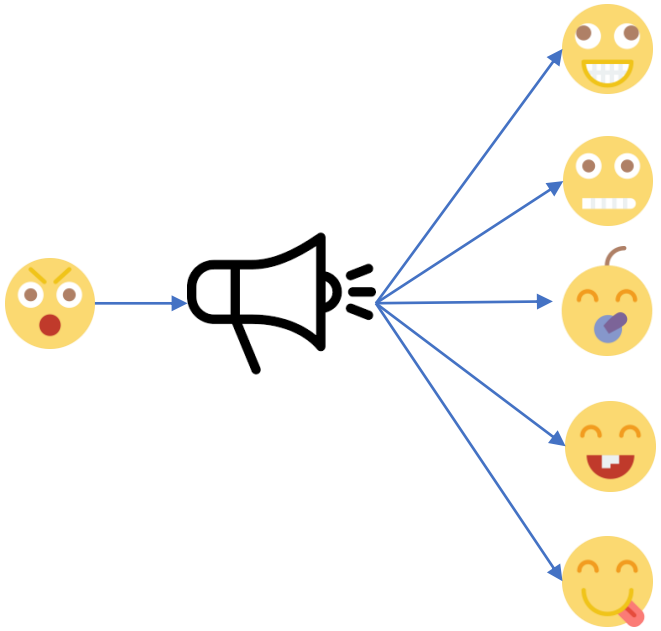
Synchronous Constructive Cryptography

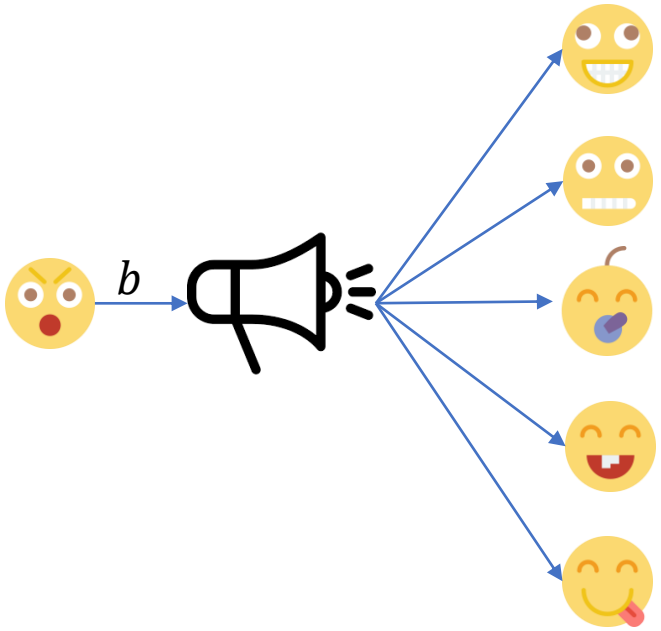
Chen-Da Liu-Zhang

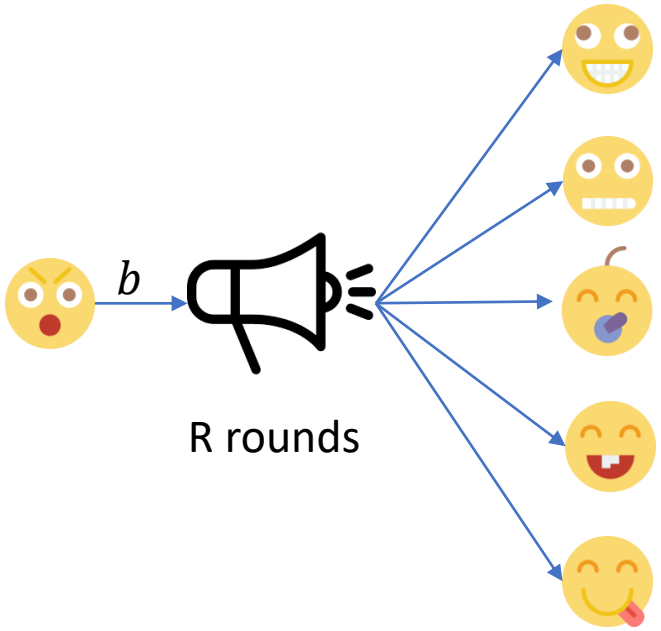
ETH Zurich

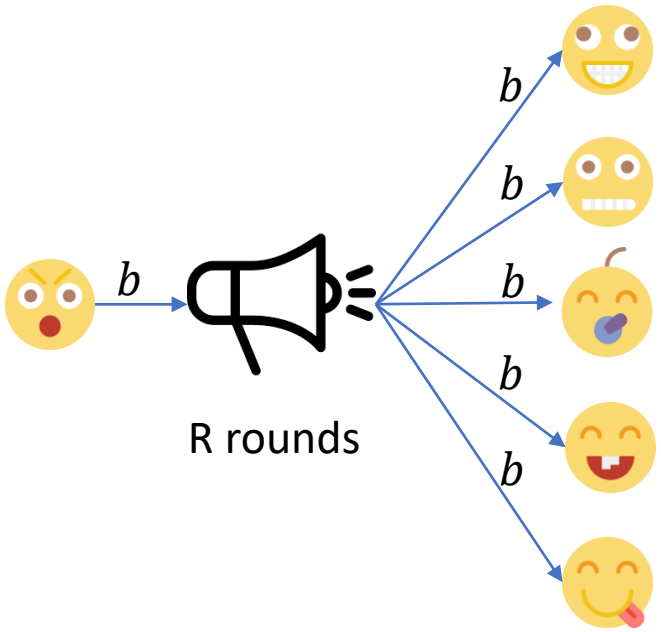
Ueli Maurer

ETH Zurich

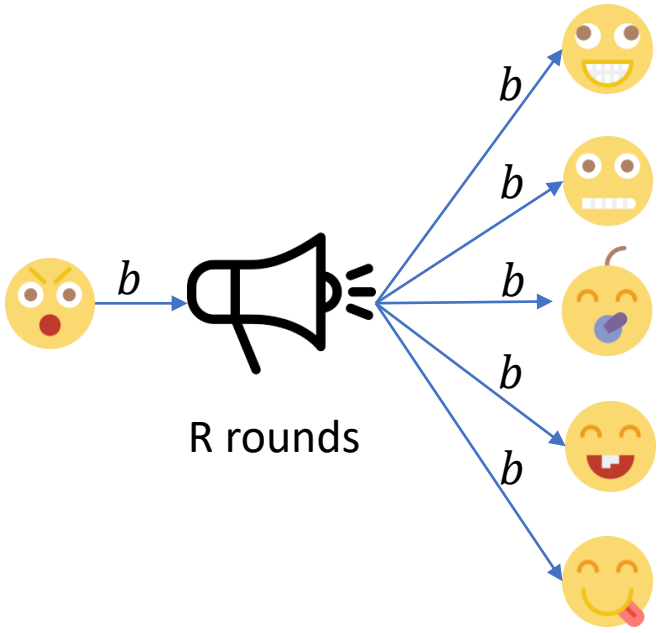


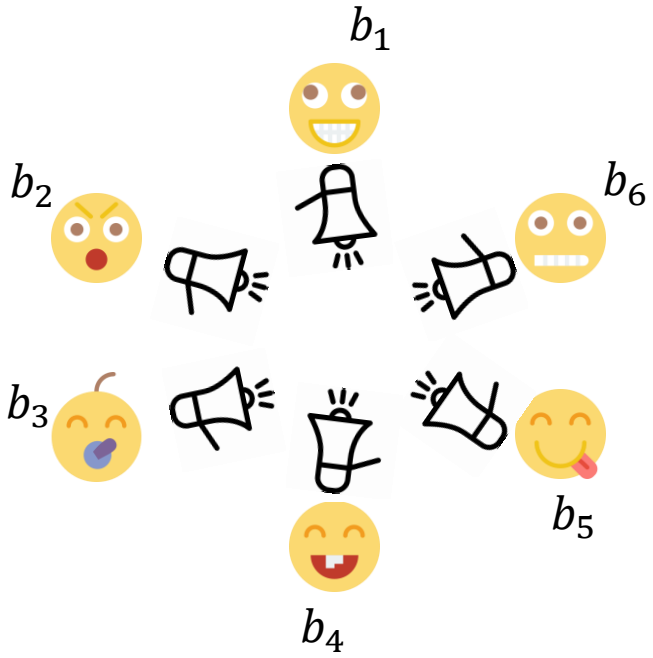

b


b
R rounds

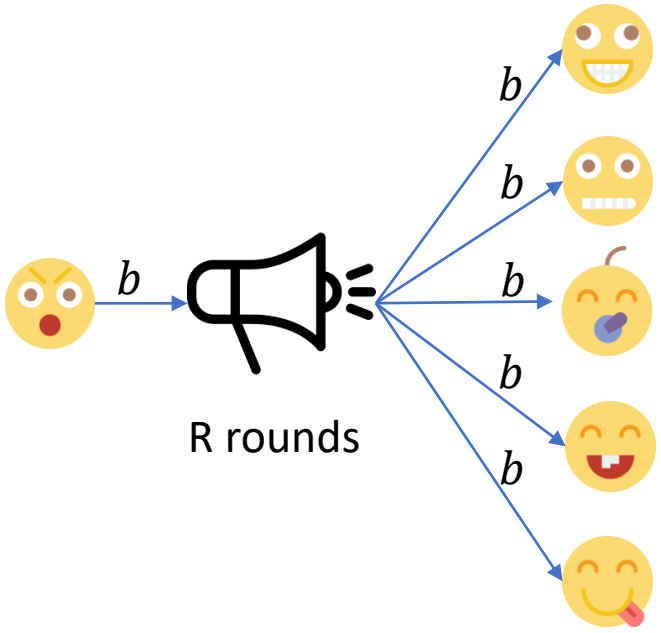

b
b
b
b
b
b
R rounds

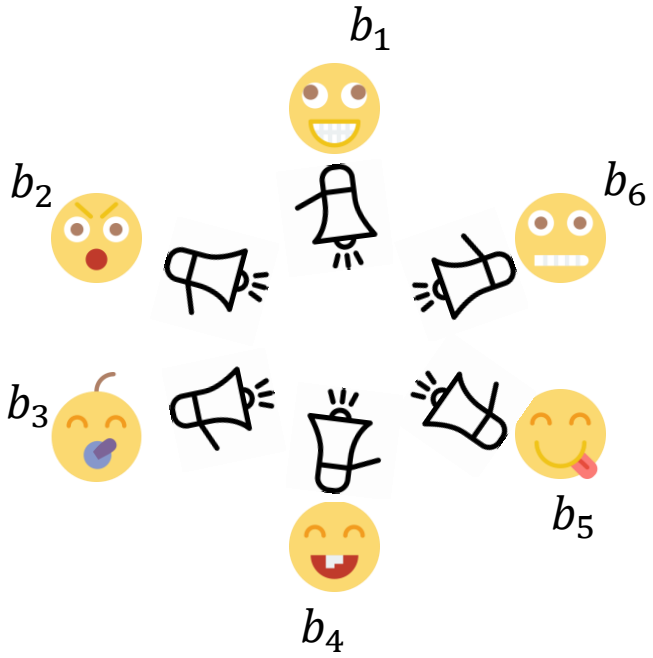
Naïve randomness generation

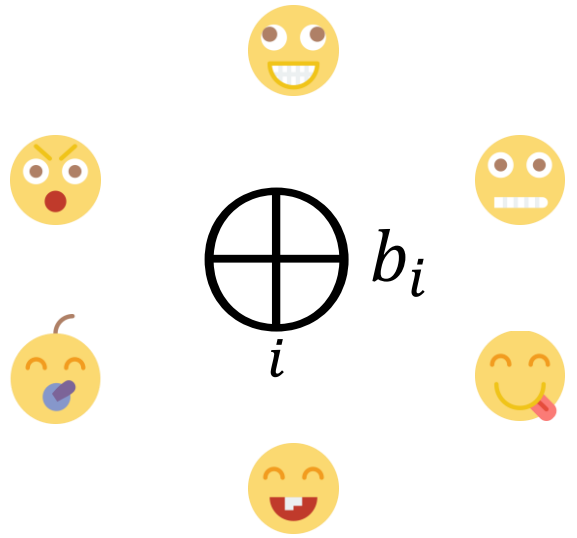




Naïve randomness generation







Composable Security

Modularization

Clean guarantees

Composition

Many existing composable frameworks [PW94,C01,N03,MR11,KT13,…]

Generality vs Simplicity

Generality vs Simplicity

Can we design a simple framework for a meaningful restricted setting?

Generality vs Simplicity

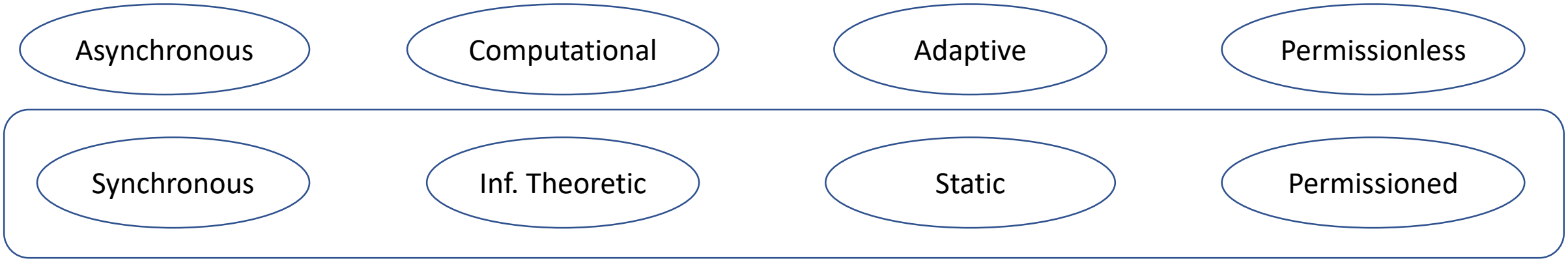
Can we design a simple framework for a meaningful restricted setting?

Precision

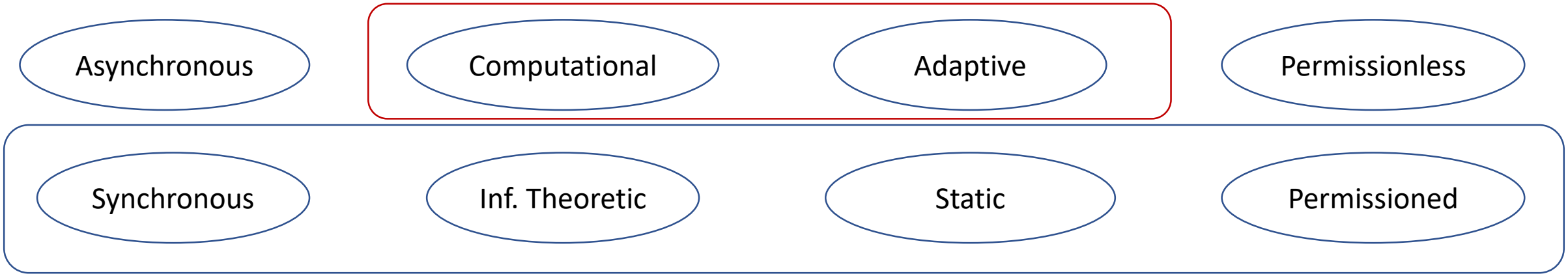
Teaching

Formal Verification

Our Focus


Asynchronous
Computational
Adaptive
Permissionless
Synchronous
Inf. Theoretic
Static
Permissioned

Future Extension



Composable Synchronous Models

Current models* are built on top of asynchronous model

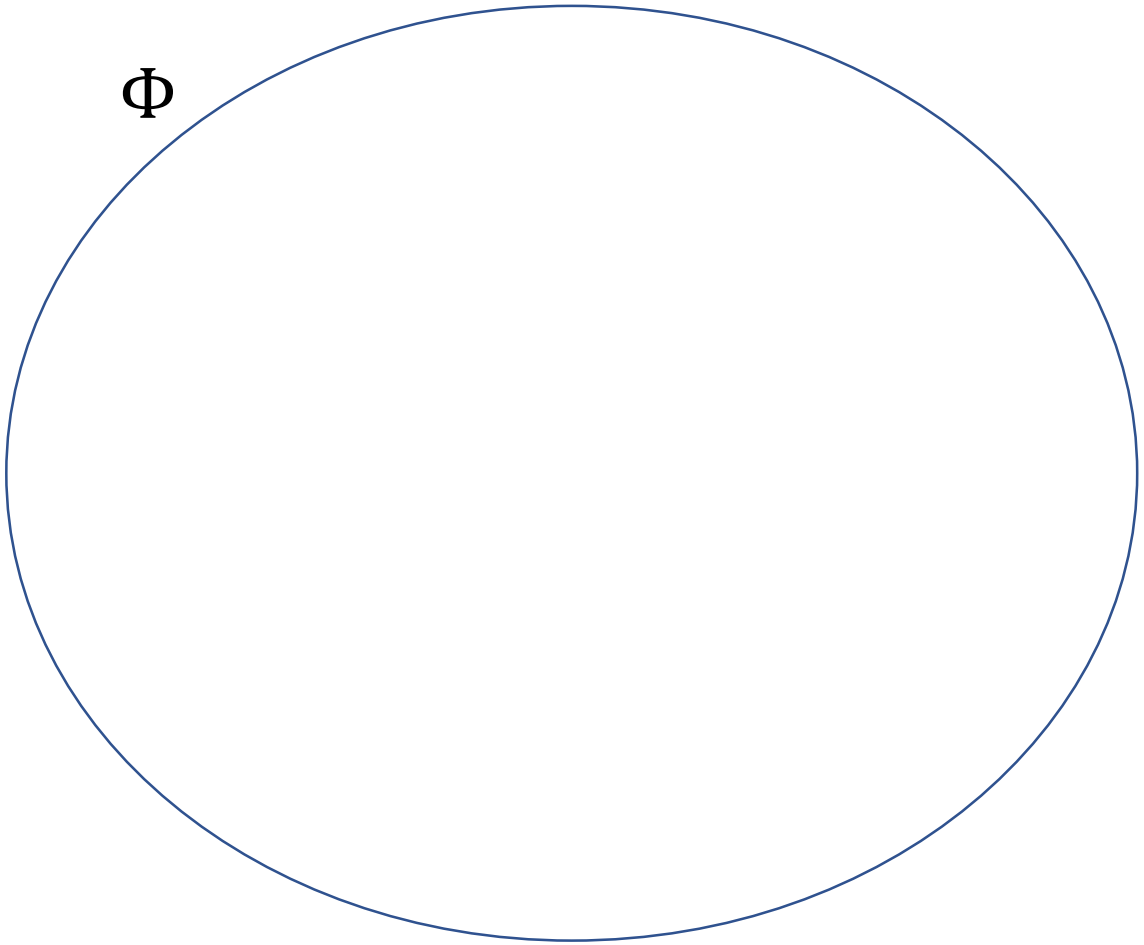
1. Extra functionalities
2. Activation token
3. Message scheduling

Our goal: *minimal* framework

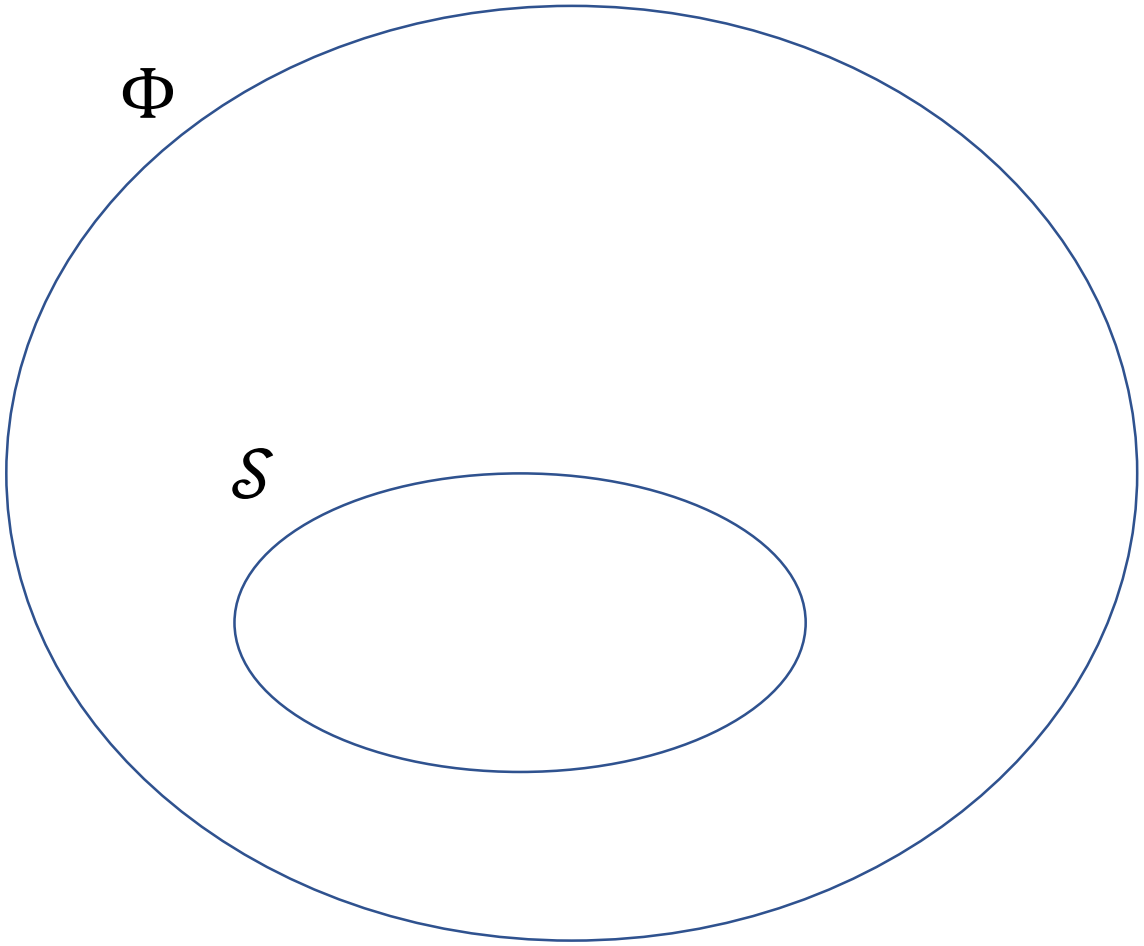
1. Intuitive descriptions
2. Simple proofs

*[Can01,Nie03,HofMul04,KMTZ13]

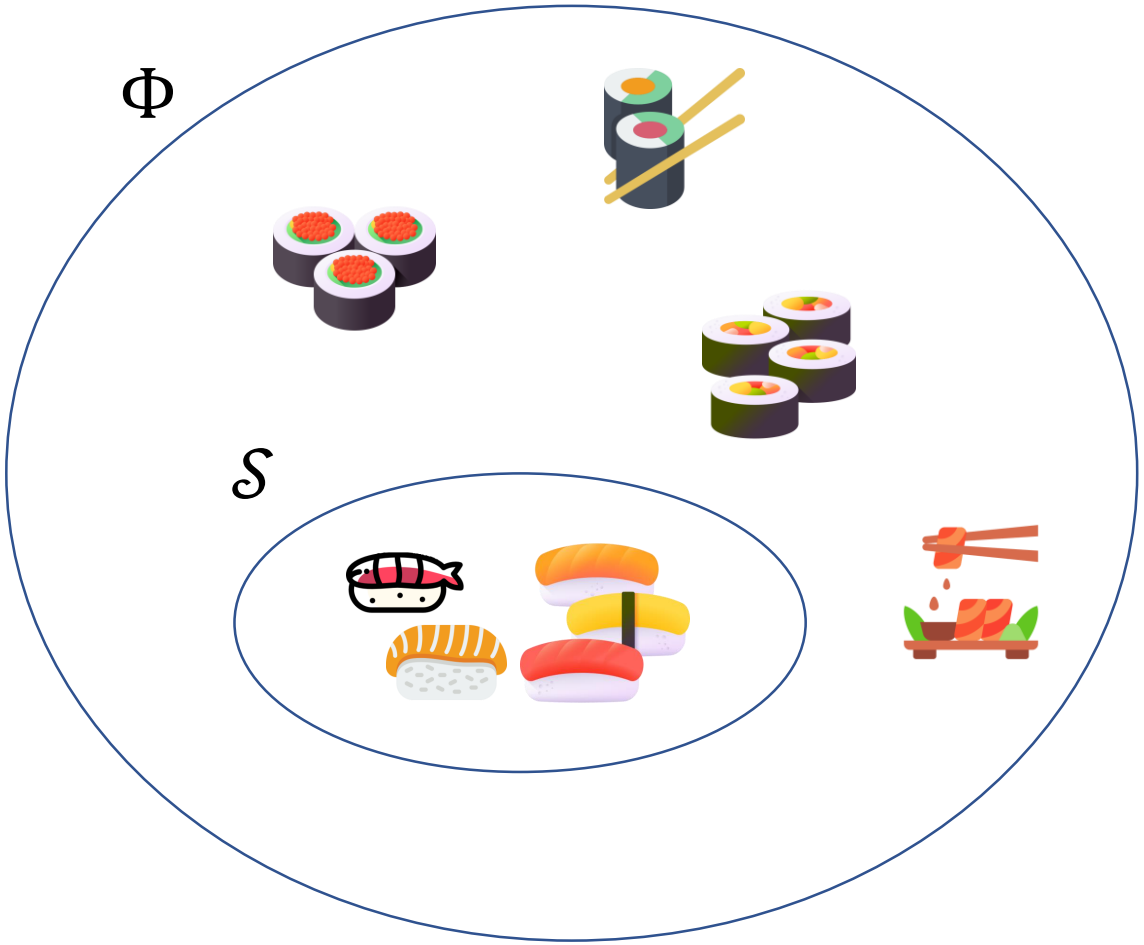
Specifications



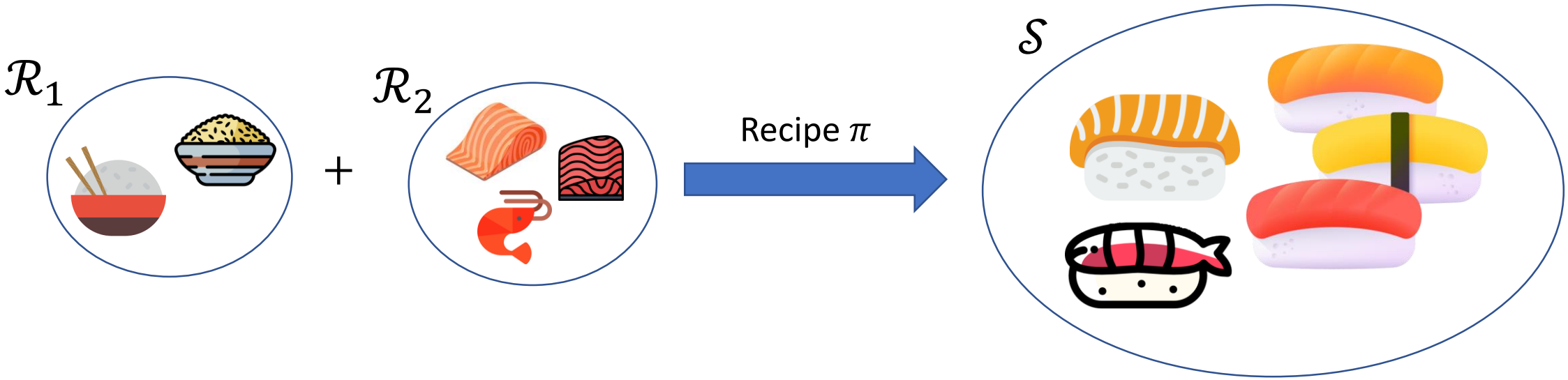
Specifications



Specifications



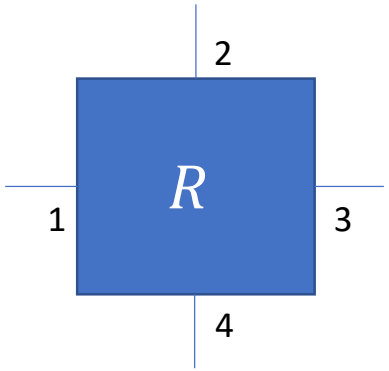
Constructions



$\mathcal{R} \xrightarrow{\pi} \mathcal{S}$, or equivalently, $\pi(\mathcal{R}) \subseteq \mathcal{S}$

Constructive Cryptography

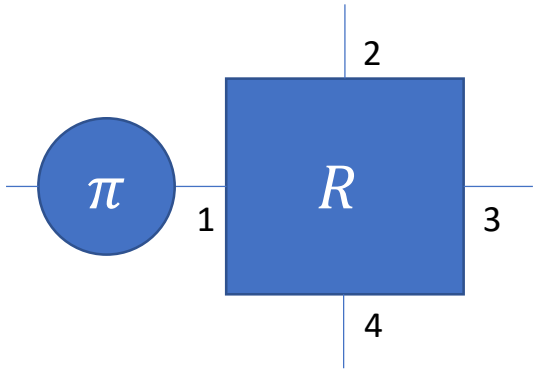
Φ resources



*[Mau11,MR11,MR16]

Constructive Cryptography

Φ resources
Σ converters

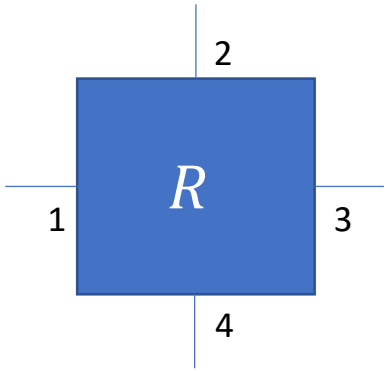


*[Mau11,MR11,MR16]

Multi-Party Constructive Cryptography

$\mathcal{P} = \{1, \ldots, n\}$

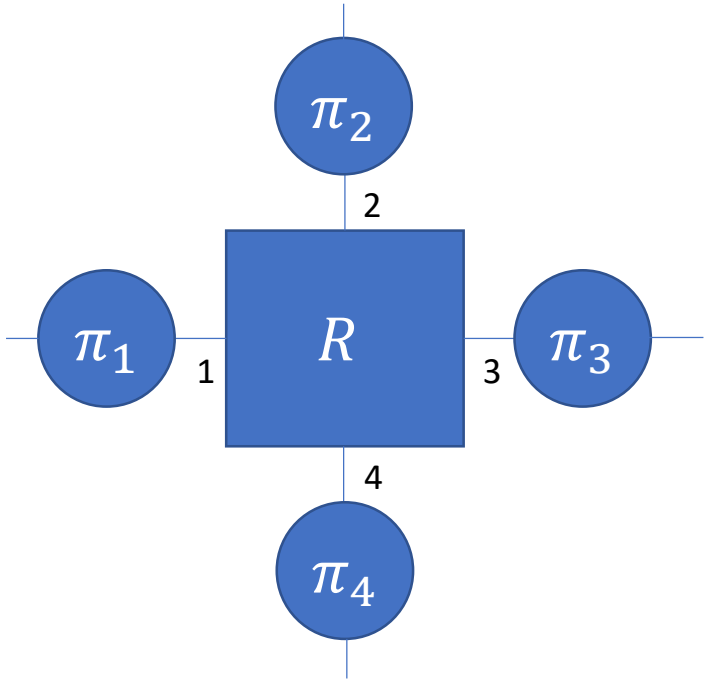
Protocol $\pi = (\pi_1, \ldots, \pi_n)$



Multi-Party Constructive Cryptography

$\mathcal{P} = \{1, \ldots, n\}$

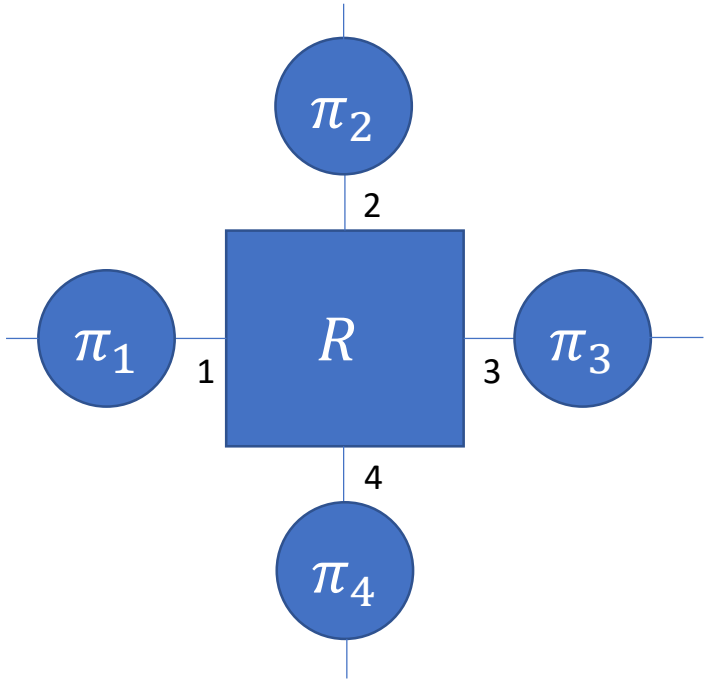
Protocol $\pi = (\pi_1, \ldots, \pi_n)$



Multi-Party Constructive Cryptography

$\mathcal{P} = \{1, \ldots, n\}$

Protocol $\pi = (\pi_1, \ldots, \pi_n)$

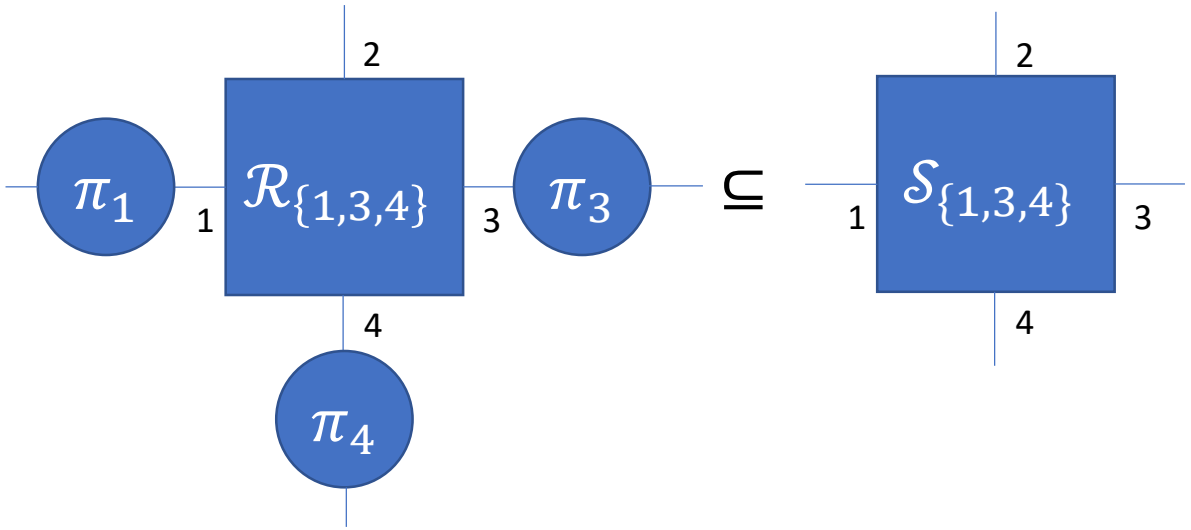


$\forall H \subseteq P \;\; \mathcal{R}_H \xrightarrow{\{\pi_i \mid i \in H\}} \mathcal{S}_H$

Multi-Party Constructive Cryptography

$\mathcal{P} = \{1, \ldots, n\}$

Protocol $\pi = (\pi_1, \ldots, \pi_n)$

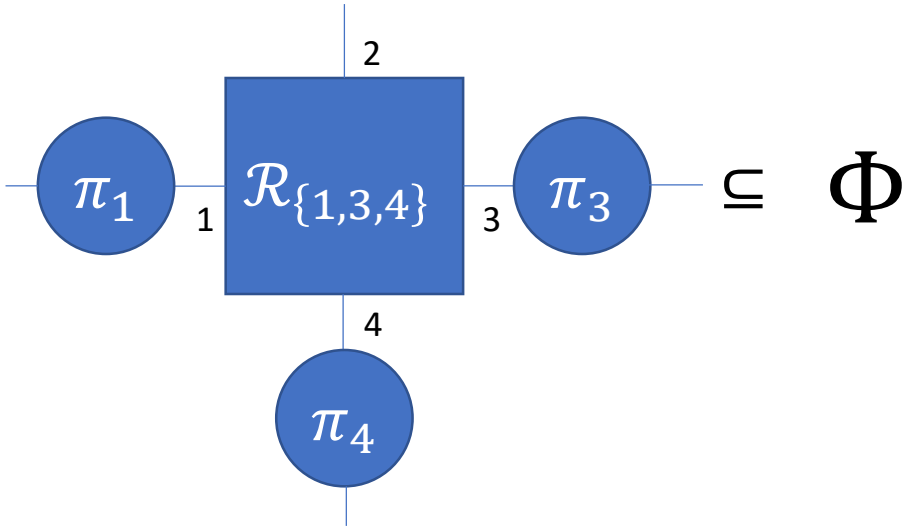


$\forall H \subseteq P \;\; \mathcal{R}_H \xrightarrow{\{\pi_i \mid i \in H\}} \mathcal{S}_H$

Multi-Party Constructive Cryptography

$\mathcal{P} = \{1, \ldots, n\}$

Protocol $\pi = (\pi_1, \ldots, \pi_n)$

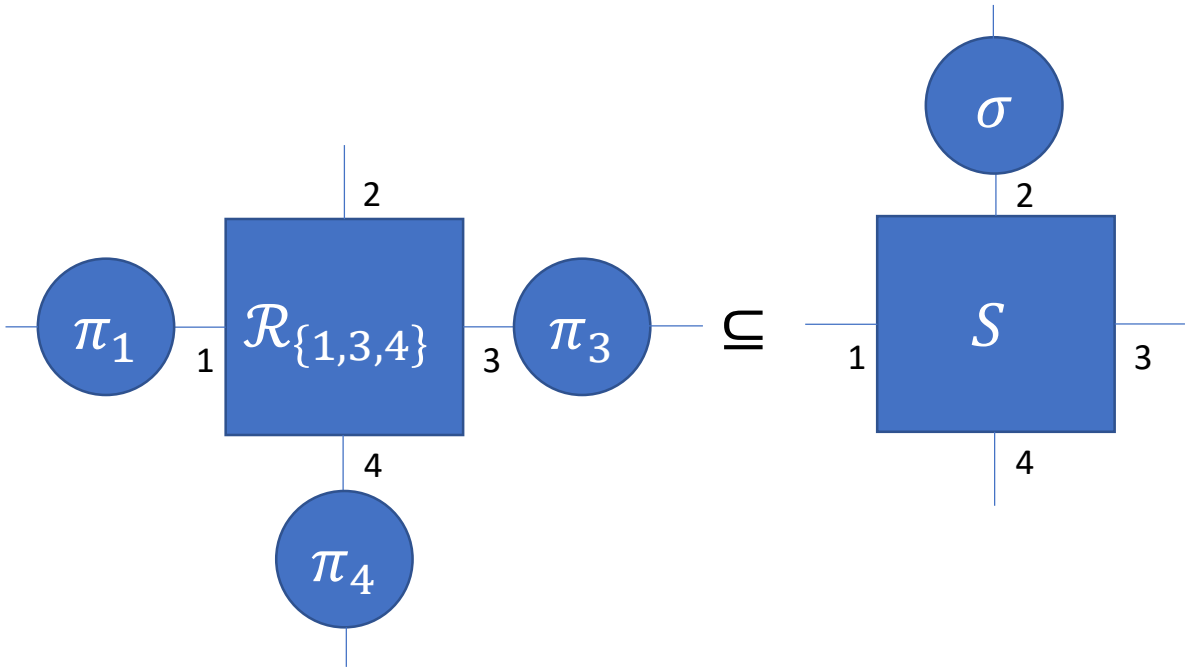


$\subseteq \quad \Phi$

$\forall H \subseteq P \;\; \mathcal{R}_H \xrightarrow{\{\pi_i \mid i \in H\}} \mathcal{S}_H$

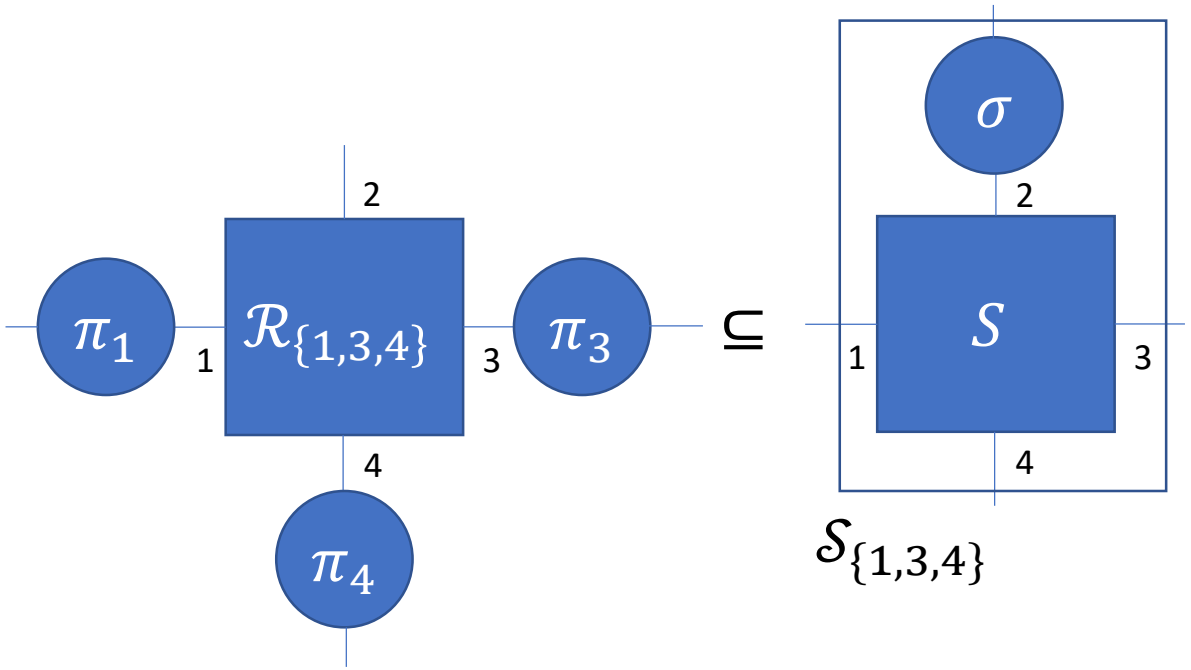
Multi-Party Constructive Cryptography

Traditional simulation-based notion



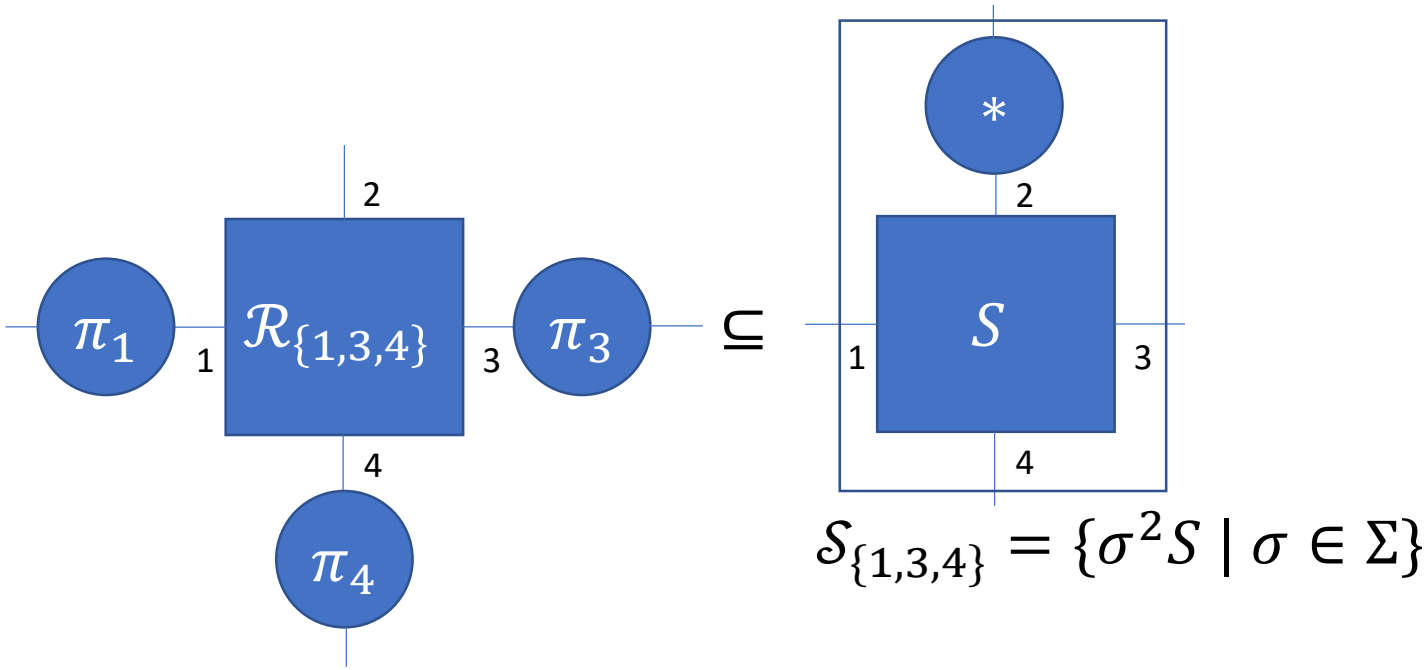
Multi-Party Constructive Cryptography

Traditional simulation-based notion

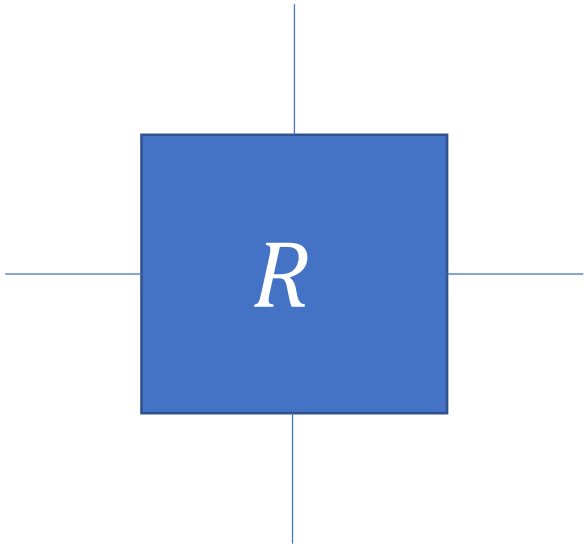


Multi-Party Constructive Cryptography

Traditional simulation-based notion

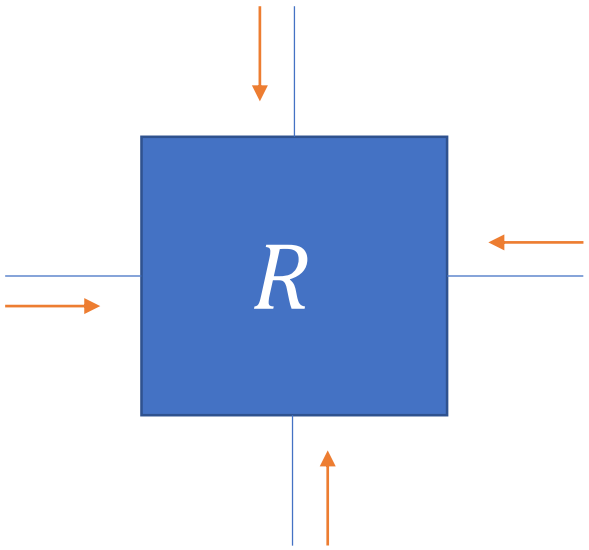


Synchronous Systems



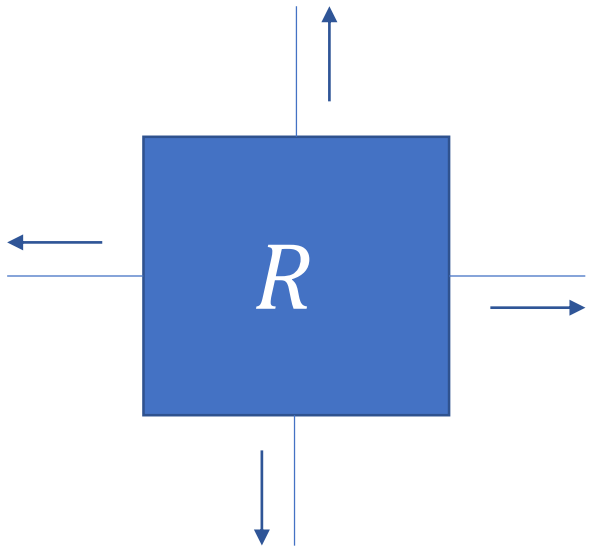
Resource

Synchronous Systems



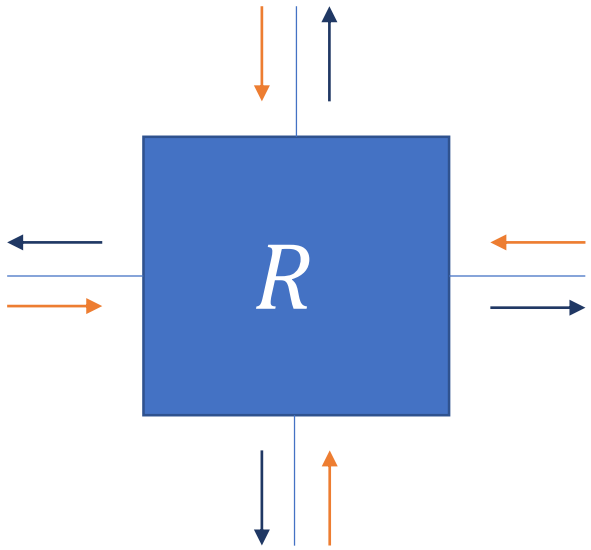
Resource

Synchronous Systems

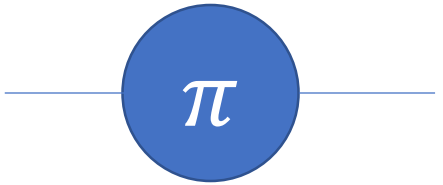


Resource

Synchronous Systems

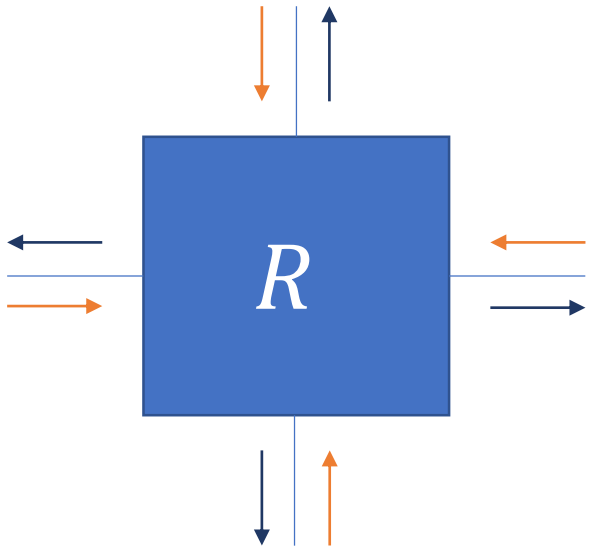


Resource

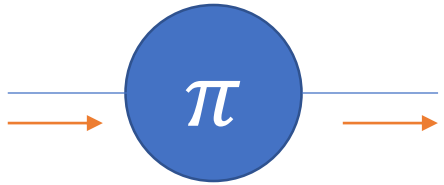


Converter

Synchronous Systems

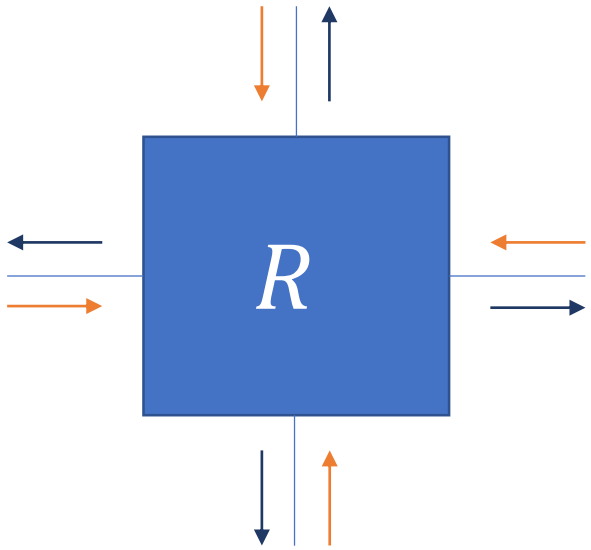


Resource

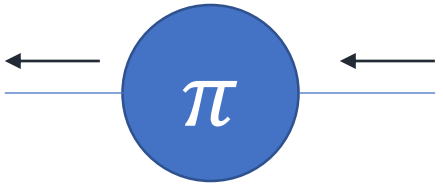


Converter

Synchronous Systems

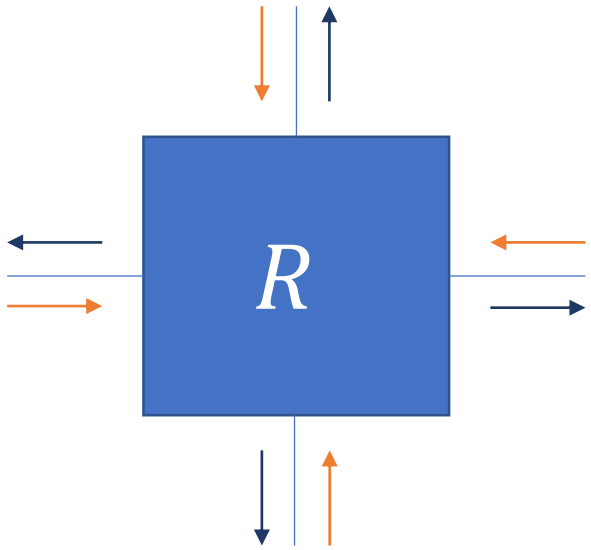


Resource

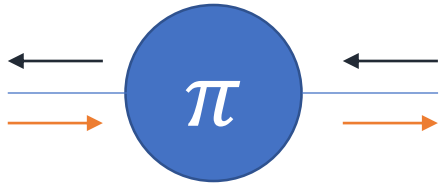


Converter

Synchronous Systems

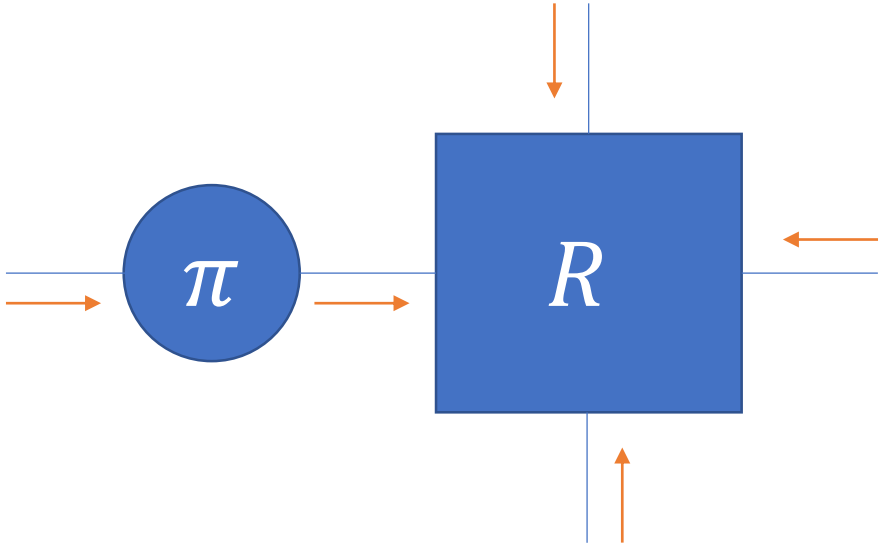


Resource

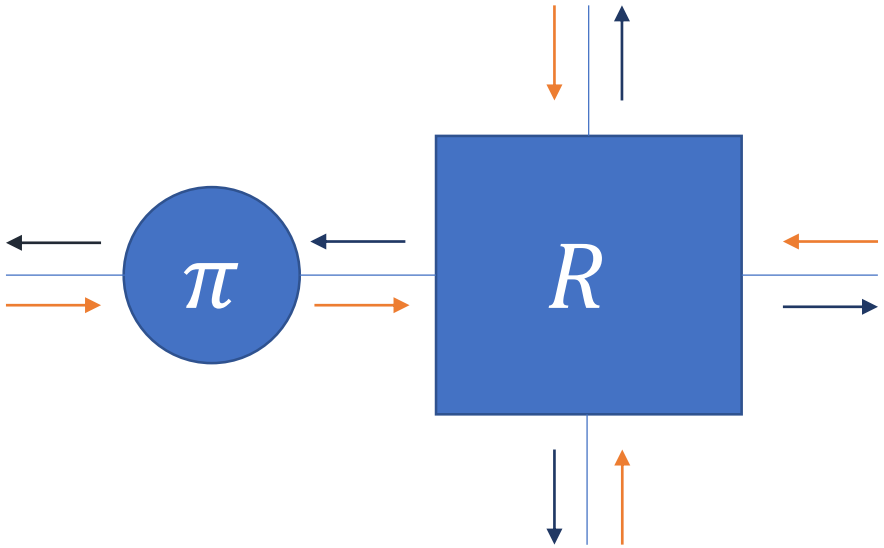


Converter

Synchronous Systems

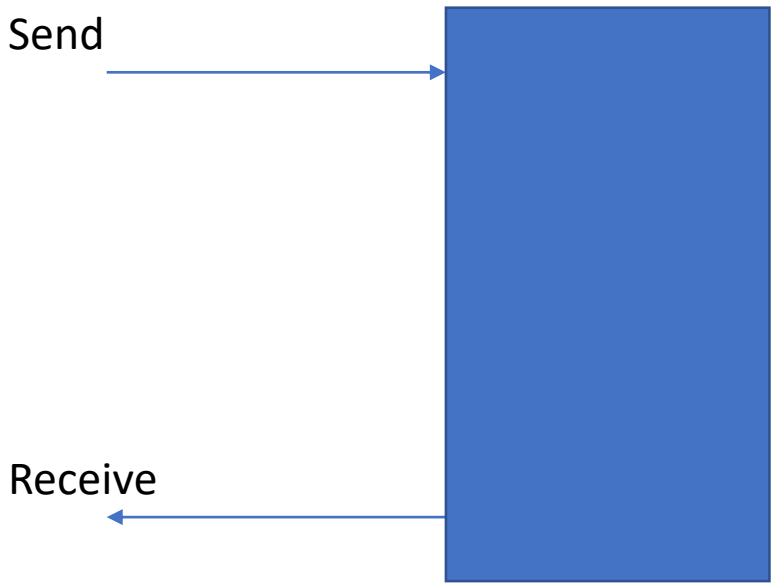


Synchronous Systems



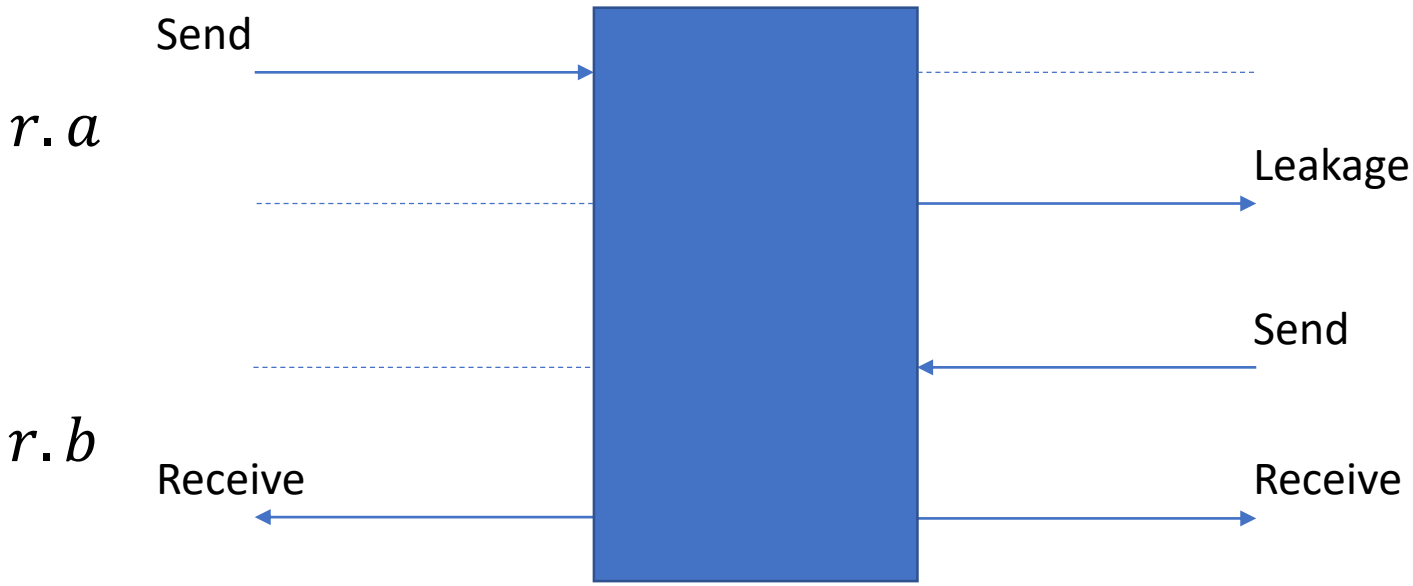
Round Structure

Round r

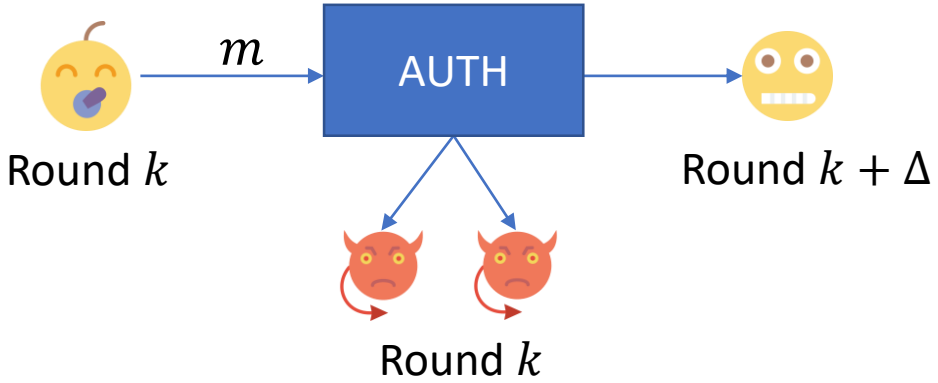


Round Structure

Round r

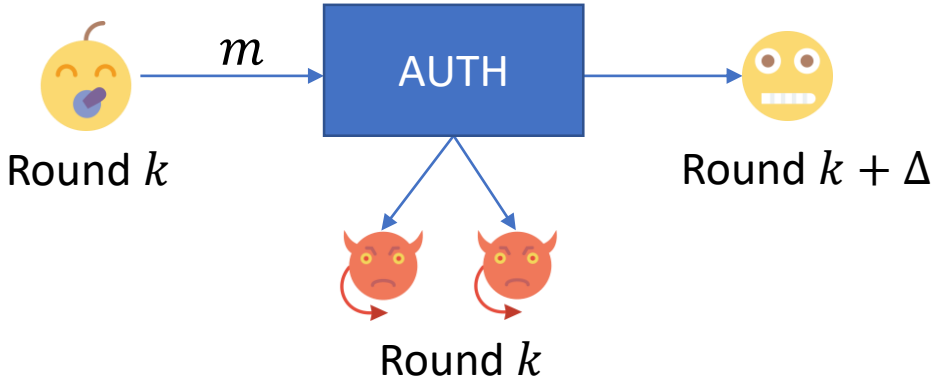


Authenticated Channel with Upper Bound Δ



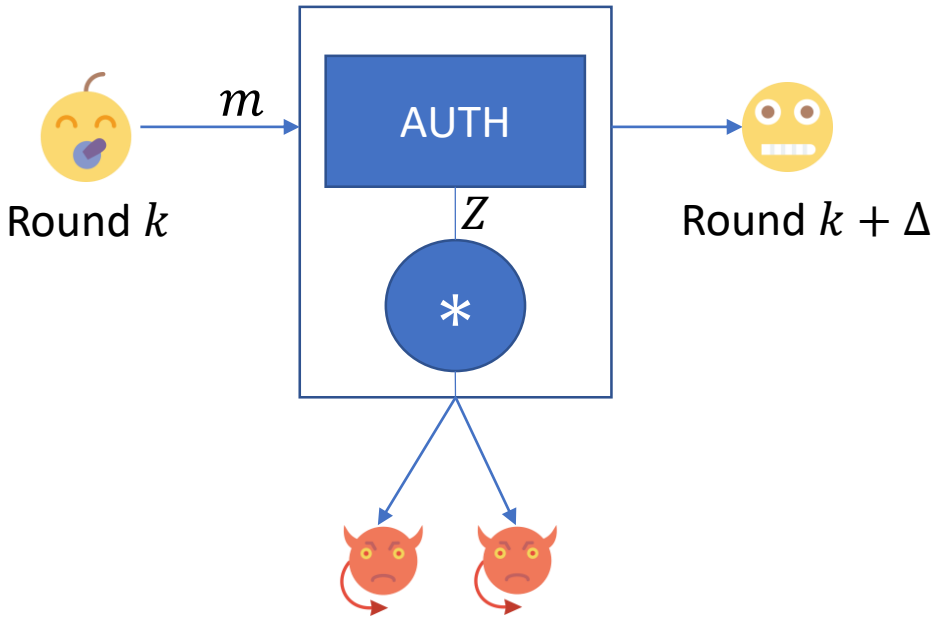
- Honest parties are guaranteed to get m after Δ rounds

Authenticated Channel with Upper Bound Δ



- Honest parties are guaranteed to get m after Δ rounds
- Dishonest parties are guaranteed to get m in the same round

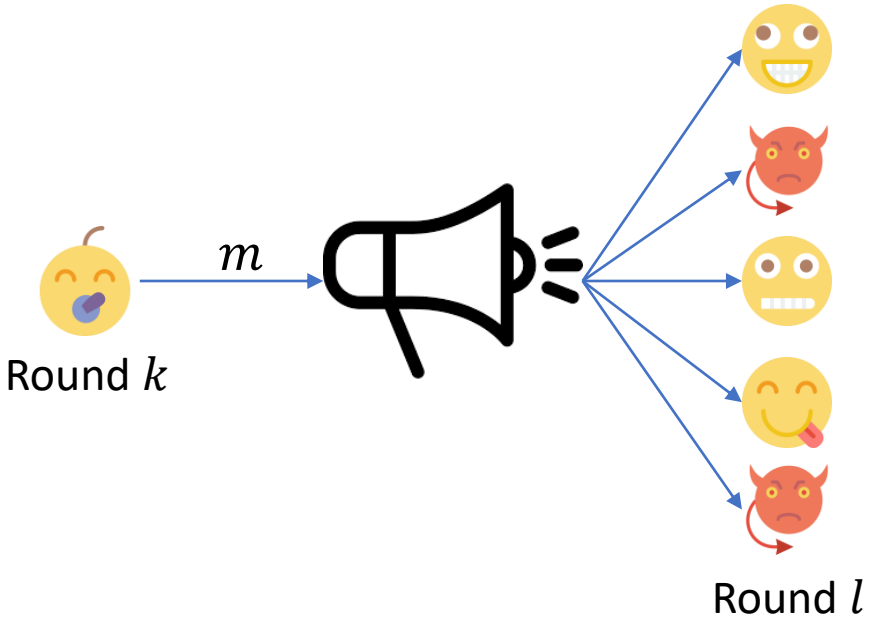
Authenticated Channel with Upper Bound Δ



- Honest parties are guaranteed to get m after Δ rounds
- Dishonest parties do not have any guarantee

$$\mathcal{AUTH}_{\Delta,Z} = \{\pi_Z \text{ AUTH}_\Delta \mid \pi \in \Sigma\}$$

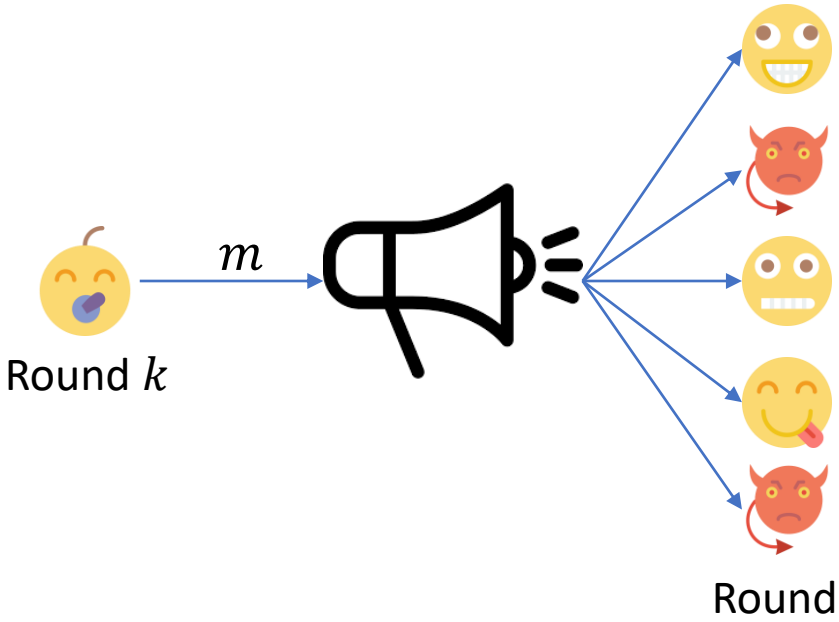
Broadcast



Validity: If sender is honest, all honest receivers output m

Consistency: All honest receivers output the same m’

Broadcast



Validity: If sender is honest, all honest receivers output m (at round l)

Consistency: All honest receivers output the same m' (at round l)

$$\mathcal{BC}_{k,l,H} = \{R \mid \exists v \, [\underbrace{(\forall P_j \in H \; y_j^l = v)}_{\text{Consistency}} \wedge \underbrace{(P_s \in H \rightarrow v = x_s^k)}_{\text{Validity}}]\}$$

Broadcast

$$\mathcal{BC}_{k,l,H}=\left\{R \mid \exists v \left[\left(\forall P_j \in H\ y_j^l = v\right) \wedge \left(P_s \in H \rightarrow v = x_s^k\right)\right]\right\}$$

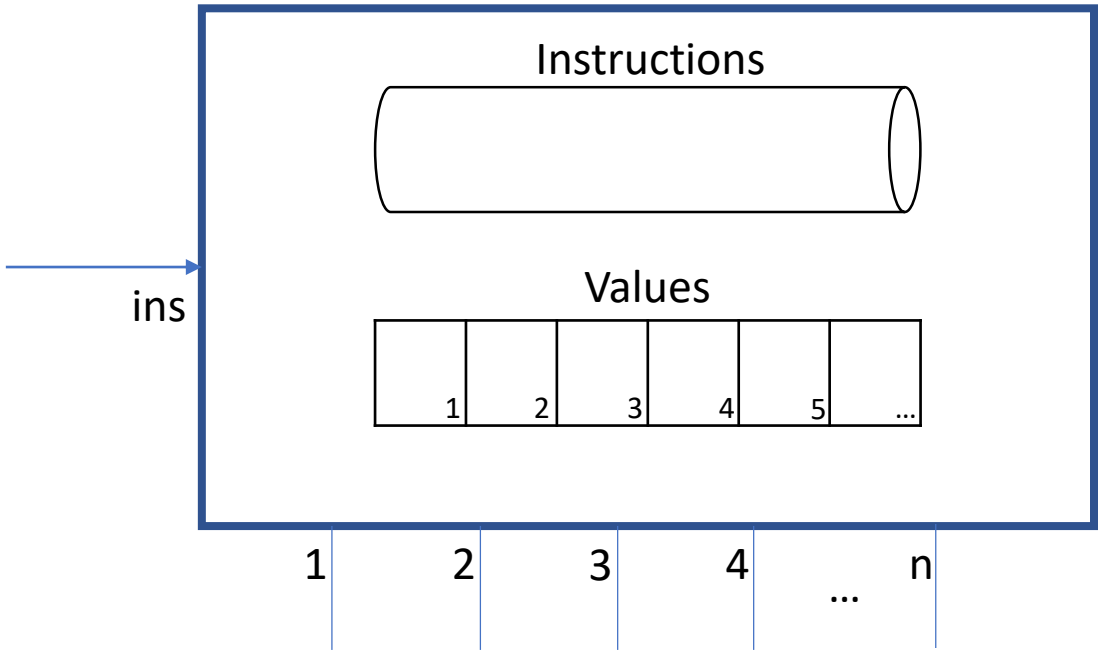
Let $\pi = (\pi_1, \ldots, \pi_n)$ be a (standard) broadcast protocol that takes Δ rounds

$$\pi_H \mathcal{NET} \subseteq \mathcal{BC}_{k,k+\Delta,H}$$

Standard proof

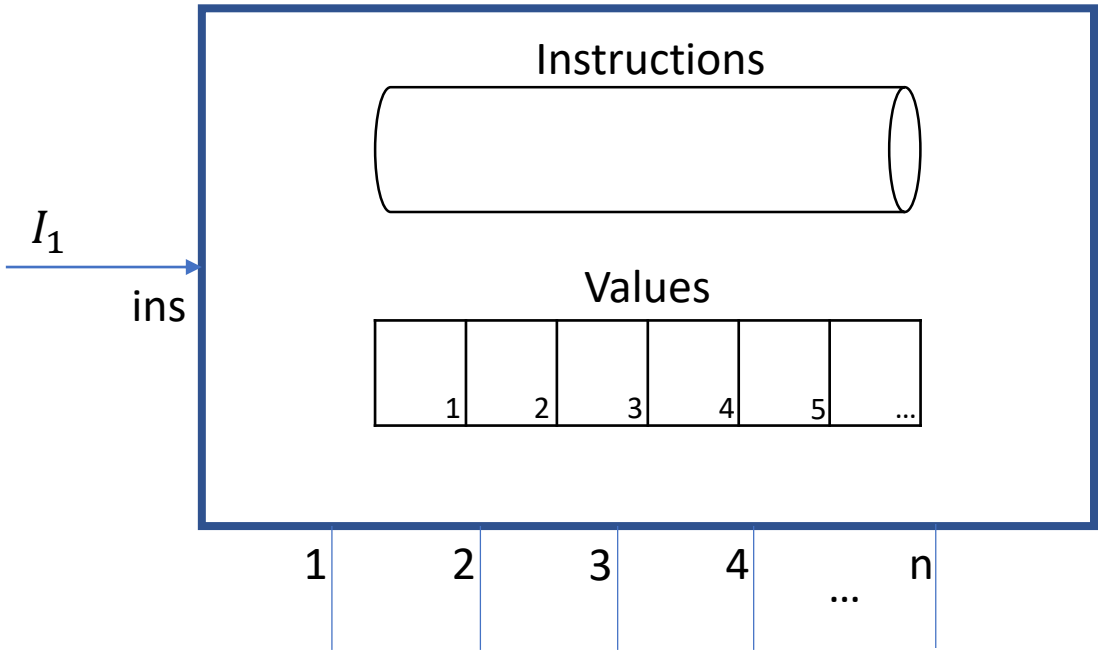
- Validity: If $P_s \in H$, all honest parties obtain m at round $k + \Delta$
- Consistency: All honest parties obtain the same value at round $k + \Delta$

Computer Resource

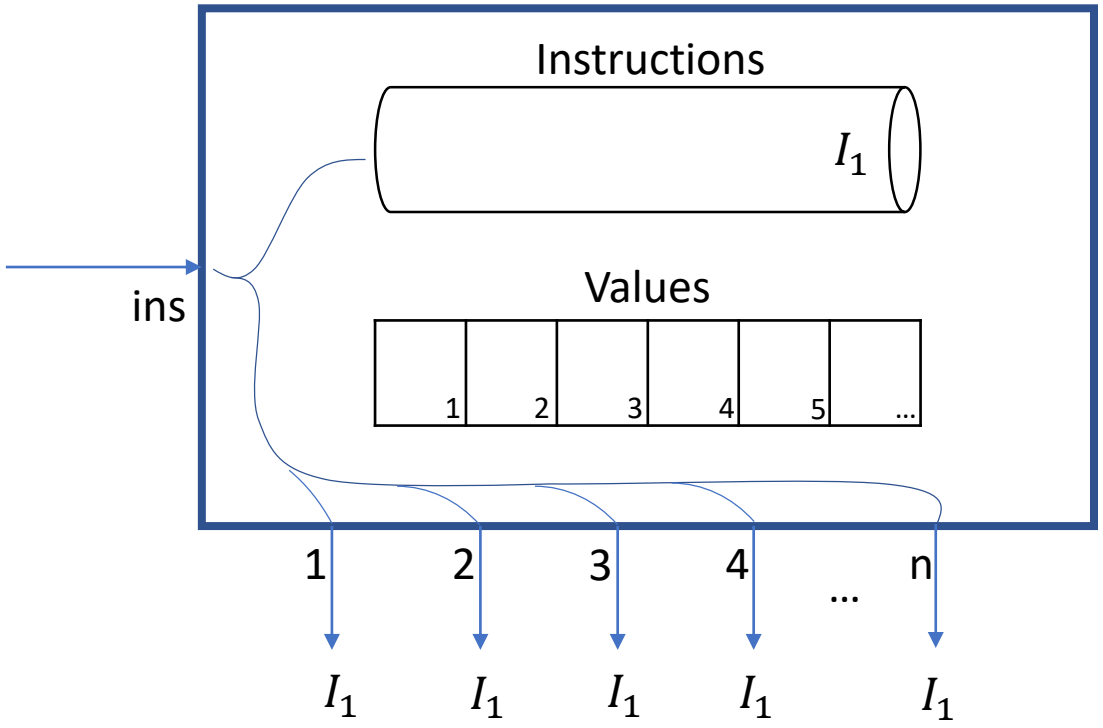


*see also *arithmetic black-box* [DN03]

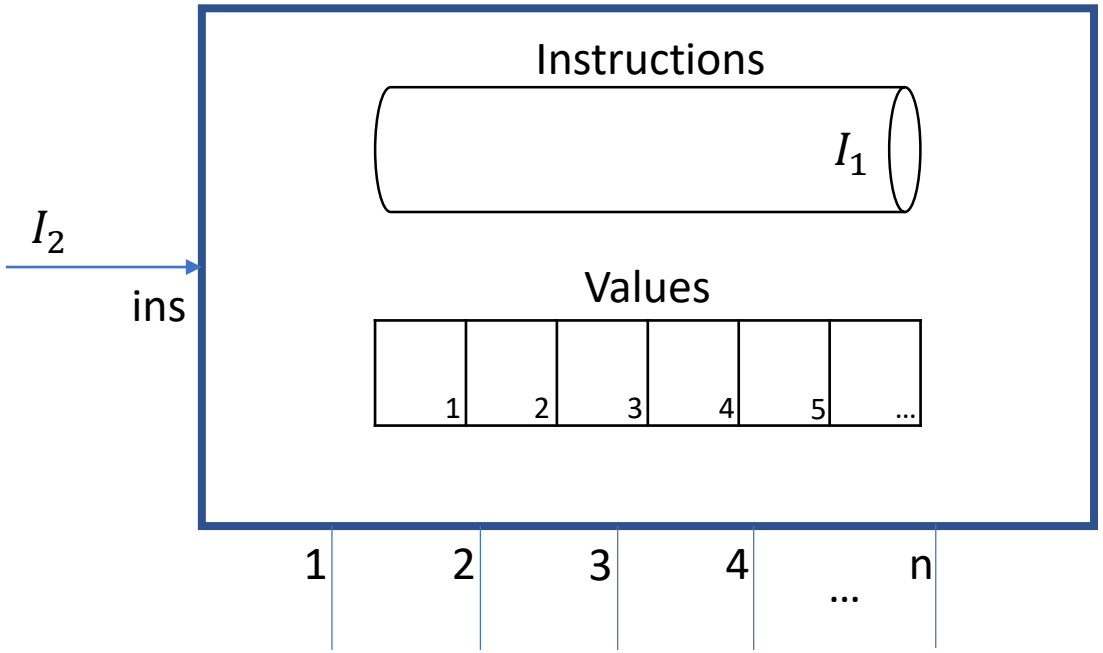
Computer Resource



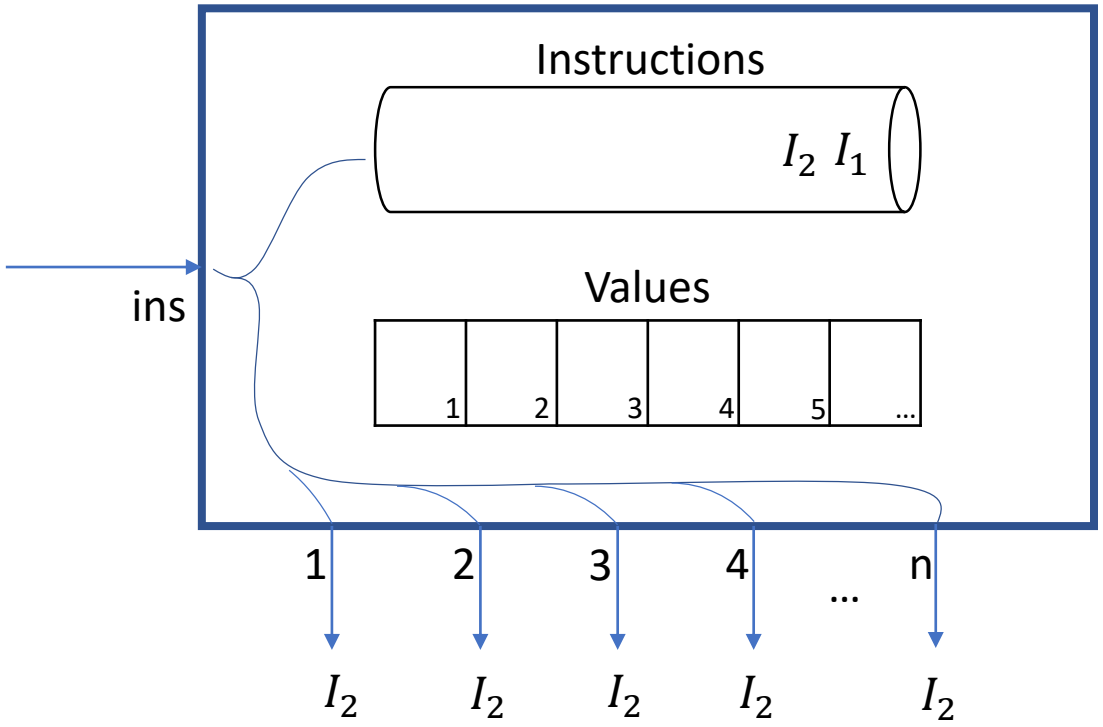
Computer Resource



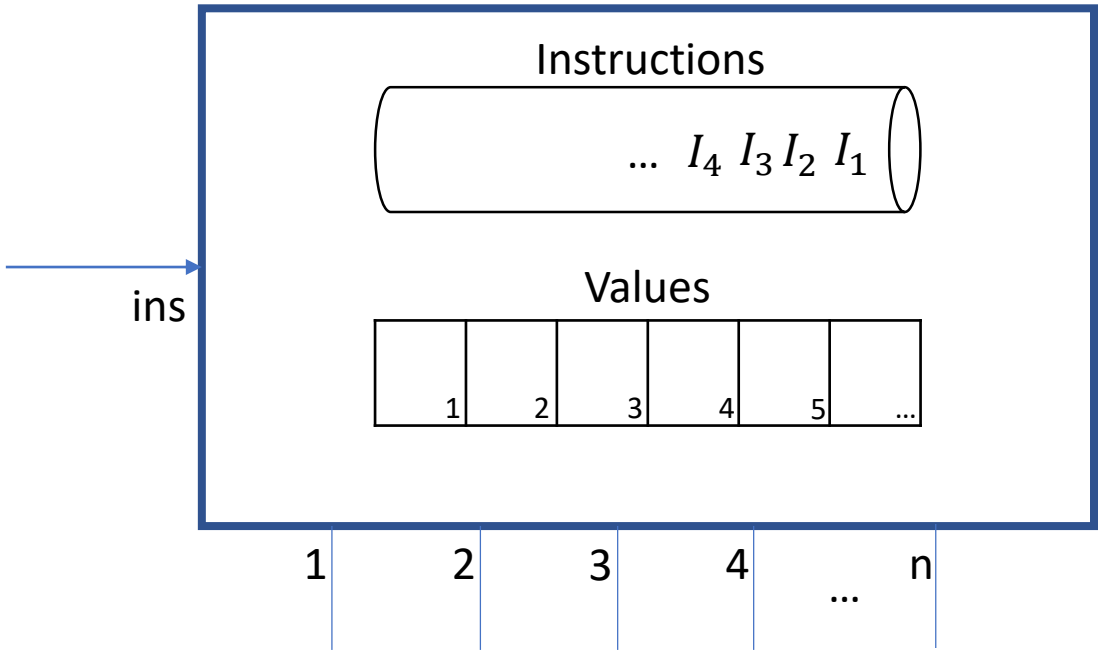
Computer Resource



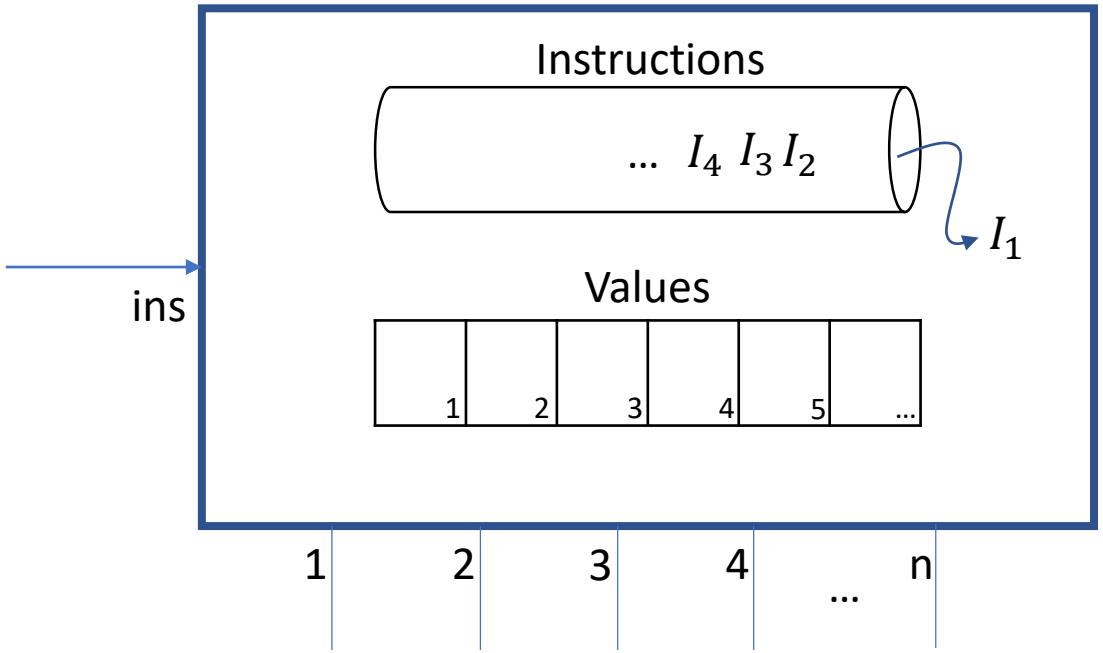
Computer Resource



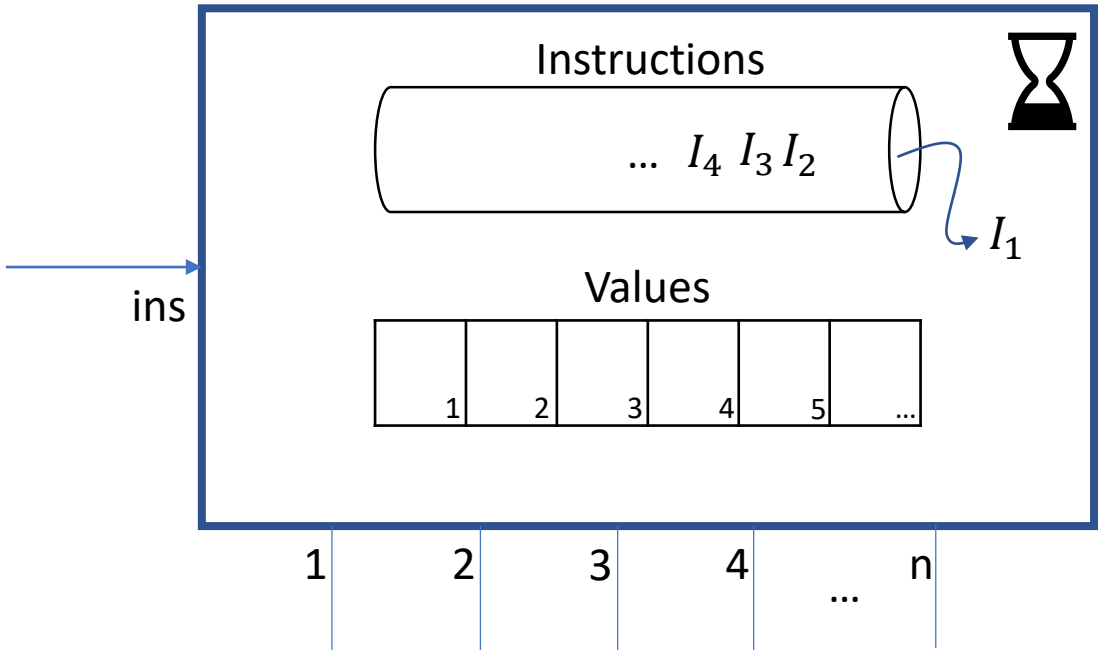
Computer Resource



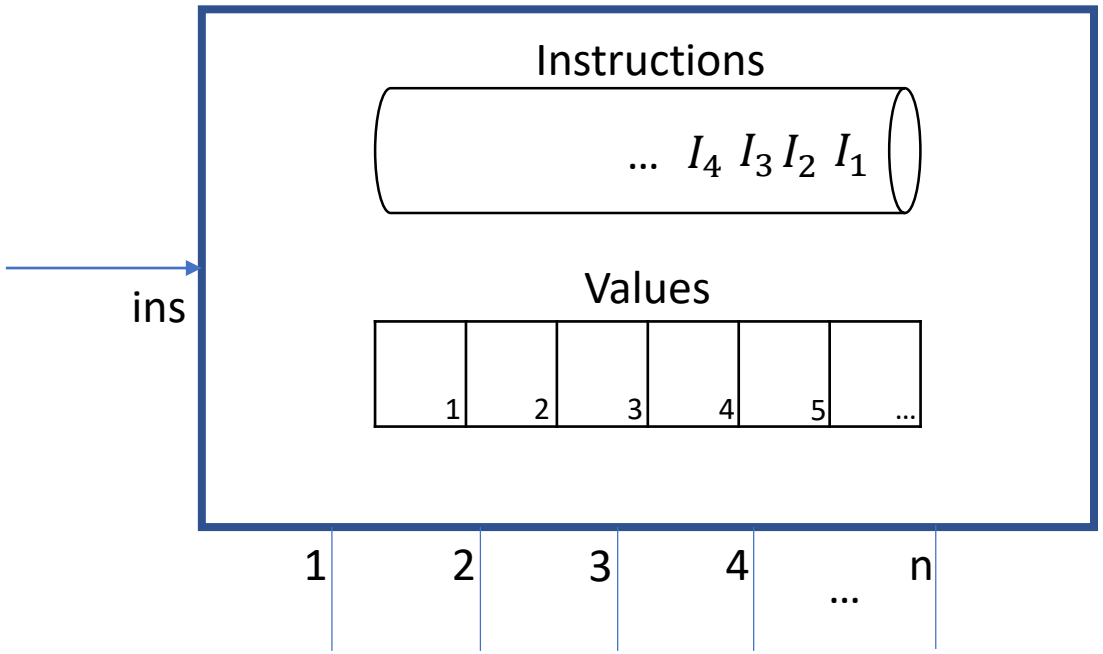
Computer Resource



Computer Resource



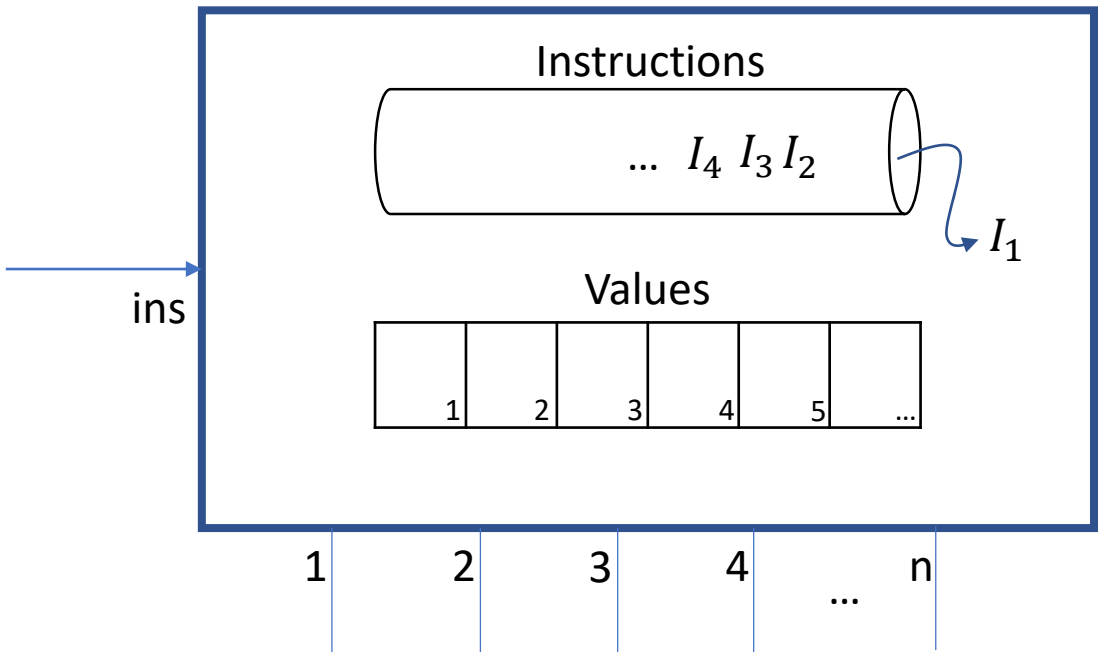
Computer Resource



Instruction I:

1. (input, i, p)
2. (output,i,p)
3. (add,p_1,p_2,p_3)
4. (mult,p_1,p_2,p_3)

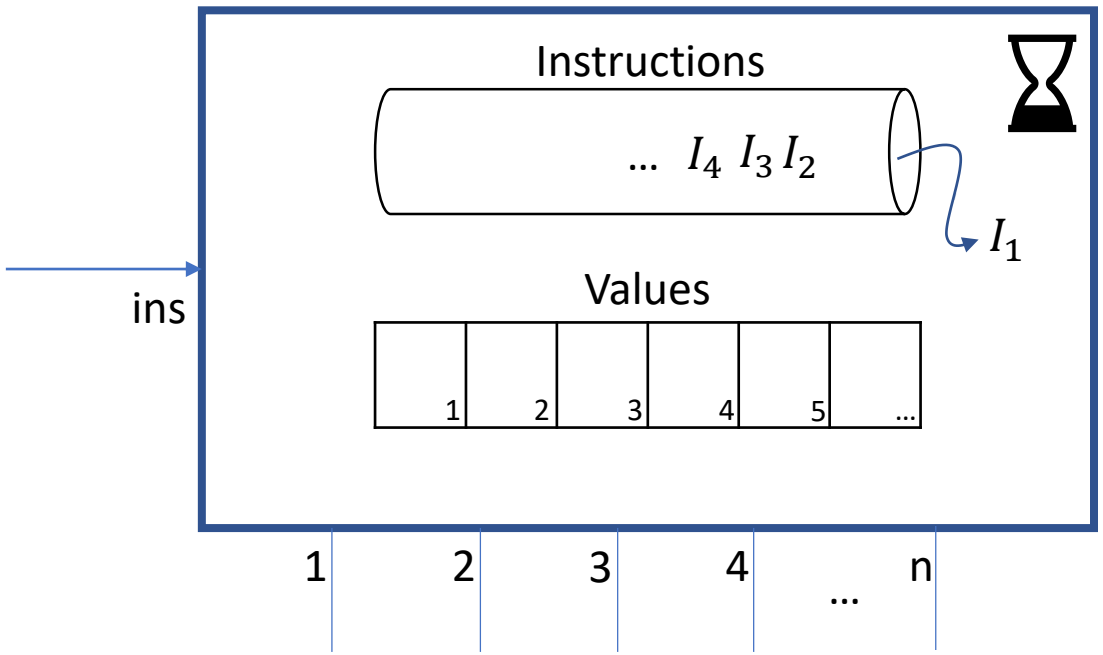
Computer Resource



Instruction I:

1. **(input, i, p)**
2. (output,i,p)
3. (add,p_1,p_2,p_3)
4. (mult,p_1,p_2,p_3)

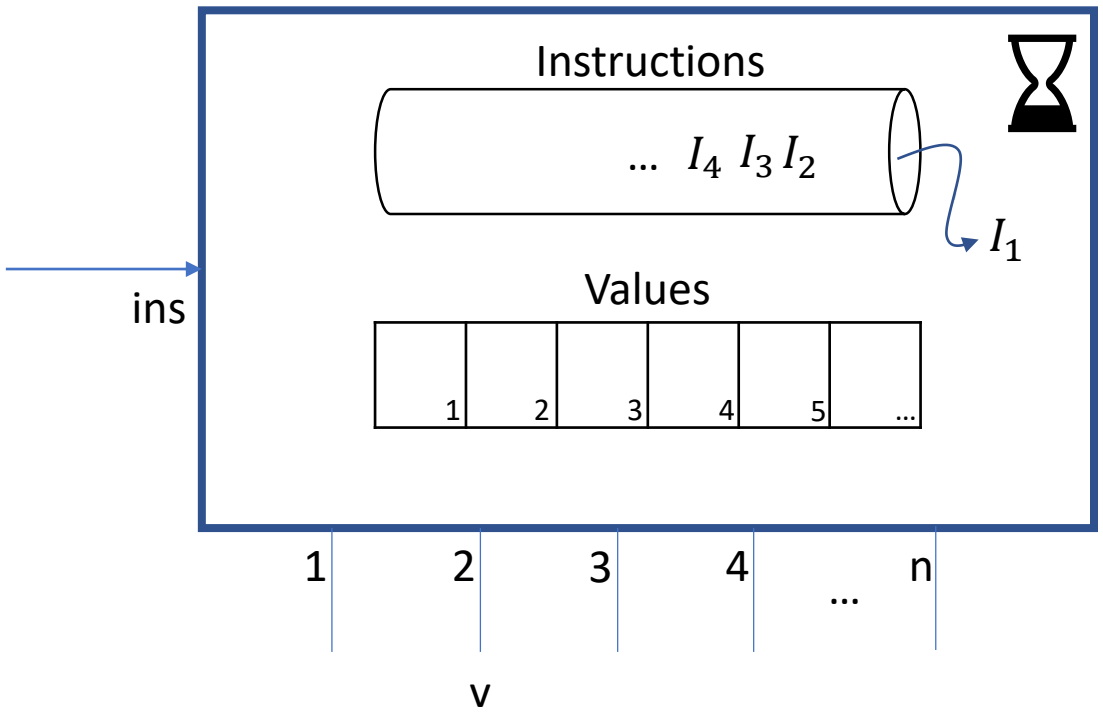
Computer Resource



Instruction I:

1. **(input, i, p)**
2. (output,i,p)
3. (add,p_1,p_2,p_3)
4. (mult,p_1,p_2,p_3)

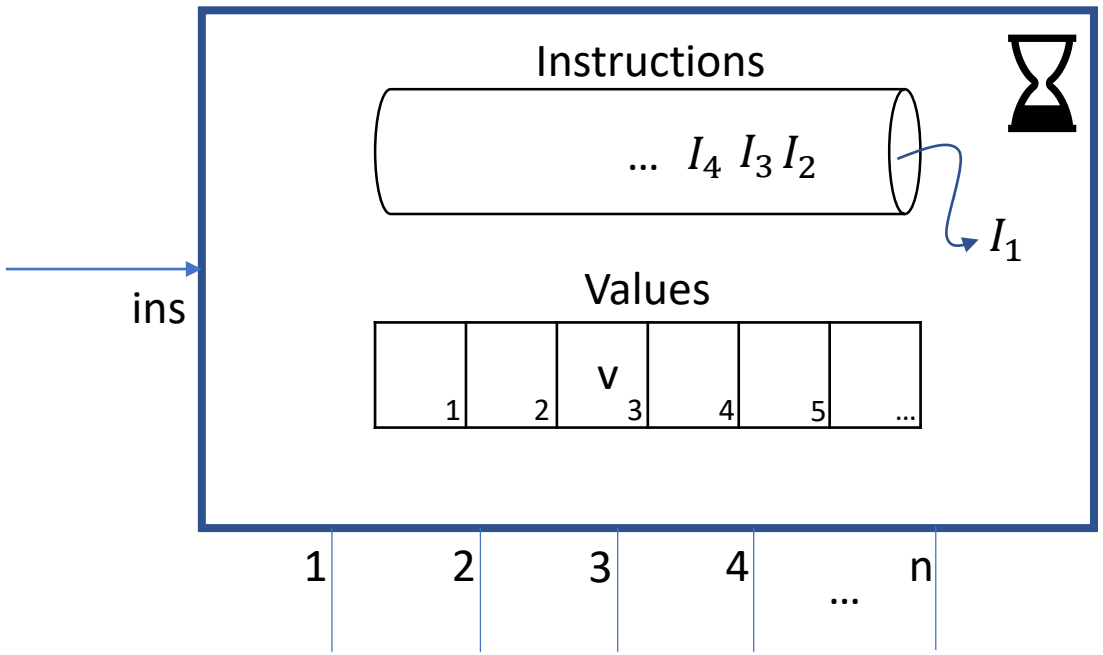
Computer Resource



Instruction I:

1. **(input, i, p)**
2. (output,i,p)
3. (add,p_1,p_2,p_3)
4. (mult,p_1,p_2,p_3)

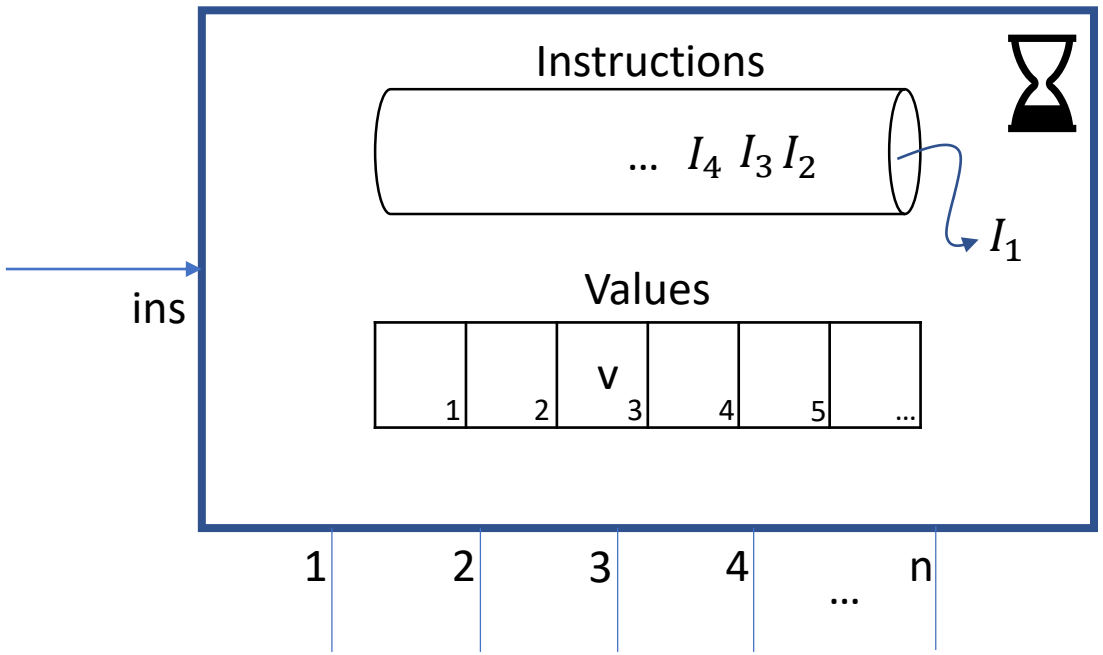
Computer Resource



Instruction I:

1. **(input, i, p)**
2. (output,i,p)
3. (add,p_1,p_2,p_3)
4. (mult,p_1,p_2,p_3)

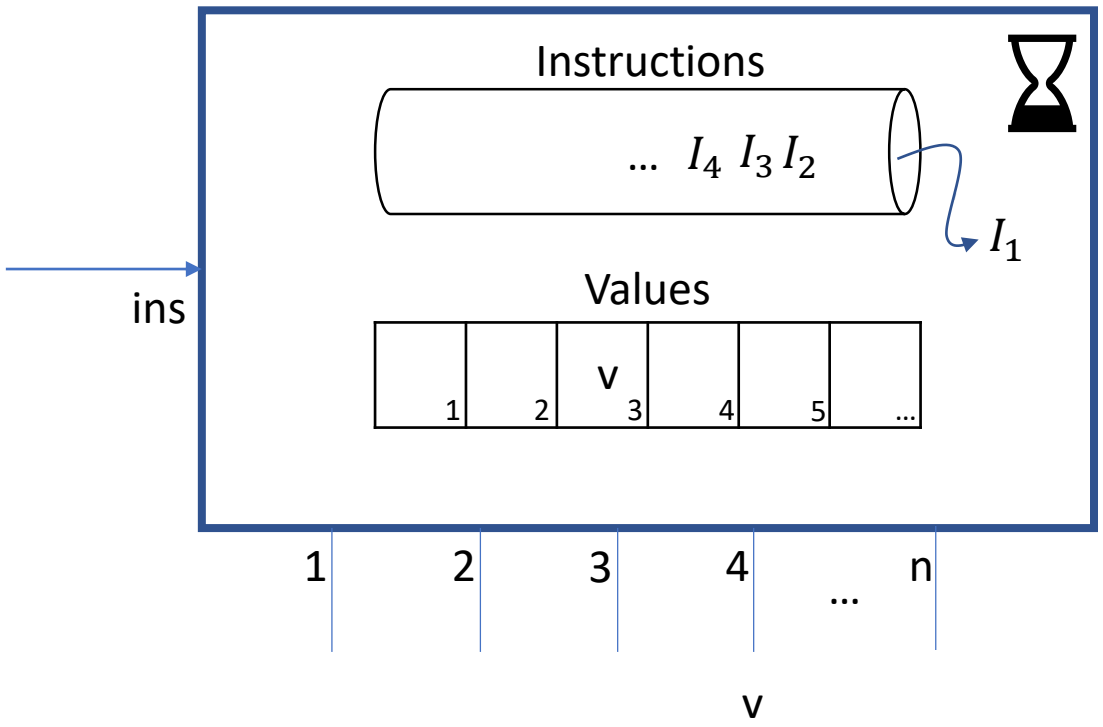
Computer Resource



Instruction I:

1. (input, i, p)
2. **(output,i,p)**
3. (add,p_1,p_2,p_3)
4. (mult,p_1,p_2,p_3)

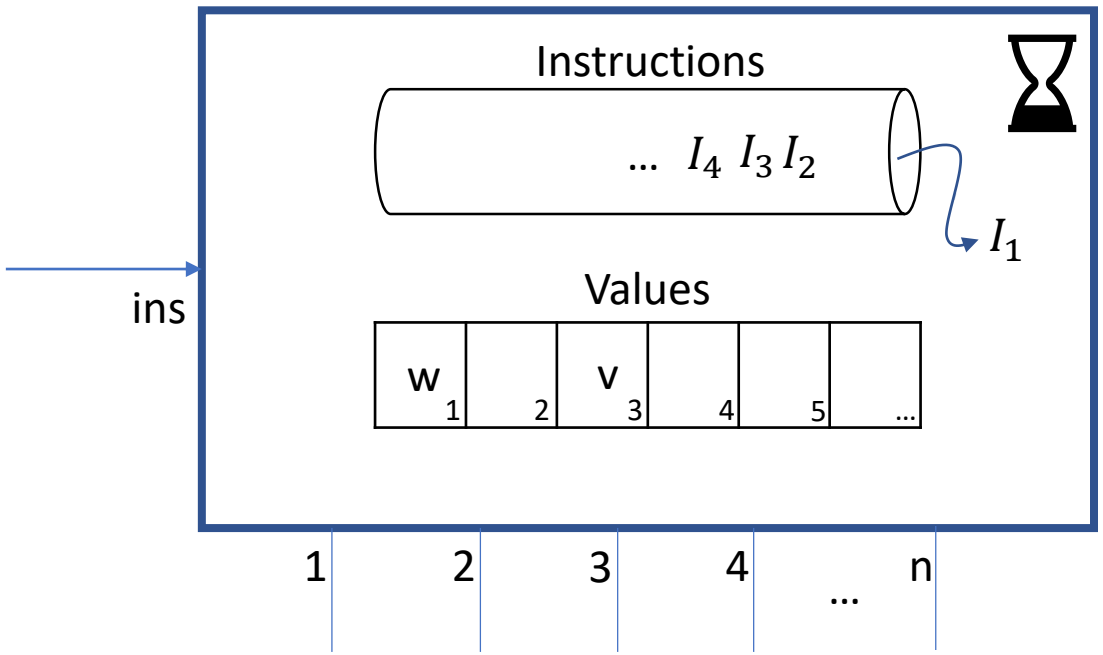
Computer Resource



Instruction I:

1. (input, i, p)
2. **(output,i,p)**
3. (add,p_1,p_2,p_3)
4. (mult,p_1,p_2,p_3)

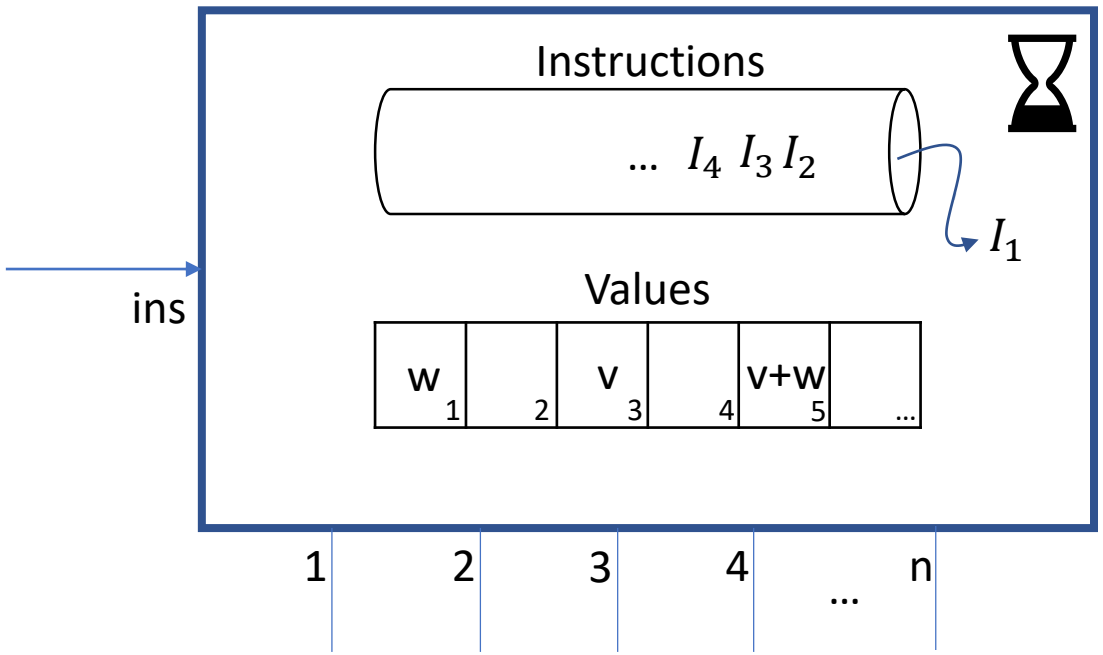
Computer Resource



Instruction I:

1. (input, i, p)
2. (output,i,p)
3. **(add,p_1,p_2,p_3)**
4. (mult,p_1,p_2,p_3)

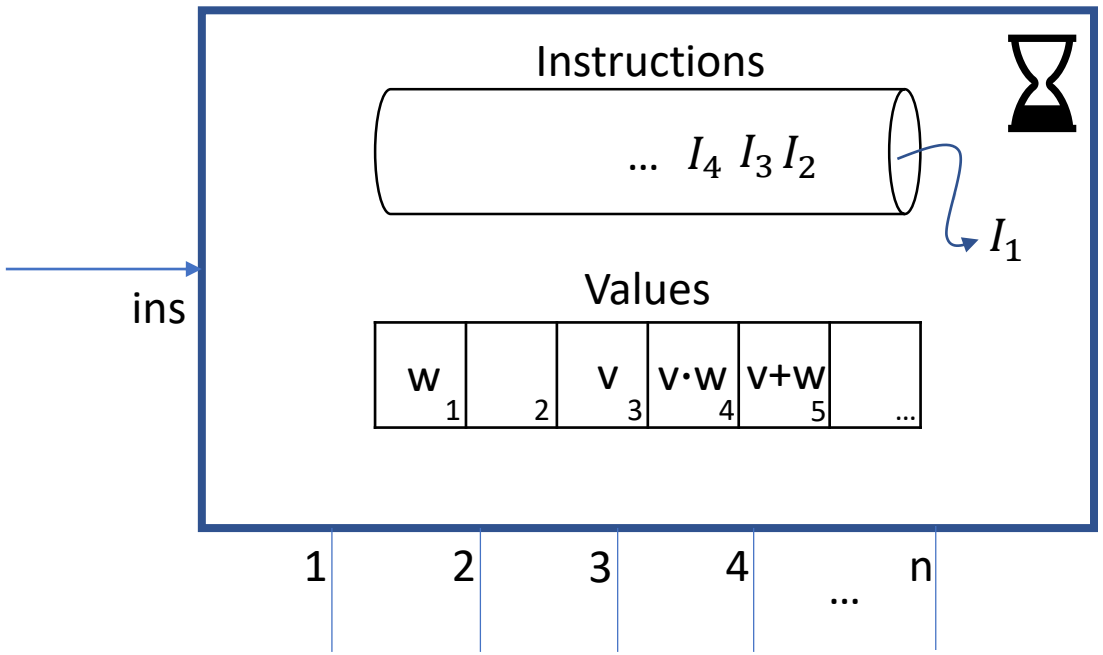
Computer Resource



Instruction I:

1. (input, i, p)
2. (output,i,p)
3. **(add,p_1,p_2,p_3)**
4. (mult,p_1,p_2,p_3)

Computer Resource



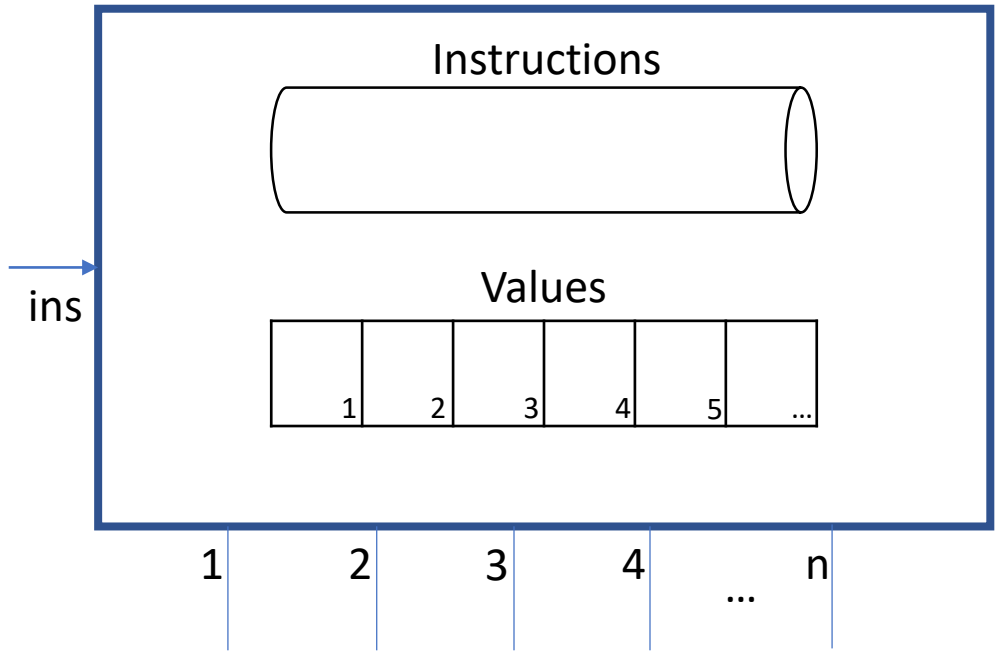
Instruction I:

1. (input, i, p)
2. (output,i,p)
3. (add,p_1,p_2,p_3)
4. **(mult,p_1,p_2,p_3)**

MPC as Computer

$[\mathcal{BC}, \mathcal{NET}]$

[BGW88]
[Mau06]



Conclusions

- Simple model for synchronous protocols
- Parties are honest/dishonest
- Information-theoretic statements
- Flexible to capture property-based formalizations

Full version: https://eprint.iacr.org/2020/1226

Credits:
Icons: https://www.flaticon.com/