

# Advanced Tools from Modern Cryptography

## Lecture 14

MPC: Fairness and Guaranteed Output Delivery  
Some Impossibility Results

# General MPC

## Information-theoretic security

- Passive with corruption threshold  $t < n/2$

Passive BGW/CCD

- Passive with OT setup

Passive GMW

- Guaranteed Output UC with  $t < n/3$

BGW

- Guaranteed Output UC with  $t < n/2$  and Broadcast

Rabin-BenOr

- Selective Abort UC, with OT

Kilian (Also IPS)

## Computational security

- Passive

Composing Yao or Passive GMW with a passive-secure OT protocol

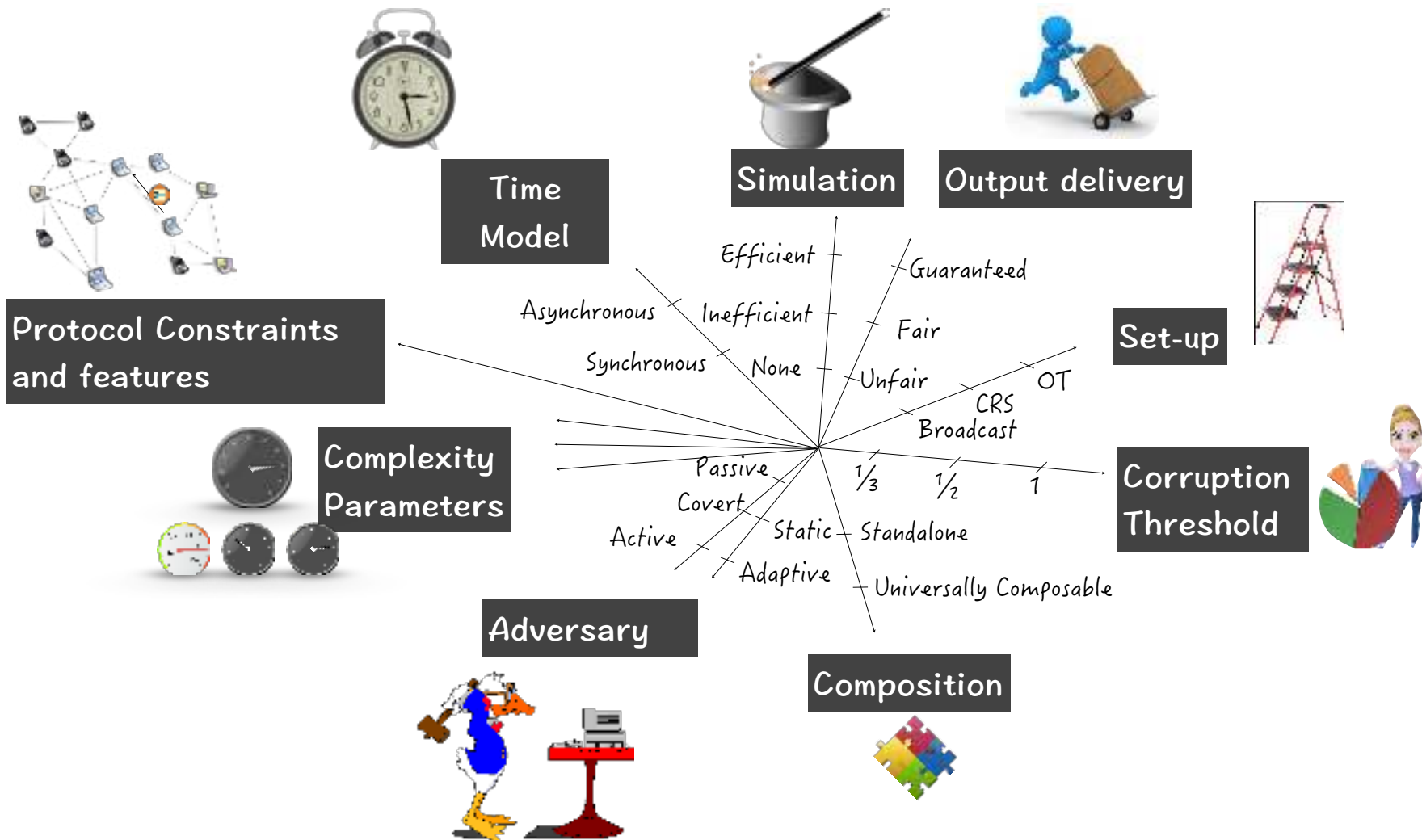
- Standalone

GMW: using ZK proofs

- Selective Abort UC, with CRS

Composing Kilian with a CRS-based UC-secure OT protocol

# MPC Dimensions



# Basic Dimensions

- Adversary's computational power: PPT adversary, Information-theoretic security
- Honest majority: Thresholds 1 (no honest majority),  $\frac{1}{2}$  and  $\frac{1}{3}$
- Security Level: Passive security, UC security with selective abort, or UC security with guaranteed output delivery
- Trusted setup: Point-to-point channels, Broadcast, Common Reference String (CRS), OT



# Output Delivery

- 3 levels:
  - Unfair (a.k.a., selective abort)
    - Adversary can see its output and decide which set of honest parties receive theirs
  - Fair
    - Adversary can cause abort for all parties before seeing its output. To get its output, should let all parties get theirs.
  - Guaranteed output delivery
    - Adversary cannot prevent honest parties from producing an output. (Adversary will have well-defined inputs no matter what it does.)
- Today: Impossibility of fairness/guaranteed output delivery even for some very simple tasks, if there is no honest majority

# Fair Coin-Tossing

- For 2-party functions, fair protocol  $\Rightarrow$  guaranteed output delivery
  - Modify a fair protocol: if aborted, locally compute the output (with a fixed input for the other party)
- 2-party fair coin-tossing from commitment?
  - Alice commits to a random bit  $a$ , Bob sends a bit  $b$ , Alice opens and they output  $a \oplus b$
  - Unfair: Alice can abort after learning the outcome
- Two parties can never obtain a fair coin, given only unfair setups, even under computational assumptions, even for standalone security, even against fail-stop adversaries
  - Unfair setup: Sends outputs to the parties one at a time.  
Adversary can abort at any point.

# Fair Coin-Tossing

- Guaranteed output delivery: Each party has a tentative output after each message it receives, if an abort happens right after it
- Best possible unfair setup:  $F$  runs the protocol on behalf of the parties; at each round, sends each party its tentative output.
  - $X_0, Y_0$  if abort before start. Then  $F$  Sends  $X_1$  (to Alice),  $Y_1$  (to Bob),  $X_2, Y_2, \dots, X_n, Y_n$ . Allows adversary to abort at any point.
- $X_0, Y_0$  independent; also uniform (by correctness for abort at start)
- Correctness when no abort:  $\Pr[X_n=b, Y_n=b]=1/2$ , for  $b \in \{0,1\}$
- $\Pr[X_i=Y_i]$  went from  $1/2$  to  $1$ : So  $\exists i$  s.t.  $\Pr[X_i=Y_i] - \Pr[X_{i-1}=Y_{i-1}] \geq 1/(2n)$ .  
i.e.,  $\Pr[X_i=Y_i] - \Pr[X_i=Y_{i-1}] + \Pr[X_i=Y_{i-1}] - \Pr[X_{i-1}=Y_{i-1}] \geq 1/(2n)$ 
  - So, some  $i$  s.t. either  $\Pr[X_i=Y_i] - \Pr[X_i=Y_{i-1}] \geq 1/(4n)$  or  $\Pr[X_i=Y_{i-1}] - \Pr[X_{i-1}=Y_{i-1}] \geq 1/(4n)$
- But will show,  $\forall i \Pr[X_i=Y_i] \approx \Pr[X_i=Y_{i-1}]$  and  $\Pr[X_i=Y_{i-1}] \approx \Pr[X_{i-1}=Y_{i-1}]$  !



# Fair Coin-Tossing

- To show  $\Pr[X_i=Y_i] \approx \Pr[X_i=Y_{i-1}]$  (and similarly  $\Pr[X_i=Y_{i-1}] \approx \Pr[X_{i-1}=Y_{i-1}]$ )
  - Firstly,  $\Pr[Y_{i-1}=0] \approx 1/2$ ,  $\Pr[Y_i=0] \approx 1/2$  (by correctness against Alice who **aborts after  $Y_{i-1}$**  and one who **aborts after  $Y_i$** )
  - Consider two more attackers for corrupt Alice:  
 **$A_0$ : If  $X_i=0$ , abort immediately, else abort after  $Y_i$  delivered**  
 **$A_1$ : If  $X_i=1$ , abort immediately, else abort after  $Y_i$  delivered**
  - Under attack by  $A_0$ ,  
$$\begin{aligned}\Pr[\text{Bob outputs } 0] &= \Pr[X_i=0, Y_{i-1}=0] + \Pr[X_i=1, Y_i=0] \\ &= \Pr[X_i=0, Y_{i-1}=0] - \Pr[X_i=0, Y_i=0] + \Pr[Y_i=0] \\ \Rightarrow \Pr[X_i=0, Y_{i-1}=0] &\approx \Pr[X_i=0, Y_i=0]\end{aligned}$$
  - Similarly, from  $A_1$ ,  $\Pr[X_i=1, Y_{i-1}=1] \approx \Pr[X_i=1, Y_i=1]$
  - So,  $\Pr[X_i=Y_{i-1}] \approx \Pr[X_i=Y_i]$



Recall

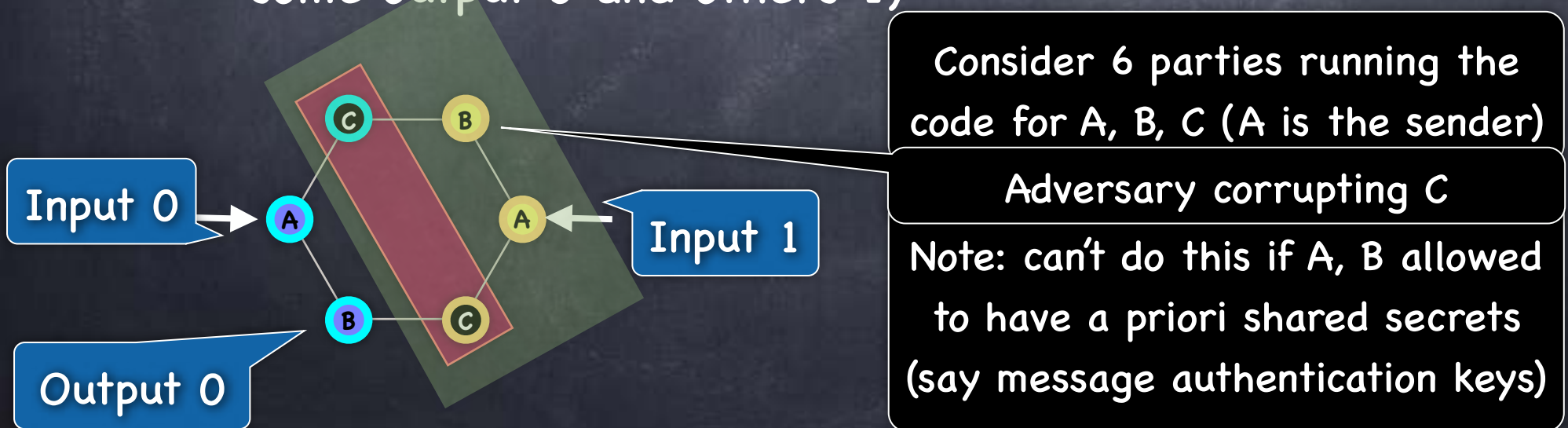
# Broadcast

- BGW protocol relied on broadcast to ensure all honest parties have the same view of disputes, resolution etc.
- Concern addressed by broadcast: a corrupt sender can send different values to different honest parties
- Broadcast with selective abort can be implemented easily, even without honest majority
  - Sender sends message to everyone. Every party cross-checks with everyone else, and aborts if there is any inconsistency.
- If corruption threshold  $t < n/3$ , then it turns out that broadcast with guaranteed output delivery can be implemented
- If broadcast given as a setup, can do MPC with guaranteed output delivery for up to  $t < n/2$

Otherwise not!

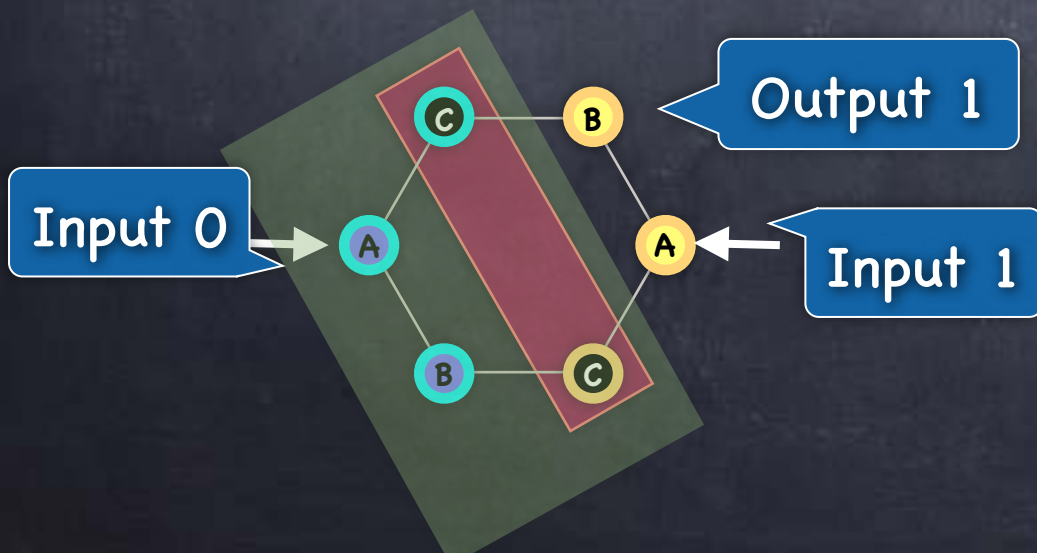
# No Broadcast with Guaranteed Output if $1/3$ Corrupt

- Broadcast requirements (message being a single bit):
  - If sender honest, all honest parties should output the bit it sends (can't abort)
  - All honest parties should agree on the outcome (can't have some output 0 and others 1)



# No Broadcast with Guaranteed Output if $1/3$ Corrupt

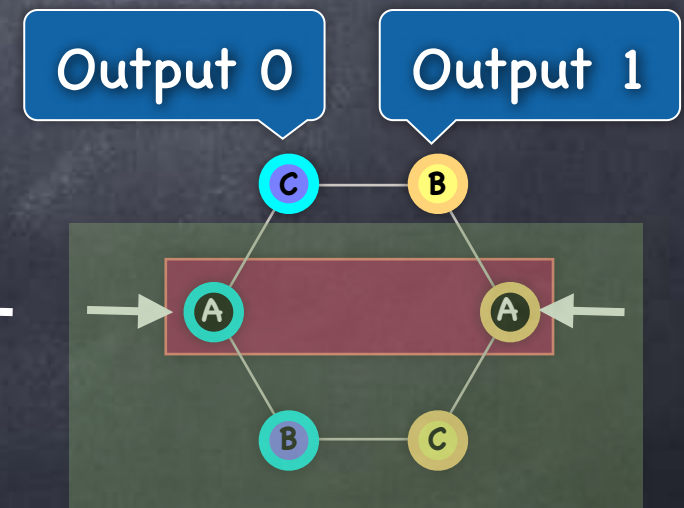
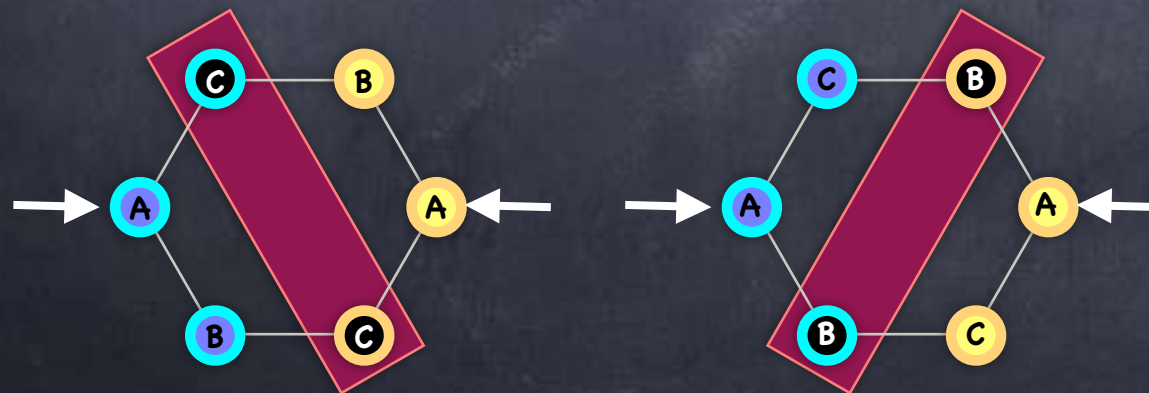
- Broadcast requirements (message being a single bit):
  - If sender honest, all honest parties should output the bit it sends (can't abort)
  - All honest parties should agree on the outcome (can't have some output 0 and others 1)





# No Broadcast with Guaranteed Output if 1/3 Corrupt

- Broadcast requirements (message being a single bit):
  - If sender honest, all honest parties should output the bit it sends (can't abort)
  - All honest parties should agree on the outcome (can't have some output 0 and others 1)



# No Broadcast with Guaranteed Output if $1/3$ Corrupt

- Broadcast requirements (message being a single bit):
  - If sender honest, all honest parties should output the bit it sends (can't abort)
  - All honest parties should agree on the outcome (can't have some output 0 and others 1)
- Impossible to satisfy both constraints simultaneously, if  $1/3$  can be corrupt
  - Irrespective of what computational assumptions are used!
  - But a priori shared keys can give broadcast with guaranteed output delivery against unrestricted corruption (in the synchronous model)