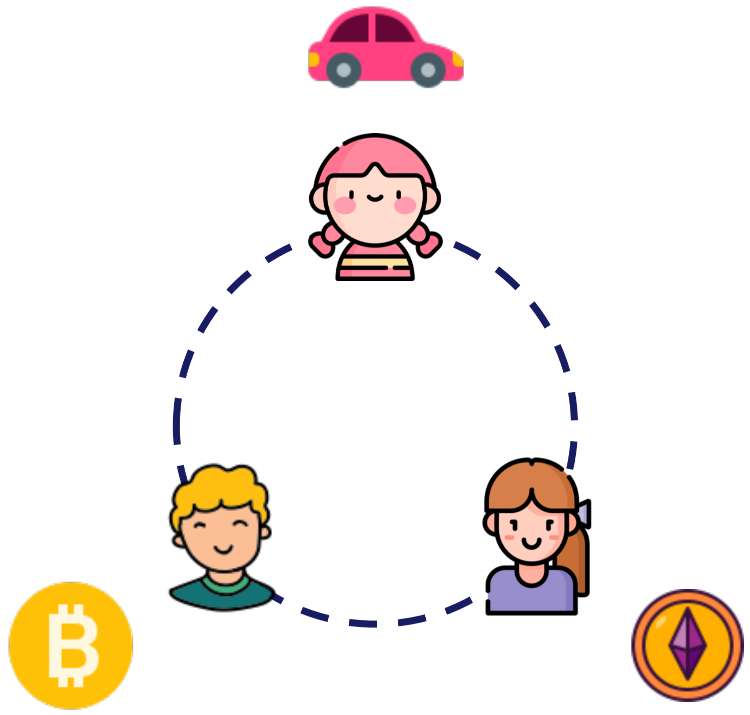
CE7490 Paper Presentation: **Atomic Cross-Chain Swaps**

Author: Maurice Herlihy, Brown University

Presenters: Zhang Siyue, Liu Chaoqun



Problem Statement (1)

- The future is envisioned with many blockchains
- People needs to exchange digital assets from separate blockchains
- Besides,
  - Cross-chain update for blockchain sharding
  - Software upgrade for the decentralized distributed system
- We need an atomic swap protocol

Problem Statement (2)

An *atomic swap protocol* guarantees

1) If all parties conform to the protocol, then all swaps take place
2) If some parties deviate from the protocol, then no conforming party ends up worse off
3) No coalition has an incentive to deviate from the protocol

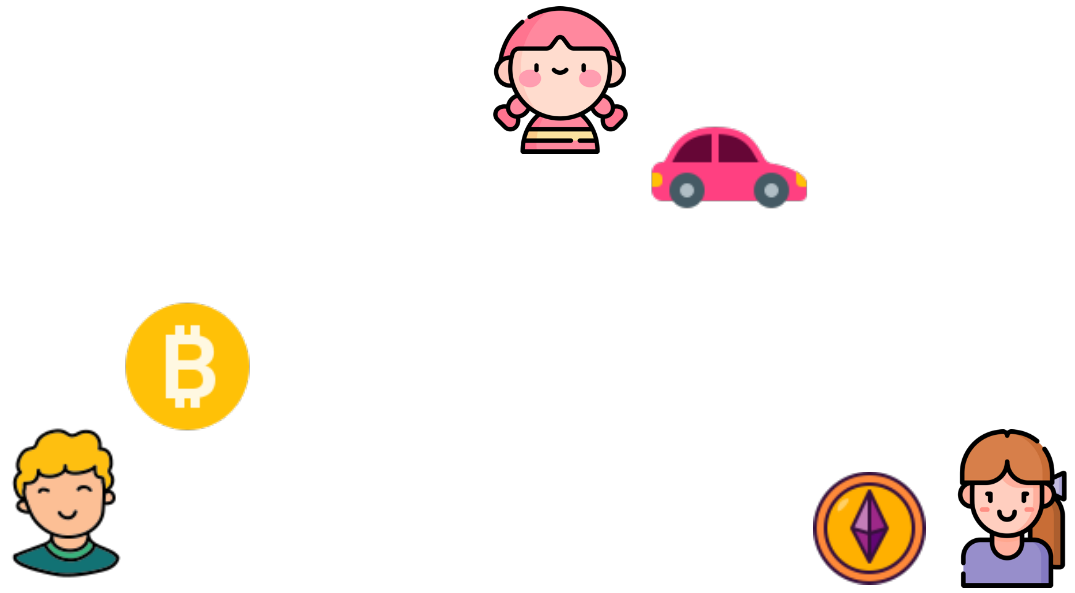
Key Questions:

When such swaps possible? How to implement? What do they cost?

A Simple Three-Way Swap Protocol

Carol:

- I want to sell my car for BTCs.



Bob:

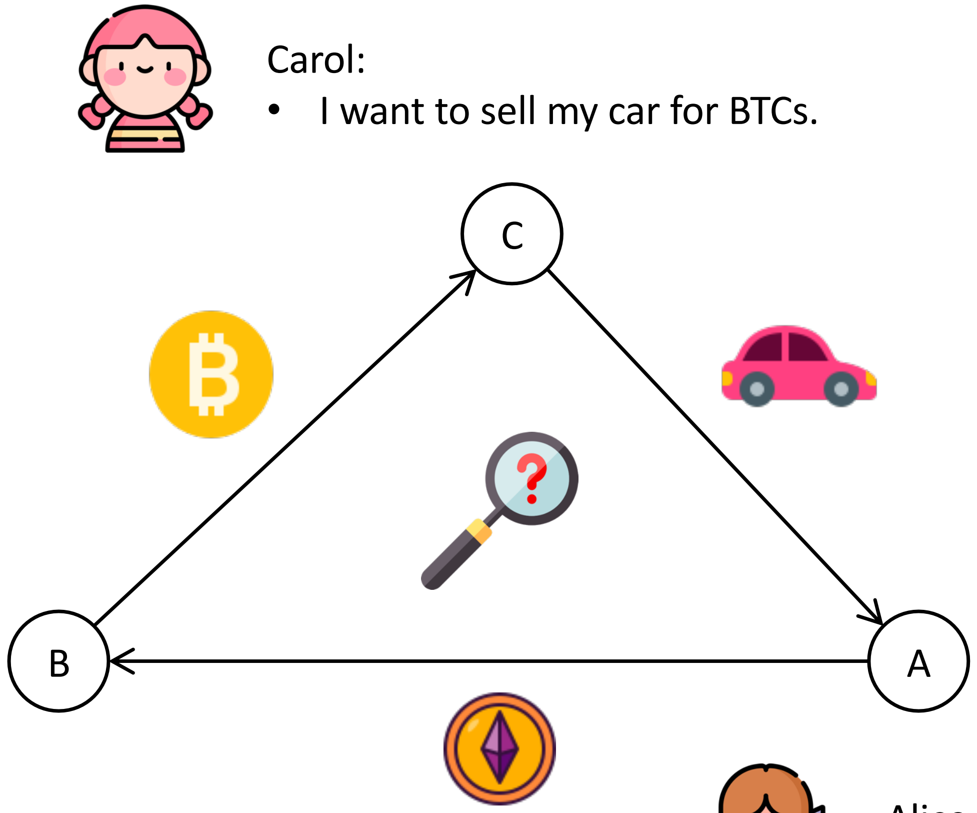
- I want to buy ETHs using BTCs.

Alice:

- I want to buy Carol's car using ETHs.

A Simple Three-Way Swap Protocol

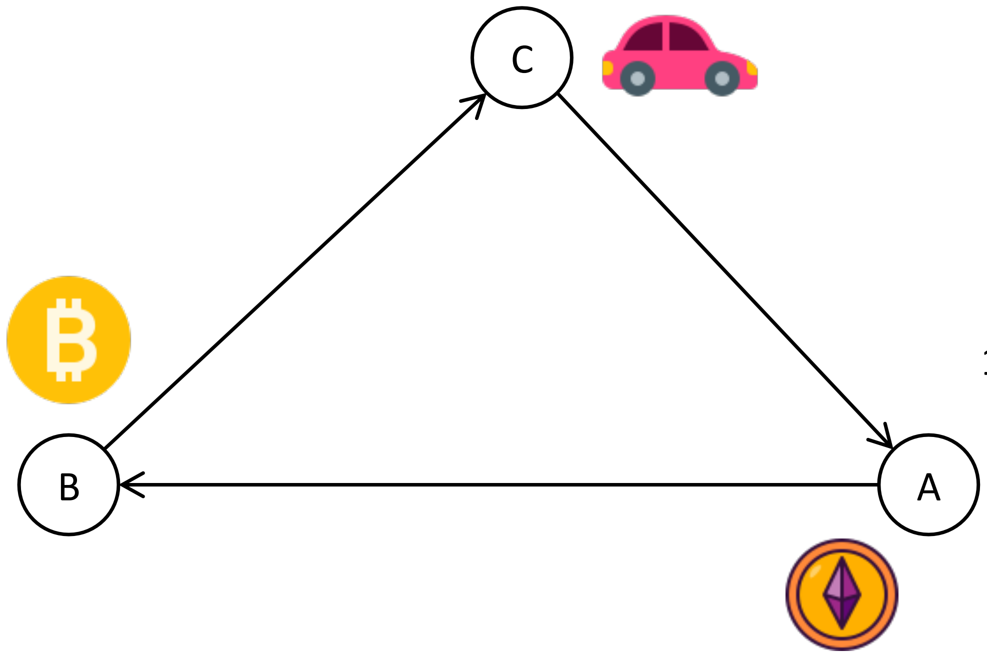
Carol:
- I want to sell my car for BTCs.



Bob:
- I want to buy ETHs using BTCs.

Alice:
- I want to buy Carol's car using ETHs.

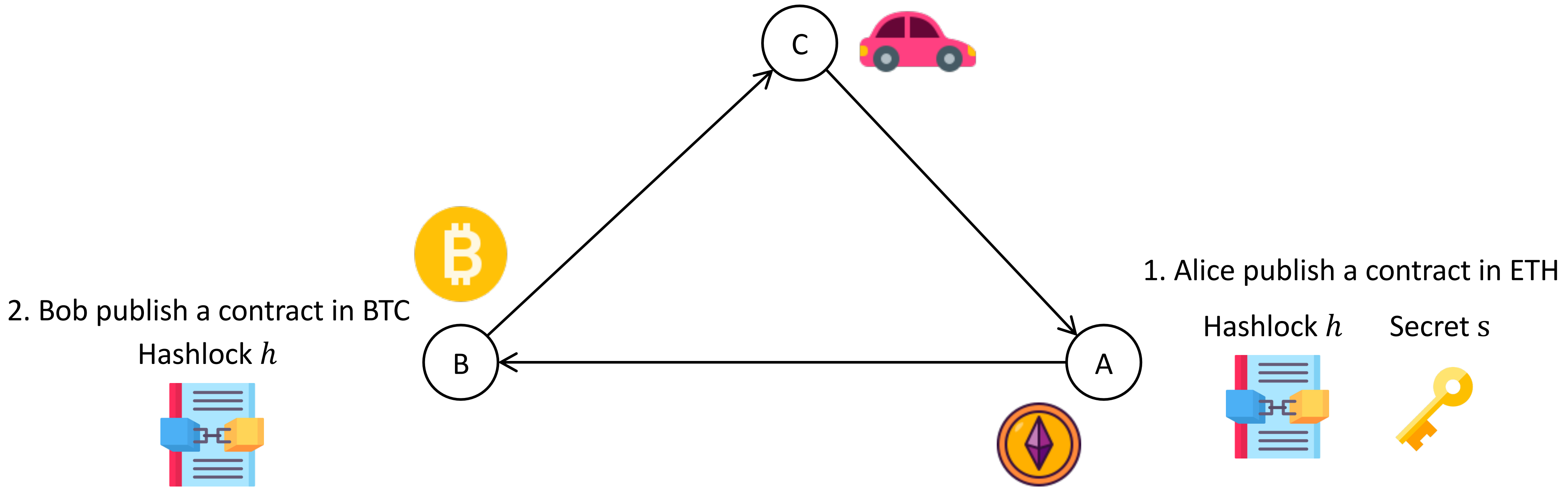
A Simple Three-Way Swap Protocol (1)



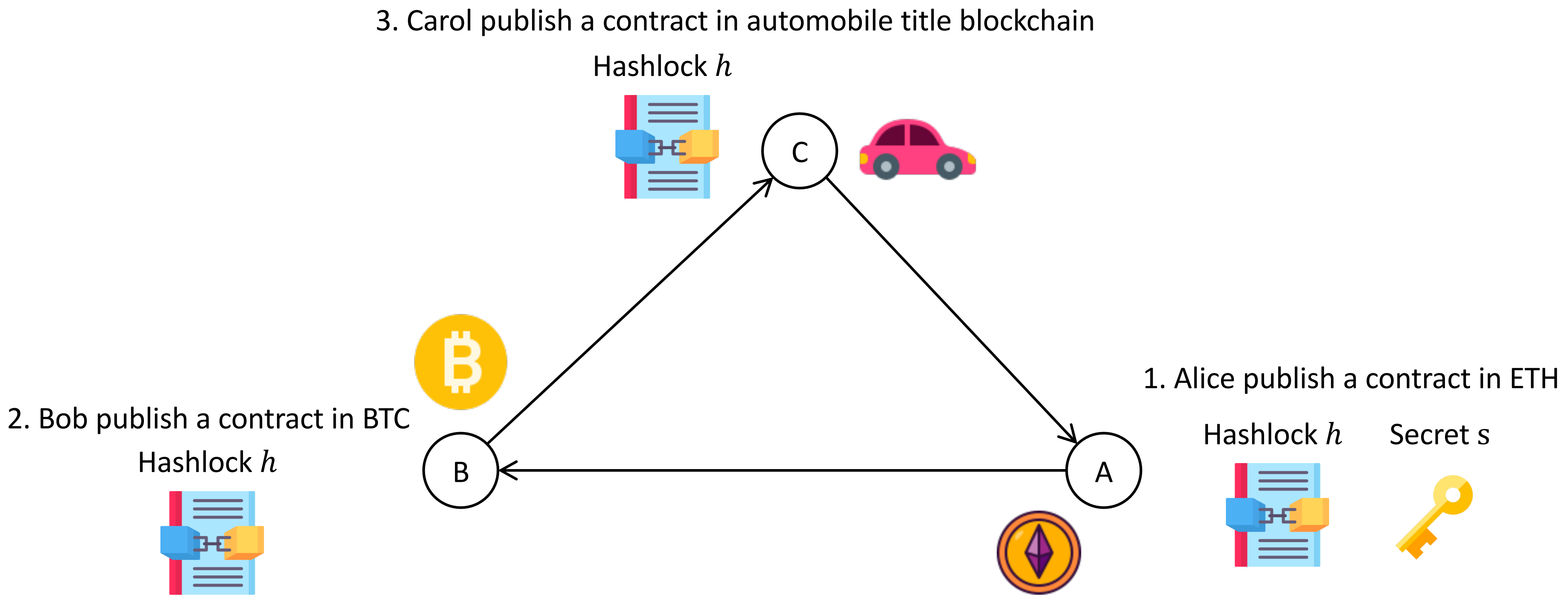
1. Alice publish a contract in ETH

Hashlock h Secret s

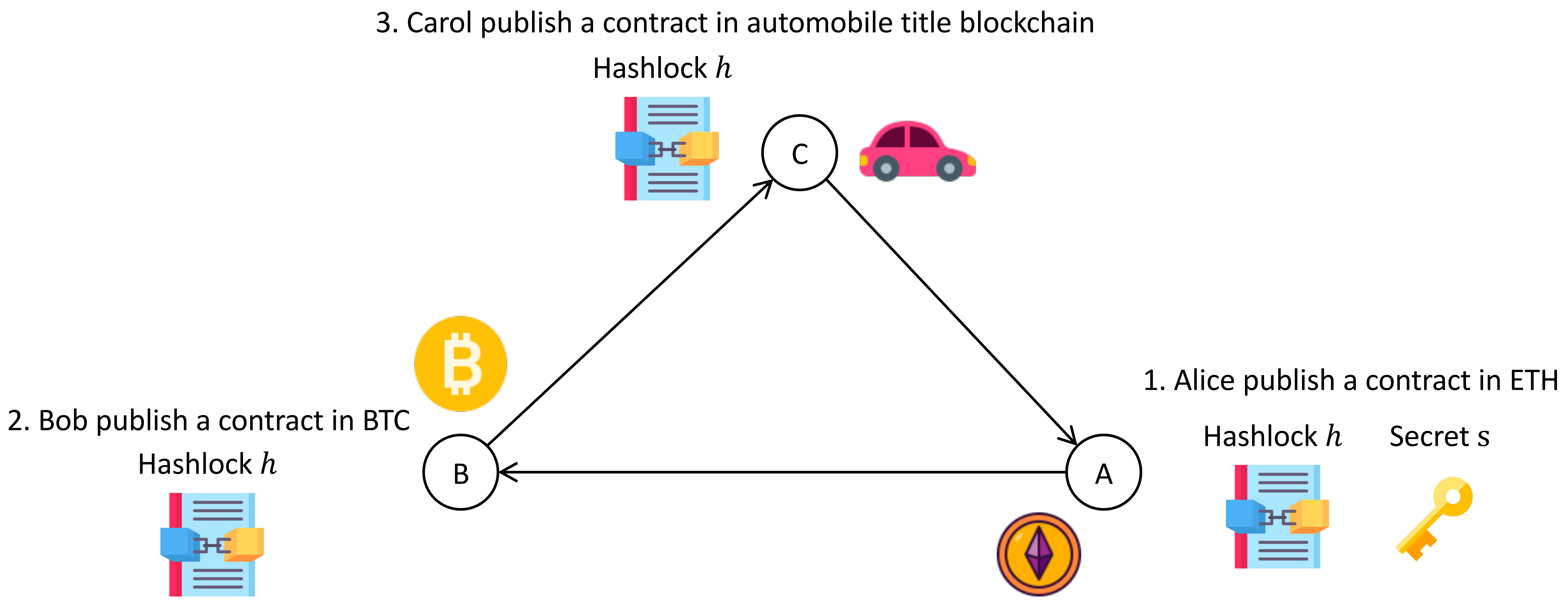
A Simple Three-Way Swap Protocol (2)


C
B
A
2. Bob publish a contract in BTC
Hashlock h
1. Alice publish a contract in ETH
Hashlock h
Secret s

A Simple Three-Way Swap Protocol (3)

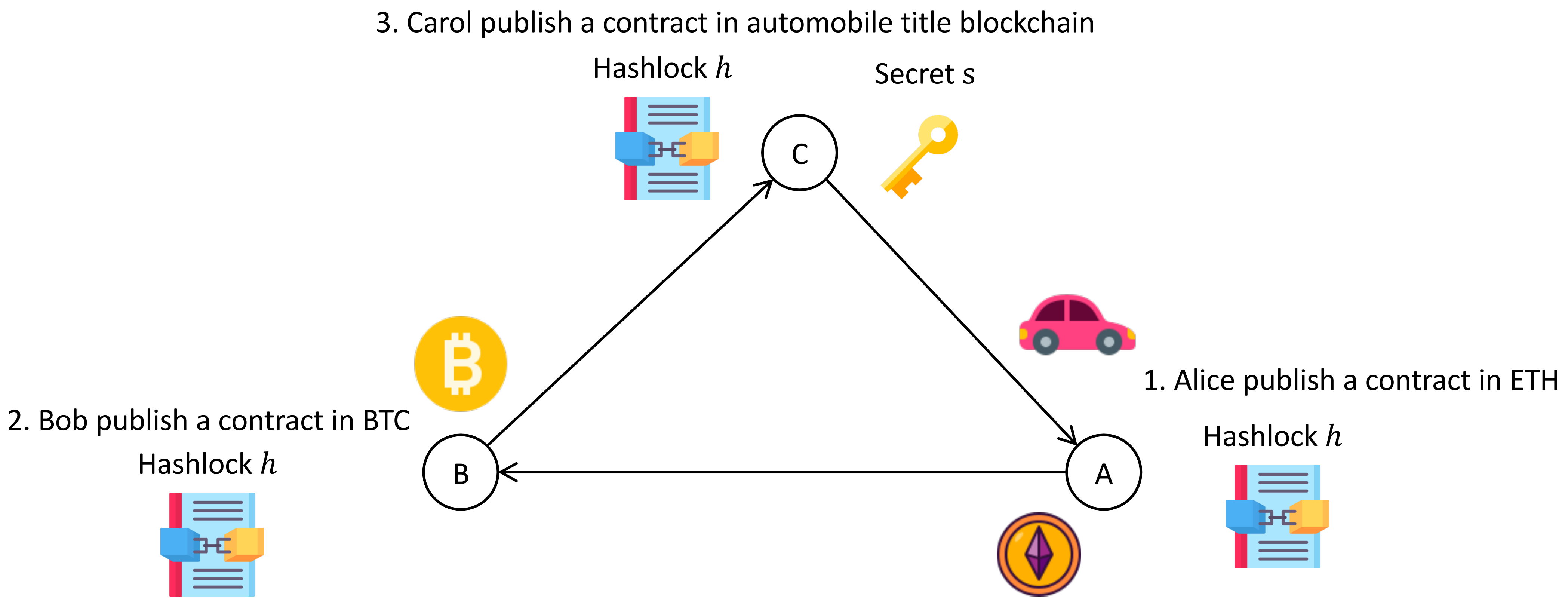

3. Carol publish a contract in automobile title blockchain
Hashlock h
C
2. Bob publish a contract in BTC
Hashlock h
B
A
1. Alice publish a contract in ETH
Hashlock h
Secret s

A Simple Three-Way Swap Protocol (4)

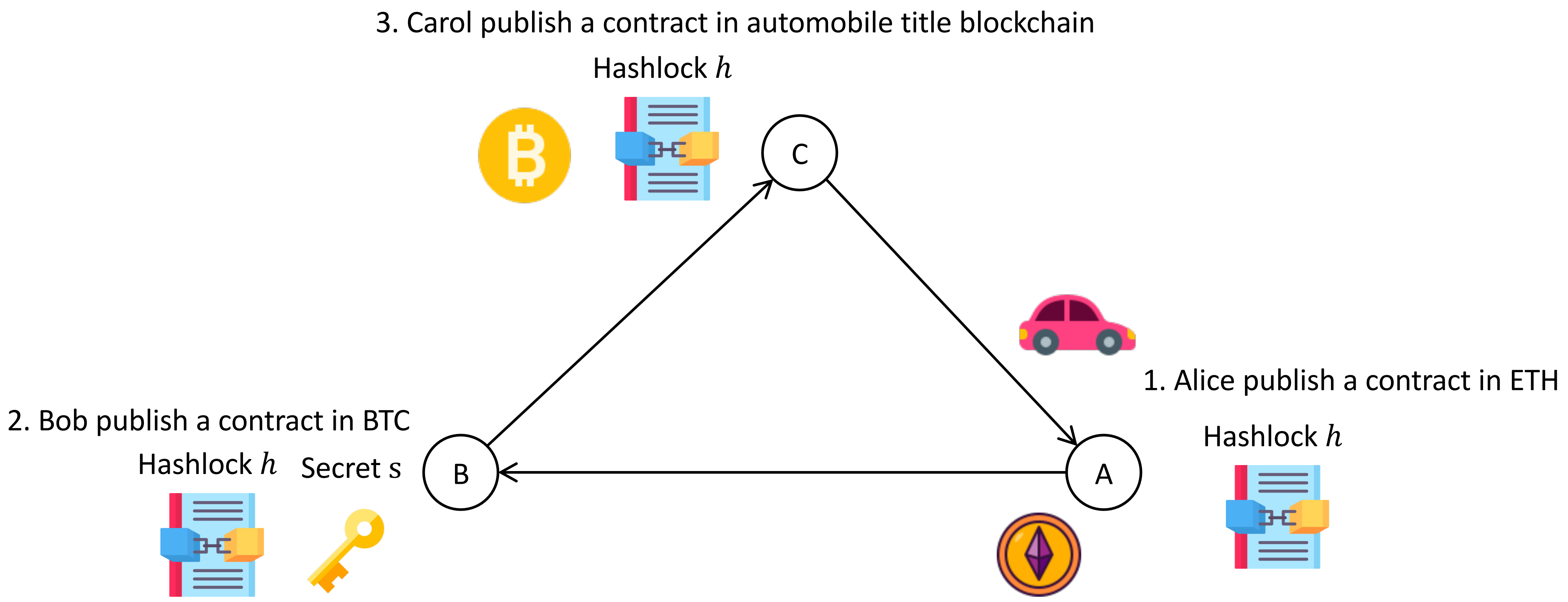


3. Carol publish a contract in automobile title blockchain
Hashlock h
C
2. Bob publish a contract in BTC
Hashlock h
B
1. Alice publish a contract in ETH
Hashlock h
Secret s
A

A Simple Three-Way Swap Protocol (5)

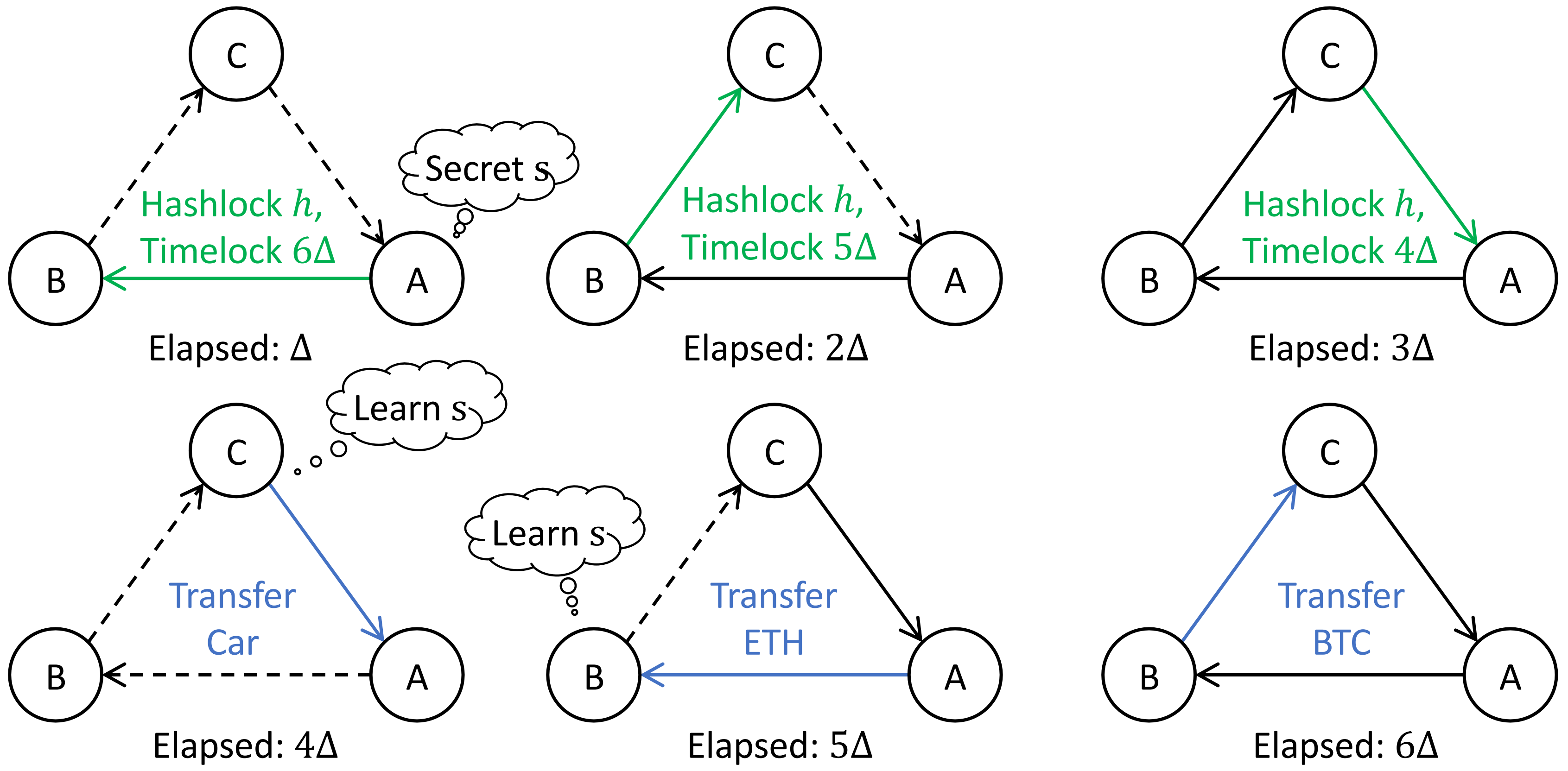

3. Carol publish a contract in automobile title blockchain
Hashlock h
Secret s
C
2. Bob publish a contract in BTC
Hashlock h
B
1. Alice publish a contract in ETH
Hashlock h
A

A Simple Three-Way Swap Protocol (6)

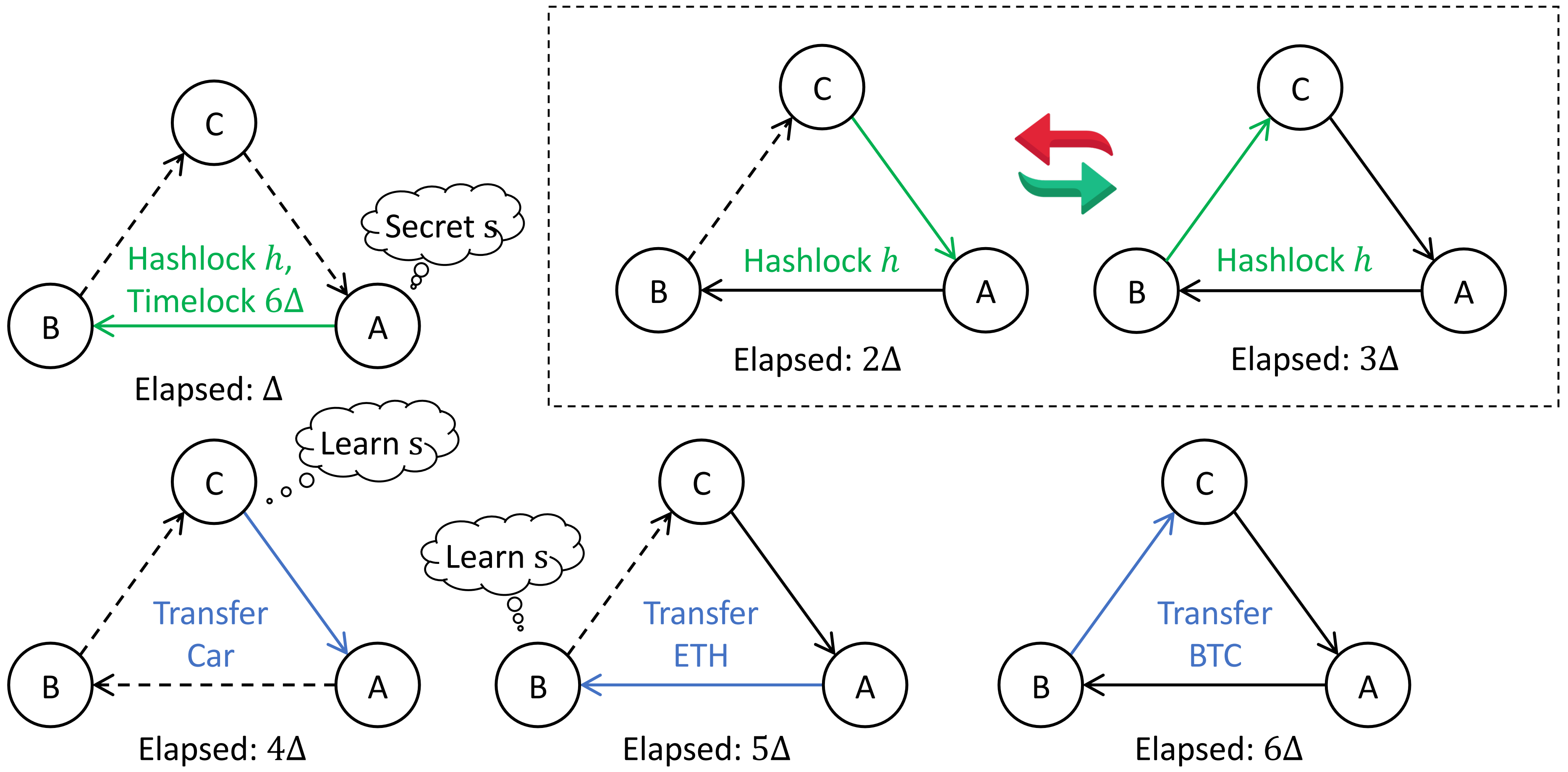

3. Carol publish a contract in automobile title blockchain
Hashlock h
C
1. Alice publish a contract in ETH
Hashlock h
A
2. Bob publish a contract in BTC
Hashlock h
Secret s
B

A Simple Three-Way Swap Protocol (7)

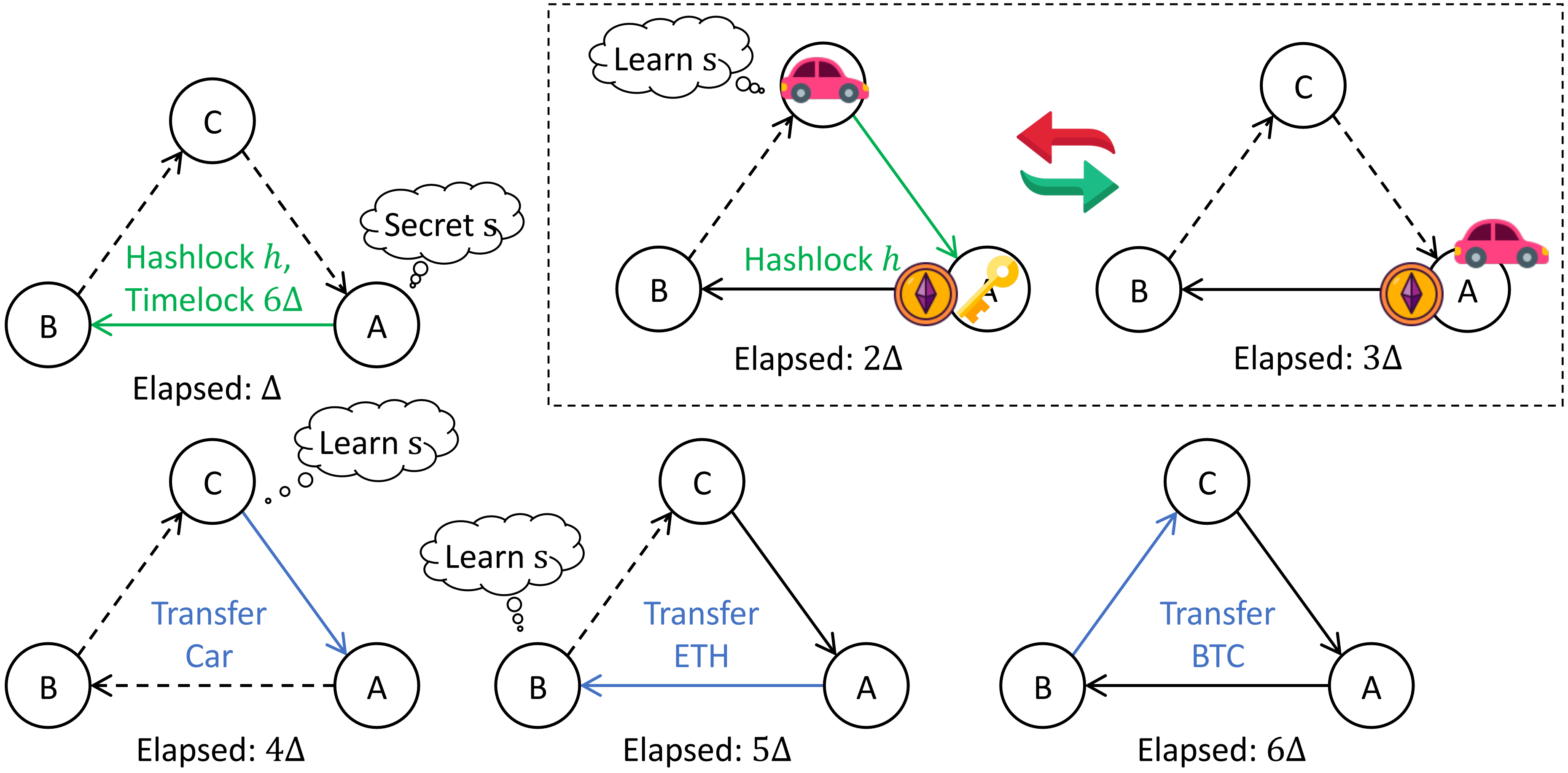
Δ: enough time for one to publish & notice a smart contract



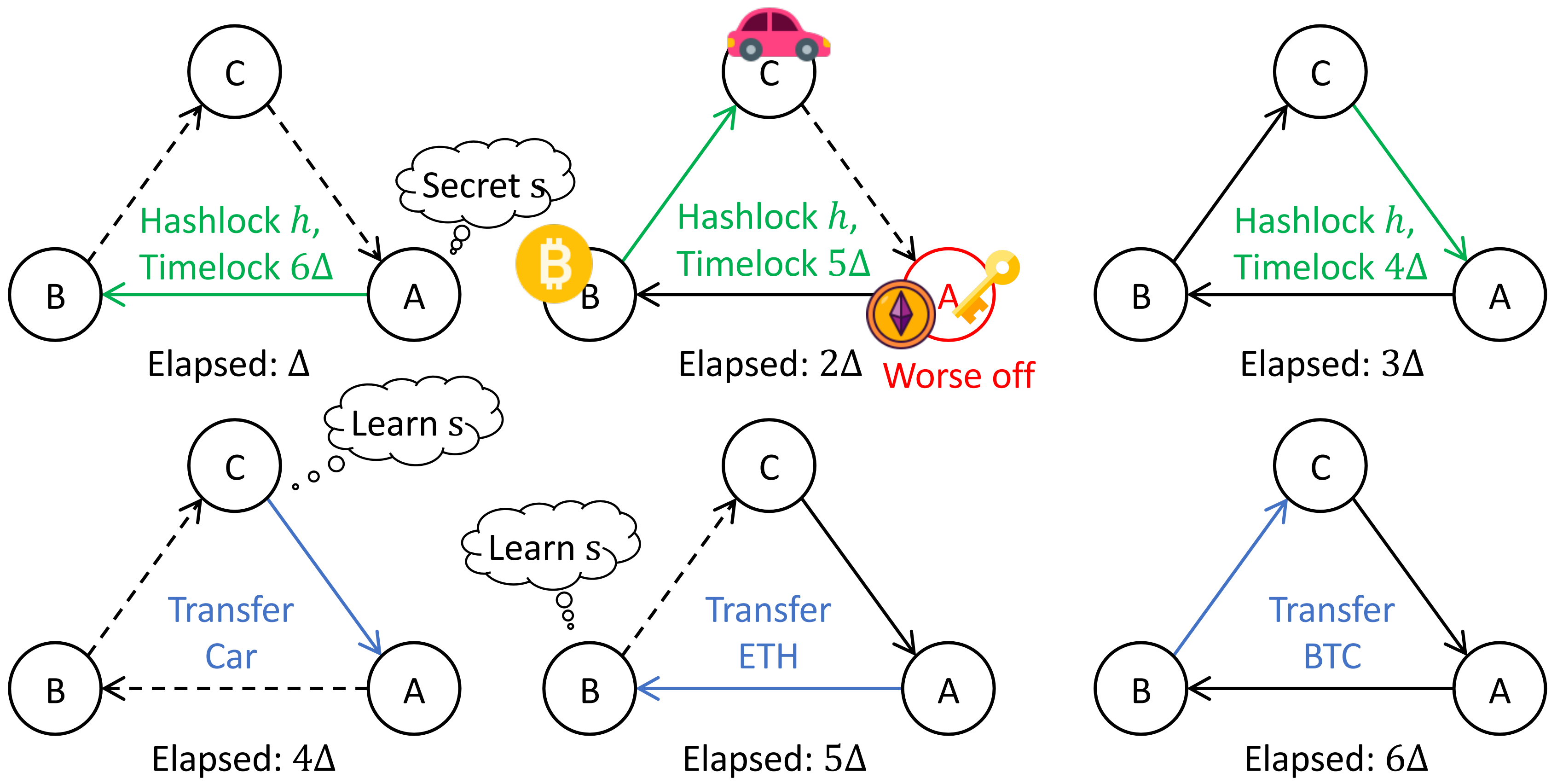
A Simple Three-Way Swap Protocol – Order Matters (1)



A Simple Three-Way Swap Protocol – Order Matters (2)



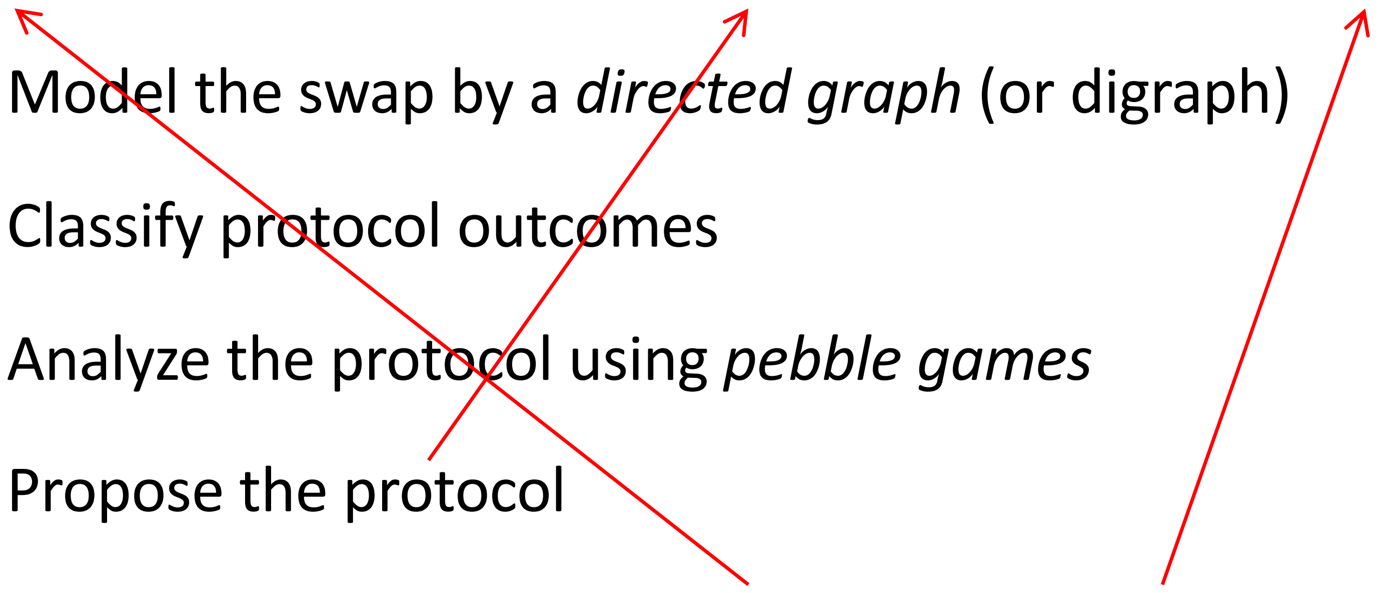
A Simple Three-Way Swap Protocol – Irrational Behavior



Problem Statement (3)

Key Questions:

When such swaps possible? How to implement? What do they cost?

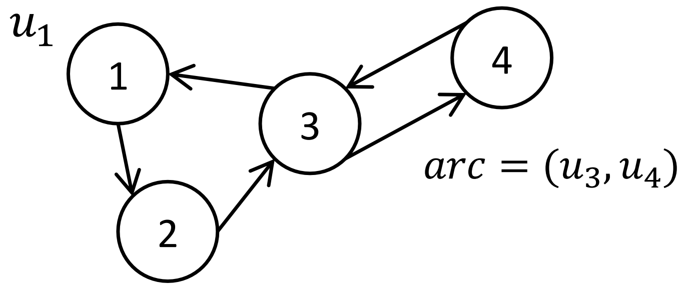


1. Model the swap by a *directed graph* (or digraph)
2. Classify protocol outcomes
3. Analyze the protocol using *pebble games*
4. Propose the protocol
5. Conclude with existence condition, time and space complexity

1. Digraph Model

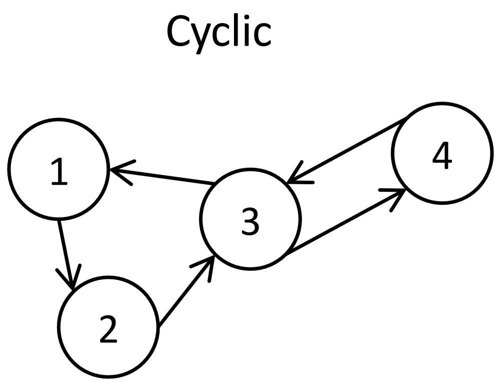
- A digraph $\mathcal{D}$ is a pair (V, A), V: vertexes, A: arcs
- A path p is a sequence of vertexes $(u_0, \dots, u_l)$, with length $l = |p|$
- $D(u, v)$ is the length of the longest path from vertex u to vertex v
- $diam(\mathcal{D})$ is the length of longest path from any vertex to any other
- *Strongly connected*:

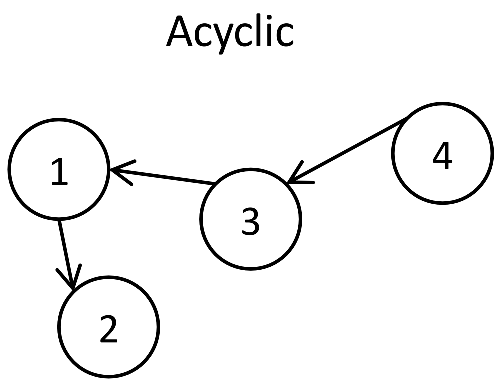
    for every pair u, v, u is reachable from v, v is reachable from u

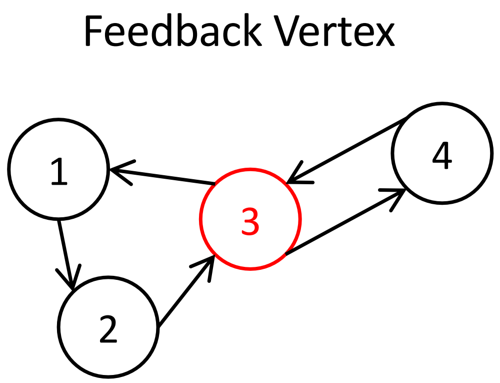


1. Digraph Model

- Acyclic if $\mathcal{D}$ has no cycles
- A *feedback vertex* set is a subset of V whose deletion leaves $\mathcal{D}$ acyclic
- Transposing $\mathcal{D}^T$ is reversing all arcs
- If $\mathcal{D}$ is strongly connected, $\mathcal{D}^T$ is strongly connected too
- Any feedback vertex set for $\mathcal{D}$ is also a feedback vertex set for $\mathcal{D}^T$







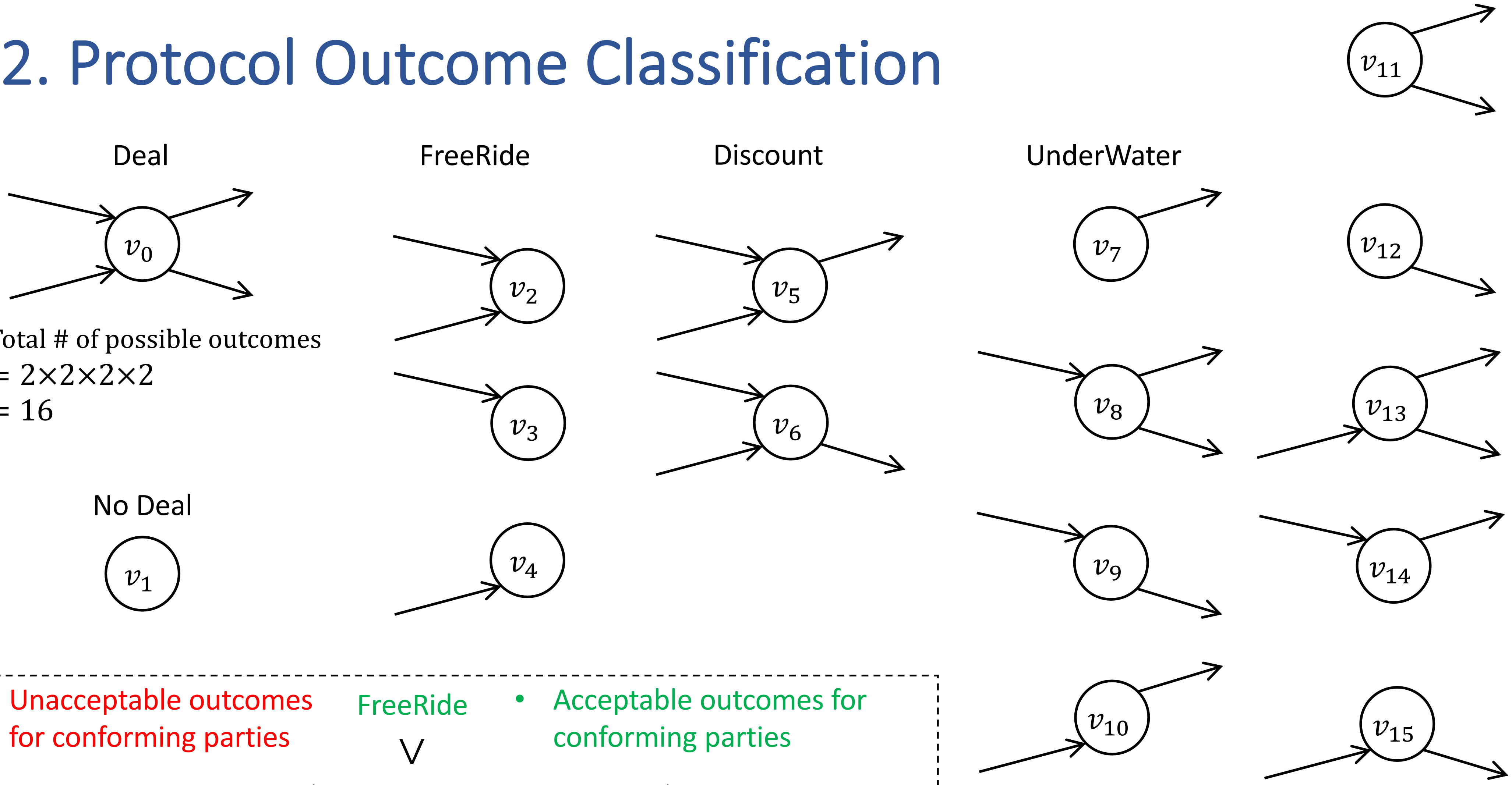
Problem Statement (3)

Key Questions:

When such swaps possible? How to implement? What do they cost?

1. Model the swap by a *directed graph* (or digraph)
2. **Classify protocol outcomes**
3. Analyze the protocol using *pebble games*
4. Propose the protocol
5. Conclude with existence condition, time and space complexity

2. Protocol Outcome Classification



Total # of possible outcomes
$= 2\times2\times2\times2$
$= 16$

- Unacceptable outcomes for conforming parties
- Acceptable outcomes for conforming parties

FreeRide
$\vee$
UnderWater $<$ No Deal $<$ Deal $<$ Discount

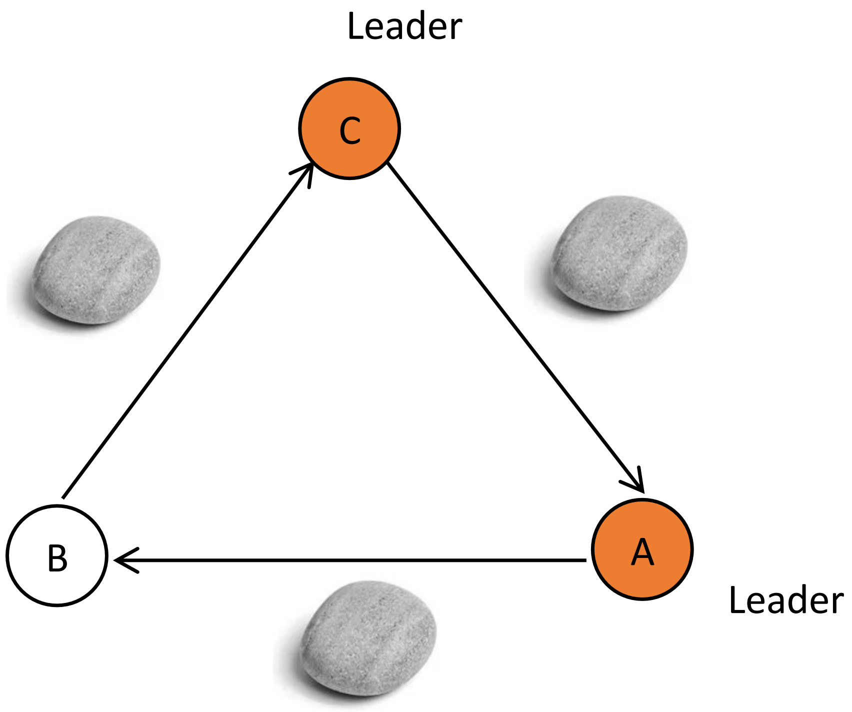
Problem Statement (3)

Key Questions:

When such swaps possible? How to implement? What do they cost?

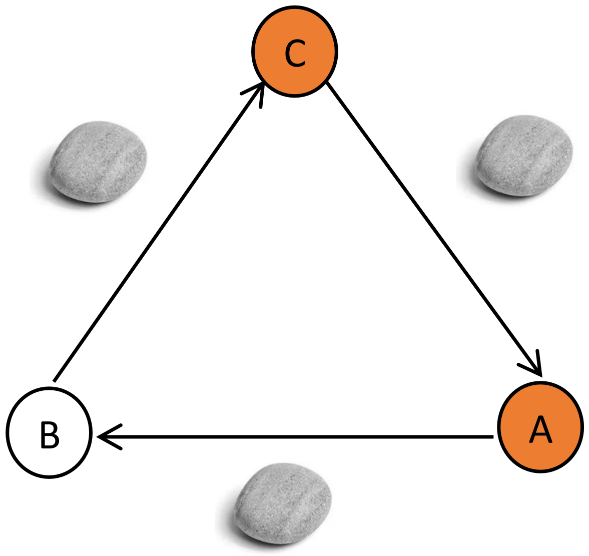
1. Model the swap by a *directed graph* (or digraph)
2. Classify protocol outcomes
3. **Analyze the protocol using *pebble games***
4. Propose the protocol
5. Conclude with existence condition, time and space complexity

3. Pebble Games (lazy game)



Lemma 4.1. In the lazy game, every arc in $\mathcal{D}$ eventually has a pebble.

3. Pebble Games (eager game)



Lemma 4.2. In the eager game, every arc in $\mathcal{D}$ eventually has a pebble.
Lemma 4.3. In both pebble games, every arc will have a pebble in time at most $diam(\mathcal{D}) \cdot \Delta$ from when the game started.

Problem Statement (3)

Key Questions:

When such swaps possible?   How to implement?   What do they cost?

1. Model the swap by a *directed graph* (or digraph)
2. Classify protocol outcomes
3. Analyze the protocol using *pebble games*
4. **Propose the protocol**
5. Conclude with existence condition, time and space complexity

4. The Protocol: Swap Contract

```
contract Swap {
  Asset asset;                     /* asset to be transferred or refunded */
  Digraph digraph;                 /* swap digraph */
  address[] leaders;               /* leaders */
  address party;                   /* transfer asset from */
  address counterparty;            /* transfer asset to */
  uint[] timelock;                 /* vector of timelocks */
  uint[] hashlock;                 /* vector of hashlocks */
  bool[] unlocked;                 /* which hashlocks unlocked? */
  uint   start;                    /* protocol starting time */
  /* constructor */
  function Swap (Asset _asset;  /* asset to be transferred or refunded */
                 Digraph   _digraph;     /* swap digraph */
                 address[] _leaders;     /* leaders */
                 address   _party;       /* transfer asset from */
                 address   _counterparty; /* transfer asset to */
                 uint[]    _timelock;    /* vector of timelocks */
                 uint[]    _hashlock;    /* vector of hashlocks */
                 uint      _start        /* protocol starting time */
                 ) {
    asset = _asset;                                /* copy */
    party = _party; counterparty = _counterparty;  /* copy */
    timelock = _timelock; hashlock = _hashlock;    /* copy */
    unlocked = [false, ..., false];                /* all unlocked */
  }
  function unlock (int i, uint s, Path path, Sig sig) {
    require (msg.sender == counterparty);  /* only from counterparty */
    if (now < start + (diam(digraph) + |path|) * Δ /* hashkey still valid? */
        && hashlock[i] == H(s)                    /* secret correct? */
        && isPath(path, digraph, leader[i], counterparty) /* path valid? */
        && verifySigs(sig, s, path) { /* signatures valid? */
      unlocked[i] = true;
    }
  }
  function refund () {
    require (msg.sender == party);  /* only from party */
    if (any hashlock unlocked and timed out) {
      transfer asset to party;
      halt;
    }
  }
  function claim () {
    require (msg.sender == counterparty);  /* only from counterparty */
    if (every hashlock unlocked) {
      transfer asset to counterparty;
      halt;
    }
  }
}
```

4. The Protocol

Phase one:

For leaders:

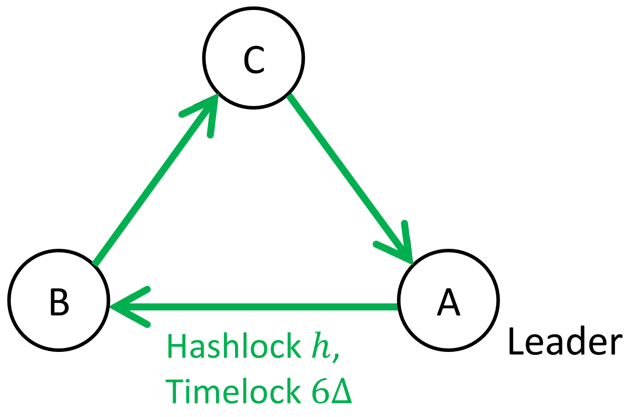
(1) Publish a contract on every arc leaving the leader, then
(2) wait until contracts have been published on all arcs entering the leader.

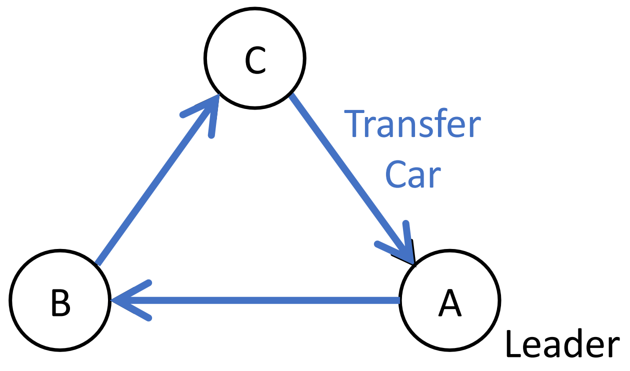
For followers:

(1) wait until correct contracts have been published on all arcs entering the vertex, then
(2) publish a contract on every arc leaving the vertex.

Phase two:

The parties disseminate secrets via hashkeys in the opposite direction of the arcs.





Problem Statement (3)

Key Questions:

When such swaps possible? How to implement? What do they cost?

1. Model the swap by a *directed graph* (or digraph)
2. Classify protocol outcomes
3. Analyze the protocol using *pebble games*
4. Propose the protocol
5. **Conclude with existence condition, time and space complexity**

5. Conclusion

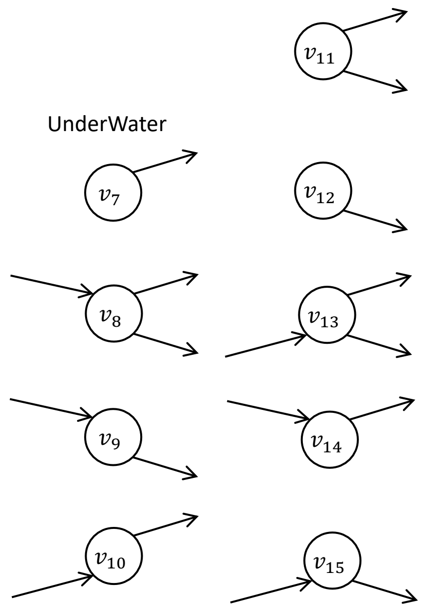
1. For any pair $(\mathcal{D}, L)$, where $\mathcal{D} = (V, A)$ is a strongly-connected digraph and $L \subset V$ a feedback vertex set, the paper gives an atomic cross-chain swap protocol using hashed timelock contracts, where vertexes in L generate the hashlocked secrets.
2. No such protocol is possible if $\mathcal{D}$ is not strongly connected, or if $\mathcal{D}$ is strongly connected but L is not a feedback vertex set.
3. The protocol has time complexity $\mathcal{O}(diam(\mathcal{D}))$ and space complexity (bits stored on all blockchains) $\mathcal{O}(|A|^2)$.

Limitations (1)

1. The swap protocol is still vulnerable to denial-of-service attacks

-> Post bonds following a failed swap and examine the blockchains to determine who was at fault

2. The classification of outcomes (e.g., UnderWater) can be more fine-grained so that customized objective functions could be proposed to accept certain outcomes



Limitations (2)

3. The leaders and their hashlocks who initiate the smart contracts are common knowledge among the participants

-> Design protocol for constructing and propagating this information dynamically

4. No privacy consideration

-> Hide account balances using the zero-knowledge proof, keep privacy from transaction hub

5. No opportunity cost consideration

-> When digital asset is escrowed, no conforming party could be punished to pay opportunity cost

NANYANG TECHNOLOGICAL UNIVERSITY

Thank you.

Q&A

Background

Atomic Cross-Chain Swap

An exchange of cryptocurrencies from separate blockchains, ~2017

Decentralized

Conducted between two entities without a third party's involvement

Atomic

The transaction either happens or it doesn't

Cross-Chain Swap Provider

Special wallets or exchange services are needed to conduct an atomic swap because the technique is still being developed and refined

Hash Timelock Contracts (HTLC)

A time-bound smart contract generating one hash on each end