CS265 Research: Introduction to Blockchain and Projects

Adaptive Blockchain

Hao Jiang
Blockchain Backgrounds

Blockchain Systems and HyperLedger Fabric

Projects
Blockchain is much older than Bitcoin
Blockchain is much older than Bitcoin
Blockchain is much older than Bitcoin

Chain of blocks

Merkle Tree

Blockchain is much older than Bitcoin
Blockchain is much older than Bitcoin
Blockchain is much older than Bitcoin
Blockchain is much older than Bitcoin
Blockchain is much older than Bitcoin

Decentralization, Mutual-Trust

Chain of blocks 1982
Merkle Tree 1992
HashCash 2002
Blockchain 2008 2013 2015
Decentralization, Mutual-Trust

A Data Structure

A Distributed System
Blockchain as a Data Structure
Blockchain as a Data Structure

Signature
Hash: 2abbcda
Content: 0110...
Blockchain as a Data Structure

Signature
Hash: 2abbcd
Content: 0110...

Signature
Prev Hash: 2abbcd
Hash: 1cde9f
Content: 0010...
Blockchain as a Data Structure

Signature
Hash: 2abbcd
Content: 0110...

Signature
Prev Hash: 2abbcd
Hash: 1cde9f
Content: 0010...

Signature
Prev Hash: 1cde9f
Hash: 3aa667
Content: 1100...
Prevent Data Tampering
Prevent Data Tampering

Sender

Data block
Signature: A

Receiver
Prevent Data Tampering

Sender

Data block
Signature: A

Receiver
Prevent Data Tampering

Sender

Attacker

Data block
Signature: A

Receiver
Prevent Data Tampering

Sender

Attacker

Data block
Signature: A

Receiver
Prevent Data Tampering

Sender

Attacker

Data block
Signature: A

Receiver
Prevent Data Tampering
Prevent Data Tampering

Sender

Attacker

Data block

Signature: A

Reject

Receiver
Make Sure Data is Ordered

Sender 1

New block
Prev Hash: X
Hash: Y

Sender 2

New block
Prev Hash: X
Hash: Z

Receiver
Make Sure Data is Ordered

Sender 1

Sender 2

Receiver

New block
Prev Hash: X
Hash: Y

New block
Prev Hash: X
Hash: Z
Make Sure Data is Ordered

Sender 1

Sender 2

New block
Prev Hash: X
Hash: Y

New block
Prev Hash: X
Hash: Z

Receiver
Make Sure Data is Ordered

Sender 1

Sender 2

New block
Prev Hash: X
Hash: Y

New block
Prev Hash: X
Hash: Z

Receiver

Last block
Hash: X
Make Sure Data is Ordered

Sender 1

Sender 2

New block
Prev Hash: X
Hash:         Y

New block
Prev Hash: X
Hash:         Z

New block
Prev Hash: X
Hash:         Y

Receiver

Prev block
Hash:         X
Make Sure Data is Ordered

Sender 1

Sender 2

Last block
Prev Hash: X
Hash: Y

New block
Prev Hash: X
Hash: Z

Receiver

Prev block
Hash: X
Make Sure Data is Ordered

Sender 1

Sender 2

New block
Prev Hash: X
Hash: Z

Receiver

Last block
Prev Hash: X
Hash: Y

Prev block
Hash: X

Hash: Z

Hash: Y

Hash: X
Make Sure Data is Ordered

Sender 1

Sender 2

Receiver

New block
Prev Hash: X
Hash: Z

Rejected

Prev block
Hash: X

Last block
Prev Hash: X
Hash: Y
Blockchain as A Database System
Blockchain as A Database System

Key-Value or Relational
Blockchain as A Database System

Key-Value or Relational

Logged Database
Blockchain as A Database System

Key-Value or Relational

Logged Database

Distributed Database
How a blockchain system works
How a blockchain system works

Clients

Transaction

Transaction

Transaction

Servers
How a blockchain system works
How a blockchain system works

Servers

Transaction

Transaction

Transaction
How a blockchain system works

Transaction Pool

Servers

Transaction

Transaction

Transaction
How a blockchain system works
How a blockchain system works

Transaction Pool

Some Server

Other Servers
How a blockchain system works
How a blockchain system works

Transaction Pool

Some Server

Other Servers
How a blockchain system works

- Transaction Pool
- Some Server
- Other Servers
Two Popular Types of Blockchain

Public/Permissionless Blockchain  Private/Permissioned Blockchain
Two Popular Types of Blockchain

Public/Permissionless Blockchain

- Runs on Internet
- Many participants
- Open to the public

Private/Permissioned Blockchain
Two Popular Types of Blockchain

<table>
<thead>
<tr>
<th>Public/Permissionless Blockchain</th>
<th>Private/Permissioned Blockchain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Runs on Internet</td>
<td></td>
</tr>
<tr>
<td>Many participants</td>
<td></td>
</tr>
<tr>
<td>Open to the public</td>
<td></td>
</tr>
<tr>
<td>Bitcoin</td>
<td>Ethereum</td>
</tr>
<tr>
<td>Ethereum</td>
<td></td>
</tr>
</tbody>
</table>

Bitcoin runs on the Internet, is open to the public, and has many participants. Ethereum is also open to the public and has many participants.
Two Popular Types of Blockchain

Public/Permissionless Blockchain
- Runs on Internet
- Many participants
- Open to the public

Private/Permissioned Blockchain
- Runs on a private network
- Small Groups
- Need Authentication to join

Bitcoin
Ethereum
Two Popular Types of Blockchain

Public/Permissionless Blockchain
- Runs on Internet
- Many participants
- Open to the public

Private/Permissioned Blockchain
- Runs on a private network
- Small Groups
- Need Authentication to join

Applications:
- Bitcoin
- Ethereum
- Hyperledger
- Oracle Blockchain
- IBM Blockchain
Who can create the next block?

Transaction Pool

- T1
- T2
- T3
Who can create the next block?

Transaction Pool

T1
T2
T3
Who can create the next block?

Transaction Pool

T1

T2

T3

Hash: ABC

Hash: 123
Synchronization of Servers
Synchronization of Servers
Synchronization of Servers

Blocked
Synchronization of Servers

In a distributed system, a **Consensus Protocol** synchronizes nodes.
In a distributed system, a **Consensus Protocol** synchronizes nodes.

Consensus Protocol is an algorithm determines:

1. Which node create the next block
2. How to resolve discrepancy between nodes
Consensus Protocol in Bitcoin

Who can create next block? Resolve Discrepancy?
Consensus Protocol in Bitcoin

Who can create next block?  Resolve Discrepancy?

\[ H \]

0000011100101....

HashCash
Consensus Protocol in Bitcoin

Who can create next block?

\[ \mathcal{H} \]

0000011100101….

HashCash

Resolve Discrepancy?

\[ \begin{align*}
  & B1 \\
  & B2 \\
  & B3 \\
  & B4 \\
\end{align*} \quad + \quad \begin{align*}
  & B1 \\
  & B2 \\
  & B3^* \\
  & B4^* \\
  & B5^* 
\end{align*} \]
Consensus Protocol in Bitcoin

Who can create next block?

\[ H \]

\[ 0000011100101 \ldots \]

HashCash

Resolve Discrepancy?

- B1
- B2
- B3*
- B4*
- B5*
Consensus Protocol in Bitcoin

Who can create next block?

\[ \mathcal{H} \]

0000011100101….

HashCash

Resolve Discrepancy?

B1

B2

B3*

B4*

B5*

When two chains meet, Longer Chain Wins
Bitcoin takes 10 min to confirm a transaction
Bitcoin takes 10 min to confirm a transaction

The first person solving the HashCash can add a new block.

Get a reward for adding the block.
Bitcoin takes 10 min to confirm a transaction

The first person solving the HashCash can add a new block.

Get a reward for adding the block.

Blockchain cannot grow too fast.

Controlled by the hardness of HashCash

Limit to ~10 min a new block
Bitcoin takes 10 min to confirm a transaction

Bitcoin (Cryptocurrency) is designed to be slow

Blockchain cannot grow too fast.

Controlled by the hardness of HashCash

Limit to ~10 min a new block

00000000000
Easier

0000000000000
Harder
Bitcoin (Cryptocurrency) is designed to be slow

Permissioned Blockchain is not
We focus on studies on Permissioned Blockchain
Example: Food Trust Supply Chain
Transactions
Transactions

2023-01-01 05:10, Farm 001, Truck 001, Potato, 100lb, Good
Transactions

2023-01-01 05:10, Farm 001, Truck 001, Potato, 100lb, Good
2023-01-01 23:10, Truck 001, Store 001, Potato, 100lb, Good
Transactions

2023-01-01 05:10, Farm 001, Truck 001, Potato, 100lb, Good
2023-01-01 23:10, Truck 001, Store 001, Potato, 100lb, Good
2023-01-02 17:10, Store 001, Customer 001, Potato, 5lb, Good
Consensus Protocol in a Permissioned Chain

Who can create next block?  Resolve Discrepancy?
Consensus Protocol in a Permissioned Chain

Who can create next block?

Resolve Discrepancy?

Leader

Follower

Follower
Consensus Protocol in a Permissioned Chain

Who can create next block?  
Leader

Resolve Discrepancy?
Follower  Follower
Consensus Protocol in a Permissioned Chain

Who can create next block?

Leader

Resolve Discrepancy?

Follower  Follower
Consensus Protocol in a Permissioned Chain

Who can create next block?

Leader

Follow the leader

Follower  Follower

Resolve Discrepancy?

Leader

Follow the leader

Follower  Follower
Two Families of Consensus Protocols
Two Families of Consensus Protocols

**Crash Fault Tolerance**

- Can detect offline leaders
- Elect new leader
- Assume all participants are benign

- Paxos
- Raft
Two Families of Consensus Protocols

**Crash Fault Tolerance**
- Can detect offline leaders
- Elect new leader
- Assume all participants are benign

**Byzantine Fault Tolerance**
- Crash Fault Tolerant
- Resist malicious participants

- **Paxos**
- **Raft**
- **PBFT**
Fabric Architecture

End User

Fabric Network
Fabric Architecture

End User

Peer

Peer

Peer

Fabric Network
Fabric Architecture

End User

Peer

Leader

Followers

Orderers

Peer

Peer

Peer

Fabric Network
Fabric Architecture

End User → Transaction → Peer → Leader, Followers, Orderers → Fabric Network
Fabric Architecture

- End User
- Peer
- Leader
- Followers
- Orderers
- Fabric Network

New Block
Develop Applications with Fabric

Developers Write Business Functions (Smart Contract)

```javascript
function issueInvoice(String customer, double amount) {
    let inv = new Invoice()
    inv.number = 'I001';
    inv.amount = amount;
    inv.customer = customer;
    inv.status = 'ISSUED';
    getStorage().put(inv.number, json_serialize(inv));
    return inv;
}

function payInvoice(String number) {
    let inv = getStorage().get(number);
    inv.status = 'PAID';
    getStorage().put(number, json_serialize(inv));
    return inv;
}
```
Develop Applications with Fabric

Developers Write Business Functions (Smart Contract)

```javascript
function issueInvoice(String customer, double amount) {
    let inv = new Invoice()
    inv.number = 'I001';
    inv.amount = amount;
    inv.customer = customer;
    inv.status = 'ISSUED';
    getStorage().put(inv.number, json_serialize(inv));
    return inv;
}

function payInvoice(String number) {
    let inv = getStorage().get(number);
    inv.status = 'PAID';
    getStorage().put(number, json_serialize(inv));
    return inv;
}
```

getStorage is an API to a key-value store
Develop Applications with Fabric

Admin Deploy the code to a Fabric Network

Clients Invoke the Business Function from a Peer
Develop Applications with Fabric

Admin Deploy the code to a Fabric Network

Clients Invoke the Business Function from a Peer

Clients

issueInvoice

getStorage().put(…)

Peer

Transaction

To Orderer
Develop Applications with Fabric

Admin Deploy the code to a Fabric Network

Clients Invoke the Business Function from a Peer

Clients

issueInvoice

getStorage().put(....)

getStorage().get(....)

Peer
Develop Applications with Fabric

Admin Deploy the code to a Fabric Network

Clients Invoke the Business Function from a Peer
Our goal: Self-Designed Adaptive Blockchain
Our goal: Self-Designed Adaptive Blockchain
Big Picture of Self-Designed Blockchain
Big Picture of Self-Designed Blockchain

Consensus Protocol
Big Picture of Self-Designed Blockchain

- Consensus Protocol
- Block Creation
Big Picture of Self-Designed Blockchain

- Consensus Protocol
- Block Creation
- Storage
Big Picture of Self-Designed Blockchain

- Consensus Protocol
- Block Creation
- Storage
  - Cosine
Bottlenecks
Bottlenecks

Block Creation Delays Transaction
Bottlenecks

Block Creation Delays Transaction

Consensus Protocol is Time-Consuming
Bottlenecks

Block Creation Delays Transaction

Project 1: Streaming Chain

Consensus Protocol is Time-Consuming
Bottlenecks

Block Creation Delays Transaction
Project 1: Streaming Chain

Consensus Protocol is Time-Consuming
Project 2: Continuum of Consensus Protocol
Project 1: Streaming Chain
Review of Transaction Execution

Peer

Orderer
Review of Transaction Execution

- Call Function
- Submit Transaction
- Add to Block

Peer — Orderer
Review of Transaction Execution

Peer

Call Function → Submit Transaction

Orderer

Call Function → Submit Transaction → Add to Block

Add to Block
Review of Transaction Execution

Peer

Call Function ➔ Submit Transaction ➔ Add to Block
Call Function ➔ Submit Transaction ➔ Add to Block

Orderer

Send Block ➔ Add to Block ➔ Block is Full
Confirm All Transactions ➔ Send Block ➔ Block is Full
Key Idea: Reduce the waiting on block creation
Key Idea: Reduce the waiting on block creation
Key Idea: Reduce the waiting on block creation

Speed up Block Creation
Key Idea: Reduce the waiting on block creation

- Speed up Block Creation
- Do Not Wait for Block Creation
Key Idea: Reduce the waiting on block creation

Pros and Cons?  Performance?

Do Not Wait for Block Creation
Thoughts: How would Block size affect these Metrics?

Number of Blocks

Block Size
Thoughts: How would Block size affect these Metrics?

We learned this in class: Computation/Data Movement
Project Tasks and Timelines
Project Tasks and Timelines

📖 Preparation (2 weeks)
Project Tasks and Timelines

**Preparation (2 weeks)**

- Read Papers and HyperLedger Document
- Get Familiar with HyperLedger Env and Blockbench Tool
- Setup and Run a HyperLedger Application
Project Tasks and Timelines

- Preparation (2 weeks)
  - Read Papers and HyperLedger Document
  - Get Familiar with HyperLedger Env and Blockbench Tool
  - Setup and Run a HyperLedger Application

- Design Experiments (1 week)
Project Tasks and Timelines

📚 Preparation (2 weeks)
- Read Papers and HyperLedger Document
- Get Familiar with HyperLedger Env and Blockbench Tool
- Setup and Run a HyperLedger Application

🔧 Design Experiments (1 week)
- List the performance metrics to collect
- List the experiments to run
- Explain the purpose of each experiment
Project Tasks and Timelines

- **Preparation (2 weeks)**
  - Read Papers and HyperLedger Document
  - Get Familiar with HyperLedger Env and Blockbench Tool
  - Setup and Run a HyperLedger Application

- **Design Experiments (1 week)**
  - List the performance metrics to collect
  - List the experiments to run
  - Explain the purpose of each experiment

- **Data Collection (1-2 weeks)**
Project Tasks and Timelines

Preparation (2 weeks)  
Read Papers and HyperLedger Document  
Get Familiar with HyperLedger Env and Blockbench Tool  
Setup and Run a HyperLedger Application

Design Experiments (1 week)  
List the performance metrics to collect  
List the experiments to run  
Explain the purpose of each experiment

Data Collection (1-2 weeks)  
Run the experiments and collect data
Project Tasks and Timelines

- **Preparation (2 weeks)**
  - Read Papers and HyperLedger Document
  - Get Familiar with HyperLedger Env and Blockbench Tool
  - Setup and Run a HyperLedger Application

- **Design Experiments (1 week)**
  - List the performance metrics to collect
  - List the experiments to run
  - Explain the purpose of each experiment

- **Data Collection (1-2 weeks)**
  - Run the experiments and collect data

- **Analyze and Modeling (2-3 weeks)**
Project Tasks and Timelines

- **Preparation (2 weeks)**
  - Read Papers and HyperLedger Document
  - Get Familiar with HyperLedger Env and Blockbench Tool
  - Setup and Run a HyperLedger Application

- **Design Experiments (1 week)**
  - List the performance metrics to collect
  - List the experiments to run
  - Explain the purpose of each experiment

- **Data Collection (1-2 weeks)**
  - Run the experiments and collect data

- **Analyze and Modeling (2-3 weeks)**
  - Compare the results from two methods
  - Identify key features that distinguish the workloads
  - Design a classification model to classify the workload
  - Design a decision model to choose a method
  - (Optional) Reasoning of the model result
What is Success
What is Success

Finish the experiment and collect the result
What is Success

Finish the experiment and collect the result

Find a decision model of which method is better
What is Success

- Finish the experiment and collect the result
- Find a decision model of which method is better
- Reasoning of the result from the model
Necessary Skills
Necessary Skills

Basic Linux Operations
- Bash scripts
- Text editor
Necessary Skills

**Basic Linux Operations**
- Bash scripts
- Text editor

**Docker Operations**
- Common Docker Commands
- Container Performance Monitoring
Necessary Skills

**Basic Linux Operations**
- Bash scripts
- Text editor

**Docker Operations**
- Common Docker Commands
- Container Performance Monitoring

**Programming Language**
- Experience with Java/Python/Node.js
- Experience with Golang
Project 2: Continuum of Consensus Protocol
Review: What does Consensus Protocol do
Key Ideas

PBFT -> Raft

Use PBFT
Key Ideas

PBFT -> Raft

Use PBFT

Raft
Fast
Less Secure

PBFT
Slower
More Secure
Key Ideas

PBFT -> Raft

Use PBFT

Raft
Fast
Less Secure

PBFT
Slower
More Secure

What’s in here?
Key Idea: Configurable Protocols
Project Description

Design a Configurable Consensus Protocol
Between Raft and PBFT

Raft
Fast
Less Secure

PBFT
Slower
More Secure
Project Description

Design a Configurable Consensus Protocol
Between Raft and PBFT

Raft
Fast
Less Secure

PBFT
Slower
More Secure
Project Description

Design a Configurable Consensus Protocol
Between Raft and PBFT

Raft
Fast
Less Secure

PBFT
Slower
More Secure
Project Description

Design a Configurable Consensus Protocol
Between Raft and PBFT
Project Description

Design a Configurable Consensus Protocol
Between Raft and PBFT

We learned this in Class: Design Continuum

Raft
Fast
Less Secure

PBFT
Slower
More Secure
Thoughts and Examples
Thoughts and Examples

Use Cosine idea?
Thoughts and Examples

Use Cosine idea?

Find Primitives
Combines them
Thoughts and Examples

Use Cosine idea?
Find Primitives
Combines them

Switch Between the Protocols?
Thoughts and Examples

Use Cosine idea?
Find Primitives
Combines them

Switch Between the Protocols?
With a probability of which to use
Project Challenges

Sync

Recovery
Project Challenges

Sync

Recovery
Project Challenges

Sync

Recovery
Project Tasks and Timelines
Project Tasks and Timelines

Preparation (2 weeks)
Project Tasks and Timelines

Preparation (2 weeks)

- Get Familiar with Raft and PBFT
- Read a paper about designing and test new protocol
- Implement the protocols
Project Tasks and Timelines

- Preparation (2 weeks)
  - Get Familiar with Raft and PBFT
  - Read a paper about designing and test new protocol
  - Implement the protocols

- System Design (2 weeks)
Project Tasks and Timelines

- **Preparation (2 weeks)**
  - Get Familiar with Raft and PBFT
  - Read a paper about designing and test new protocol
  - Implement the protocols

- **System Design (2 weeks)**
  - Detail Design of the Protocol
  - How does the protocol resembles Raft/PBFT?
  - How does the protocol recover from attacks?
Project Tasks and Timelines

- **Preparation (2 weeks)**
  - Get Familiar with Raft and PBFT
  - Read a paper about designing and test new protocol
  - Implement the protocols

- **System Design (2 weeks)**
  - Detail Design of the Protocol
  - How does the protocol resembles Raft/PBFT?
  - How does the protocol recover from attacks?

- **Implementation (2-3 weeks)**
Project Tasks and Timelines

- **Preparation (2 weeks)**
  - Get Familiar with Raft and PBFT
  - Read a paper about designing and test new protocol
  - Implement the protocols

- **System Design (2 weeks)**
  - Detail Design of the Protocol
  - How does the protocol resembles Raft/PBFT?
  - How does the protocol recover from attacks?

- **Implementation (2-3 weeks)**
  - Implement the new Protocol
  - Implement Raft/PBFT in the same language for comparison
## Project Tasks and Timelines

<table>
<thead>
<tr>
<th>Task</th>
<th>Timeline</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preparation</td>
<td>2 weeks</td>
<td>Get Familiar with Raft and PBFT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Read a paper about designing and test new protocol</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Implement the protocols</td>
</tr>
<tr>
<td>System Design</td>
<td>2 weeks</td>
<td>Detail Design of the Protocol</td>
</tr>
<tr>
<td></td>
<td></td>
<td>How does the protocol resembles Raft/PBFT?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>How does the protocol recover from attacks?</td>
</tr>
<tr>
<td>Implementation</td>
<td>2-3 weeks</td>
<td>Implement the new Protocol</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Implement Raft/PBFT in the same language for comparison</td>
</tr>
<tr>
<td>Experiment and Analysis</td>
<td>2-3 weeks</td>
<td></td>
</tr>
</tbody>
</table>
Project Tasks and Timelines

- **Preparation (2 weeks)**
  - Get Familiar with Raft and PBFT
  - Read a paper about designing and test new protocol
  - Implement the protocols

- **System Design (2 weeks)**
  - Detail Design of the Protocol
  - How does the protocol resembles Raft/PBFT?
  - How does the protocol recover from attacks?

- **Implementation (2-3 weeks)**
  - Implement the new Protocol
  - Implement Raft/PBFT in the same language for comparison

- **Experiment and Analysis (2-3 weeks)**
  - Run performance experiments with the new Protocol
  - Verify that the new protocol can simulate Raft/PBFT
  - Verify that the new protocol can sit in between
  - Run attack experiments against the new protocol
What is Success
What is Success

Design and Implementation of such a protocol
What is Success

Design and Implementation of such a protocol

Verify that the new protocol’s performance goes in the spectrum
Necessary Skills
Necessary Skills

Programming Language
One Programming Language (Preferred Go)
Necessary Skills

**Programming Language**
One Programming Language (Preferred Go)

We want to test our new protocol in Fabric, who uses Go
Necessary Skills

Programming Language
One Programming Language (Preferred Go)

We want to test our new protocol in Fabric, who uses Go

Distributed System
Experience with Distributed Systems