L9

Hard to agree

abhi shelat

The simple model

n parties, party *i* holds private value v_i. m are faulty.

The simple model

n parties, party *i* holds private value v_i. m are faulty.

Parties communicate via 2-party msgs.
Network fail-safe, low delay, authenticated.

The simple model

n parties, party *i* holds private value v_i. m are faulty.

Parties communicate via 2-party msgs.
Network fail-safe, low delay, authenticated.

Goal: compute a value for each party such that:

The simple model

n parties, party *i* holds private value v_i. m are faulty.

Parties communicate via 2-party msgs.
Network fail-safe, low delay, authenticated.

Goal: compute a value for each party such that:

All non-faulty parties compute same vector **V**

The simple model

n parties, party *i* holds private value v_i. m are faulty.

Parties communicate via 2-party msgs.
Network fail-safe, low delay, authenticated.

Goal: compute a value for each party such that:

All non-faulty parties compute same vector **V**

If party *i* is not faulty, value $\mathbf{V_i}$ = private value v_i

Reaching Consensus

[PLS protocol for n=4]

1. Pick your value v_i
2. Send v_i to every party.

Reaching Consensus

[PLS protocol for n=4]

1. Pick your value v_i
2. Send v_i to every party.

3. After receiving $a_i=(v_1,\ldots,v_4)$ from each other party, send your list of received values a_i every party.

Reaching Consensus

[PLS protocol for n=4]

1. Pick your value v_i
2. Send v_i to every party.

3. After receiving $a_i=(v_1,\ldots,v_4)$ from each other party, send your list of received values a_i every party.

4. After receiving $(a_1,\ldots,a_4)$ take a majority.

Why does PLS n=4 work?

What happens if everyone is honest?

P1 output: (_, _, _, _)

P2 output: (_, _, _, _)

P3 output: (_, _, _, _)

P4 output: (_, _, _, _)

Why does PLS n=4 work?

What happens if one party, say P1, is faulty?

P2 output: (__, __, __, __)

P3 output: (__, __, __, __)

P4 output: (__, __, __, __)

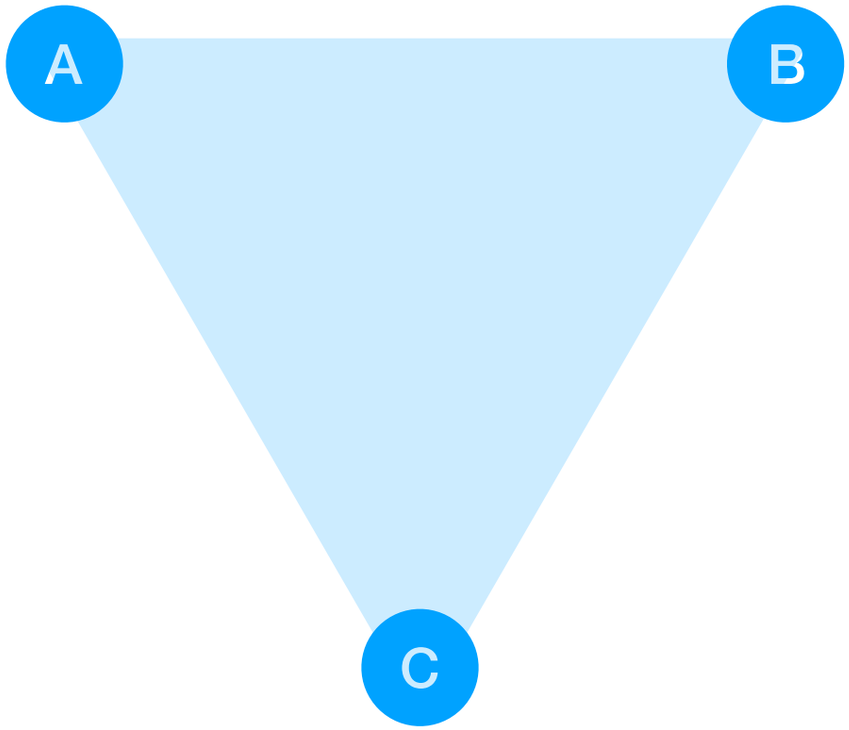
PLS can be generalized

Works whenever n>3m

Works whenever n>3m

Only works with n>3m

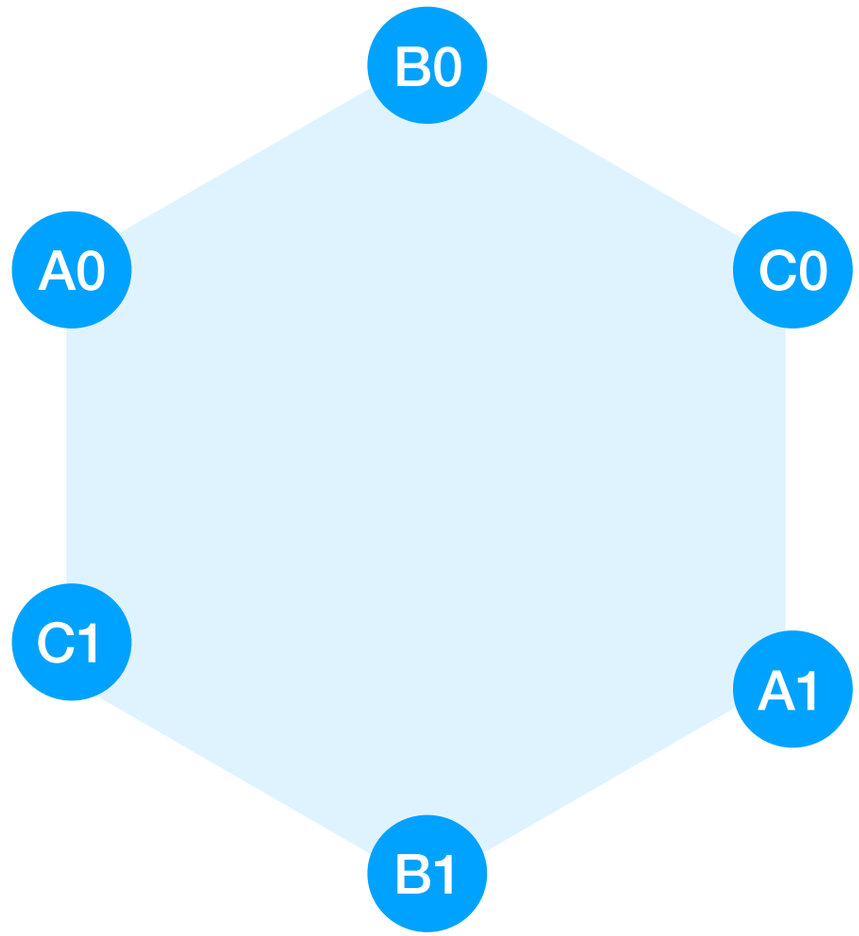
[Fischer-Lynch-Merrit proof]



Consider n=3, m=1 parties.

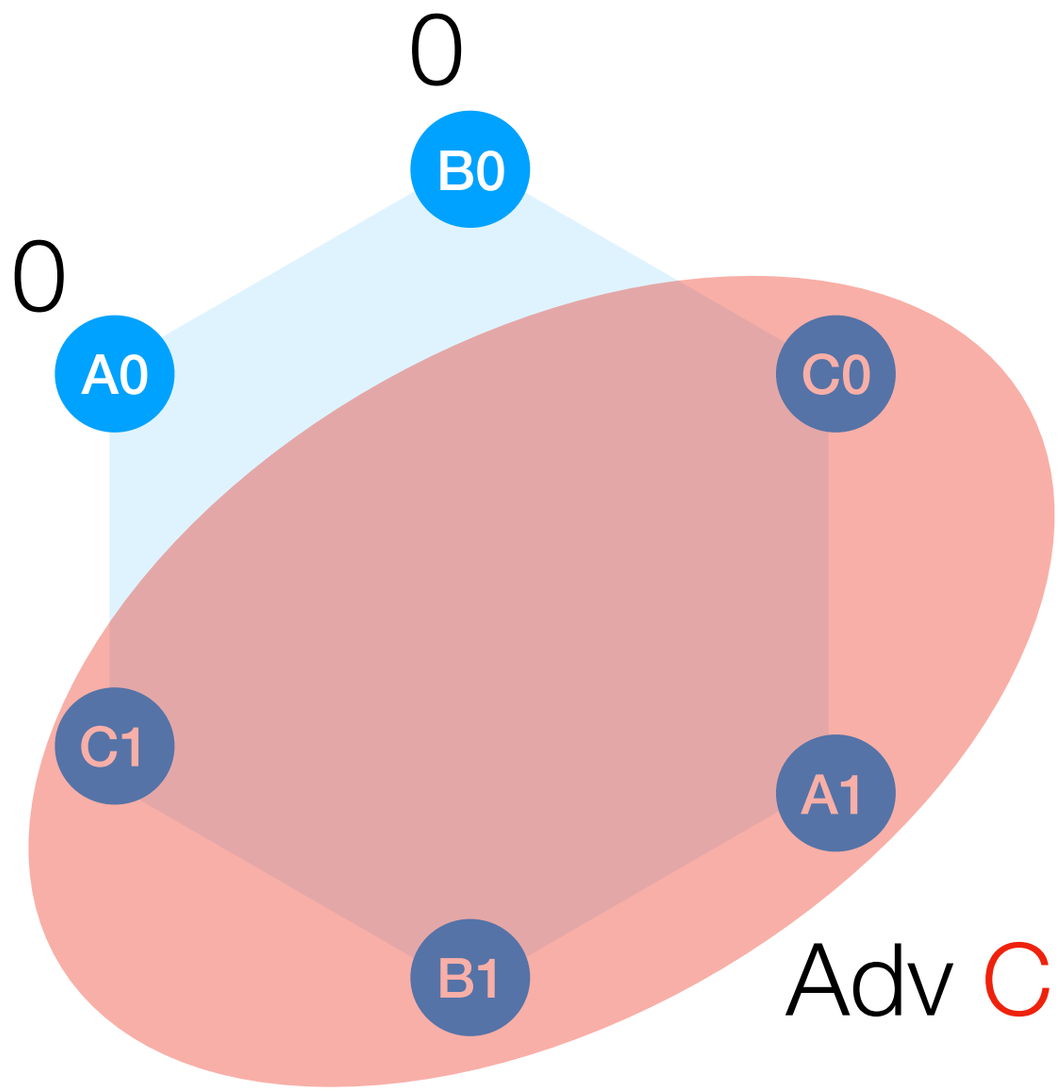
Only works with n>3m

[Fischer-Lynch-Merrit proof]



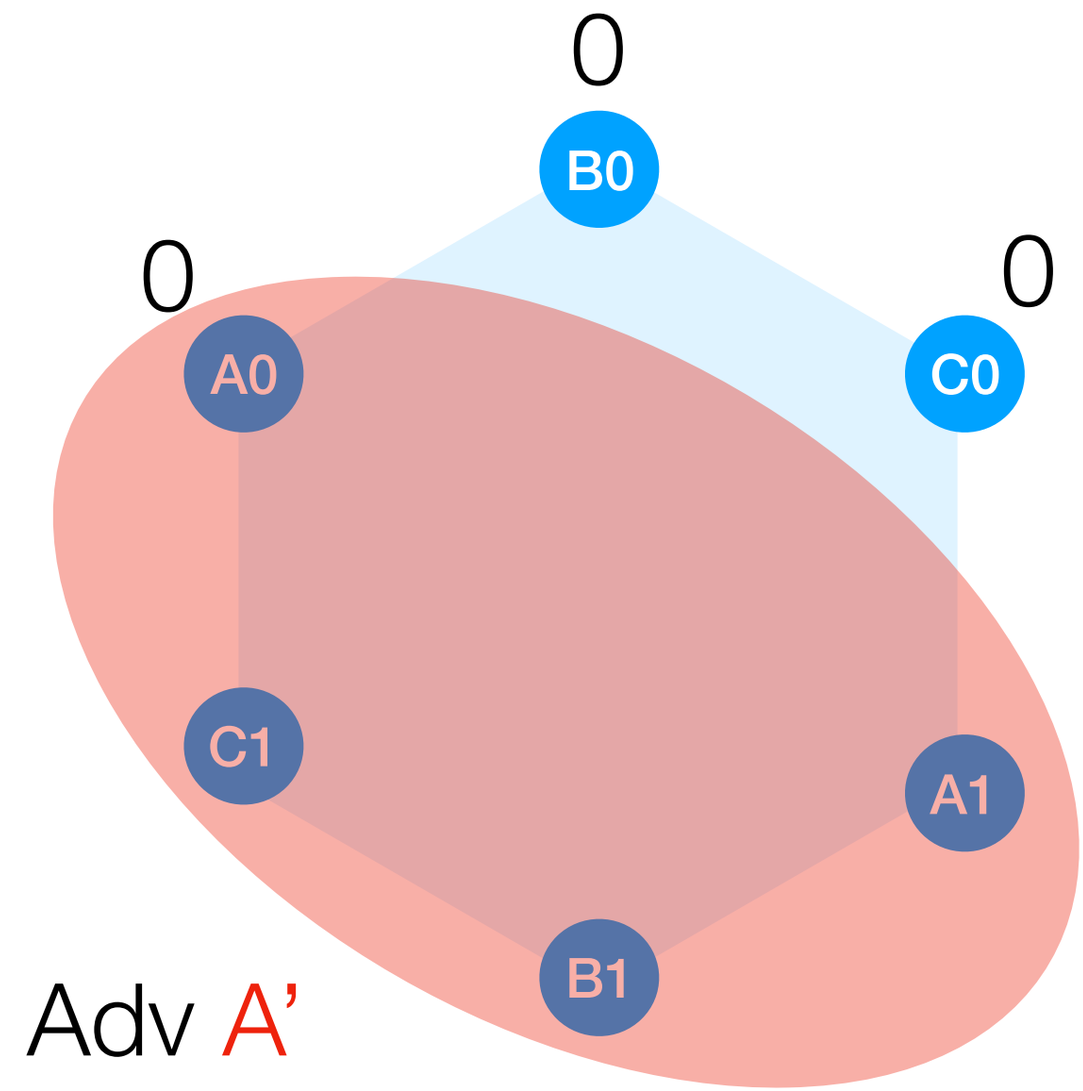
Duplicate parties, Pi running protocol with input i.

Only works with n>3m



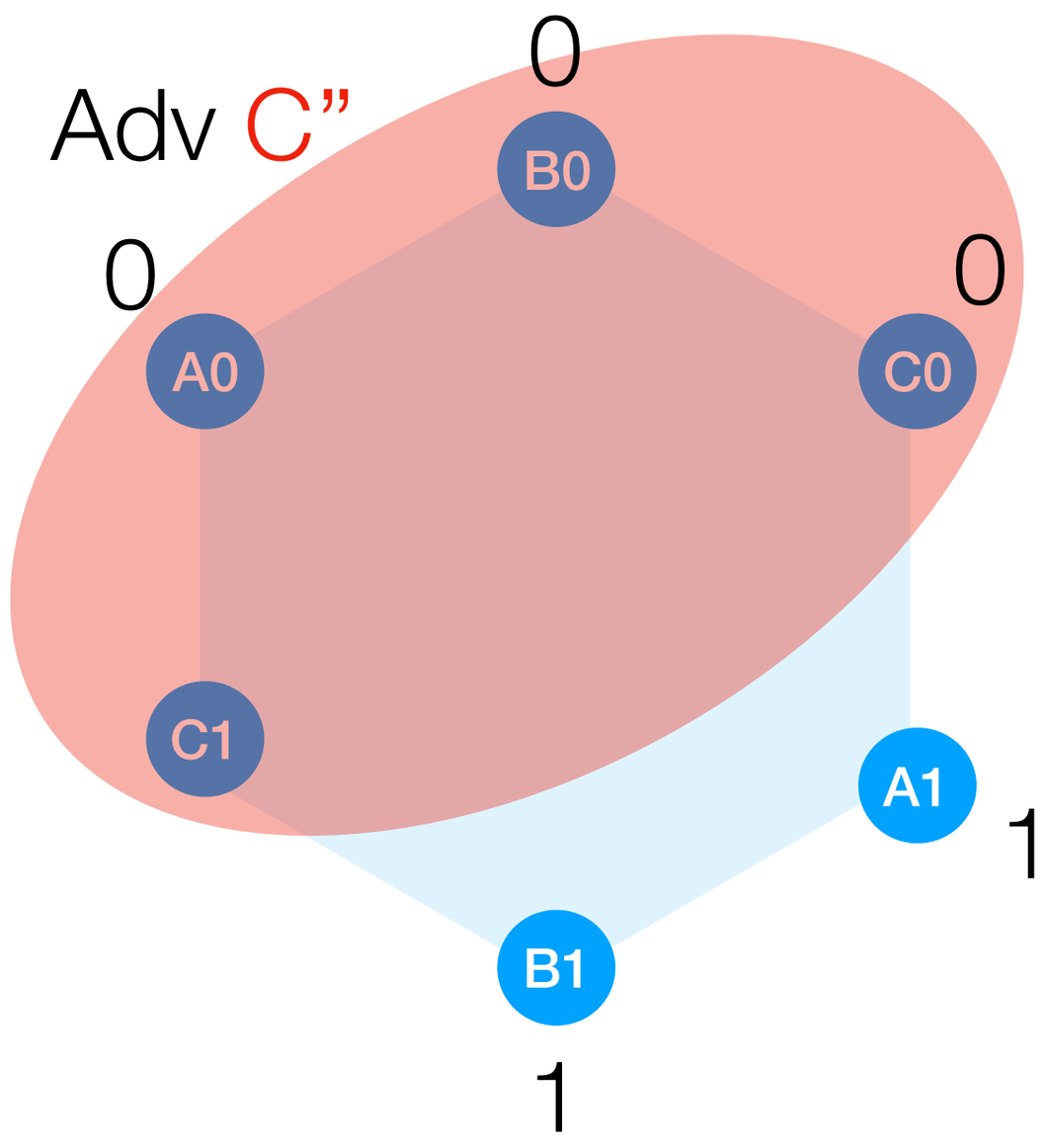
By consensus, honest parties output 0.

Only works with n>3m



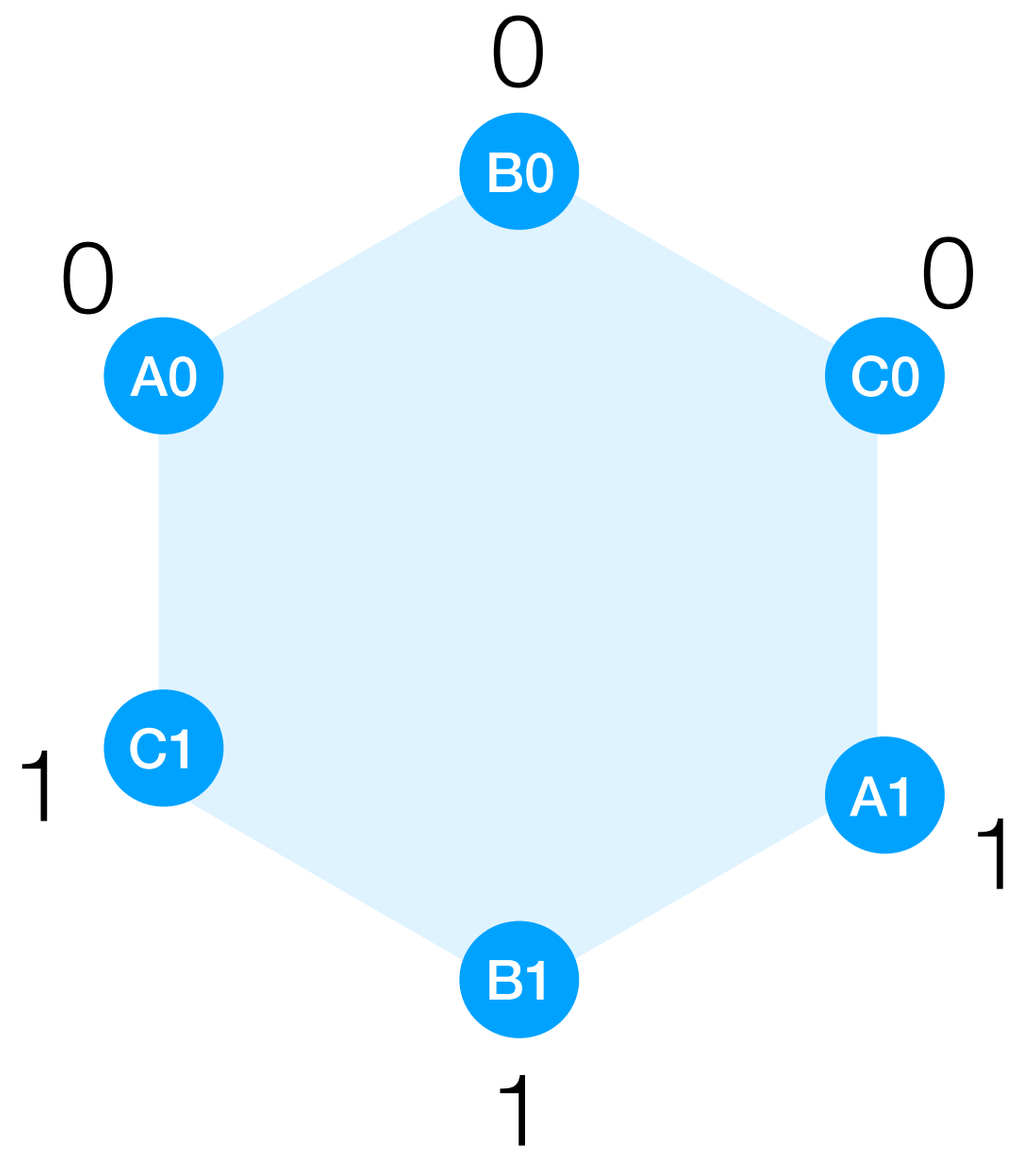
By consensus, honest parties output 0.

Only works with n>3m



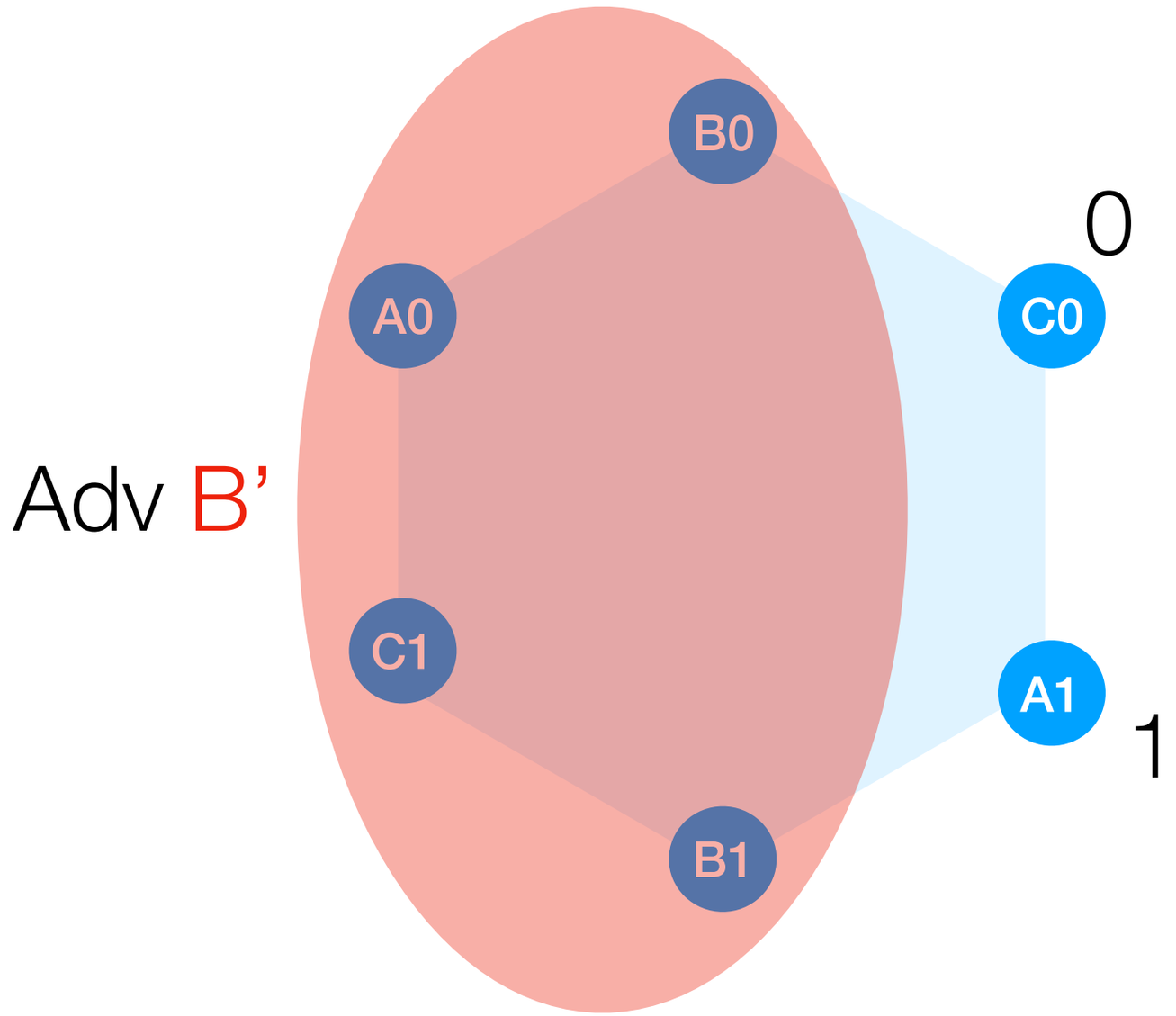
By consensus, honest parties output 0.

Only works with n>3m



These must be the outputs.

Only works with n>3m



Thus, there exists Adv B’ which violates consistency.