output futuri sulla blockchain di Bitcoin. [35]
Firmando le *commitment transactions*, entrambi i partner *si impegnano reciprocamente* a mantenere il saldo attuale e a consentire il recupero dei propri fondi all'occorrenza, in questo modo entrambe le parti sono protette in caso di:

- Disconnessione involontaria del proprio nodo Lightnig
- Rifiuto di collaborare della controparte
- Tentativi di imbroglio da parte della controparte

Il meccanismo della *commitment transaction* permette ad **Alice** e **Bob** di non doversi fidare direttamente l'uno dell'altro, ma di affidarsi solamente al protocollo, poiché la *commitment transaction* è **asimmetrica** in virtù di due meccanismi di sicurezza, [32] ovvero:

- **timelock**
- **revocation secret** (o *revoacation key*)

Timelock

Il **timelock** è un numero di blocchi da attendere – ovvero la misura *del tempo* sulla blockchain di Bitcoin – prima che il nodo, che possiede quella *commitment transaction*, possa effettivamente spendere quello specifico output. [32] [36]

Revocation secret

Se un nodo pubblica una *commitment transaction* obsoleta, la controparte di quel canale può utilizzare il **revocation secret** per invalidarla e

prenderne **tutti i fondi**, inclusi quelli dell'utente malevolo.
Questa operazione dev'essere svolta dal nodo onesto, prima che il timelock scada per il nodo malevolo, rendendo il meccanismo punitivo un deterrente efficace contro comportamenti scorretti, permettendo ai due nodi di non doversi fidare l'uno dell'altro, poichè, finché entrambi seguono il protocollo, non hanno il rischio di perdere fondi. Ogni volta che viene creata una nuova commitment transaction, i due nodi si inviano il *revocation secret* della precedente commitment transaction, in modo che entrambe le parti siano pronte a punirsi in caso di tentativo di imbroglio. [32] [36]

3.3 Gossip protocol

Per conoscere la topologia della rete, i nodi Lightning utilizzano il **Gossip Protocol** indicato nel **BOLT #7** [37]. Questo protocollo permette ai nodi di comunicare tra loro e scoprire altri nodi e canali, per poter costruire un grafo della rete.

3.3.1 Messaggi del gossip protocol

I messaggi utili ad un nodo per costruire il proprio grafo sono:

- `node_announcement`
- `channel_announcement`
- `channel_update`

node_announcement

Comunica la presenza di un nuovo nodo; grazie a questo messaggio un nodo può rilevare nuovi nodi all'interno del network e aggiungerli al proprio grafo.

channel_announcement

Comunica la presenza di un nuovo canale e include i dati che collegano i bitcoin *on-chain* ai nodi Lightning associati a quel canale. Questo messaggio è essenziale per aggiornare il proprio grafo con nuovi canali utili al *pathfinding*.

channel_update

Dopo l'annuncio di un canale, entrambi i nodi coinvolti (in modo indipendente) inviano al network informazioni sul canale, come le *commissioni per il routing* oppure il parametro ctlv_expiry_delta (serve a garantire che il

nodo abbia tempo sufficiente per completare la propria parte del processo di inoltro del pagamento).

3.3.2 Consenso del grafo

Poiché non esiste un consenso generale sul grafo della rete (come invece esiste per Bitcoin, ovvero la blockchain) e nemmeno un'autorità centrale o un coordinatore, l'istanza del grafo sarà potenzialmente diversa per ogni nodo del network. Inoltre, alcuni nodi potrebbero anche partecipare alla rete decidendo di **non** comunicare la loro presenza o i loro canali, quindi la rete sarà sempre più estesa rispetto a quella determinata dai dati *annunciati* nei messaggi pubblici.

3.4 La gestione di un canale

3.4.1 Apertura

La creazione del canale avviene tramite la **funding transaction**, ovvero una transazione *on-chain* che "blocca" una determinata quantità di bitcoin all'interno di un indirizzo *multi-signature 2-di-2*, ovvero un conto cointestato da **Alice** e **Bob** rappresentato *on-chain* tramite una *UTXO*. Prima di inviare questa transazione on-chain **Alice** chiede a **Bob** di firmare una **commitment transaction**, ovvero una transazione *off-chain* firmata da entrambi che codifica il saldo corrente del canale. [32]

3.4.2 Chiusura cooperativa

Il processo di chiusura di un canale Lightning avviene *on-chain*, in quanto solo la *funding transaction* e la transazione di chiusura vengono scritte sulla blockchain. Quando **Alice** decide di chiudere il canale, informa **Bob** della propria intenzione; a questo punto i loro nodi interrompono l'aggiornamento delle *commitment transactions* e, di comune accordo, pubblicano insieme una transazione *on-chain* di chiusura, simile ad una commitment transaction, ma senza *timelock* e *revocation secret*. Le commissioni per la transazione *on-chain* di chiusura sono generalmente concordate tra le parti; in alcuni casi la parte che ha originato l'apertura del canale può assumersi l'intero costo. Quando la UTXO associata al canale è stata spesa, il canale viene considerato definitivamente chiuso. [32]

3.4.3 Force-close

Quando **Alice** decide di chiudere unilateralmente un canale senza il consenso di **Bob**, la *commitment transaction* include un *timelock* che impedisce ad **Alice** di spendere immediatamente l'output associato a **Bob**. Questo timelock obbliga **Alice** ad attendere un certo numero di blocchi (specificato alla creazione del canale) prima di poter spendere tali fondi, affinché **Bob** abbia il tempo di contestare eventuali tentativi di chiusura fraudolenta. Il timelock offre un vantaggio reciproco poiché **Alice**, che avvia la chiusura forzata del canale (**force-close**), dovrà attendere lo sblocco del timelock dei fondi di **Bob**, mentre **Bob** potrà spendere il proprio output immediatamente grazie al *revocation secret*. Questo dà **Bob** il tempo di rilevare un inganno da parte di **Alice** e pubblicare una transazione punitiva prima che il timelock per **Alice** scada, permettendogli di prendere anche i fondi di **Alice**. [32]

3.5 Routing

Un canale di pagamento può essere utilizzato anche da altri nodi per **instradare** un pagamento attraverso quel canale, consentendo ad un mittente di inviare fondi ad un destinatario, senza la necessità di un canale diretto tra loro. Immaginiamo quattro nodi Lightning: **Alice**, **Bob**, **Carl** e **Dina**. Supponiamo esistano i seguenti canali:

- **Alice** $\longleftrightarrow$ **Bob**
- **Bob** $\longleftrightarrow$ **Carl**
- **Carl** $\longleftrightarrow$ **Dina**

In questo caso **Alice** non ha un canale aperto con **Dina**, ma può chiedere a **Bob** di instradare un pagamento verso **Carl**. Quello che succede è che **Alice** paga **Bob**, che paga **Carl**, che a sua volta paga **Dina**. Questo è possibile grazie ad uno *smart-contract*, ovvero **HTLC (Hashed Time-Locked Contracts)**.

Il percorso effettivo sarà:

$$Alice \longrightarrow Bob \longrightarrow Carl \longrightarrow Dina$$

3.5.1 Pathfinding

Prima di poter inviare il pagamento, il nodo di **Alice** deve identificare un percorso disponibile verso il nodo di **Dina**: questo processo di ricerca è chiamato **pathfinding** e implica l'esplorazione del grafo dei canali per identificare un percorso (*path*). Il *pathfinding* è gestito dal mittente e quindi ogni nodo può implementare il proprio algoritmo di ricerca. Per questo motivo esistono diverse implementazioni di algoritmi di pathfinding, anche se la maggior parte delle implementazioni più diffuse utilizzano l'algoritmo di Dijkstra o una sua versione modificata. Questa scelta non è vincolante in quanto il pathfinding non rientra nei **BOLT**.[35][38]

Implementazione	Algoritmo di Pathfinding
LND	Algoritmo di Dijkstra modificato
CLN	Algoritmo di Dijkstra standard
LDK	Algoritmo di Dijkstra standard
Eclair	Algoritmo K-shortest paths di Yen

Tabella 3.1: Algoritmi di pathfinding utilizzati da alcune implementazioni di nodi Lightning

3.5.2 HTLC e Onion Routing

Gli **HTLC (Hashed Time-Locked Contracts)** sono *smart-contract* che permettono l'instradamento dei pagamenti attraverso più canali garantendo sicurezza e atomicità. Questo sistema si basa sulla creazione di una **preimage**, ovvero un *segreto* r che viene creato dal destinatario (**Dina**) e rappresenta la chiave che sblocca il *timelock* del HTLC. Senza la **preimage**, l'HTLC non può essere sbloccato e il pagamento non viene riscosso; in questo modo, i nodi intermedi non hanno modo di trattenere fondi in maniera fraudolenta, poiché la preimage viene rilasciata da **Dina** solo **al termine** del pagamento.

Update message

Il mittente (**Alice**) crea un messaggio m chiamato `update_add_htlc` che contiene tra le altre cose:

- Quantità di fondi del pagamento (`amount_msat`)
- Tempo di scadenza del pagamento (`cltv_expiry_delta`)
- Hash H del pagamento (`payment_hash`)
- Pacchetto **onion** (ovvero istruzioni cifrate che indicano a ciascun nodo intermedio a quale nodo inoltrare il messaggio m)

3.5.3 Fasi di un pagamento

Le fasi di un pagamento tra **Alice** e **Dina** sono le seguenti:

1. **Dina** genera una *preimage* r e ne calcola l'hash $H = \text{SHA256}(r)$ che poi invia ad **Alice**.

2. **Alice** avvia il pagamento creando un $HTLC$ con **Bob**, utilizzando H come `payment_hash`, e gli invia un `update_add_htlc` m.

3. **Bob**, ricevuto il messaggio m, utilizza le informazioni per creare un $HTLC$ con **Carl** e inoltrargli il pagamento.

4. **Carl** ripete lo stesso procedimento con **Dina**, creando un $HTLC$ a sua volta con `payment_hash` H.

5. **Dina** nota che il pagamento è destinato a lei poiché `payment_hash` è H e rivela a **Carl** la *preimage* r, sbloccando così l'$HTLC$.

6. **Carl**, ottenuta la *preimage*, la utilizza per sbloccare l'$HTLC$ con **Bob** e gli invia r.

7. **Bob** ripete il passaggio con **Alice**, inviandole r.

8. **Alice**, ora in possesso di r, sblocca la sua $HTLC$ con **Bob**. Il pagamento è stato **completato**.

Capitolo 4

Data Retrieval

4.1 Nodo Bitcoin

A cosa serve un nodo Bitcoin?

Un nodo Bitcoin è l'entità che gestisce le regole di consenso distribuito della rete, validando i blocchi e le firme delle transazioni, rifiutando blocchi e transazioni che non rispettano il protocollo. Un **full node**, inoltre, conserva l'*intero storico delle transazioni* eseguite, oltre allo *UTXO set* corrente, contribuendo alla resilienza e immutabilità, nonché all'incensurabilità della rete. Una grande quantità di nodi rende più complesso e costoso, per un attaccante, corrompere o danneggiare una quantità sufficiente di nodi per compromettere il sistema. L'utilizzo di un nodo Bitcoin permette l'esecuzione di transazioni senza intermediari, garantendo sicurezza, privacy e proteggendo da potenziali censure che potrebbero essere attuate da altri nodi. In questo modo vengono rispettati i principi di decentralizzazione che caratterizzano la rete Bitcoin.

Come accennato in precedenza, quando un utente desidera eseguire una transazione *on-chain*, la inoltra ad un nodo che, successivamente, la trasmette a tutti gli altri nodi della rete. Tuttavia, questo nodo potrebbe essere in malafede e potrebbe decidere di censurare questa specifica transazione rifiutandosi di trasmetterla al network. Quindi, per garantire che la transazione

venga effettivamente trasmessa alla rete e possa essere inclusa in un blocco, è necessario utilizzare un proprio nodo Bitcoin.

4.1.1 Tipi di nodo Bitcoin

Esistono tre tipi di nodo Bitcoin:

- Full node
- Pruned node
- SPV node

Full Node

È un nodo completo, salva localmente l'intero storico delle transazioni a partire dal blocco genesi. Può validare nuove transazioni poiché possiede lo UTXO set corrente e può verificare tutte le transazioni passate.

Pruned Node

Ha un parametro di configurazione che permette di limitare lo spazio su disco occupato, rimuovendo le informazioni relative ai blocchi più vecchi. Può validare nuove transazioni poiché possiede lo UTXO set corrente, ma non può verificare le transazioni che ha rimosso dal suo storico: per quelle transazioni deve riporre fiducia in un full node.

SPV Node

È un tipo di nodo estremamente leggero, ideale per gli smartphone, poiché non scarica l'intero storico delle transazioni e non conserva l'intero set di UTXO. Tuttavia, per verificare la validità delle transazioni, deve fare affidamento su un full node. Un *SPV node* può solo verificare se una transazione è confermata, ma per farlo deve chiedere delle informazioni ai full node (*Merkle Proof* [39]) per confermare che una specifica transazione sia inclusa in un determinato blocco.

Tipo di Nodo	Storico	UTXO set	Verifica nuove transazioni
Full Node	Completo	Sì	Sì
Pruned Node	Limitato	Sì	Sì
SPV Node	Solo header	Nessuno	Parzialmente

Tabella 4.1: Caratteristiche dei nodi Bitcoin

4.1.2 Bitcoin Core

Per installare un full-node sul mio server ho deciso di utilizzare l'implementazione di **Bitcoin Core** [30], ovvero l'implementazione ufficiale e più diffusa di un nodo Bitcoin. Dopo aver scaricato il software dal sito di Bitcoin Core, ho configurato il nodo attraverso la modifica del file di configurazione `bitcoin.conf`.

```
server=1
txindex=1
daemon=1
maxconnections=20

rpcport=8332
rpcbind=0.0.0.0
rpcallowip=127.0.0.1
rpcallowip=192.0.0.0/8
rpcworkqueue=160

zmqpubrawblock=tcp://0.0.0.0:28332
zmqpubrawtx=tcp://0.0.0.0:28333
zmqpubhashblock=tcp://0.0.0.0:28334
whitelist=127.0.0.1
whitelist=192.168.1.0/24

proxy=127.0.0.1:9050
listen=1
listenonion=1
bind=127.0.0.1
onlynet=onion

addnode=bipshow4n2uhgfb7my5ddmeqn3ysxgtsfwlzp2yhqg6ex7z7guzkkwqd.onion:8333
```

Figura 4.1: File di configurazione del nodo Bitcoin

TOR

Per aumentare la privacy, ho configurato il nodo affinché utilizzi **TOR**, rendendolo accessibile esclusivamente tramite connessioni **onion**, grazie al parametro `onlynet=onion`. La maggior parte dei nodi all'interno del network utilizza questa impostazione per evitare di esporre il proprio indirizzo IP, in questo modo si ottiene un ottimo livello di privacy.

Esecuzione del nodo Bitcoin

Una volta installato il nodo, ho eseguito il comando `bitcoind -daemon` per avviarlo; in questo modo il nodo ha iniziato a scaricare la blockchain per sincronizzarsi con il network. La sincronizzazione ha impiegato circa una settimana di tempo per essere completata, poiché la dimensione della blockchain di Bitcoin supera i 700 GB e il nodo ha dovuto scaricare tutti i blocchi, verificandone la validità. Sebbene il processo non imponga un carico eccessivo al sistema, richiede comunque un tempo considerevole. Una volta terminata la sincronizzazione, il nodo è pronto a svolgere tutte le sue funzioni di validazione di nuovi blocchi e transazioni, oltre a mantenere una copia della blockchain.
Eseguendo il comando `bitcoin-cli getblockchaininfo` otteniamo lo stato attuale della blockchain secondo il nodo.


```
{
  "chain": "main",
  "blocks": 886743,
  "headers": 886743,
  "bestblockhash": "00000000000000000001d47d5706caf15fd835cfbe4a190192ec9b76ba67d24c",
  "difficulty": 110568428300952.7,
  "time": 1741365117,
  "mediantime": 1741362206,
  "verificationprogress": 0.9999955440541821,
  "initialblockdownload": false,
  "chainwork": "0000000000000000000000000000000000000000000b2d9eb1c886e9147bf75d810",
  "size_on_disk": 731563210235,
  "pruned": false,
  "warnings": ""
}
```


Figura 4.2: Output del comando `bitcoin-cli getblockchaininfo`

4.2 Nodo Lightning

4.2.1 Utilità di un nodo Lightning

Per studiare il grafo di Lightning Network e analizzarne le proprietà, è necessario avere un nodo Lightning attivo e aggiornato. Ogni nodo Lightning ha una visione parziale del network; poiché non esiste un consenso diffuso sulla forma del grafo, ho deciso di utilizzare un'architettura scalabile che mi permettesse di utilizzare diversi nodi Lightning contemporaneamente, collegandoli tutti allo stesso nodo Bitcoin.

4.2.2 Architettura e configurazione

Questa architettura utilizza l'implementazione **LND** (*Lightning Network Daemon*) in un container, tramite l'immagine *Docker* di Lightning Labs, oltre ad un sistema di network *Docker* e una VPN, ovvero *Tailscale*. Per la gestione tramite interfaccia grafica ho utilizzato un container con *Ride The Lightning* (**RTL**). Anche LND utilizza un file di configurazione come Bitcoin Core, chiamato `lnd.conf`.

Al primo avvio, il nodo chiede di creare un wallet e di impostare una password per sbloccarlo. Per comodità ho configurato un file all'interno del container per sbloccare automaticamente il wallet tramite `wallet-unlock-allow-create=true`. In generale, una volta sbloccato il wallet del nodo, sarà possibile gestire la liquidità e i canali su Lightning, oltre a poter instradare pagamenti per altri utenti ed essere remunerati con le *routing fee*.

LNDConnect

Ho configurato **LNDConnect** su uno dei miei due nodi Lightning. **LNDConnect** è uno strumento che consente di creare un URI o un QR Code per stabilire una connessione sicura con un altro dispositivo. In questo modo può essere utilizzare il nodo Lightning attraverso l'uso dell'app *BitBanana*.

```
[Application Options]
alias=NODINO_1
; Automatically unlock the wallet after lnd starts up.
wallet-unlock-password-file=root/.lnd/password.txt
wallet-unlock-allow-create=true

listen=:9735

rpclisten=0.0.0.0:10009

tlsextraip=192.168.1.204
tlsextradomain=lnd-1.mammut-tetra.ts.net

restlisten=0.0.0.0:8000
tlsextradomain=0.0.0.0
tlsextraip=0.0.0.0
tlsautorefresh=1
tlsdisableautofill=false

debuglevel=info

[Bitcoin]
bitcoin.mainnet=1
bitcoin.node=bitcoind

[Bitcoind]
bitcoind.rpccookie=root/.bitcoin/.cookie
bitcoind.rpchost=192.168.1.204:8332
bitcoind.zmqpubrawblock=tcp://192.168.1.204:28332
bitcoind.zmqpubrawtx=tcp://192.168.1.204:28333

[tor]
tor.active=true
tor.socks=192.168.1.204:9050
tor.control=192.168.1.204:9051
tor.streamisolation=true
tor.v3=true
```

Figura 4.3: File di configurazione del nodo Lightning

4.3 Raccolta dati

4.3.1 Grafo di Lightning Network

L'implementazione **LND** di un nodo Lightning consente di esportare il grafo in formato *JSON*. Grazie a questa funzionalità ho potuto impostare un *job* ricorrente, tramite *systemd* e un *timer*, per salvare automaticamente il grafo, generando un grafo giornaliero per ogni nodo. Utilizzando il nodo per un periodo prolungato, è possibile analizzare l'evoluzione del grafo di Lightning: nel mio caso, ho potuto osservare variazioni nell'arco di un paio di mesi.

```
[Unit]
Description=Run daily tasks for LND nodes and backup

[Service]
Type=oneshot
ExecStart=/usr/local/bin/lnd_backup.sh

[Install]
WantedBy=multi-user.target
```

Figura 4.4: Configurazione di un job ricorrente con systemd

Il *job* è configurato in modo tale che estragga il grafo dai due nodi Lightning attraverso il comando `lncli describegraph`. Questo job viene eseguito nello stesso momento su entrambi i nodi e viene salvato il log all'interno di un apposito file. Grazie al *mounting* dei volumi Docker [40], il grafo viene archiviato in un file *JSON*, all'interno di una cartella esterna ai containers dei nodi, per avere persistenza dei dati.

4.3.2 Timemachine

Un nodo Lightning non salva lo storico del grafo, tiene solo in memoria lo stato attuale del network, pertanto non è possibile utilizzare un nodo Lightning per ottenere informazioni sullo stato passato della rete. Per ottenere informazioni in merito allo stato passato del network, ho dovuto utilizzare

```
#!/bin/bash

# Calcolare la data
DATE=$(date "+%Y-%m-%d_%H-%M-%S")

# Esegui il comando per lnd-1
/usr/bin/docker exec lnd-1 sh -c "lncli describegraph > /root/.lnd/graph-snapshot/lnd_1_graph_$DATE.json"

# Esegui il comando per lnd-2
/usr/bin/docker exec lnd-2 sh -c "lncli describegraph > /root/.lnd/graph-snapshot/lnd_2_graph_$DATE.json"

# Log del backup
echo "Backup $(date "+%Y-%m-%d %H:%M:%S")" >> /home/samoggino/lnd_backup.log
```

Figura 4.5: Estrazione del grafo con `lncli describegraph`

[41] uno strumento in *Python*, che permette di ricostruire il grafo di Lightning Network a partire dai messaggi di gossip raccolti fino ad un determinato momento. Questo strumento è stato sviluppato nel 2020 da Christian Decker (uno dei principali sviluppatori di Bitcoin e di Lightning Network, nonché scrittore di diversi articoli scientifici da cui ho studiato per questa tesi) e raccoglie i dati di gossip tra i nodi, ovvero le informazioni che utilizzano per costruire il proprio grafo. Utilizzando questo strumento mi è stato possibile studiare lo stato del network anche in tempi passati. Ciò sarebbe stato impossibile utilizzando solamente un nodo Lightning, poiché, come detto in precedenza, quest'ultimo non salva nel tempo lo stato della rete, a differenza di come funziona un nodo Bitcoin, dove la cronologia delle transazioni è salvata su disco e sarà sempre accessibile nel tempo. Nel repository di **Timemachine** sono presenti i `gossip_store`, ovvero file compressi che contengono tutti i gossip message. A partire da questi, *Timemachine* riesce a ricreare l'istanza di un network in un momento specifico, simulando l'utilizzo di un nodo Lightning attivo in quel preciso istante. Tuttavia, poiché il nodo con cui sono stati salvati utilizzava l'implementazione di *Core Lightning*, ho dovuto scrivere una serie di script Python per convertire il formato del grafo da quello di **CLN** a quello del mio nodo, ovvero **LND**, per confrontarli nello stesso modo e con le stesse metriche.

Capitolo 5

Analisi del grafo di Lightning Network

5.1 Introduzione e strumenti utilizzati

Nei capitoli precedenti ho approfondito l'architettura di Bitcoin e il funzionamento di Lightning Network, mostrando il ruolo dei nodi e dei canali di pagamento. Per comprendere meglio l'efficacia e le potenzialità di Lightning, oltre ai suoi punti deboli, è fondamentale analizzarne la struttura topologica: il modo in cui sono connessi i nodi tra loro, la distribuzione dei nodi e il grado di decentralizzazione del sistema. Lo studio di Lightning Network come grafo permette di applicare **tecniche di analisi delle reti complesse** per individuare hub oppure pattern, che si sono susseguiti negli anni di sviluppo e di crescita del network. Questo capitolo descrive la mia analisi della struttura di Lightning Network, attraverso strumenti di *teoria dei grafi*. Per effettuare un'analisi approfondita sull'evoluzione e sulle caratteristiche di Lightning, ho utilizzato in particolare due software:

- Gephi
- Cytoscape

Questi programmi mi hanno aiutato in diversi modi. Gephi mi ha permesso di osservare la struttura del grafo grazie ai suoi layout e all'estetica dell'applicazione. Cytoscape, invece, mi ha aiutato generando le metriche utili ad analizzare il network come una normale rete complessa. Per eseguire la maggior parte delle operazioni ho creato script personalizzati in Python, in modo da poter automatizzare i processi di Cytoscape.

Ad oggi Lightning Network è una rete molto estesa, con più di 16 mila nodi e quasi 50 mila canali. La capacità complessiva è quasi 5 mila bitcoin, una somma che, al momento attuale, corrisponde a poco meno di 400 milioni di dollari, una quantità che rappresenta ancora una percentuale bassa rispetto a tutti i bitcoin esistenti (poco più di 19.8 milioni), circa lo 0.02%.

Bisogna considerare, però, che questa tecnologia è nata nel 2018 ed è ancora in forte fase di sviluppo, quindi siamo ancora agli albori di quello che potrebbe essere un domani Lightning Network.

Nonostante questo, è comunque possibile svolgere delle analisi su questi primi anni di vita del network. Come detto in precedenza, un nodo Lightning non salva gli stati passati del network, ed è stato, quindi, necessario fare affidamento su altre fonti per la raccolta dei dati precedenti al 27 dicembre 2024.

5.1.1 Procedimento

Dopo aver estratto i dati dal nodo in formato JSON, li ho convertiti tramite alcuni miei script e, grazie all'utilizzo di **Networkx** (una libreria Python), li ho trasformati in file *graphml*, un formato che utilizza una sintassi basata su *XML*. Questo tipo di file è molto utilizzato per rappresentare i grafi, ciò mi ha permesso di estrarre un grafo e poterlo analizzare su diverse piattaforme (**Gephi** e **Cytoscape**) e con diverse librerie (**Networkx** e **Py4cytoscape**), rendendo più semplice applicare modifiche al grafo stesso e visualizzarlo in un modo esteticamente gradevole. Per la maggior parte delle metriche, ho impiegato Cytoscape, che offre uno strumento particolarmen-

te utile, cioè `analyze_network`, un tool che calcola simultaneamente molte metriche e può essere automatizzato in python, per applicarlo su diversi grafi.

5.2 Stato dell'arte

Durante il periodo di tempo analizzato, le differenze tra il nodo 1 e il nodo 2 sono state minime, nell'ordine del centinaio di nodi e canali. Questo fenomeno è legato al fatto che le informazioni si propagano in modo indipendente tra i nodi. Può quindi accadere che due nodi non vedano un gruppo di altri nodi o di canali ad essi associati. Per questo motivo, l'analisi è stata svolta interamente su uno solo dei due nodi, poiché non vi erano differenze significative tra i due.

Lo studio è stato condotto su uno snapshot del nodo del 6 febbraio 2025, in quel momento il nodo conosceva 16688 nodi e 50565 canali. Come **peso del grafo** è stato applicato l'**inverso della capacità di un canale**, poiché nel *pathfinding* vengono preferiti i canali con maggior liquidità, quindi ho deciso di dare un peso maggiore agli archi con minor liquidità.

Figura 5.1: Grafo di Lightning Network al 6 febbraio 2025

Per iniziare l'analisi ho calcolato diverse metriche, tra cui il **Clustering Coefficient (CC)** e la **Average Path Length (APL)**. Il CC misura il grado di concentrazione tra i vicini di un nodo, mentre l'APL indica la distanza media tra i nodi del grafo, ovvero quanti **hop** deve fare mediamente

un nodo per raggiungerne un altro. Il valore di **CC** calcolato è stato 0.126, mentre l'**APL** è risultata essere 3.91. Queste due metriche sono importanti perché la loro relazione può fornire indicazioni sul comportamento e sul tipo di struttura del network.

5.2.1 CC e APL: ipotesi di Small-World

I grafi di tipo Small-World hanno un alto valore di CC e un basso APL. Questo implica forti connessioni tra i nodi locali, ma anche che questi possono essere raggiunti rapidamente attraverso la rete globale tramite pochi hop. Nel caso di Lightning Network, i valori ottenuti suggeriscono che la rete potrebbe essere una **Small-World**. Un parametro utile per verificare la validità di questa ipotesi è **sigma**, che rappresenta il rapporto tra il CC e l'APL della rete in analisi e quelli di una rete casuale con lo stesso numero di nodi e archi. La formula per ottenere **sigma** è la seguente:

$$\sigma = \frac{(CC_{LN}/CC_{rand})}{(APL_{LN}/APL_{rand})}$$

Dove:

- **APL_LN** è l'Average Path Length di Lightning Network
- **CC_LN** è il Clustering Coefficient di Lightning Network
- **APL_rand** è l'Average Path Length del grafo generato casualmente
- **CC_rand** è il Clustering Coefficient del grafo generato casualmente

Il valore di **sigma** σ permette di comprendere quanto Lightning differisca in termini di interconnessioni e struttura globale. Un *sigma* $\sigma \gg 1$ mostra una grande differenza nell'organizzazione tra un grafo generato casualmente e un grafo reale, come avviene tipicamente nelle **Small-World**, caratterizzate da *forti connessioni locali* e *brevi percorsi tra nodi distanti.*[42]. Questo comportamento è tipico dei social network, dove si verifica l'effetto *sei gradi di separazione*, secondo cui ogni individuo può essere collegato a qualsiasi

altro nella rete tramite una catena di conoscenze che non supera i sei hop. [43]

5.2.2 Scale-Free e powerlaw

Poiché la rete presentava le caratteristiche tipiche delle Small-World, ho deciso di approfondire ulteriormente l'analisi, cercando di verificare se il grafo seguisse anche una **legge di potenza** (**powerlaw**), ovvero una distribuzione del grado dei nodi che segue una legge del tipo:

$$P(k) \sim k^{-\alpha}, \quad k \geq x_{\min}$$

Dove:

- $P(k)$ è la frequenza dei nodi con grado **k**
- k è il grado di un nodo, ovvero il numero di canali che gestisce
- x_{min} è il **grado minimo** oltre il quale la distribuzione $P(k)$ segue la *legge di potenza*
- α è l'*esponente* della legge di potenza e rappresenta la **forma** della distribuzione

Utilizzando un mio script python e le librerie **matplotlib** e **plotly**, ho creato il grafico della *Degree Distribution* 5.2. Successivamente, ho utilizzato la libreria **powerlaw** per verificare che la distribuzione seguisse una *legge di potenza*, confermando la mia ipotesi iniziale. Inoltre, ho calcolato l'esponente **alfa** α. Le reti con un valore di *alfa* compreso tra 2 e 3 possono essere classificate come **Scale-Free**, ovvero reti caratterizzate dalla presenza di pochi nodi altamente connessi, detti **hub**, e da un gran numero di nodi con gradi bassi, cioè con pochi canali.

Nell'analisi condotta per l'anno 2025, risultano $x_{min} = 36$ e un $\alpha = 2.367$. Questo configura una rete Scale-Free in cui la Degree Distribution segue una legge di potenza per valori del degree $k \geq 36$. Ciò conferma che la rete è

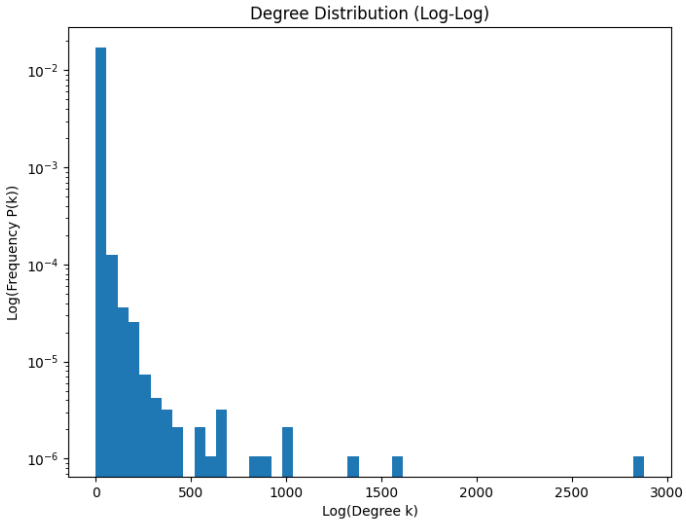

Figura 5.2: Degree Distribution di Lightning Network

caratterizzata da pochi nodi che gestiscono la maggior parte dei canali del network. Questi **hub** favoriscono una connettività altamente efficiente, ma allo stesso tempo possono rendere il network più vulnerabile ad attacchi, poiché danni o guasti a questi hub potrebbero causare una significativa diminuzione dell'efficienza e della affidabilità complessiva della rete. Per verificare il trend nel corso degli anni, ho condotto la stessa analisi anche sui grafi degli anni passati e ho notato che il trend tende ad una centralizzazione, poiché x_{min} e α sono in crescita da quando è nato Lightning Network.

Come si può osservare dalla tabella 5.1 e dal grafico dell'andamento di x_{min} e α nell'immagine 5.3, nel corso degli anni c'è stato un aumento significativo di entrambi i parametri. Questo suggerisce una possibile dinamica di centralizzazione del grafo, con una crescente tendenza ad avere pochi hub molto connessi, insieme ad un elevato numero di nodi con pochi canali.

Anno	α	$x_{\min}$
2019	2.123	9
2020	2.086	11
2021	2.108	11
2022	2.251	20
2023	2.254	31
2024	2.344	36
2025	2.366	36

Tabella 5.1: Evoluzione dell'esponente α e del valore $x_{\min}$ della distribuzione dei gradi nella Lightning Network (2019-2025).

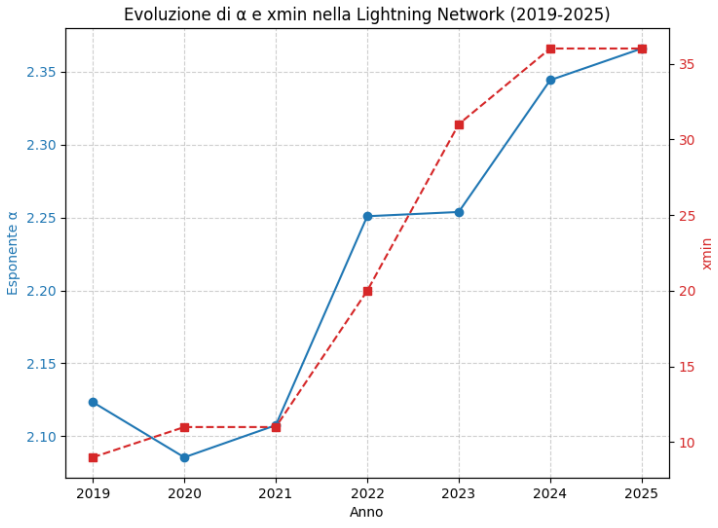

Figura 5.3: Andamento di x_{min} e α nel tempo

5.2.3 Possibile centralizzazione

Per approfondire l'analisi della possibile centralizzazione, ho deciso di visualizzare la relazione tra **Degree** e **Betweenness Centrality** 5.5, per la *componente connessa* più grande 5.4.

Figura 5.4: Componente connessa più grande del 2025

In questo modo, è possibile osservare come i nodi con maggiori canali (più alto degree) si distribuiscono rispetto alla loro *betweenness centrality*, che indica la frequenza con cui un nodo appare sui percorsi più brevi tra tutte le coppie. Analizzando la componente connessa più grande vengono rimossi dei nodi e dei canali, portando il numero di nodi e canali rispettivamente a 14899 e 48020.

Nel grafico 5.5 è possibile vedere la relazione tra Degree e Betweenness Centrality. La *correlazione* è pari a 0.9067, questo significa che i nodi con più canali sono anche nodi estremamente utilizzati. Quindi, i nodi che possiedono un **degree** più elevato — ossia con un maggior numero di connessioni —

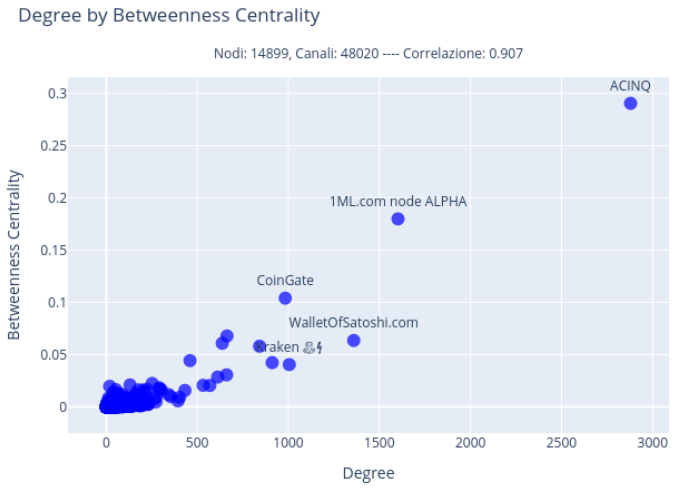

Figura 5.5: Relazione tra Degree e Betweenness Centrality

tendono a essere coinvolti frequentemente nei percorsi più brevi tra le coppie di nodi della rete. Il grafico, inoltre, evidenzia come alcuni nodi dominino sia per numero di canali, sia per la loro posizione strategica all'interno dei percorsi di pagamento. Questa alta correlazione è un'ulteriore prova che Lightning Network possiede una struttura **Scale-Free**. Il nodo di *ACINQ* risulta essere il nodo più connesso, con quasi 3000 canali.

Network Centralization (NC)

Ho calcolato anche la **Network Centralization (NC)**, ovvero una misura della concentrazione della centralità nella rete. In questo caso, la **NC** è risultata essere pari a 0.198, ciò indica un livello di centralizzazione moderato. Questo significa che, nonostante il grafo presenti alcuni nodi molto connessi, non sembra dipendere in modo eccessivo da questi, anche se la loro influenza non può essere considerata marginale.

Average Neighbors Degree (AND)

Infine ho utilizzato la metrica **Average Neighbors Degree (AND)**, che rappresenta una misura del grado medio dei vicini di ciascun nodo, cioè

il numero medio di canali di ciascun vicino. Nella situazione di partenza, il valore di **AND** è risultato essere 6.51, ovvero un valore in linea con le ipotesi di Scale-Free e Small-World.

5.3 Studio sulla centralizzazione

In questa sezione, ho utilizzato quattro etichette:

- **unfiltered**: grafo senza alcun filtro applicato.
- **min_2**: grafo con nodi aventi grado ≥ 2, ovvero almeno due canali.
- **top10**: grafo con i 10 nodi con il grado maggiore rimossi, mantenendo solo i nodi con grado ≥ 2.
- **top100**: grafo con i 100 nodi con il grado maggiore rimossi, mantenendo solo i nodi con grado ≥ 2.

5.3.1 Rimozione dei Nodi con Degree < 2

Per approfondire la possibile centralizzazione della rete, ho creato altri grafi rimuovendo i nodi con meno di 2 canali, ovvero con $Degree < 2$. In questo modo, ho considerato solo i nodi che partecipano attivamente al network, ossia quelli che consentono il routing dei pagamenti. I nodi con un solo canale, infatti, non instradano pagamenti, ma sono semplicemente utilizzatori della rete, senza apportare un contributo significativo. Dopo aver mantenuto solo la componente connessa più grande, si verifica un dimezzamento del numero dei nodi e anche una riduzione significativa dei canali.

In questo caso il Clustering Coefficient (CC) è pari a 0.246 e il APL 3.40, in questo caso la nostra σ è aumentata oltre i 1100. Possiamo notare inoltre un aumento significativo del CC e del APL. Ciò è coerente con le aspettative perché, rimuovendo i nodi poco connessi, miglioriamo la APL e soprattutto rimuoviamo una parte significativa di nodi, che portava ad una maggior dispersione. Questo comportamento è visibile anche osservando la Network Centralization (NC), poiché questa passa da un valore di 0.198 a 0.270, un aumento di circa il 36%. Anche questo aspetto è piuttosto in linea con le aspettative, perché rimuovendo i nodi più piccoli è normale che aumenti la centralità dei nodi più grandi. Lo stesso fenomeno avviene per il

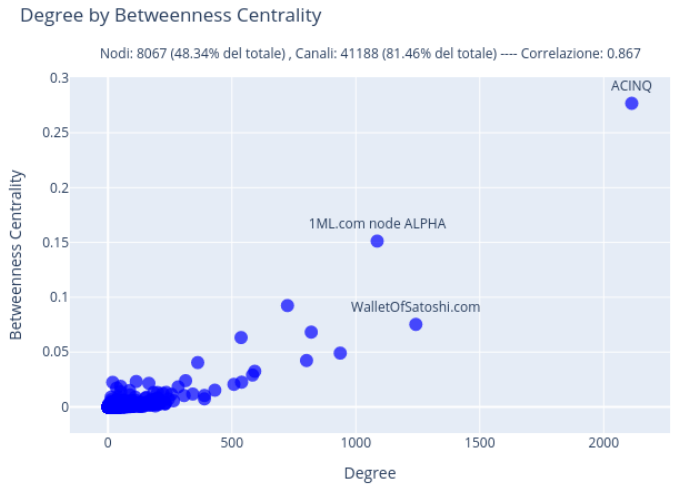

Figura 5.6: Grafo di Lightning Network con Degree ≥ 2

Average Neighbors Degree (AND), che subisce una variazione da 6.51 a 10.31, poiché rimuovendo i nodi con solo un canale, i nodi rimanenti tendono ad avere vicini più connessi tra loro, poiché i nodi con un solo canale abbassano il livello di connessioni locali in proporzione al numero di nodi.

Inoltre, osservando il grafico della Degree Distribution 5.7 possiamo notare come ACINQ, che prima era connesso a poco meno di 3000 nodi (visibile nel grafico precedente 5.2), ora è connesso a poco più di 2000 nodi. Ciò dimostra che quasi 1000 nodi del network sono connessi solo ad ACINQ o ad altri nodi che, a loro volta, sono connessi esclusivamente ad ACINQ. Questo rappresenta circa il 6% di tutti i nodi del network, un dato in linea con la differenza nelle componenti connesse che ho potuto osservare: 566 senza applicare filtri e 68 rimuovendo i nodi con un solo canale, per un totale di 500 componenti connesse in meno.

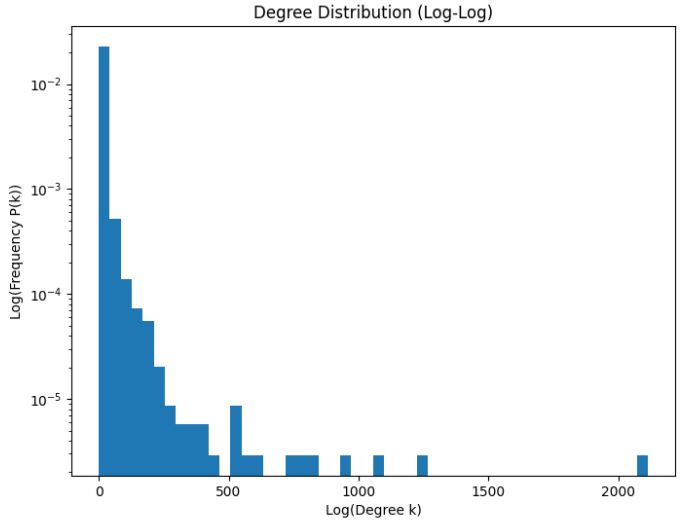

Figura 5.7: Degree Distribution di Lightning Network con Degree ≥ 2

5.3.2 Rimozione Top10

Data la sua natura di rete Scale-Free e Small-World, mi sarei aspettato che Lightning Network fosse particolarmente sensibile alla rimozione degli hub più grandi. Per verificare la resilienza del network, ho rimosso i 10 nodi più grossi e ricalcolato le metriche precedenti, continuando ad ignorare i nodi con meno di due canali. I dati ottenuti sono rappresentati nella tabella 5.2.

Nodi	7135
Canali	31957
APL	4.12
CC	0.14
NC	0.08

Tabella 5.2: Risultati della rimozione dei 10 nodi più grossi

Questi dati rappresentano un cambiamento significativo rispetto alle metriche precedenti poiché l'Average Path Length (APL) è aumentato, indicando che la rete è diventata meno efficiente nel collegare i nodi tra loro. Allo stesso tempo, il Clustering Coefficient (CC) è diminuito, dimostrando che la rete ha perso il livello di coesione iniziale, dovuto a quei 10 nodi che tendevano a formare cluster più densi. Inoltre, anche la Network Centralization (NC) è diminuita fino a 0.08, mostrando una maggior distribuzione della centralità, riducendo il peso degli hub rimanenti sulla rete globale. Infine, come si può notare nell'immagine 5.8, sono ancora presenti numerosi nodi con più di 300 canali. Questo potrebbe indicare che alcuni nodi continuano a gestire la maggior parte delle connessioni del grafo.

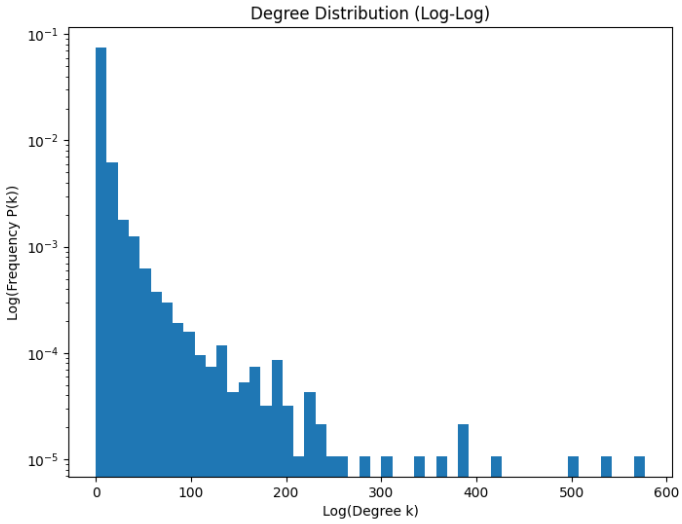

Figura 5.8: Degree Distribution di Lightning Network con i 10 nodi più grossi rimossi

5.3.3 Rimozione Top100

Per verificare ulteriormente il comportamento di Lightning Network e mostrarne la sensibilità alla rimozione degli hub più rilevanti, ho deciso di rimuovere anche la Top100, ovvero i 100 nodi più connessi. La curva della distribuzione è visibile nell'immagine 5.9.

Come era facile ipotizzare, anche in questo caso si verifica un calo del CC a 0.077. Ciò mostra quanto diventi poco clusterizzato il network ma, nonostante ciò, il dato di 5.29 di APL mostra come il livello di connessione sia rimasto decisamente alto. Anche il livello della NC si è drasticamente ridotto, passando a 0.013, come il AND, che è passato da 8.82 a 5.65, evidenziando come localmente la rete sia meno densa.

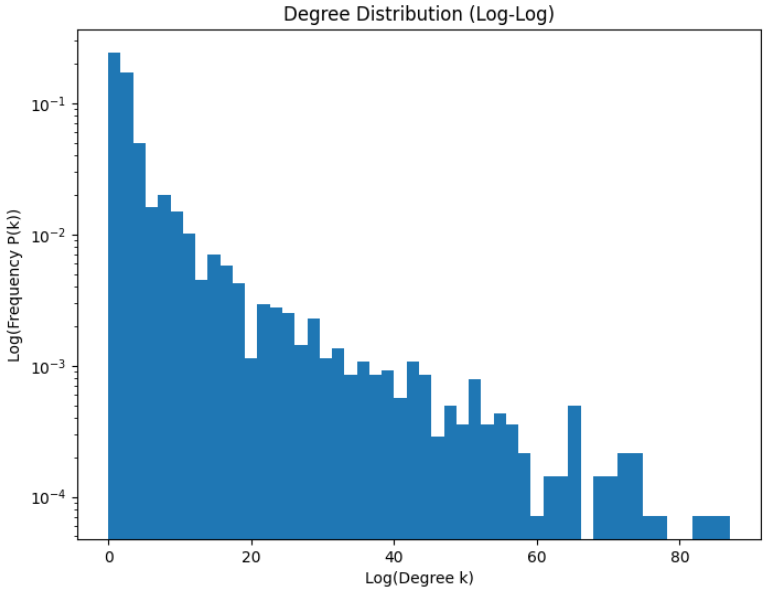

Figura 5.9: Degree Distribution di Lightning Network con i 100 nodi più grossi rimossi

5.4 Sezione grafici

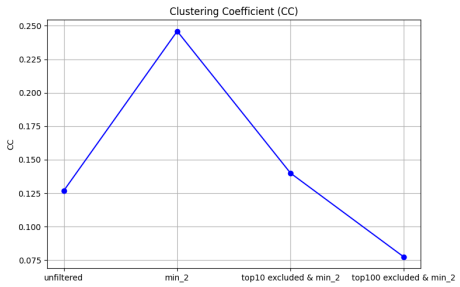

Figura 5.10: Clustering Coefficient

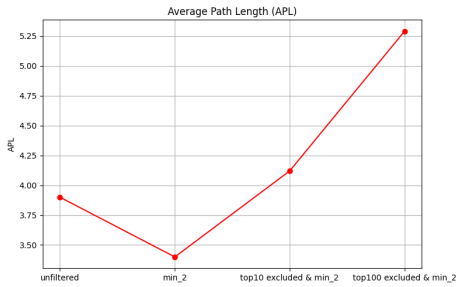

Figura 5.11: Average Path Length

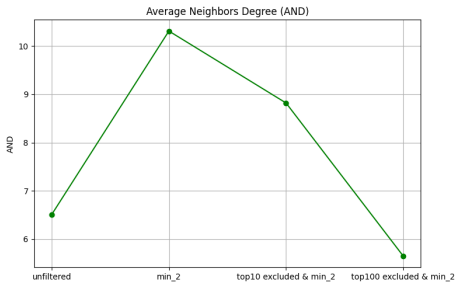

Figura 5.12: Average Neighbors Degree

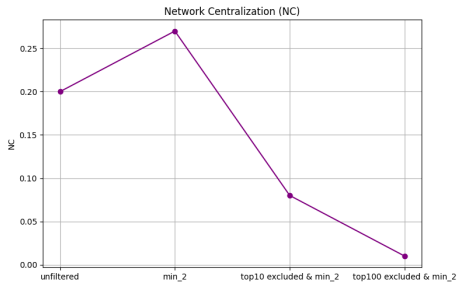

Figura 5.13: Network Centralization

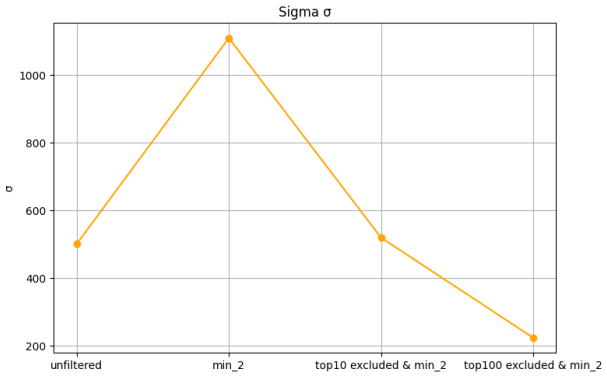

Figura 5.14: Sigma di Lightning Network

Figura 5.15: Metriche di Lightning Network nel 2025

5.5 Analisi sul passato

Ho effettuato le stesse analisi calcolando le medesime metriche, ottenendo dati molto coerenti con i dati di oggi. Questo evidenzia come i pattern osservati nel 2025 siano strettamente collegati alle caratteristiche intrinseche di Lightning Network.

Di seguito vengono presentati i risultati delle principali metriche analizzate, suddivisi per categorie di filtro:

Unfiltered

Anno	Nodi	Canali	AND	APL	CC	NC	σ	Comp. Conn.
2019	4.704	28.411	12,09	2,91	2,85E-01	0,22	1,50E+03	3
2020	5.973	29.966	10,07	3,01	3,07E-01	0,26	1,56E+03	12
2021	6.672	29.444	8,86	3,21	2,25E-01	0,19	1,08E+03	11
2022	19.151	82.605	8,69	3,49	1,12E-01	0,14	4,91E+02	76
2023	15.111	64.289	8,52	3,43	1,16E-01	0,15	5,19E+02	15
2024	17.043	51.875	6,52	3,89	1,26E-01	0,20	4,98E+02	557
2025	16.688	50.565	6,51	3,90	1,27E-01	0,20	5,01E+02	566

Tabella 5.3: Metriche principali per la categoria **unfiltered**

Almeno 2 canali

Anno	Nodi	Canali	AND	APL	CC	NC	σ	Comp. Conn.
2019	3.387	27.096	16,00	2,67	4,01E-01	0,27	2,31E+03	1
2020	3.972	27.976	14,09	2,68	4,66E-01	0,38	2,67E+03	1
2021	3.784	26.565	14,04	2,73	4,03E-01	0,31	2,26E+03	2
2022	9.746	73.271	14,18	3,02	2,10E-01	0,25	1,07E+03	17
2023	8.493	57.684	13,59	2,96	2,11E-01	0,24	1,09E+03	2
2024	8.597	43.922	10,31	3,38	2,43E-01	0,27	1,10E+03	64
2025	8.378	42.753	10,31	3,40	2,46E-01	0,27	1,11E+03	68

Tabella 5.4: Metriche principali per la categoria **min_2**

Top10 e nodi con almeno 2 canali

Anno	Nodi	Canali	AND	APL	CC	NC	σ	Comp. Conn.
2019	3.377	20.745	12,65	2,98	2,30E-01	0,12	1,18E+03	98
2020	3.962	20.711	10,80	3,19	2,18E-01	0,11	1,05E+03	125
2021	3.774	19.730	10,96	3,14	2,36E-01	0,12	1,15E+03	172
2022	9.283	51.987	11,81	3,43	1,47E-01	0,09	6,58E+02	454
2023	8.483	45.716	11,30	3,42	1,26E-01	0,07	5,65E+02	380
2024	8.587	34.010	8,74	4,10	1,35E-01	0,08	5,07E+02	729
2025	8.368	33.276	8,82	4,12	1,40E-01	0,08	5,20E+02	746

Tabella 5.5: Metriche principali per la categoria **top10**

Top100 e nodi con almeno 2 canali

Anno	Nodi	Canali	AND	APL	CC	NC	σ	Comp. Conn.
2019	3.287	6.914	5,37	4,30	4,11E-02	0,02	1,47E+02	696
2020	3.874	6.807	5,10	4,42	4,46E-02	0,02	1,55E+02	1178
2021	3.685	6.593	4,96	4,46	5,06E-02	0,02	1,74E+02	996
2022	8.261	24.838	7,82	4,24	7,89E-02	0,02	2,85E+02	1817
2023	8.393	25.063	8,07	4,15	7,46E-02	0,02	2,76E+02	2112
2024	8.498	19.007	5,64	5,29	7,58E-02	0,01	2,20E+02	1641
2025	8.279	18.494	5,65	5,29	7,74E-02	0,01	2,24E+02	1619

Tabella 5.6: Metriche principali per la categoria **top100**

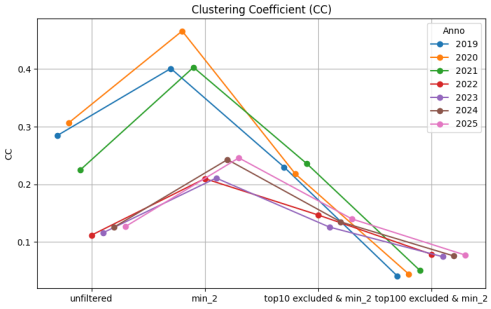

Figura 5.16: Clustering Coefficient nel tempo

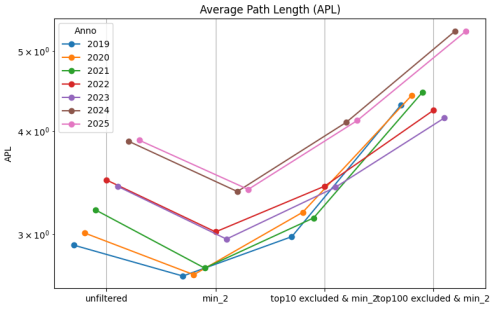

Figura 5.17: Average Path Length nel tempo

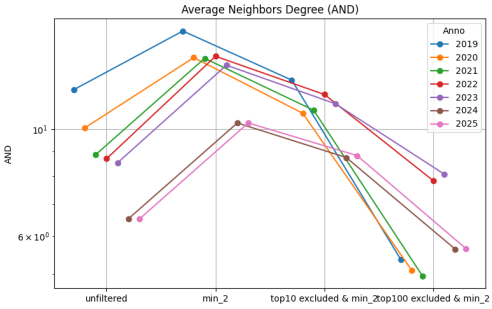

Figura 5.18: Average Neighbors Degree nel tempo

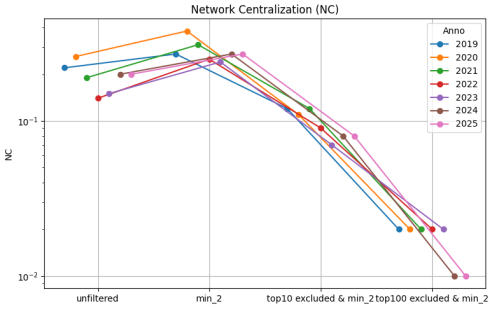

Figura 5.19: Network Centralization nel tempo

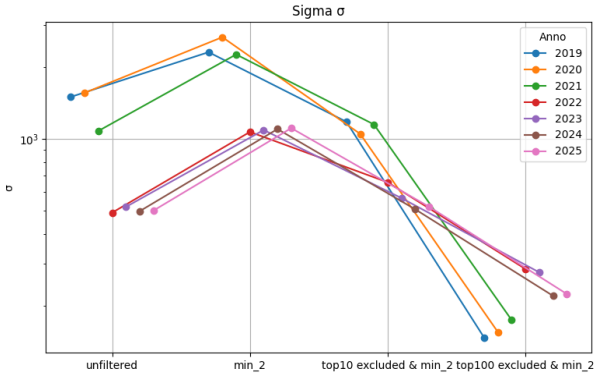

Figura 5.20: Sigma di Lightning Network nel tempo

Figura 5.21: Evoluzione delle metriche di Lightning Network nel tempo

5.6 Ragionamenti sulle analisi

Dalle analisi emerge che, nonostante il notevole aumento di nodi e canali, le caratteristiche fondamentali del network restano sostanzialmente invariate, confermando la sua natura Small-World e Scale-Free. In questo contesto, gli hub assumono un ruolo cruciale, come evidenziato dall'incremento significativo delle componenti connesse (vedi immagine 5.22) osservato dopo la rimozione di soli 10 o 100 nodi – una quota trascurabile rispetto ai circa 17.000 nodi iniziali o ai più di 8.000 nodi dotati di almeno due canali. Ciò implica che quasi la metà dei nodi dipende da una piccola parte del network, per motivi ovvi di efficienza, portando però a una minor decentralizzazione.

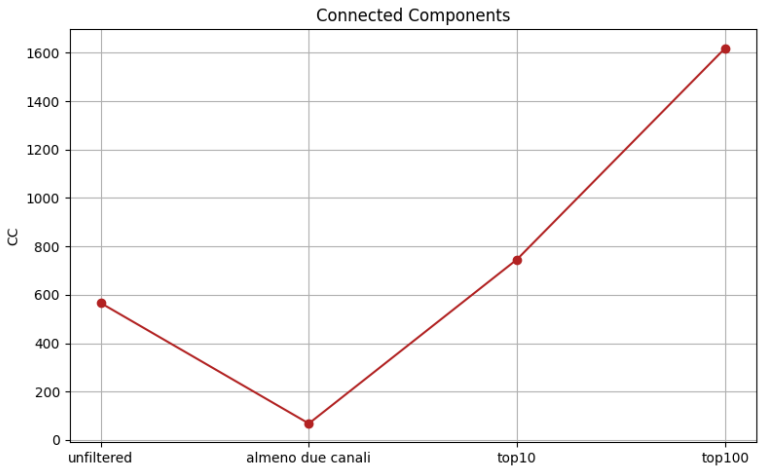

Figura 5.22: Componenti connesse nel 2025

Nonostante tutto, la situazione risulta meno critica rispetto a quello che potrebbe sembrare: anche eliminando la top100, il grafo resta comunque ben collegato, consentendo di attraversarlo in pochi hop.

Distribuzione della liquidità

Sebbene sia possibile percorrere gran parte del grafo iniziale, in realtà i percorsi tendono a coinvolgere sempre un piccolo numero di nodi; questo perchè la liquidità non è distribuita in modo omogeneo all'interno della rete. Rimuovendo la top10 e la top100, la distribuzione della liquidità all'interno dei canali risulta essere poco equilibrata, con pochi nodi che gestiscono la maggior parte dei canali (vedi immagine 5.23).

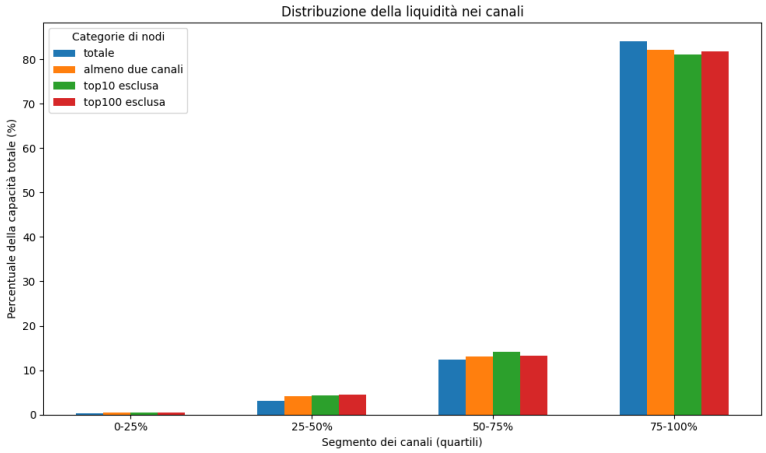

Figura 5.23: Distribuzione della liquidità dei canali

Trend della centralità

Questi dati mostrano come Lightning Network sia ancora in una fase embrionale del suo sviluppo. L'aumento nel tempo dei parametri α e x_{min} potrebbe essere indice di una possibile tendenza ad una maggiore centralizzazione, come suggerisce anche la Network Centralization (NC) nel tempo (vedi immagine 5.24).

Un valore di 0.2 per la Centralization non è molto elevato, ma è comunque significativo. Questo indica che la rete è ancora abbastanza distribuita, ma che la rete fa molto affidamento a pochi nodi centrali, come quelli di ACINQ, 1ML e WalletOfSatoshi, come visto nel precedente grafico 5.5. Inoltre, la

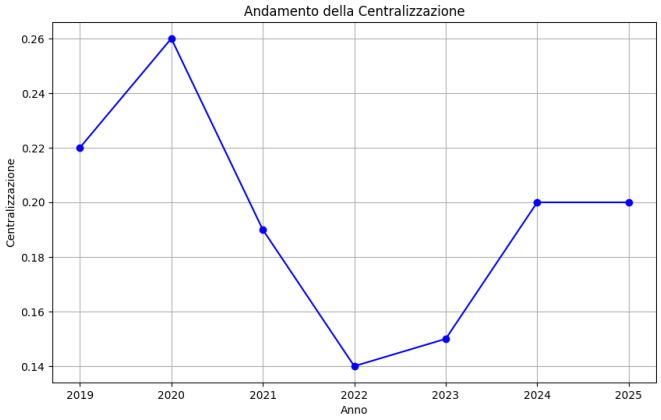

Figura 5.24: Network Centralization (NC) nel tempo

diminuzione della Network Centralization dopo la rimozione dei nodi più grossi, come mostrato nel grafico 5.19, indica che la rete è in grado di adattarsi e di mantenere una certa resilienza, anche in seguito alla rimozione di nodi centrali, non perdendo troppo in termini di efficienza, come mostrato nel grafico della APL 5.17.

5.7 Approfondimento: ACINQ, WalletOfSatoshi e 1ML

ACINQ, WalletOfSatoshi e 1ML sono tre dei nodi maggiormente connessi all'interno del network, con un numero di canali molto elevato. Questo avviene per motivi diversi.

ACINQ (Phoenix)

ACINQ gestisce Phoenix, un wallet Lightning molto apprezzato dagli utenti di Lightning Network per la facilità d'uso. L'applicazione semplifica notevolmente la gestione di un nodo Lightning, permettendo agli utenti di evitare la gestione manuale dei canali, delegando gran parte della complessità al software. Inoltre Phoenix permette lo splicing, ovvero una funzionalità che consente di aumentare (splice-in) o diminuire (splice-out) la capacità di un canale senza doverlo chiudere per poi riaprirlo. ACINQ permette anche i **Trampoline Payments**, ovvero un sistema che consente agli utenti di effettuare pagamenti anche a nodi con i quali non si ha un canale diretto, utilizzando un nodo intermediario e riducendo il carico sul nodo nello smartphone. Per svolgere questa funzionalità, Phoenix utilizza il nodo di ACINQ come Trampoline Node, aumentando complessivamente le sue connessioni. [44]

WalletOfSatoshi

A differenza di ACINQ, WalletOfSatoshi offre un'applicazione per smartphone che si concentra esclusivamente sulla gestione dei pagamenti Lightning, senza richiedere la gestione di un nodo direttamente sul dispositivo dell'utente. Non dover gestire un nodo Lightning permette una migliore esperienza utente, ma introduce un livello di centralizzazione maggiore, poiché i pagamenti vengono gestiti interamente da un unico nodo centrale, cioè il nodo di WalletOfSatoshi. Inoltre, i fondi all'interno di WalletOfSatoshi non sono

custoditi direttamente dall'utente, ma dal servizio stesso, introducendo un elemento di fiducia contrario ai principi di decentralizzazione di Bitcoin e Lightning Network.

1ML

1ML è una piattaforma che fornisce una dashboard con statistiche e informazioni sulla rete Lightning Network. Permette agli utenti di visualizzare i nodi più connessi, i canali più liquidi e altre informazioni utili per la gestione di un nodo Lightning e non solo. Grazie alla sua funzione di aggregatore di informazioni, 1ML è ampiamente utilizzato dagli utenti che vogliono ottenere una panoramica della rete, per decidere quali nodi connettersi e quali canali aprire. Questa popolarità porta a un aumento dei canali al nodo di 1ML, facendo sì che diventi uno dei nodi più connessi all'interno della rete.

5.7.1 Preferential Attachment

Quando un nuovo nodo si collega alla rete, tende a connettersi preferibilmente ai nodi più connessi. Questo accade perchè avere un canale con un nodo molto connesso consente di raggiungere un maggior numero di nodi, aumentando la scelta tra i diversi percorsi possibili per il routing di un pagamento, riducendo così le commissioni complessive per instradare un pagamento. Questo fenomeno è noto come **Preferential Attachment** ed è alla base della formazione delle reti **Scale-Free**. Queste reti si distinguono per una rapida espansione, in cui alcuni nodi ottengono un numero di connessioni molto più elevato rispetto agli altri, principalmente perché sono stati i primi a gestire un numero significativo di connessioni. Questo fenomeno, nel tempo, crea hub che gestiscono sempre più canali e liquidità, ed è alimentato dalla preferenza dei nuovi nodi di connettersi a quelli già altamente connessi, proprio come sta succedendo per ACINQ, WalletOfSatoshi e 1ML. [45]

Conclusioni e sviluppi futuri

In questa tesi ho esaminato la struttura di Bitcoin e del suo principale *layer 2*: Lightning Network. Dopo un approfondimento tecnico di entrambi, ho analizzato dettagliatamente la topologia del grafo di Lightning, studiando diversi aspetti attraverso molteplici metriche. Lightning si distingue per un'elevata di efficienza, grazie alla rapidità dei pagamenti, alla dimensione e interconnessione del grafo, mostrando, però, una preoccupante tendenza alla creazione di hub. Attualmente, i dati mostrano come pochi nodi gestiscano un elevato numero di canali e una grande liquidità, nonostante il network potrebbe continuare a funzionare ugualmente anche senza di essi. È importante evidenziare che, nonostante il grande numero di nodi, circa la metà non contribuisce attivamente ad instradare i pagamenti, essendone solo fruitore. Di conseguenza, il semplice conteggio dei nodi non rappresenta una metrica significativa. Infatti, il numero dei nodi non ha avuto un ruolo determinante nell'analisi di questo studio. Lo stesso vale per il numero dei canali, poiché, come visto in precedenza, la maggior parte della liquidità è concentrata in una piccola parte dell'intero network. Tuttavia, Lightning Network è ancora molto giovane, quindi risultano ancora diversi scenari per il suo sviluppo futuro, come l'aumento dell'adozione da parte degli utenti, ma anche da parte di aziende e imprese. Inoltre, lo sviluppo tecnico potrebbe portare ad un ottimizzazione nella gestione di un nodo, favorendo nuovi wallet come *Phoenix*, stimolando una competizione e, possibilmente, una maggior distribuzione della liquidità tra i canali.

Covenants

Anche Bitcoin è ancora giovane e, nonostante la sua evoluzione proceda con cautela per motivi di una *paranoica* sicurezza, potrebbe ricevere aggiornamenti in grado di potenziare Lightning e la scalabilità complessiva. I futuri sviluppi di Bitcoin potrebbero includere i *covenants*, meccanismi che permettono di imporre restrizioni sulle modalità di spesa di un UTXO, vincolando le transazioni future secondo regole prestabilite dall'utente che spende l'UTXO. Le proposte più discusse sono `OP_CHECKTEMPLATEVERIFY` (**CTV**) [46] e `SIGHASH_ANYPREVOUT` (**APO**) [47], oltre alla reintroduzione di `OP_CAT` (disattivato nel 2010 da Satoshi Nakamoto a causa di preoccupazioni relative alla sicurezza) [48]. Queste proposte amplierebbero le potenzialità degli script e degli smart-contract on-chain e aumenterebbero le possibilità di sviluppo per Lightning. Questi possibili aggiornamenti sono, però, oggetto di discussione da una parte della comunità degli sviluppatori di Bitcoin, perché l'applicazione dei covenants potrebbe introdurre bug o applicazioni dannose per Bitcoin. Questi sviluppatori sostengono l'*ossificazione* del protocollo Bitcoin, ovvero desiderano che non vengano introdotte ulteriori modifiche sostanziali, simili a quelle apportate con *SegWit* [23] e *Taproot* [27].

Sidechains

Esistono anche alternative a Lightning Network, come **Liquid** [49] e **Rootstock** [50], che adottano una propria gestione del consenso o si affidano a coordinatori. Queste alternative sono chiamate *sidechains*, perché svolgono la funzione di blockchain parallele, per rendere più rapide ed economiche le transazioni, favorendo la possibilità di nuove funzionalità, senza dover modificare il protocollo Bitcoin, a differenza dei *covenants*. Ad esempio Rootstock permette l'esecuzione di smart-contract analogamente ad Ethereum, pur basandosi sulla blockchain di Bitcoin.

ARK

Inoltre, nel febbraio 2024 è stato avviato lo sviluppo di **ARK**, un altro *layer 2* di Bitcoin, che consentirebbe di partecipare alla rete senza dover mantenere un nodo online, oltre a risolvere problemi tecnici di Lightning, come l'utilizzo delle VTXO invece delle UTXO, riponendo, però, fiducia in server esterni. [51]

FediMint e Cashu

Infine, esistono altri protocolli in fase embrionale, come **FediMint** e **Cashu**, implementazioni del *Chaumian ECash*. *FediMint* mira ad una gestione dei fondi in comunità federate, con la gestione dei fondi condivisa, basata interamente sulla fiducia verso la federazione. [52] *Cashu*, invece, permette di conservare i fondi fisicamente sul proprio dispositivo, utilizzando direttamente Lightning per inviare i pagamenti. [53] Entrambi mirano a garantire un miglior livello di privacy, grazie alle potenzialità del *Chaumian ECash* in combinazione con la blockchain di Bitcoin e Lightning Network.

In conclusione, mentre Bitcoin continua il suo lento sviluppo, sia attraverso innovazioni nel protocollo, sia con soluzioni come Lightning Network, il futuro della rete si prospetta sempre più promettente, con nuove tecnologie in grado di migliorare la scalabilità, la privacy e l'efficienza, senza mai sacrificare i principi fondamentali di sicurezza, privacy e decentralizzazione.

The computer can be used as a tool to liberate and protect people, rather than to control them.

Hal Finney

Bibliografia

[1] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. 2008. URL: https://bitcoin.org/bitcoin.pdf.

[2] Andreas M. Antonopoulos. *Mastering Bitcoin*. O'Reilly Media, 2014.

[3] *Unraveling Reorgs Problems: The Chainbase Approach*. Chainbase. 2023. URL: https://platform.chainbase.com/blog/article/unraveling-reorgs-problems-the-chainbase-approach.

[4] *Chain Reorganization*. Learn Me A Bitcoin. 2024. URL: https://learnmeabitcoin.com/technical/blockchain/chain-reorganization/.

[5] Learn Me A Bitcoin. *What is a UTXO?* 2024. URL: https://learnmeabitcoin.com/technical/transaction/utxo/.

[6] Young Platform. *Cosa sono le UTXO?* 2025. URL: https://youngplatform.com/glossary/utxo/.

[7] Kaleido. *UTXO vs Account Model*. 2025. URL: https://www.kaleido.io/blockchain-blog/utxo-vs-account-model.

[8] *Doppia Spesa*. Wikipedia. 2025. URL: https://it.wikipedia.org/wiki/Doppia_spesa.

[9] *Litecoin*. Wikipedia. 2024. URL: https://it.wikipedia.org/wiki/Litecoin.

[10] *Bitcoin Cash*. Wikipedia. 2024. URL: https://en.wikipedia.org/wiki/Bitcoin_Cash.

[11] *Bitcoin SV*. Crypto.com. 2024. URL: https://crypto.com/price/bitcoin-sv.

[12] *Bitcoin SV 2GB Block*. cryptonomist.ch. 2024. URL: https://cryptonomist.ch/2021/08/18/bitcoin-sv-blocco-2-gigabyte/.

[13] *BSV Node Status*. Whatsonchain. 2024. URL: https://whatsonchain.com/node-status.

[14] *Bitcoin Cash Nodes*. Cashnodes.io. 2024. URL: https://cashnodes.io/.

[15] *Litecoin Nodes*. Bitaps. 2024. URL: https://ltc.bitaps.com/.

[16] *Ethereum Nodes*. Etherscan. 2024. URL: https://etherscan.io/nodetracker.

[17] *Solana Nodes*. Solana RPC Map. 2024. URL: https://www.solanarpcmap.fyi/.

[18] *Block Reward*. IQ Wiki. 2025. URL: https://iq.wiki/wiki/block-reward.

[19] *Tor Browser*. Wikipedia. 2025. URL: https://it.wikipedia.org/wiki/Tor_(software).

[20] *Bitcoin Nodes Estimate*. Bitnodes. 2024. URL: https://bitnodes.io/nodes/all/#global-bitcoin-nodes.

[21] *Bitcoin Nodes Software*. Luke Dashjr. 2024. URL: https://luke.dashjr.org/programs/bitcoin/files/charts/software.html.

[22] *Bitcoin Public Nodes*. Bitnodes. 2024. URL: https://bitnodes.io/nodes/#network-snapshot.

[23] *Segregated Witness*. Learn Me A Bitcoin. 2024. URL: https://learnmeabitcoin.com/technical/upgrades/segregated-witness/.

[24] *Bitcoin Core Performance Evolution*. Jameson Lopp. 2025. URL: https://blog.lopp.net/bitcoin-core-performance-evolution/.

[25] *Quadratic Hashing*. Wikipedia. 2025. URL: https://en.wikipedia.org/wiki/Quadratic_hashing.

[26] *The Lightning Network Whitepaper*. Lightning Network. 2016. URL: https://lightning.network/lightning-network-paper.pdf.

[27] *Taproot e Beyond*. Around Bitcoin. 2024. URL: https://aroundbitcoin.io/blog/taproot-e-beyond-miglioramenti-nella-scalabilita-privacy-e-smart-contract-su-bitcoin/.

[28] *Firma di Schnorr*. Wikipedia. 2024. URL: https://it.wikipedia.org/wiki/Firma_di_Schnorr.

[29] *Overview*. Lightning Network. 2025. URL: https://docs.lightning.engineering/the-lightning-network/overview.

[30] *Bitcoin Core*. 2025. URL: https://bitcoin.org/en/download.

[31] *Run LND*. Lightning Network. 2025. URL: https://docs.lightning.engineering/lightning-network-tools/liglnd/run-lnd.

[32] Riccardo Masutti. *Mastering Lightning Network*. 2025. URL: https://riccardomasutti.com/Mastering%20Lightning%20Network%20-%20Protocollo%20di%20Secondo%20Livello%20per%20Pagamenti%20Bitcoin%20Istantanei.pdf.

[33] *Bitcoin Smart Contracts*. Trust Machines. 2025. URL: https://trustmachines.co/learn/bitcoin-smart-contracts/.

[34] *Lifecycle of a Payment Channel*. Lightning Network. 2025. URL: https://docs.lightning.engineering/the-lightning-network/payment-channels/lifecycle-of-a-payment-channel.

[35] *BOLT: Basis of Lightning Technology (Lightning Network Specifications)*. GitHub. 2025. URL: https://github.com/lightning/bolts.

[36] *An Introduction to the Lightning Network*. Lightspark. 2025. URL: https://www.lightspark.com/learn/lightning/an-introduction-to-the-lightning-network.

[37] Lightning Network Developers. *BOLT #7: Gossip Protocol*. 2016. URL: https://github.com/lightningnetwork/lightning-rfc/blob/master/07-routing-gossip.md.

[38] *Pathfinding in the Lightning Network*. arXiv. 2025. URL: https://arxiv.org/pdf/2410.13784.

[39] *Merkle Trees Ensure Transaction Integrity*. CCN. 2024. URL: https://www.ccn.com/education/crypto/bitcoin-merkle-trees-ensure-transaction-integrity/.

[40] *Use bind mounts*. Docker. 2025. URL: https://docs.docker.com/engine/storage/bind-mounts/.

[41] Christian Decker. *Lightning Network Research; Topology Datasets*. https://github.com/lnresearch/topology. Accessed: 2020-10-01. DOI: 10.5281/zenodo.4088530.

[42] *Small-world network*. Wikipedia. 2025. URL: https://en.wikipedia.org/wiki/Small-world_network.

[43] *Six degrees of separation*. Wikipedia. 2025. URL: https://en.wikipedia.org/wiki/Six_degrees_of_separation.

[44] *Trampoline Payments*. Zone Bitcoin. 2025. URL: https://zonebitcoin.co/it/Portafoglio-Phoenix-Bitcoin-Lightning/.

[45] *Preferential attachment*. Wikipedia. 2025. URL: https://en.wikipedia.org/wiki/Preferential_attachment.

[46] *CheckTemplateVerify*. Bitcoin Improvement Proposals. 2025. URL: https://github.com/bitcoin/bips/blob/master/bip-0119.mediawiki.

[47] *AnyPrevOut*. Bitcoin Improvement Proposals. 2025. URL: https://github.com/bitcoin/bips/blob/master/bip-0118.mediawiki.

[48] *OPCAT*. Bitcoin Improvement Proposals. 2025. URL: https://github.com/bip420/bip420.

[49] *Liquid*. Liquid. 2025. URL: https://docs.liquid.net/docs/welcome-to-liquid-developer-documentation-portal.

[50] *Rootstock*. Rootstock. 2025. URL: https://dev.rootstock.io.

[51] *ARK*. ARK. 2025. URL: https://arkdev.info/docs/developers/get-started.

[52] *Fedimint*. Fedimint. 2025. URL: https://fedimint.org/docs/intro.

[53] *CashU*. CashU. 2025. URL: https://cashu.space.