ZeroAuction: Zero-Deposit Sealed-bid Auction via Delayed Execution

Haoqian Zhang, Michelle Yeo, Vero Estrada-Galinanes, Bryan Ford

Swiss Federal Institute of Technology Lausanne (EPFL) &

National University of Singapore (NUS)

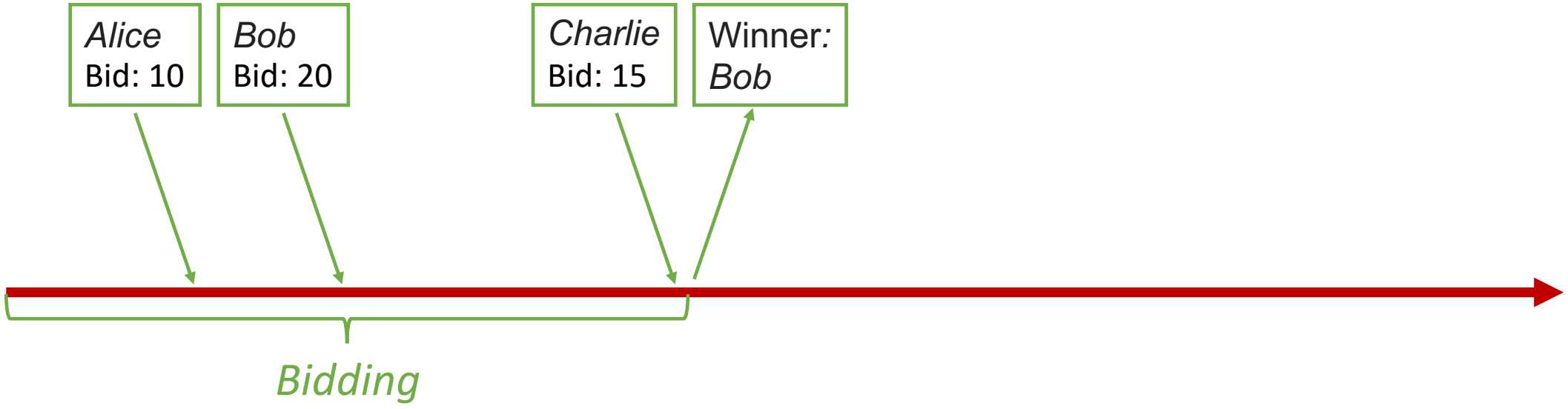
Workshop on Trusted Smart Contracts - WTSC 2024

March 8, 2024

Outline

- Sealed-bid Auction
- Impossibilities
- ZeroAuction
- Experimental Results
- Conclusion

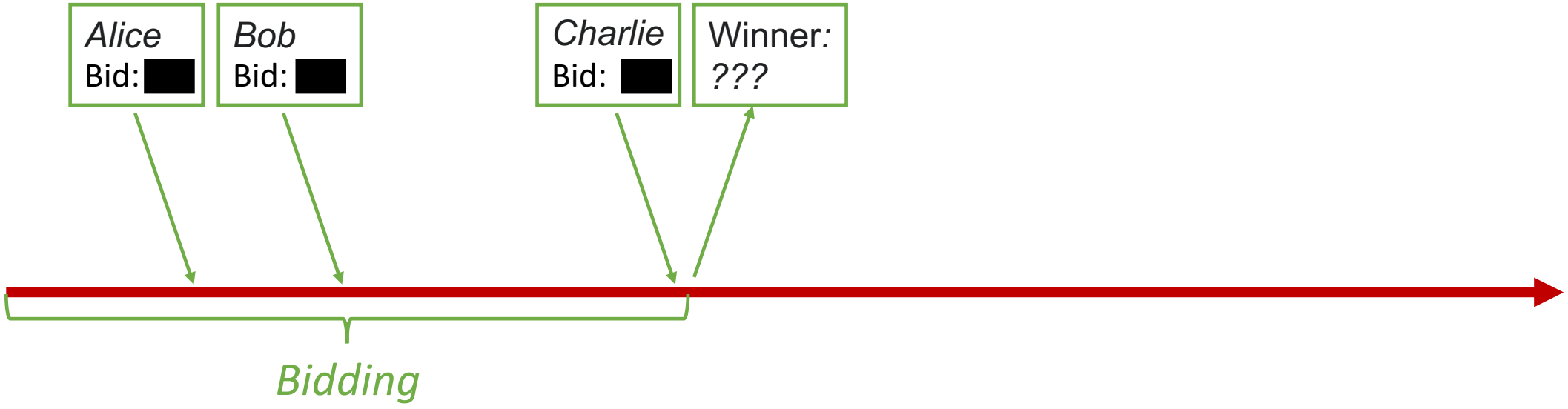
Open-bid Auction



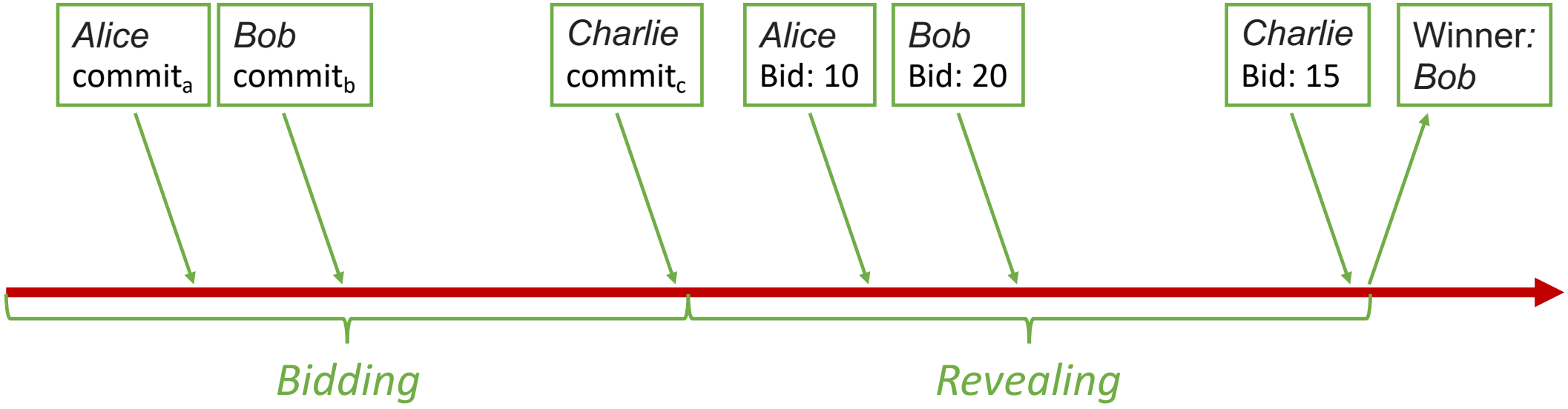
Open-bid Auction

```
1  Init Upon creating the auction smart contract:
2  |   highest ← 0, winner ← ∅
3
4  Bid Upon receiving i's bid b_i in the bidding period:
5  |   if b_i > highest then
6  |   |   Assert(i transfers b_i)
7  |   |   Distribute highest to winner when winner ≠ ∅;
8  |   |   highest ← b_i
9  |   |   winner ← i
10 |   end
```

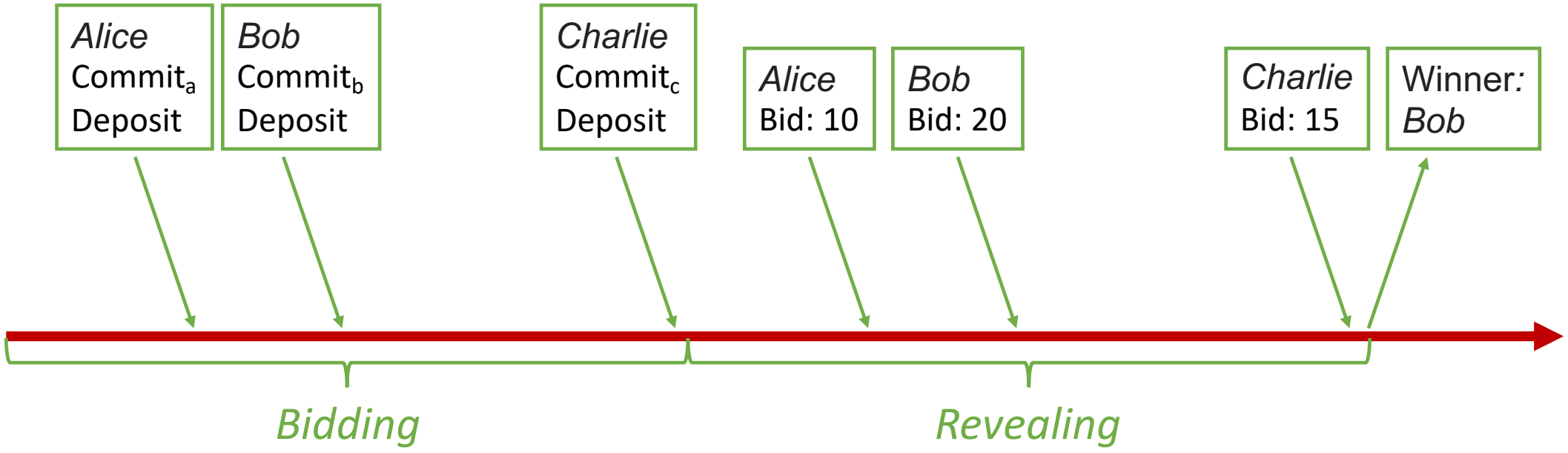
Sealed-bid Auction



Commit-and-Reveal



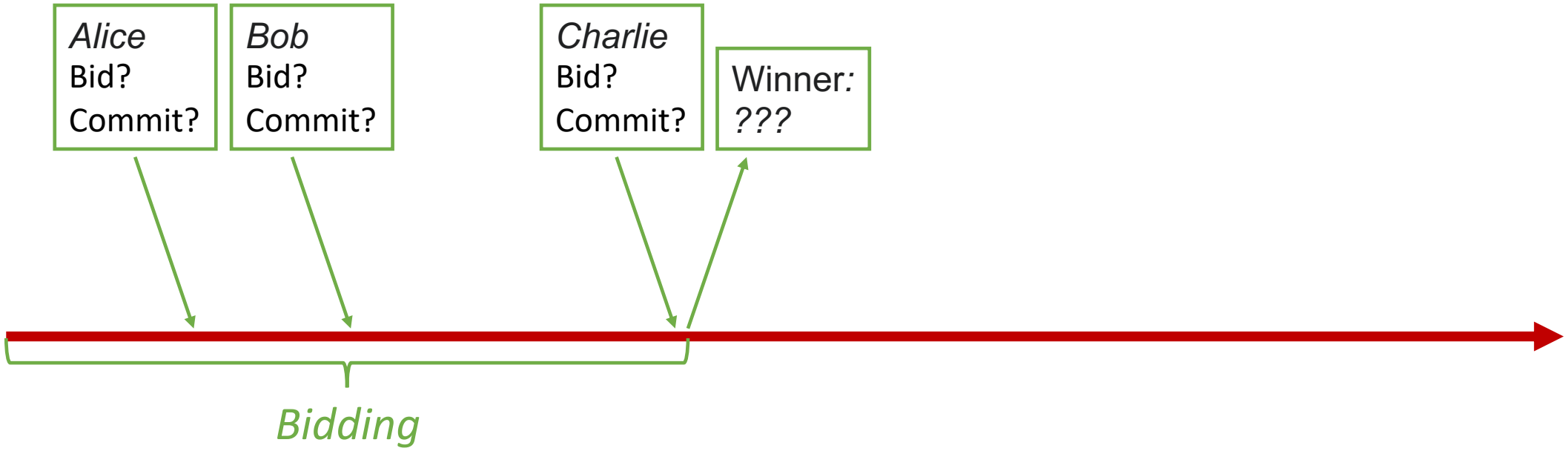
Deposit


Alice
Commit$_a$
Deposit
Bob
Commit$_b$
Deposit
Charlie
Commit$_c$
Deposit
Alice
Bid: 10
Bob
Bid: 20
Charlie
Bid: 15
Winner:
Bob
Bidding
Revealing

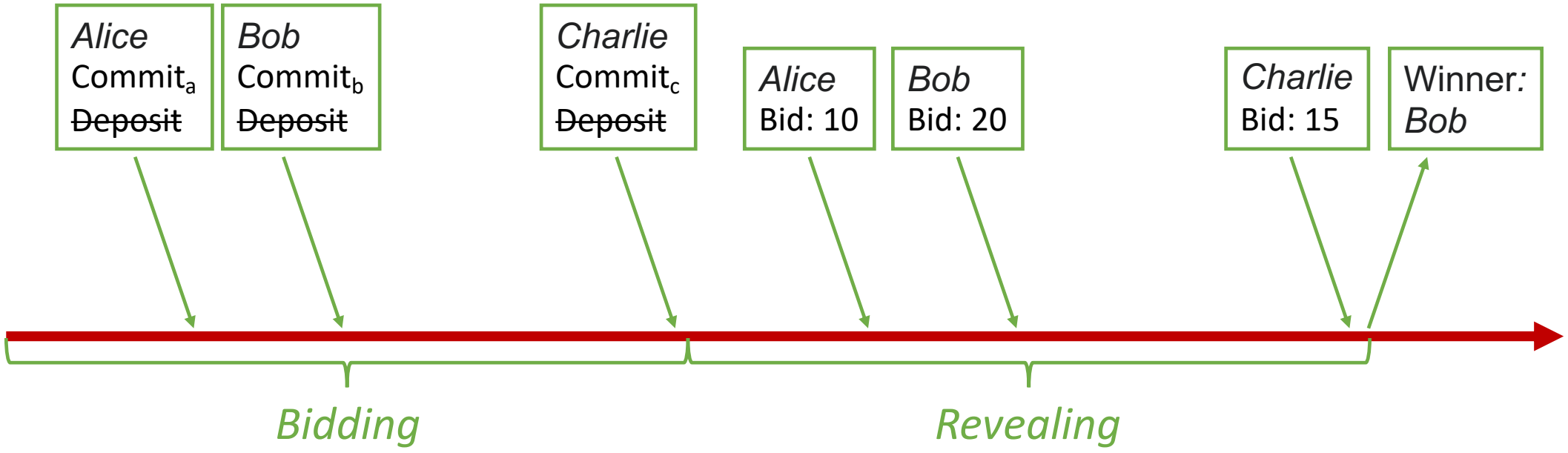
Sealed-bid Auction

```
1  Init Upon creating the auction smart contract:
2  |  Set d as required deposit for the auction
3  |  highest ← 0, winner ← ∅, hash ← []
4
5  Bid Upon receiving i's commitment c_i first time in bidding period:
6  |  Assert(i transfers d)
7  |  hash[i] ← c_i
8
9  Reveal Upon receiving i's bid b_i and salt r_i first time in revealing period:
10 |  Assert(Hash(b_i, r_i) = hash[i])
11 |  Assert(b_i ≤ d)
12 |  if b_i > highest then
13 |  |  Distribute highest to winner when winner ≠ ∅;
14 |  |  Distribute d − b_i to i
15 |  |  highest ← b_i
16 |  |  winner ← i
17 |  else
18 |  |  Distribute d to i;
19 |  end
```

Impossibility 1: One Round Communication

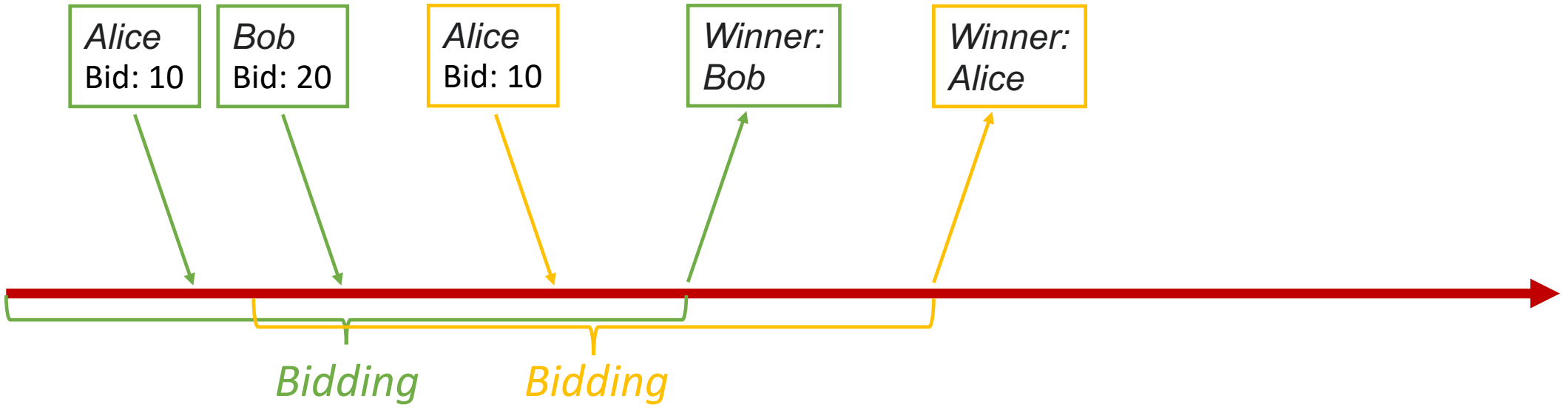


Impossibility 2: Eliminating Deposit



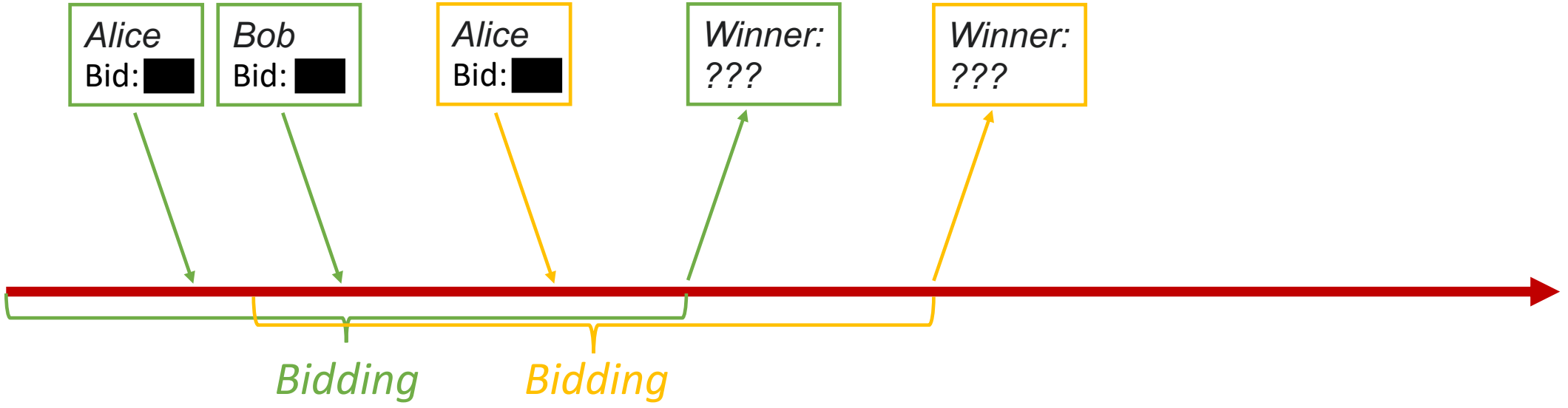
Impossibility 3: Multiple Auctions

	Balance	Auction	Auction
Alice	10	10	10
Bob	20	20	

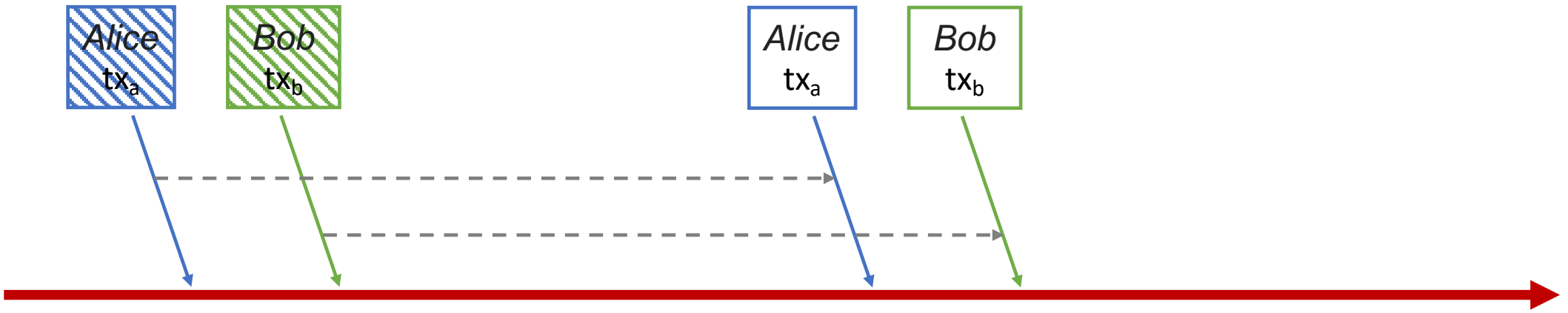


Impossibility 3: Multiple Auctions

	Balance	Auction	Auction
Alice	10	10	10
Bob	20	20	



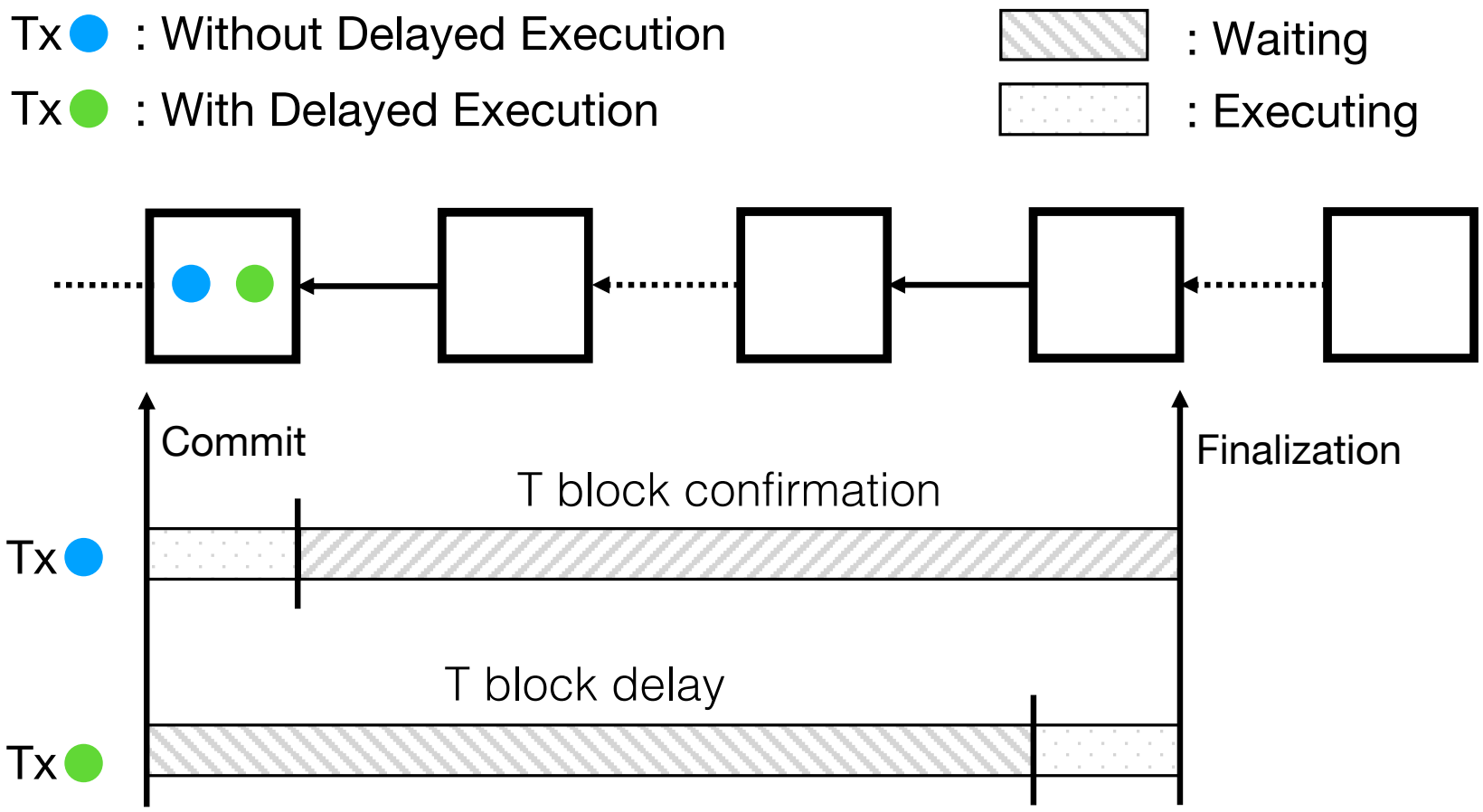
Delayed Execution*



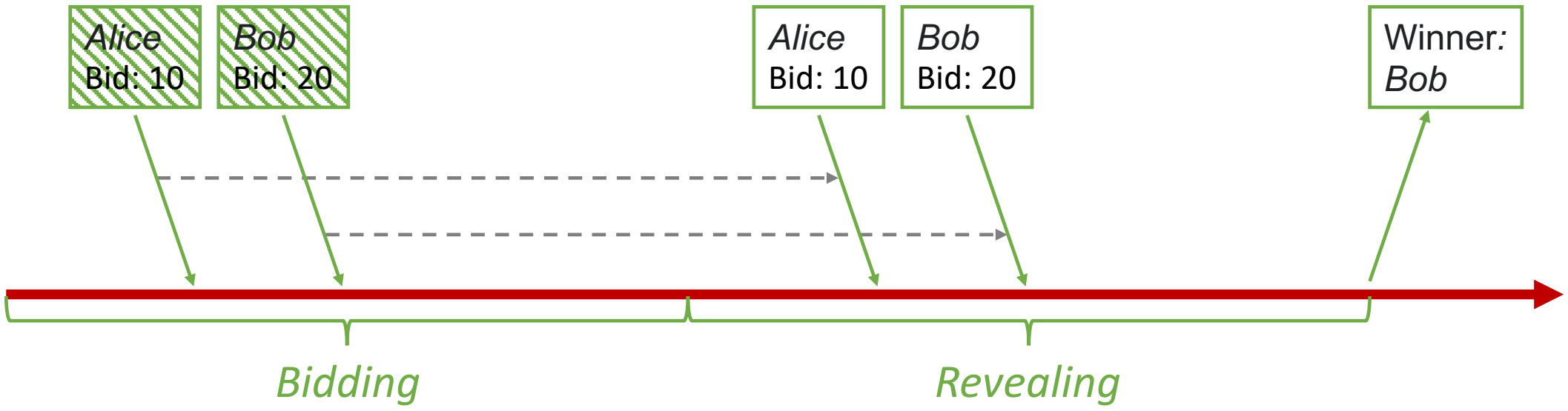
Global Delay Time

*Zhang, Haoqian, et al. "F3B: A low-overhead blockchain architecture with per-transaction front-running protection." 5th Conference on Advances in Financial Technologies (AFT 2023)

Delayed Execution without Latency Overhead



ZeroAuction



Global Delay Time

Open-bid Auction

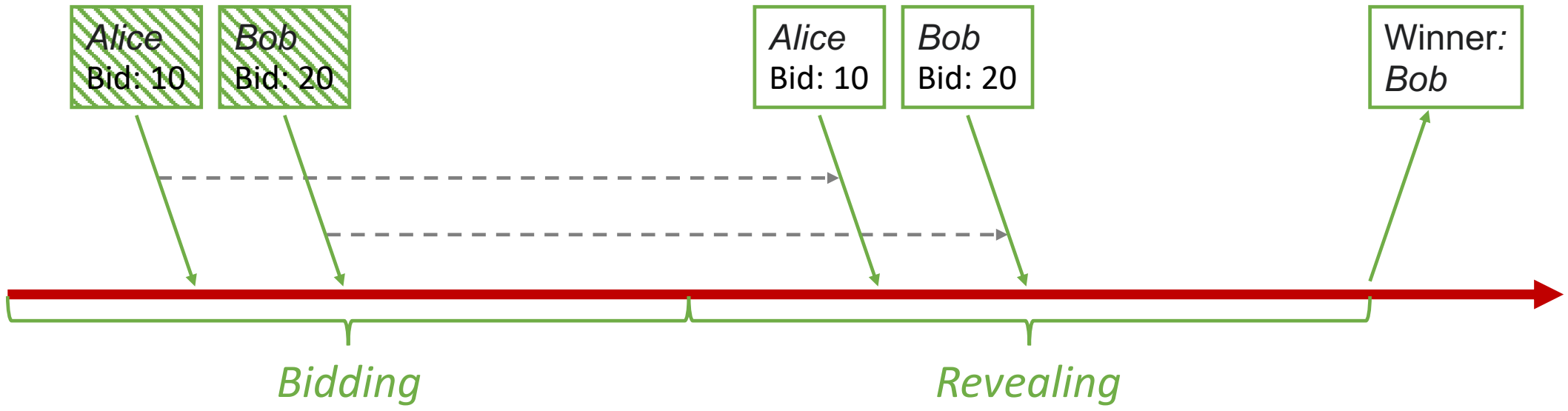
```
1  Init Upon creating the auction smart contract:
2  |   highest ← 0, winner ← ∅
3
4  Bid Upon receiving i's bid b_i in the bidding period:
5  |   if b_i > highest then
6  |   |   Assert(i transfers b_i)
7  |   |   Distribute highest to winner when winner ≠ ∅;
8  |   |   highest ← b_i
9  |   |   winner ← i
10 |   end
```

ZeroAuction

Init *Upon creating the auction smart contract*:
  $highest \leftarrow 0$, $winner \leftarrow \varnothing$

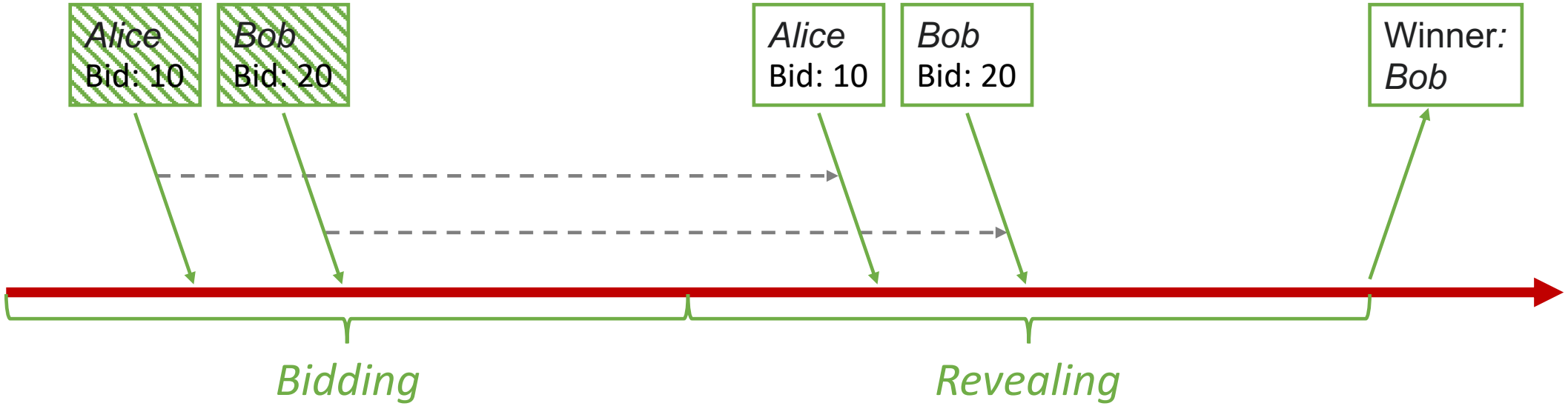
Bid *Upon receiving i's bid b_i in the bidding period*:
  if $b_i > highest$ **then**
    Assert(i transfers b_i)
    Distribute $highest$ to $winner$ when $winner \neq \varnothing$;
    $highest \leftarrow b_i$
    $winner \leftarrow i$
  end

~~Impossibility 1~~: One Round Communication



~~Impossibility 2~~: Eliminating Deposit

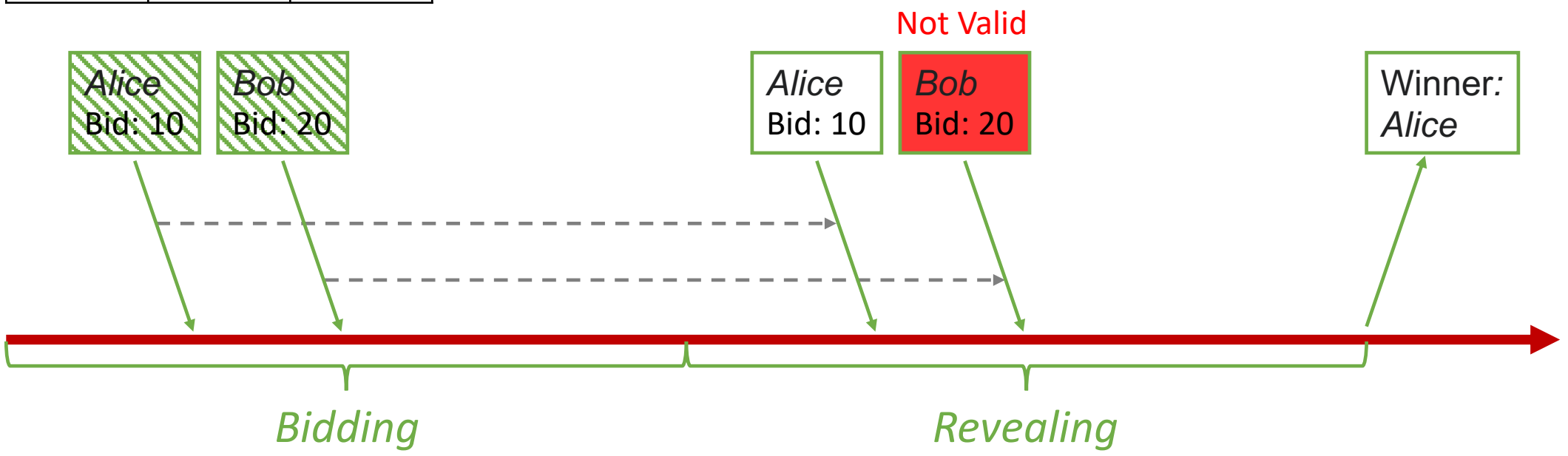
	Balance	Auction
Alice	10	10
Bob	20	20



Global Delay Time

~~Impossibility 2~~: Eliminating Deposit

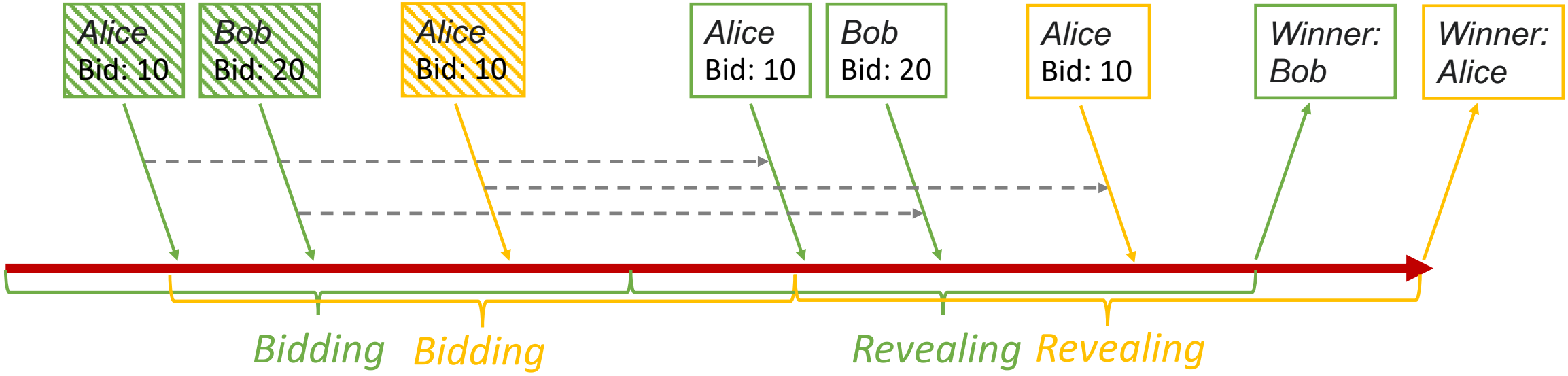
	Balance	Auction
Alice	10	10
Bob	**0**	20



Global Delay Time

~~Impossibility 3~~: Multiple Auctions

	Balance	Auction	Auction
Alice	10	10	10
Bob	20	20	



Global Delay Time

Experimental Results

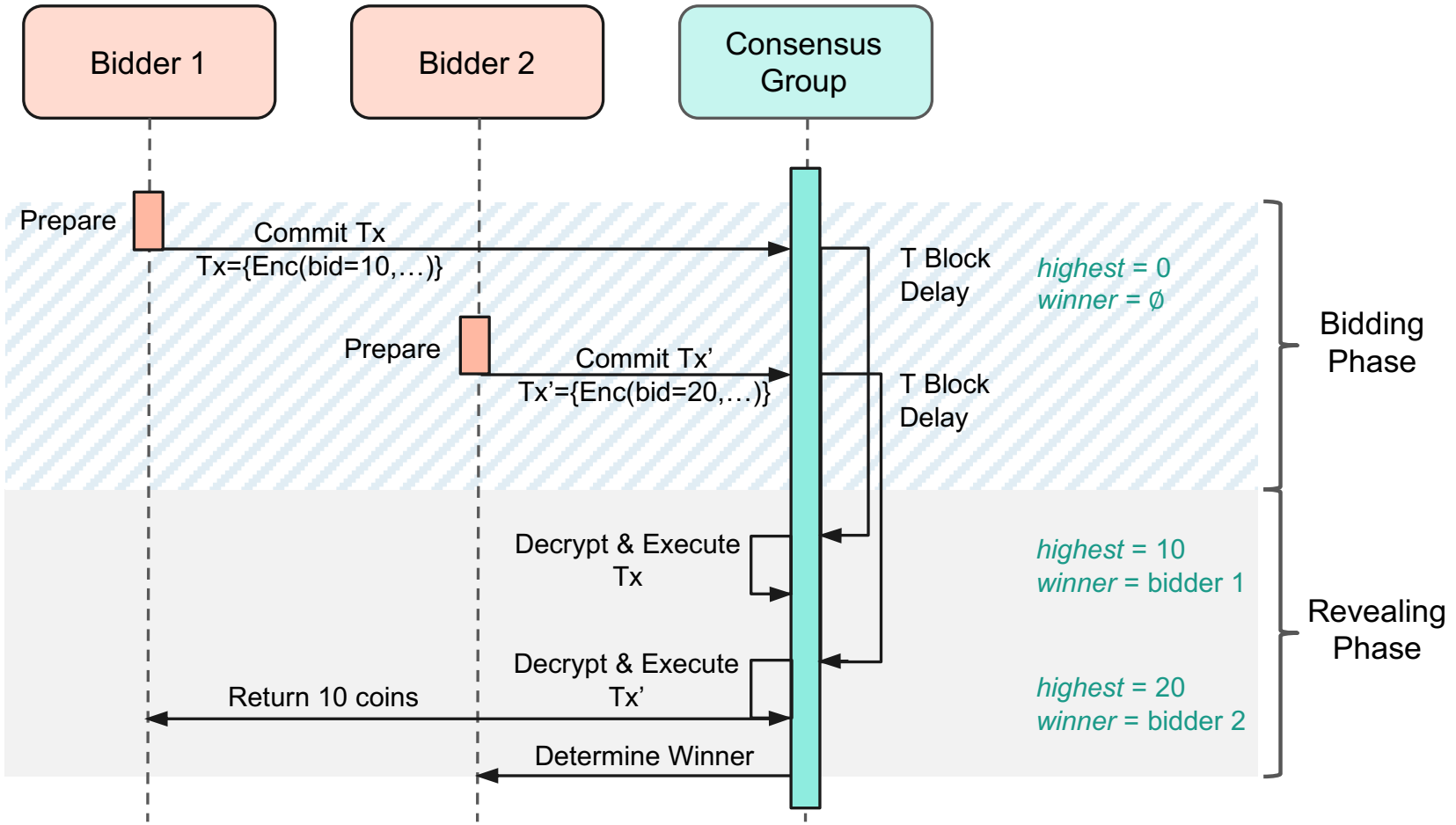
Under Delayed Execution

Conclusion

- ZeroAuction achieves
  - One round of communication
  - Zero deposit requirement
  - Same fund for multiple auctions
- ZeroAuction requires
  - Delayed execution for all transactions

Workshop Paper

Protocol


Bidder 1
Bidder 2
Consensus Group
Prepare
Commit Tx
Tx={Enc(bid=10,…)}
T Block Delay
highest = 0
winner = ∅
Prepare
Commit Tx'
Tx'={Enc(bid=20,…)}
T Block Delay
Bidding Phase
Decrypt & Execute Tx
highest = 10
winner = bidder 1
Decrypt & Execute Tx'
Return 10 coins
highest = 20
winner = bidder 2
Revealing Phase
Determine Winner