8.1 An MPC Protocol

Parties $P_1, \ldots, P_n$ would like to conduct a majority vote. However, no one wants to reveal his voting behaviour.

a) Suppose the parties plan to use Sum Protocol II modulo $Z_m$ from the slides to solve this problem. Describe the precise specification that is implemented by this protocol.

b) Show that the sum protocol is secure against up to $n - 1$ passively corrupted parties.

c) What happens with your protocol if some party $P_i$ starts with input $x_i = n$. Is the protocol insecure?

d) Is the sum protocol secure against actively corrupted parties?

8.2 Types of Oblivious Transfer

Oblivious transfer (OT) comes in several variants:

- **Rabin OT**: Alice transmits a bit $b$ to Bob, who receives $b$ with probability $1/2$ while Alice does not know which is the case. That is, the output of Bob is either $b$ or $\perp$ (indicating that the bit was not received).

- **1-out-of-2 OT**: Alice holds two bits $b_0$ and $b_1$. For a bit $c \in \{0, 1\}$ of Bob’s choice, he can learn $b_c$ but not $b_{1-c}$, and Alice does not learn $c$.

- **1-out-of-$k$ OT for $k > 2$**: Alice holds $k$ bits $b_1, \ldots, b_k$. For $c \in \{1, \ldots, k\}$ of Bob’s choice, he can learn $b_c$ but none of the others, and Alice does not learn $c$.

Prove the equivalence of these three variants, by providing the following reductions:

a) 1-out-of-$k$ OT $\Rightarrow$ 1-out-of-2 OT

b) 1-out-of-2 OT $\Rightarrow$ 1-out-of-$k$ OT

**Hint**: In your protocol, the sender should choose $k$ random bits and invoke the 1-out-of-2 OT protocol $k$ times.

c) 1-out-of-2 $\Rightarrow$ Rabin OT

d) Rabin OT $\Rightarrow$ 1-out-of-2 OT

**Hint**: Use Rabin OT to send sufficiently many random bits. In your protocol, the receiver might learn both bits, but with negligible probability only.
8.3 Multi-Party Computation with Oblivious Transfer

In the lecture, it was shown that 1-out-of-\( k \) oblivious string transfer (OST) can be used by two parties \( A \) and \( B \) to securely evaluate an arbitrary function \( g : \mathcal{X} \times \mathcal{Y} \to \Omega \), where \( \mathcal{X} \) is \( A \)'s input domain, \( \mathcal{Y} \) is \( B \)'s input domain with \( |\mathcal{Y}| = k \), and \( \Omega \) is the output domain.

a) Let \( \mathcal{Z} \) be a finite (and small) domain. Generalize the above protocol to the case of three parties \( A, B, \) and \( C \), with inputs \( x \in \mathcal{X}, y \in \mathcal{X}, \) and \( z \in \mathcal{Z} \), respectively, who wish to compute a function \( f : \mathcal{X} \times \mathcal{Y} \times \mathcal{Z} \to \Omega \).

**Hint:** Which function table should \( A \) send to \( B \)? Which entry should \( B \) choose, and what should he send to \( C \)?

b) Is your protocol from a) secure against a passive adversary? If not, give an example of a function \( f \) where some party receives too much information by executing the protocol.

c) Modify your protocol to make it secure against a passive adversary.