Principles of Blockchains
Princeton University,
Professor: Pramod Viswanath

Lecture 4

Peer to Peer Network and Bitcoin system

July 22, 2023

Contents

Chapter 1

Peer to Peer Network and Bitcoin system

1.1 Blockchains and Networking

A blockchain network is a system that provides ledger and smart contract services to applications without relying on a centralized server or authority. A centralized server can be a single point of failure or a target for censorship, which can compromise the security and availability of the network. Therefore, a blockchain network has the following networking requirements:

- The key primitive of the network is to **broadcast blocks and transactions to all nodes**.
  Blocks are data structures that contain transactions, which are records of value transfers or contract executions. Transactions are validated and appended to the ledger by consensus algorithms. Broadcasting ensures that all nodes have the same view of the ledger and can verify its integrity.
- The network also needs to be **robust and resilient to node failures or attacks**.
  Some nodes may go offline due to various reasons, such as power outages, network disruptions, or malicious attacks. The network should be able to tolerate a certain percentage of node failures without affecting its functionality. Moreover, the network should allow new nodes to join and synchronize with the existing nodes, as well as handle node churns and partitions.

1.1.1 Types of Network Architecture

There are two types of network architecture:

1. **Client Server** :
   This is a type of network architecture where one or more servers store most of the data and resources, and the clients access them through requests. The server is the central authority that controls the network, and the clients are dependent on the server for their functionality.
2. **Peer to Peer** :
   This is a type of network architecture where each node acts as both a client and a server, and there is no central authority or hierarchy. The nodes share their data and resources directly with each other, without relying on a server.

The need for robustness implies that we do not want a client-server relationship; we settle for a peer-to-peer (P2P) network where each node has identical behavior.

1.1.2 Overlay Networks

An overlay network depicts the connections between nodes, and is represented as a graph. It abstracts out the physical network switches and routers and defines virtual links between nodes. Two nodes that are connected by a link can exchange messages directly. Those that are not connected by a link must find a path connecting them on this overlay network in order to communicate messages.

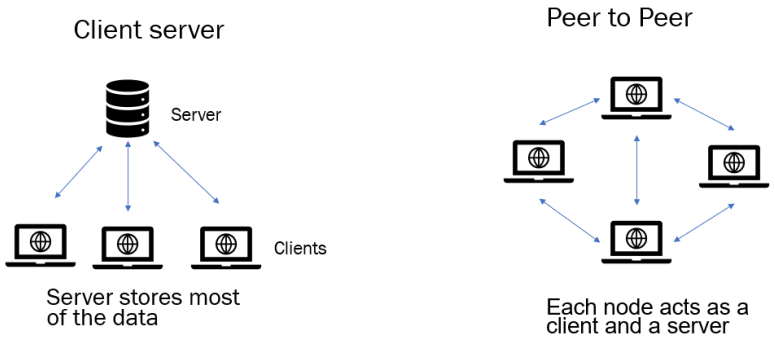

Figure 1.1: Types of Network Architecture

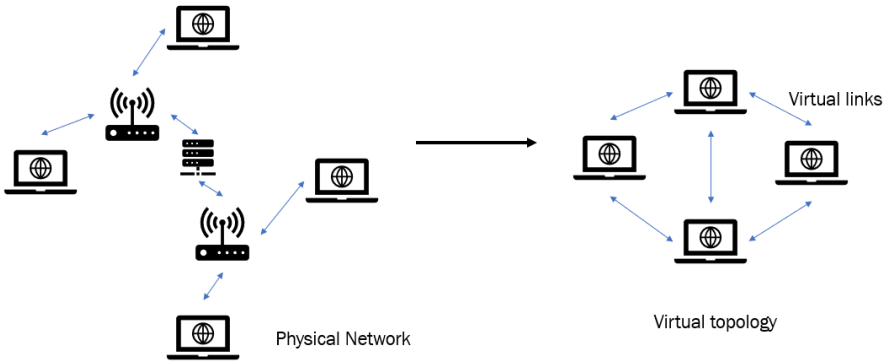

Figure 1.2: Overlay Network

There are different ways to classify overlay networks, but one common way is to distinguish between two types of overlay networks

- **structured** : These are overlay networks that have a specific topology or organization, such as a ring, a tree, or a grid. Structured overlay networks, like CHORD, assign an identifier to each node and uses that to construct welldefined routing rules. These networks are excellent for routing sending point to point messages, a use case not required for Bitcoin. Structured networks are suitable for broadcast, too; any message transmission takes O(logN) hops on CHORD with O(logN) connections per node.
- **unstructured** : These are overlay networks that have no specific topology or organization, and the nodes are connected randomly or based on some criteria, such as proximity, interest, or availability. Unstructured networks like d-regular graphs have no node identifiers; a node connects to d other nodes randomly.

Routing point to point messages is not feasible because it takes O(logN) hops and requires many peer queries to find the path from point A to point B. However, broadcast is very effective using gossip and also takes O(logN) hops. Therefore, the number of hops needed is equal to the structured network with the extra advantage of O(1) peer connections. That is why the Bitcoin network uses an unstructured d-regular overlay network.

1.1.3 Gossip and Flooding

Gossip and Flooding are two techniques for disseminating information in a network of nodes. Gossip is a technique that randomly sends the information to some of its neighbors, while Flooding sends the information to all of its neighbors. Both techniques can spread the information exponentially and reach all nodes in O(logN) time.
Gossip and Flooding can be used to achieve fast and reliable information dissemination in a network, but they differ in terms of efficiency, overhead, and robustness.

1.1.4 Expander Graph

Expander graph is a type of graph that has the property of being well connected but sparse.

Definition 1.1.1: Sparse Graph

A sparse graph $G(V, E)$ is a graph that has a small number of edges compared to the number of vertices, such that $|E| = O(|V|)$, where $|E|$ is the number of edges and $|V|$ is the number of vertices.

An Expander graph is a sparse graph that is also well connected, such that for any subset A of vertices, the number of vertices outside A that have at least one neighbor in A, denoted by $|A|$, is large.

A gossip message starts with $A(0)$ as the broadcasting node with $|A(0)| = 1$. In the next hop, it will reach $\partial A(0)$ with $|A(1)|$ being at least $(1+\epsilon)$ times $|A(0)|$. This process repeats and we get $|A(k)|$ being at least $(1+\epsilon)^k$ times $|A(0)|$ for any k. Therefore, the number of steps to get to half the number of nodes is proportional to the logarithm of the number of nodes. It can be proven that the other half of the nodes can also be reached in O(logN) time.

1.2 Bitcoin Network

Bitcoin is a peer-to-peer network that uses TCP protocol to exchange data. In Bitcoin network:

- The codebase limits the number of outgoing connections to eight and the number of incoming connections to 117.
- The network has a high churn rate (rate at which users join or leave the system); therefore, the node must be prepared to connect to new peers.
- The node maintains a large list of nodes running Bitcoin in the form of their (IP, port) pair and connects to one of them randomly when a slot becomes available.
- The node ensures that the peers it connects to are selected randomly.

1.2.1 Peer discovery

A node starts its list of peers by connecting to a group of DNS seed nodes. These are special nodes that provide a list of IP addresses and ports of active nodes in the network. A node can query a DNS seed node to get this list and then try to connect to some of the nodes in the list.
The seed nodes are not very distributed; so it is not wise to depend entirely on the peer list given by them. After connecting to the first set of peers, a node requests their peer list using **getAddr** and **Addr** messages. The node updates its peer list frequently by swapping peer lists with its peers.

1.2.2 Block transmission

The process of transmitting blocks and transactions involves the following steps:

1. A node sends an **inv** message to its peers, which is an inventory message that tells them what blocks and transactions are available.
2. When a peer receives an **inv** message, it checks if it already has the block or the transaction in its local storage. If not, it sends a **getData** message to the node to get those blocks and transactions.
3. Optionally, the peer can request only the headers of the blocks first, using a **getHeaders** message, before asking for the full blocks .
4. This header-first block transmission can reduce the bandwidth use and prevent invalid blocks from being accepted .

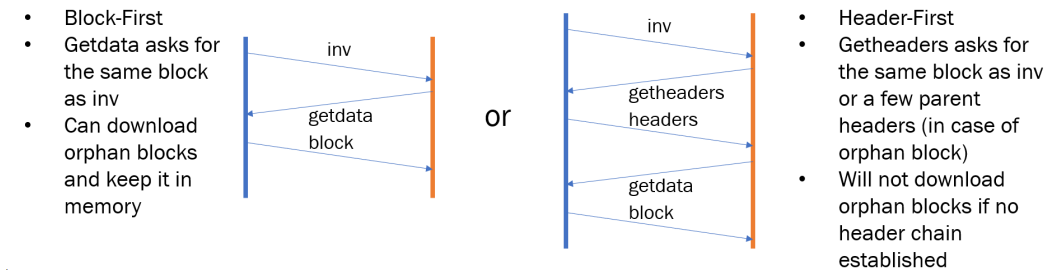

Figure 1.3: Block transmission in blockchain network

1.2.3 Data Broadcast

Now we explain how data, such as transactions, are broadcasted in a peer-to-peer network

- Each node maintains a non-persistent memory to store unconfirmed transactions, which are transactions that have not been included in a block yet. This memory is called the mempool.
- When a node receives a new transaction, it sends an inv message to its peers, which contains the transaction id (txid). The inv message is a way of checking if the peer already has the transaction in its mempool.
- If not, the peer sends a getdata message to request the full transaction data from the node. The node then sends the transaction data (tx) to the peer.
- Some unconfirmed transactions might be removed from the mempool due to various reasons, such as low fees, expiration time, or double spending.

1.2.4 Compact Blocks

Here we discuss two methods of relaying blocks in a peer-to-peer network:

1. **legacy relaying** :
   Legacy relaying is the traditional way of sending the full block data to each node.
2. **compact block relaying** :
   Compact block relaying is a protocol that reduces the amount of bandwidth and latency required to transfer a block that confirms many transactions that the nodes already have in their mempools.

There some differences between the two methods in terms of the messages exchanged between the nodes .Here are the advantages of compact block relaying over legacy relaying:

- Compact block relaying uses shortened transaction identifiers (txids) instead of full transaction data, which reduces the size of the block data.
- Compact block relaying allows nodes to request only the transactions they are missing from their mempools, which reduces the redundancy of sending transactions twice.
- Compact block relaying has two modes: low bandwidth mode and high bandwidth mode.
  Low bandwidth mode minimizes the bandwidth usage by asking for permission before sending a compact block.
  High bandwidth mode reduces the latency by sending a compact block as soon as possible without asking for permission.
- Compact block relaying can verify the proof of work (PoW) from the block header before requesting the full block data, which prevents invalid blocks from being accepted.

There is a trade-off between capacity and propagation delay in a peer-to-peer network. **Capacity** (C) is the maximum amount of data that can be transmitted in a given time, measured in bits per second (bps). **Propagation delay** (D) is the time it takes for a signal to travel from one node to another, measured in seconds. The end to end delay, which is the total time it takes for a block to reach all nodes in the network, increases with the increase

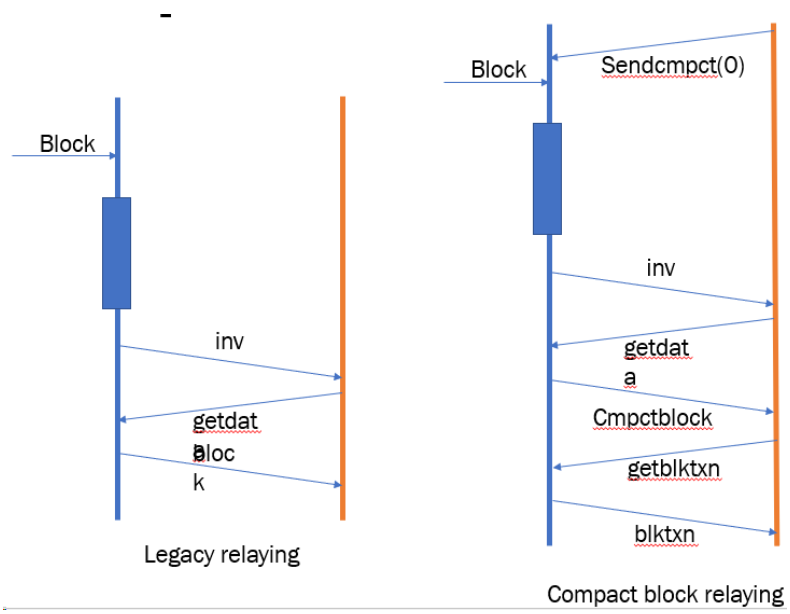

Figure 1.4: Compact Block

in block size. This is because larger blocks take longer to transmit and verify, which can cause congestion and orphaning. Therefore, there is a trade-off between increasing the capacity and reducing the propagation delay of the network. This increase in delay affect the system in following ways:

- Wasted Hash power
- Forking

1.2.5 Disadvantages of current p2p network

- **Efficiency:**
  It requires a lot of communication between nodes. The total communication is proportional to the number of nodes (N) and the average degree of connectivity (d). This means that as the network grows, the communication overhead also increases, which can affect the performance and scalability of the system.
- **Privacy:**
  It can link the transaction source to the IP address of the node that broadcasts it. This means that anyone who monitors the network traffic can potentially identify the sender and receiver of a transaction, which can compromise their anonymity and privacy.
- **Security:**
  It allows for plausible deniability for forking. Forking can cause inconsistency and double spending. Plausible deniability means that a node can claim that it did not receive a valid block from another node, even if it did, and create a fork on purpose. This can be used as an attack strategy to disrupt the consensus and integrity of the system.

1.3 Geometric Random Network

The current network topology is based on random IP addresses, which are not related to geographic distances. This means that nodes may be connected to peers that are far away from them, which can increase the latency and bandwidth usage of the network.

A better network topology would be based on a **geometric random network**, which is a network where nodes are placed according to some geometric criteria, such as proximity or similarity. This way, nodes can connect to peers that are closer or more relevant to them, which can improve the efficiency and performance of the

network. The challenges are creating a self-adapting network topology based on measurements, such as latency, bandwidth, or trust. This means that nodes can dynamically adjust their connections based on the changing conditions and preferences of the network.

1.3.1 Perigee

Perigee is a self-adaptive network topology algorithm for peer-to-peer networks. Perigee is a decentralized algorithm that selects neighbors based on past interactions. It aims to retain neighbors that relay blocks fast and disconnect from neighbors that do not relay blocks fast. It also explores unseen neighbors to discover new potential connections.

‘Perigee is motivated by the multi-armed bandit problem, which is a problem of finding the optimal strategy for choosing among several options with uncertain rewards. Perigee tries to balance between exploration and exploitation, which means finding new neighbors and using existing neighbors, respectively. Perigee can improve the efficiency and performance of the peer-to-peer network by reducing the latency and bandwidth usage.

Algorithm

The Perigee Algorithm works as follows:

- It assigns scores for each subset of neighbors based on how fast they relay blocks.
- It retains the subset of neighbors with the best score and disconnects the node that is not in the subset.
- It forms a connection to a random neighbor to explore new potential connections.