Bitcoin:
Programming the Future of Money

Topics in Computer Science - ITCS 4010/5010, Spring 2025

Dr. Christian Kümmerle

Lecture 8

Identities & Finite Fields

Main Reference for Finite Fields:
- "Programming Bitcoin: Learn How to Program Bitcoin from Scratch", Jimmy Song, 1st Edition, O'Reilly, 2019.

Basics of the Bitcoin Protocol

RECAP

- What were some of the milestones in the development and evolution of the Bitcoin network/ the Bitcoin protocol?

▷ Paper published late 2008
▷ Early 2009: Software published & first block mined
▷ Early 2010's – 2014: Silk road
▷ 2017: "Blocksize war"

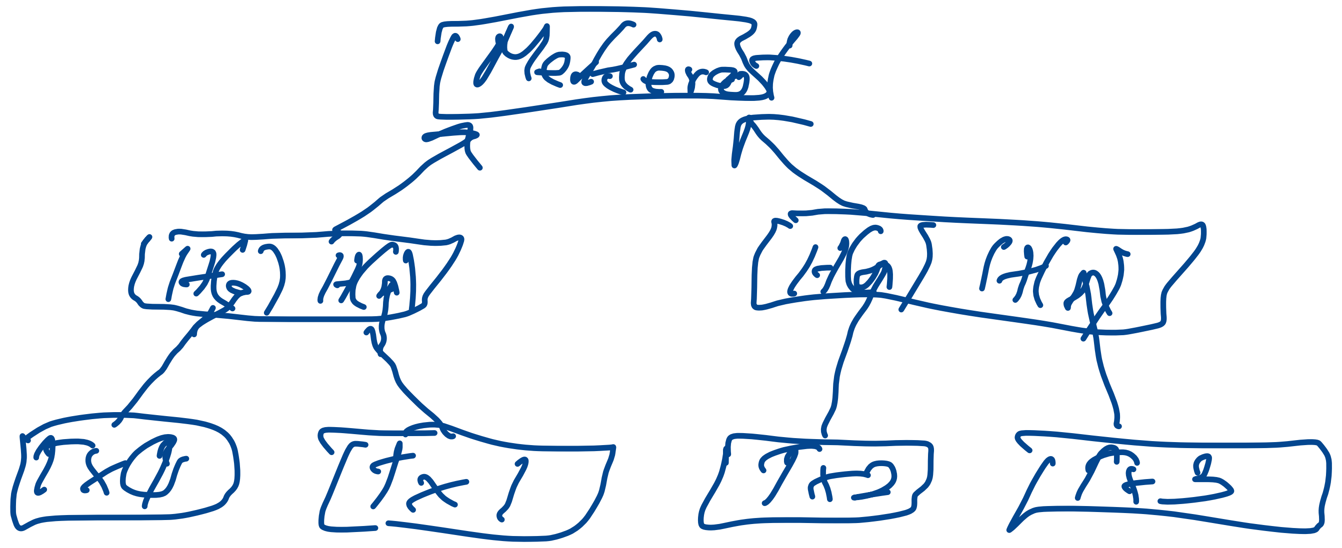
- What is a key mechanism that facilitates the “integrity” of a chain of blocks of data? What does “integrity” mean here?

▷ Hash pointers in block header

- What is a suitable data structure to store transactions of a Bitcoin block and what advantages does it have?

→ Merkle tree

Advantages:
▷ Efficient Verification of Transactions
▷ Potential reduction of data overhead



Identities in the Bitcoin Protocol

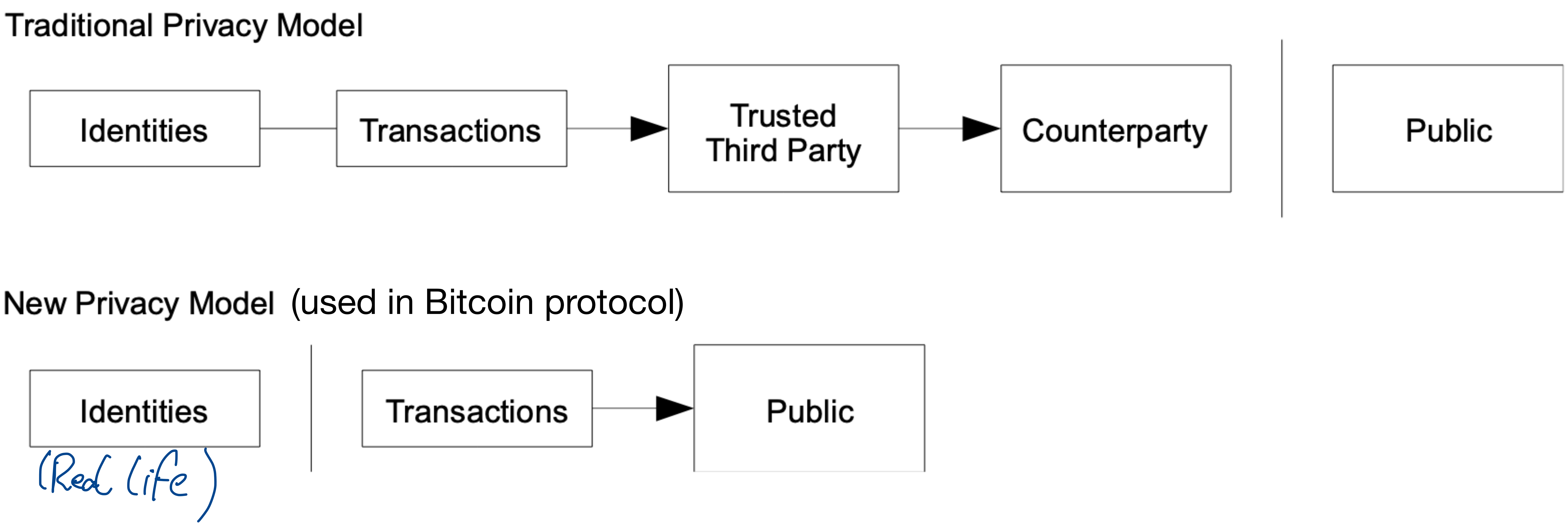
BITCOIN ADRESSES

Different Bitcoin address formats:

- **P2PK** (Pay to Public Key)
- **P2PKH** (Pay to Public Key Hash)
- **P2SH** (Pay to Script Hash)
- Bech32 (Native SegWit, **P2WPKH and P2WSH**)
- **P2TR** (Pay to Taproot)

We will learn about these later. All address types are derived from (one or more) **public-private key pair(s).**

IDENTITIES IN THE BITCOIN PROTOCOL & PRIVACY



- The Bitcoin blockchain is permissionless, permissionlessness is achieved by the ability of anyone to create new private/public key pairs.
- Transactions are public (**not encrypted**), identities are pseudonymous (can be linked with real-world identities indirectly)

RECAP: DIGITAL SIGNATURE SCHEMES

Essential functions of digital signature scheme:

Let M be finite message space and Σ finite signature space.

- A *signing function* $s_{\text{ALICE}} : \mathcal{M} \to \Sigma$ that maps messages $m \in \mathcal{M}$ from a finite *message space* $\mathcal{M}$ to an element $s_{\text{ALICE}}(m) \in \Sigma$ of a finite signature space Σ. This signing function is not publicly known, but only to the user ALICE. (private to Alice)
- A *verification function* $v_{\text{ALICE}} : \mathcal{M} \times \Sigma \to \{\text{TRUE}, \text{FALSE}\}$ which outputs

$$v_{\text{ALICE}}(m,\sigma) = \begin{cases} \text{TRUE}, & \text{if } \sigma = s_{\text{ALICE}}(m), \\ \text{FALSE}, & \text{if } \sigma \neq s_{\text{ALICE}}(m). \end{cases}$$

(public)

How to implement this within public-private key cryptography:

- **Randomized Key Generation**:
  (sk,pk) = generateKeys(keysize,nonce)
  where *sk* is the secret or private key, *pk* is the public key, and *nonce* is a random seed only to be used once. *keysize* determines the size of the private (secret) key *sk.*
- **(Randomized) Signing Function:**
  sig = sign(sk,msg,nonce)
  where *msg* $\in M$ is a finite message, *sk* is the secret key, and *nonce* is a random seed only to be used once. For certain signature protocols, no random *nonce* is needed (deterministic signing functions)
- **Verification Function:**
  verify(pk, msg, sig)
  Returns a Boolean *(True* if signature *sig* valid, *False* otherwise*)*

S_{Alice}
$\uparrow$
sk

V_{Alice}
$\uparrow$
pk

DIGITAL SIGNATURE SCHEMES USED IN BITCOIN

- **Elliptic Curve Digital Signature Algorithm (ECDSA)**
  - Concept proposed by Neal Koblitz and Victor S. Miller in 1985
  - Standardized in 2000 by NIST
  - Used in Bitcoin since 2009, was freely available
  - Used by all address formats before Taproot upgrade

- **Schnorr Signatures:**
  - Proposed and **patented** by Claus-Peter Schnorr in 1990
  - Has certain advantages over ECDSA (will see later) and simpler
  - Patent expired in 2010, so not available at inception of Bitcoin
  - Implemented in address format introduced by 2021 Taproot upgrade

Finite Fields

FINITE FIELDS

Def: A field $(F, +, \cdot)$ is a set F that together with two operations "+" (called addition) and "$\cdot$" (called multiplication) satisfy the following properties:

1) If $a, b \in F$, then $a+b \in F$, $a \cdot b \in F$ ("closedness")
2) A element $0 \in F$ called additive identity exists and satisfies $a+0=a$ for any $a \in F$.
3) An element $1 \in F$ called multiplicative identity exists and satisfies $a \cdot 1 = a$ for any $a \in F$.
4) For any element $a \in F$, there exists $-a \in F$ (additive inverse) $a+(-a)=0$
5) For any element $a \in F \setminus \{0\}$, there exists an element $a^{-1} \in F$ called multiplicative inverse with $a \cdot a^{-1} = 1$

EXAMPLES OF INFINITE & FINITE FIELDS

1) $\mathbb{R}$: real numbers $\longrightarrow$ infinite field

-5.4 additive inverse

$5.4 \Longrightarrow \frac{1}{5.4}$ multiplicative inverse etc.

2) Not a field: $\mathbb{N}_0 = \{0, 1, 2, 3, 4, \ldots\}$.

⚠: No additive inverse

3) Not a field: $\mathbb{Z} = \{\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots\}$ $\notin \mathbb{Z}$

⚠ No multiplicative inverse: E.g., $-3 \cdot \left(-\frac{1}{3}\right) = 1$

4) Field (infinite): $\mathbb{Q}$: set of rational numbers

5) Finite field: $\mathbb{F}_5 = \{0, 1, 2, 3, 4\}$ with appropriate notion of "+" and "·".

ORDER OF A FINITE FIELD

Def: The order of a field $(F, +, \cdot)$ is the number $|F|$ of elements in F

E.g.: If $\mathbb{F}_7 = \{0, 1, 2, 3, 4, 5, 6\}$, then $|\mathbb{F}_7| = 7$

Observation: For any prime number p, we can define a finite field $\mathbb{F}_p = \{0, 1, 2, \ldots, p-1\}$.

To make this work: "Redefine" "+" and "$\cdot$"

MODULO ARITHMETIC: HOW TO CALCULATE WITHIN A FINITE FIELD

$$(2+39)\%12 = 41\%12$$

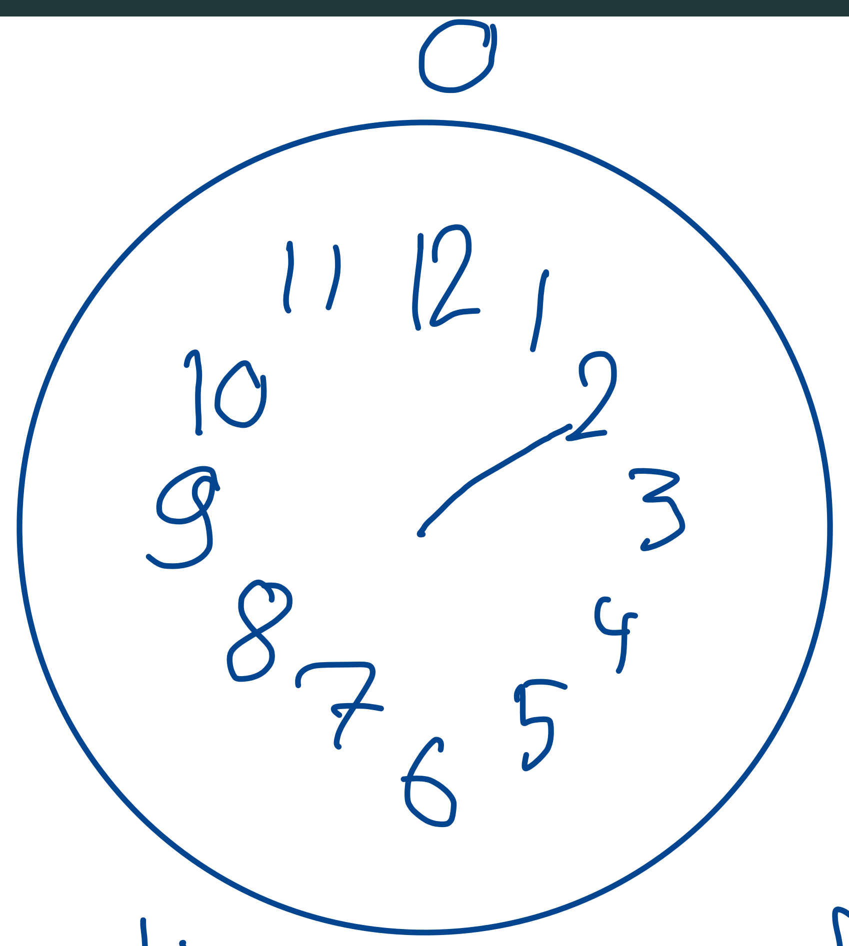
"mod"

$$= (3\cdot 12+5)\%12$$

$$= \underbrace{3\cdot 12\%12}_{0} + \underbrace{5\%12}_{5}$$

as multiple of 12

$$= 5$$



It is 2 o'clock.
What time is it 39 hours later?

$\Rightarrow$ 5 o'clock

Use convention $n\%n = 0$

↳ Rule: For any integer k, n: $(k\cdot n)\%n = 0$

Multiplication? E.g.: $3\cdot 4 = (\underbrace{4+4+4}_{3\text{ times}})$

ADDITION WITHIN FINITE FIELDS

Recall $\mathbb{F}_p = \{0, 1, 2, \dots, p-1\}$, where p prime

For $a, b \in \mathbb{F}_p$, we define $a + b := a +_f b := (a+b) \% p$

E.g.: $\triangleright$ $a = 3$, $b = 5$, $p = 11$: $a+b = (3+5) \% 11 = 8 \% 11 = 8$

$\triangleright$ $a = 3$, $b = 10$, $p = 11$: $a+b = (3+10) \% 11 = 13 \% 11 = 2$

How about additive identity? $\rightarrow$ $(a +_f 0) = (a+0) \% p = a \% p = a$ ✓

— " — additive inverse? $\rightarrow$ $-a \overset{(*)}{=} (p-a) \% p$

is additive inverse since

$$a +_f (-a) = (a + (p-a)) \% p$$
$$= (p+0) \% p = 0$$ ✓

ADDITION WITHIN FINITE FIELDS

Accordingly, we can define subtraction within finite fields:

For any $a, b \in \mathbb{F}_p$: $\boxed{a -_f b := a +_f \underbrace{(-b)_f}_{(*)} := [a + (p-b)] \% p.}$

MULTIPLICATION WITHIN FINITE FIELDS

We need:

1) $a \cdot_f b \in F_p$ if $a, b \in F_p$
2) For all $a \in F_p$, exists $1 \in F_p$ s.t. $a \cdot_f 1 = a$
3) $\boxed{\text{For all } a \in F_p, \text{ exists } a^{-1} \in F_p \text{ s.t. } a \cdot_f a^{-1} = 1}$

for this to hold, we need p prime (or order $= p^n$ with n integer)

We define multiplication within a finite field:

For any $a, b \in F_p$,

$$\boxed{a \cdot_f b := \underbrace{a +_f a +_f \dots +_f a}_{b \text{ times}}}$$

as integer $\rightarrow$ b

Examples: 1) $a = 5$, $b = 3$, $p = 11$

$$5 \cdot_f 3 = (5 +_f 5) +_f 5 = [(5+5)\% 11] +_f 5 = 10 +_f 5 = (10+5)\% 11$$

$$= 15 \% 11 = 4$$