Building a Cryptoeconomic Tool Set

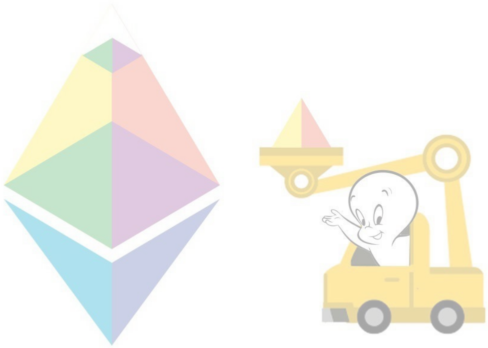
Casper CBC Team

Vlad Zamfir, Aditya Asgaonkar, Ben Jones, Georgios Piliouras, Nate Rush

Outline

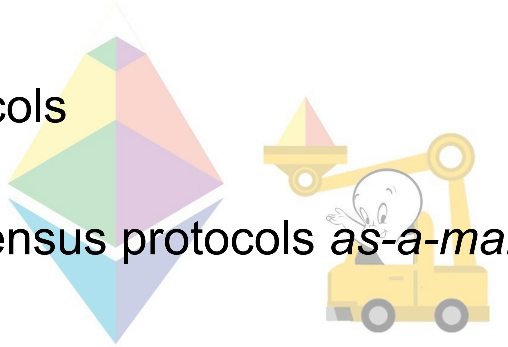
- Introduction
    - PRESTO
- Design Philosophy
- Roadmap
- Examples
    - AND-Gate Game
        - Game 1: One-shot game
        - Game 2: One-shot game with deposits
        - Game 3: One-shot game with deposits and chance of failure
        - Game 4: Iterated game with deposits and chance of failure *(time permitting)*
    - Censorship Game
- Discussion & Conclusion

Introduction



Where did Cryptoeconomics come from?

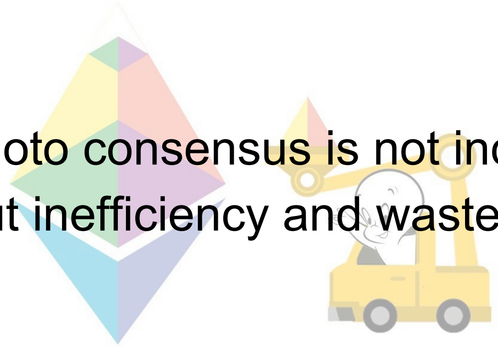
Satoshi Nakamoto's "Proof-of-Work" consensus protocol introduced two innovations:



- Forking consensus protocols
- Public, incentivized consensus protocols *as-a-marketplace*
- The second innovation was the genesis block of Cryptoeconomics
- This got us thinking about how to properly incentivize consensus protocols

Where did Cryptoeconomics come from?

- It turns out that Nakamoto Consensus is unreasonably simple



- We found out that Nakamoto consensus is not incentive compatible
- We were concerned about inefficiency and waste in proof-of-work

- “Proof-of-Stake”, an alternative to “Proof-of-Work” is an alternative model for the incentivization of public consensus protocols

Where did Cryptoeconomics come from?

- “Proof-of-Stake” refers to the use of digital assets (as opposed to proof-of-work’s computational costs) for the incentivization of consensus

- Consensus protocols are distributed systems that allow nodes to make consistent decisions out of mutually exclusive available alternatives

- But proof-of-stake (and Cryptoeconomics) is useful (interested in) a boarder set of distributed protocols

What is Cryptoeconomics?

The **goal** of Cryptoeconomics is:

Given any **distributed protocol** to be able to deploy it

1. To the public internet **(open participation)**
2. by **incentivizing** people to run nodes that **faithfully** implement roles defined in the protocol
3. in a way that is **publicly verifiable**

Formal Foundations of Cryptoeconomics: the PRESTO framework

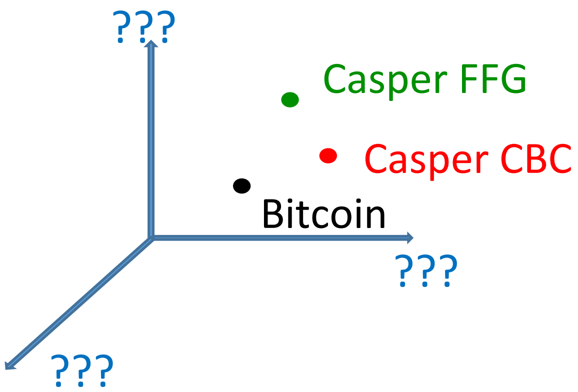
Georgios Piliouras
georgios@ethereum.org

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

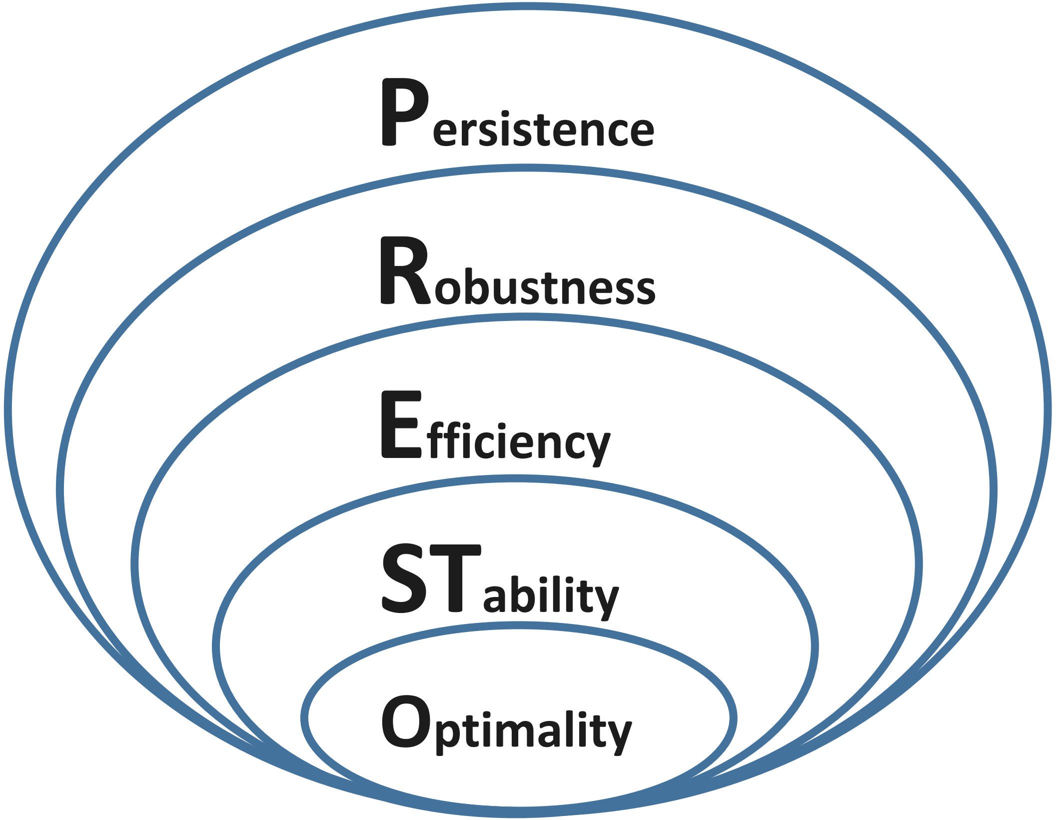
joint w. Vitalik Buterin (Ethereum) and Daniel Reijsbergen (SUTD), Vlad Zamfir (Ethereum)

Language is everything

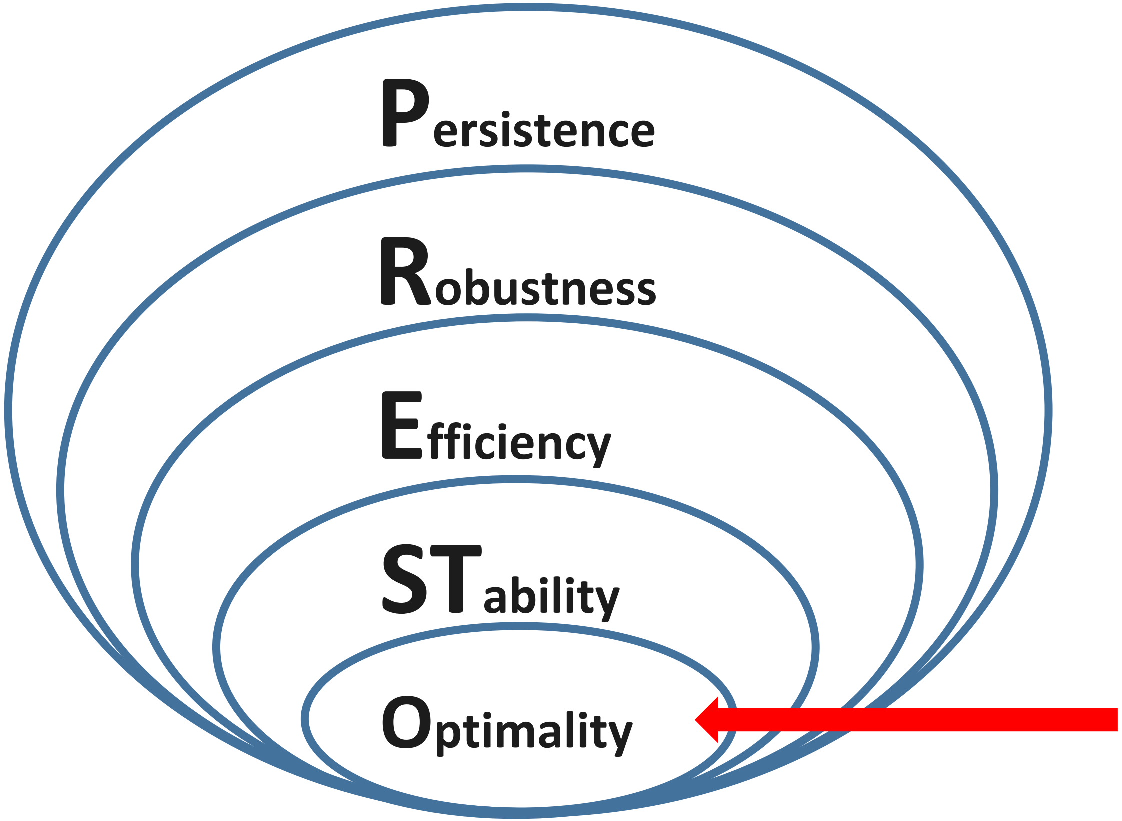
- How do we **speak/argue** about the systems we build?
- What does it mean that a specific protocol is ``**better"** than another one?
- What makes the language problem tricky is that consensus, cryptoeconomic protocols should be thought as *points in a high-dimensional space.*



The Nesting Doll (PRESTO) Design

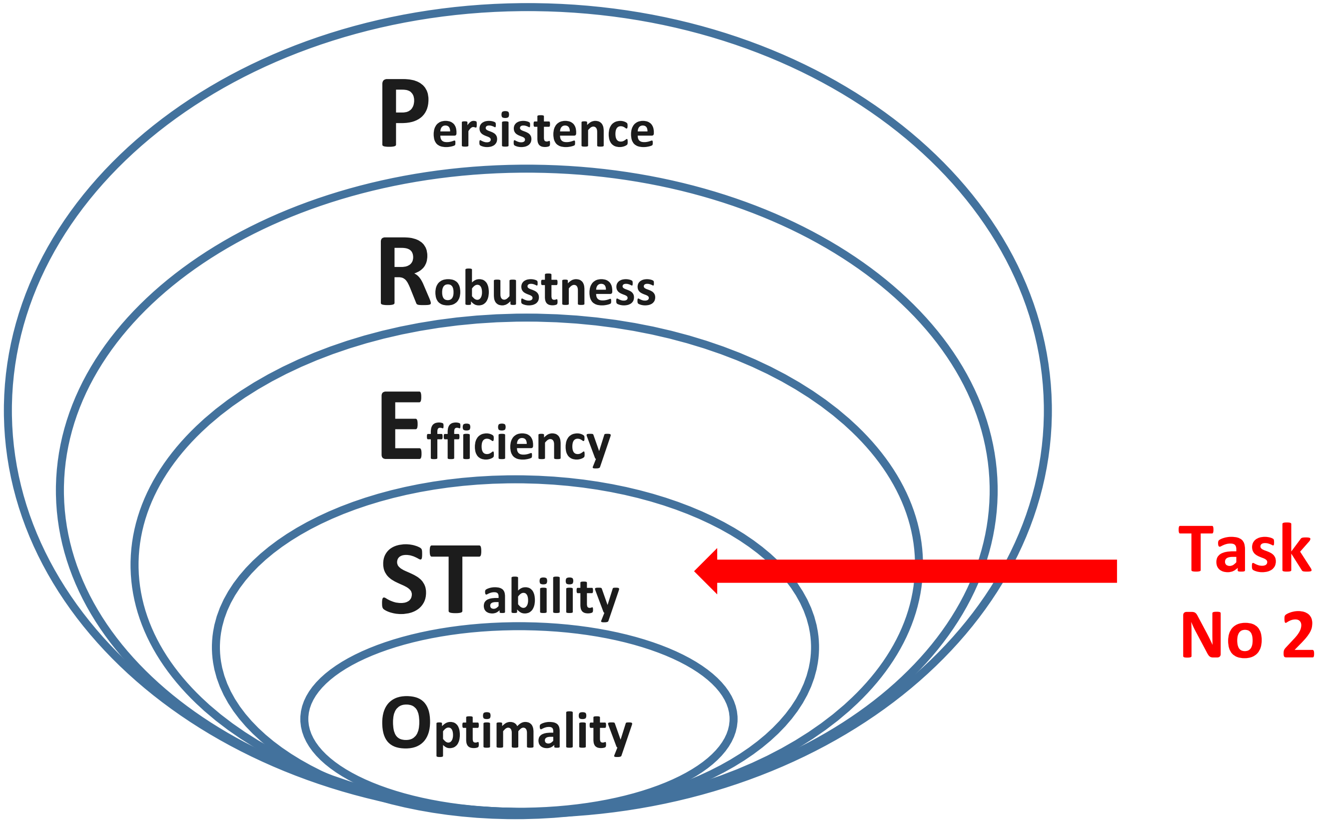


The Nesting Doll (PRESTO) Design



Task
No 1

The Nesting Doll (PRESTO) Design



The PRESTO framework

Optimality: Does the protocol maximize the quality of outcomes?

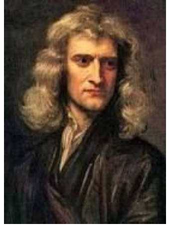
STability: Is the designed protocol an equilibrium?

Efficiency: How efficiently does the protocol utilize its difference resources (e.g. time, space, energy, e.t.c.)

Robustness: Do the protocol performance guarantees withstand perturbations to the system parameters?

Persistence: If the protocol is forced out of equilibrium, does it recover?

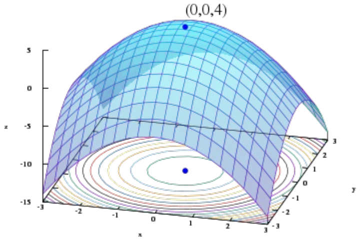
Optimality



Does the protocol maximize the quality of outcomes?

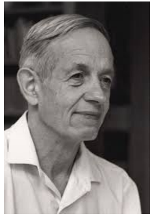
Optimization is a task of pure mathematics.

$$\max_{x \in S} f(x)$$



- The function f can express any aspect of our system that we wish to maximize (e.g. # of valid transactions per second)
- This is an ideal world where we do not have to worry about the physical implementation of the solution.
- Long history (since the **1600s**, e.g., Fermat, Lagrange, Gauss, Newton)

Stability



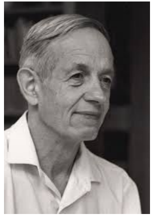
Is the protocol a Nash equilibrium? Is the protocol incentive compatible?

Equilbrium is an outcome that is optimal from the perspective of all decision makers involved.

	Deer	Rabbit
Deer	20, 20	0, 1
Rabbit	1, 0	1, 1

Reasonably thoroughly studied (since **1940s**, Nash, von Neumann)

Stability



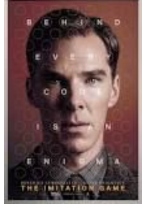
Is the protocol a Nash equilibrium? Is the protocol incentive compatible?

Equilbrium is an outcome that is optimal from the perspective of all decision makers involved.

	Deer	Rabbit
Deer	20, 20	0, 1
Rabbit	1, 0	1, 1

Stability *does not* imply optimality. Stability may be hard to enforce. **Can we design protocols that are (near) optimal equilibria?**

Efficiency



How efficiently does the protocol utilize its difference resources (e.g. time, space, energy, e.t.c.)

Efficiency is a task of computer science. Solve the problem below

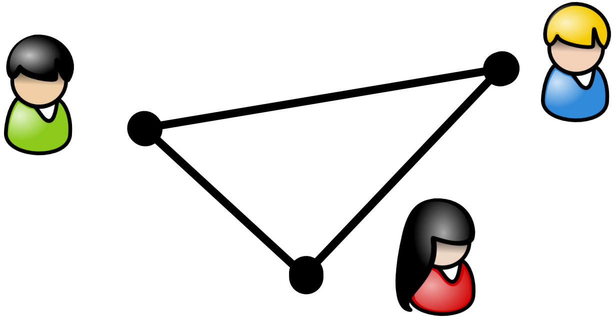
f, S $\longrightarrow$  $\longrightarrow$ $\max_{x \in S} f(x)$

- But do it as fast as possible, using as little space as possible, using as little randomness as possible, using as little energy as possible, use parallelization efficiently.
- Reasonably thoroughly studied (since **1940s**, Turing, von Neumann)
- **Not always possible:** E.g. Traveling Salesman Problem
- **Solution: Tradeoffs,** Approximate optimality vs. Speed

Optimality + Efficiency + Stability

Can we find a Nash equilibrium fast in general large games? **NO**
PPAD-complete [Daskalakis, Goldberg, Papadimitriou ’05]

Solution: Add (designed) payments to users to make equilibration easier.



This is studied by Algorithmic Game Theory (AGT) (since **1999**)

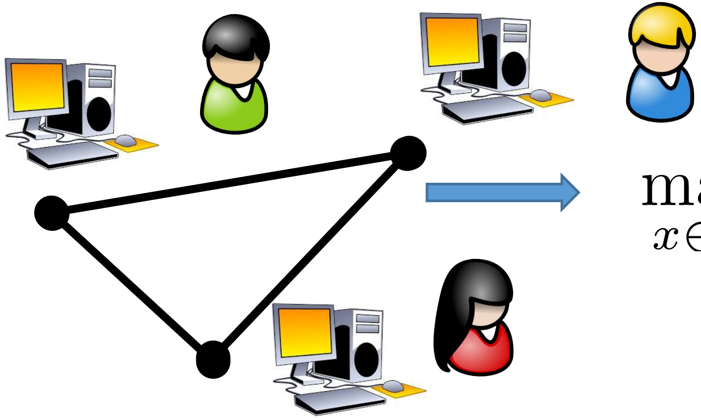
Algorithmic Mechanism Design: Design games to have equilibria at optimal states that are easy to verify/compute *(by design)*.

Optimality + Efficiency + Stability

Is the protocol a Nash equilibrium? Is the protocol incentive compatible? (i.e. are the agents willing to use it?, will some agents fork off?) Suppose it is an optimal efficient equilibrium. Are we done?

Decentralized problem:

$$f, S$$



$$\max_{x \in S} f(x)$$

Still not enough! **Real distributed systems pose more challenges** (e.g. asynchrony, communication delays, users might have different utility functions due to differences in electricity/computation costs/risk attitudes, …)

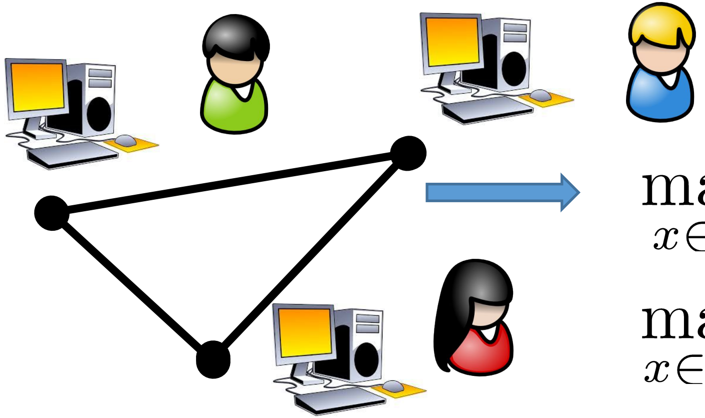
Robustness

How robust is the implemented equilibrium to perturbations of the game? How smoothly do the system properties degrade as we move away from the ideal specification?

Decentralized problem:

$f, S \quad \Rightarrow$

f', S'



$\Rightarrow \quad \max_{x \in S} f(x)$

$\max_{x \in S'} f'(x)$

No system is *truly* robust. If a system is pushed far enough from its initial specifications eventually the system will break. How far we push it?

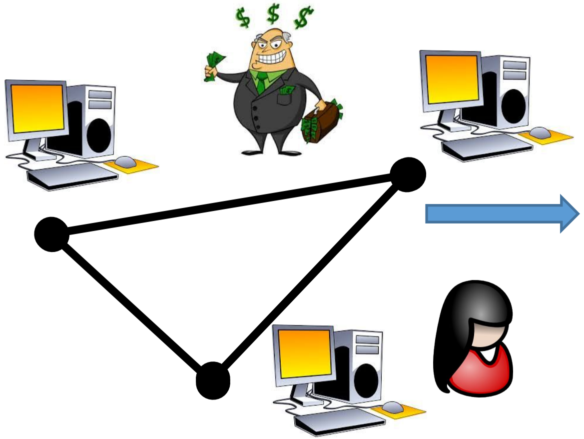
A particular question of interest is what happens in the case of **coalition formation/collusion**.

Robustness

How robust is the implemented equilibrium to perturbations of the game? How smoothly do the system properties degrade as we move away from the ideal specification?

Decentralized problem:

f, S



$$\max_{x \in S} f(x)$$

f', S'

$$\max_{x \in S'} f'(x)$$

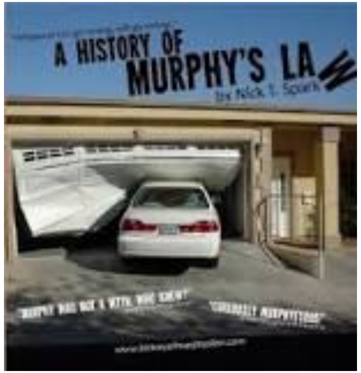
[Eyal, Sirer’14] If x > 1/3 fraction of mining power is owned by a mining pool, then the following the **Bitcoin protocol is NOT an equilibrium**.

[Kiayias, Koutsoupias, Kyropoulou, Tselekounis’16] If x < ≈30% of mining power is owned by a mining pool, then the **Bitcoin protocol is an equilibrium**.

Murphy’s Law

Systems with good robustness properties are still not enough!

“Anything that can go wrong will go wrong”.



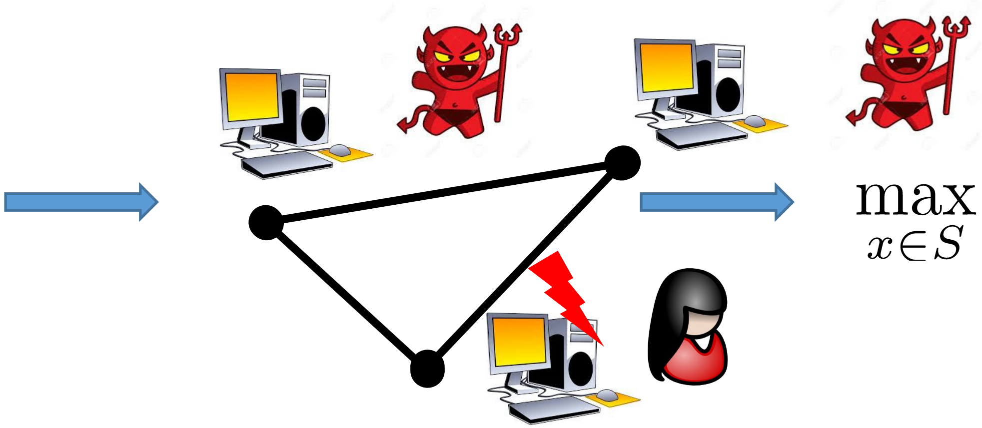
Every system eventually breaks down.

How do we design systems with this eventuality in mind?

Persistence

Example: If the system is forced out of equilibrium (e.g. via a coordinated attack that "burns" $100M) can the system **recover**? How fast? At what cost?

f, S



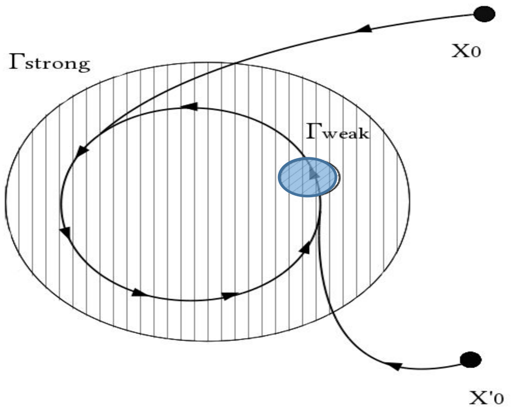
$$\max_{x \in S} f(x)$$

Thought experiment: Take this idea to its logical extreme.

Assume that the **system** may be always under attack and design it so that it **recovers** and **provides the desirable properties often**.

Persistence

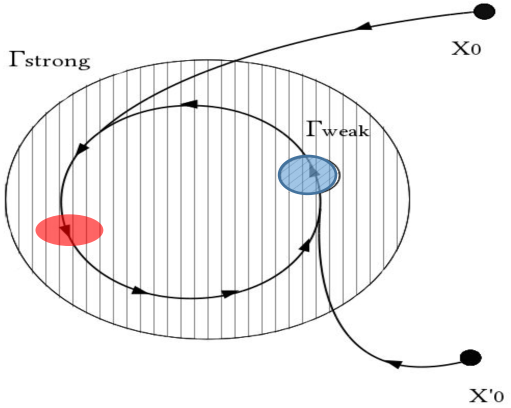
A **strongly persistent** property will eventually be *satisfied (and stay satisfied)* given any initial system condition.



A **weakly persistent** property will eventually be satisfied given any initial system condition and will become satisfied again infinitely often.
[Piliouras, Nieto-Granda, Christensen, Shamma 2014]

Persistence

A desirable property is not satisfied by a system just in equilibrium, but it is satisfied in a dynamic way.



More **flexibility** to explore **tradeoffs** between recovery/convergence time, "periodicity" & cost of implementation.

Two (or more) incompatible properties can both be supported in a weakly persistent manner.

The PRESTO framework recap

Optimality: Does the protocol maximize the quality of outcomes?

STability: Is the designed protocol an equilibrium?

Efficiency: How efficiently does the protocol utilize its difference resources (e.g. time, space, energy, e.t.c.)

Robustness: Do the protocol performance guarantees withstand perturbations to the system parameters?

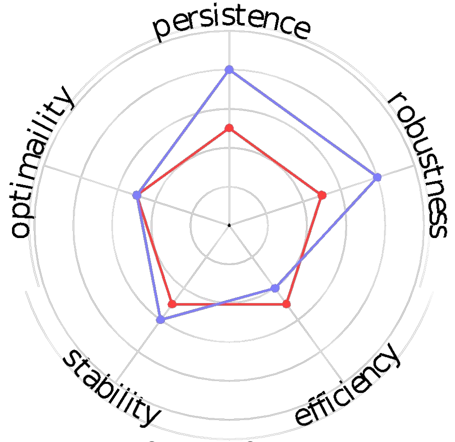
Persistence: If the protocol is forced out of equilibrium, does it recover?

Applications of the PRESTO framework

Language/Communication is key.

- Creating a **formal and intuitive specification** framework is useful for coordination between the different stakeholders (developers, researchers, community)
- All layers of PRESTO are actively being studied (some really mature with many of years of work, some work in progress). Understanding these connections allows us to **port very powerful ideas** into future iterations of blockchain protocols.
- A perfect PRESTO protocol is impossible, but understanding the fundamental limits/limitations will **show us the way forward**.

The PRESTO framework applied on Casper/Ethereum



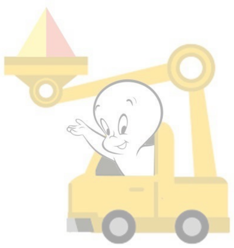
- **Casper** (PoS w. checkpoints) vs **PoW**
- Why is Casper a **Nash Equilibrium**?
- **Griefing factor analysis** (loss to the other players relative to loss to the player in absolute terms)
- **Minority fork** (recovery from 51% attack)

Incentives in Casper the Friendly Finality Gadget.
with Vitalik Buterin and Daniel Reijsbergen.

Ethresear.ch link https://bit.ly/2Of1PWN

Rethinking Blockchain Security: Position Paper. **IEEE Blockchain, 2018.** with Vincent Chia, Pieter H. Hartel, Qingze Hum, Sebastian Ma, Daniel Reijsbergen, Mark van Staalduinen and Pawel Szalachowski.

Design Philosophy



Design Philosophy

- Come up with a distributed (e.g. consensus) protocol that we want to incentivize and make public

- Create an equilibrium to follow the protocol
  - Detect and penalize deviations from protocol expectations
  - Reward nodes for meeting protocol expectations

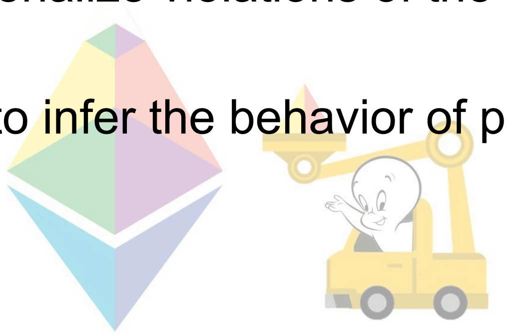
- Allow node operators to place security deposits to play a role in the system

- Maximize the cost of bribing attack by parameterizing the penalties/rewards

About the Incentive Mechanism

- The incentive mechanism observes participants’ behaviors to reward expected behavior, and penalize violations of the rules of the system.

- It must therefore be able to infer the behavior of players from its information



- But there is limited information available in distributed systems about the behaviour of programs in a distributed system

Design Philosophy

- Consider the protocol-determined map between player strategies and protocol states

- Come up with a distributed (e.g. consensus) protocol that have protocol states that reveal as much information about the behavior of the participants as possible

- And then detect and penalize deviations from the protocol

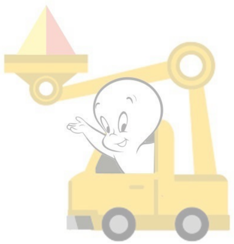
Design Philosophy

- Assume that we are in an oligopolistic market setting
  - Some players with a lot of weight have low coordination costs and can collude
  - A large number of players have a small amount of weight and high coordination costs
- Try to guarantee that it’s a coalitional dominant strategy to follow the protocol
- Make sure that every player marginally contributes to the utility of the protocol
- Make sure that there is no optimal coalition (that doesn’t include everyone)
- If we fail, try again under slightly less conservative assumptions

Roadmap

Roadmap - Series of Games!

- AND-Gate game (one-shot)
    - Without deposits and without faults
    - With deposits and without faults
    - With deposits and faults
- Iterated AND-Gate game
- Censorship game
    - Without deposits
    - With deposits
- …
- Casper CBC Protocol States game!



Examples

AND-Gate Game - Background

- Alice and Bob must both sign on a message.
- Alice is expected to sign the message, and pass the signed message to Bob.
- Bob is expected to receive the message and sign it.
- Both players may behave in an unexpected manner:
  - Alice may choose not to sign or not to send the message.
  - Bob may choose not to receive or not to sign the message.

AND-Gate Game

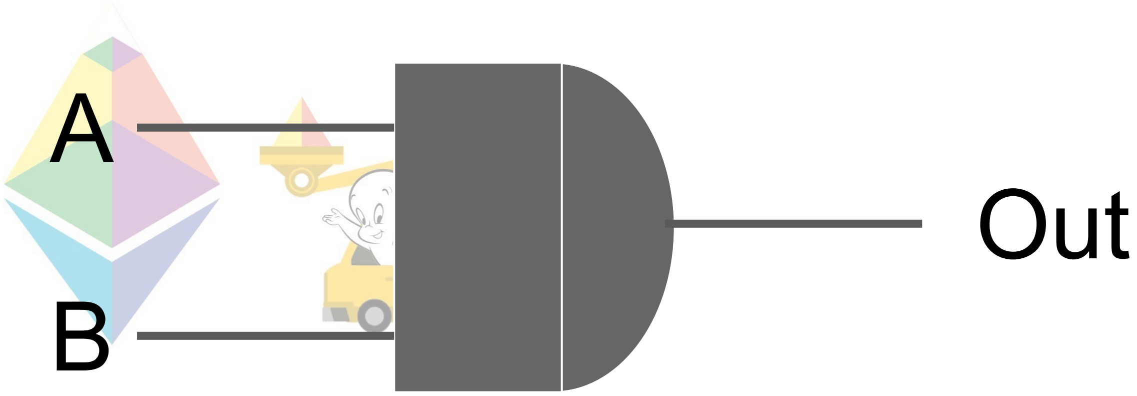
AND-Gate Game 1

- The mechanism pays both A and B an amount of R if a message signed by both A and B is seen.
- A can play either 1 (send signed message to B) or 0 (any other unexpected behavior)
- B can play either 1 (sign on message received from A) or 0 (any other unexpected behavior)
- The mechanism can only check whether there was a message signed by both A and B.

AND-Gate Game 1

A & B play values from {0, 1}

The mechanism can only see the output of the AND gate, and not the individual inputs.



AND-Gate Game 1

P	$B, 0$	$B, 1$
$A, 0$	$0,\ 0$	$0,\ 0$
$A, 1$	$0,\ 0$	$R,\ R$

AND-Gate Game 1 - Analysis

- What is the Nash Equilibrium of this game?
- How to attack this game?

P	$B,0$	$B,1$
$A,0$	$0,\ 0$	$0,\ 0$
$A,1$	$0,\ 0$	$R,\ R$

Attacker’s Model

- The attacker’s objective is to cause the the AND gate to output 0.
- The attacker can bribe any of the players, and can bribe any (non-negative) amount.
- The attacker can see the individual inputs of the AND gate.
- The bribery offer can be represented as a “bribing game matrix”

Attacker’s Model

- Budget of Attack:
  - Minimum capital required to attack the game
- Cost of Attack:
  - Cost (to the attacker) of a realized attack

AND-Gate Game 1 - Attack

L	$B, 0$	$B, 1$
$A, 0$	$R + \delta,\ 0$	$R + \delta,\ 0$
$A, 1$	$0,\ 0$	$0,\ 0$

AND-Gate Game 1 - Bribed Game

$P+L$	$B,0$	$B,1$
$A,0$	$R+\delta,\ 0$	$R+\delta,\ 0$
$A,1$	$0,\ 0$	$R,\ R$

Budget and cost of attack are both $R+\delta$

AND-Gate Game 2

- A and B each have to make a non-negative deposit while playing their respective values.
- There is a background rate of interest *r* on the deposits that the players make
- The mechanism pays out a fixed reward split in the ratio of players' deposits when the AND gate outputs 1.
- When the AND gate outputs 0, the mechanism penalizes both players a fraction *p* of their respective deposits.

AND-Gate Game 2

P	$D_B, 0$	$D_B, 1$
$D_A, 0$	$-p \cdot D_A, \ -p \cdot D_B$	$-p \cdot D_A, \ -p \cdot D_B$
$D_A, 1$	$-p \cdot D_A, \ -p \cdot D_B$	$\frac{R \cdot D_A}{D_A + D_B}, \ \frac{R \cdot D_B}{D_A + D_B}$

AND-Gate Game 2 - Analysis

- Find the equilibrium values for D_A and D_B

$$E(U_A|s = (D_A, 1), (D_B, 1)) = R \cdot D_A/(D_A + D_B) - r \cdot D_A$$

  - Find $\frac{dE(U_A)}{dD_A}$

  - Find D_A that maximizes E(U_A)

AND-Gate Game 2 - Analysis

- Find the equilibrium values for D_A and D_B

$$E(U_A|s = (D_A, 1), (D_B, 1)) = R \cdot D_A/(D_A + D_B) - r \cdot D_A$$

  - Find $\frac{dE(U_A)}{dD_A}$

  - Find D_A that maximizes E(U_A)

$$D_A = D_B = \frac{R}{4 \cdot r}$$

AND-Gate Game 2 - Attack

1. **Bonding**: Alice and Bob put down deposits D_A and D_B respectively
2. **Bribing**: The Attacker credibly commits to a bribing matrix L
3. **Players Choose Actions**: Alice and Bob each choose an action from $\{0, 1\}$

AND-Gate Game 2 - Attack

The cost of attack *C* (with objective to have A play 0) is:

$$C = E(U_A|s = (D_A, 1), (D_B, 1)) - E(U_A|s = (D_A, 0), (D_B, 1))$$

, where:

$$E(U_A|s = (D_A, 0), (D_B, 1)) = -(p + r) \cdot D_A$$

$$E(U_A|s = (D_A, 1), (D_B, 1)) = R \cdot D_A/(D_A + D_B) - r \cdot D_A$$

AND-Gate Game 2 - Attack

$$C = R \cdot D_A/(D_A + D_B) - r \cdot D_A + (p + r) \cdot D_A$$

$$C = \frac{R}{2} + \frac{p \cdot R}{4 \cdot r}$$

AND-Gate Game 2 - Attack

$$C = \frac{R}{2} + \frac{p \cdot R}{4 \cdot r}$$

- The derivative of *C* w.r.t. *p* is:

$$\frac{dC}{dp} = \frac{R}{4 \cdot r}$$

- Therefore, cost of attack is maximum when *p* = 1

AND-Gate Game 3

- We now assume that the AND gate might fail (i.e., output 0) irrespective of it’s inputs, with probability q.
- In the message passing analogy, this corresponds to node failure or network failure.

AND-Gate Game 3

With probability $1 - q$ we have this matrix:

$P_{correct}$	$D_B, 0$	$D_B, 1$
$D_A, 0$	$-p \cdot D_A, \ -p \cdot D_B$	$-p \cdot D_A, \ -p \cdot D_B$
$D_A, 1$	$-p \cdot D_A, \ -p \cdot D_B$	$\frac{R \cdot D_A}{D_A + D_B}, \ \frac{R \cdot D_B}{D_A + D_B}$

With probability q we have this matrix:

P_{faulty}	$D_B, 0$	$D_B, 1$
$D_A, 0$	$-p \cdot D_A, \ -p \cdot D_B$	$-p \cdot D_A, \ -p \cdot D_B$
$D_A, 1$	$-p \cdot D_A, \ -p \cdot D_B$	$-p \cdot D_A, \ -p \cdot D_B$

AND-Gate Game 3

$$P = \begin{cases} P_{faulty} & \text{with probability } q \\ P_{correct} & \text{with probability } 1 - q \end{cases}$$

AND-Gate Game 3 - Analysis

- Find the equilibrium values for D_A and D_B

$$E(U_A) = (1-q) \cdot R \cdot D_A/(D_A + D_B) - q \cdot p \cdot D_A - r \cdot D_A$$

  - Find $\frac{dE(U_A)}{dD_A}$

  - Find D_A that maximizes E(U_A)

AND-Gate Game 3 - Analysis

- Find the equilibrium values for D_A and D_B

$$E(U_A) = (1-q)\cdot R \cdot D_A/(D_A + D_B) - q \cdot p \cdot D_A - r \cdot D_A$$

  - Find $\dfrac{dE(U_A)}{dD_A}$
  - Find D_A that maximizes E(U_A)

$$D_A = D_B = \frac{(1-q)\cdot R}{4\cdot(p\cdot q + r)}$$

AND-Gate Game 3 - Attack

The cost of attack *C* (with objective to have A play 0) is:

$$C = E(U_A | s = (D_A, 1), (D_B, 1)) - E(U_A | s = (D_A, 0), (D_B, 1))$$

, where:

$$E(U_A | s = (D_A, 0), (D_B, 1)) = -(p + r) \cdot D_A$$

$$E(U_A | s = (D_A, 1), (D_B, 1)) = (1 - q) \cdot R \cdot D_A / (D_A + D_B) - (q \cdot p + r) \cdot D_A$$

AND-Gate Game 3 - Attack

- The cost of attack is: $$C = \frac{(1-q) \cdot R}{2} + \frac{p(1-q)^2 \cdot R}{4(p \cdot q + r)}$$

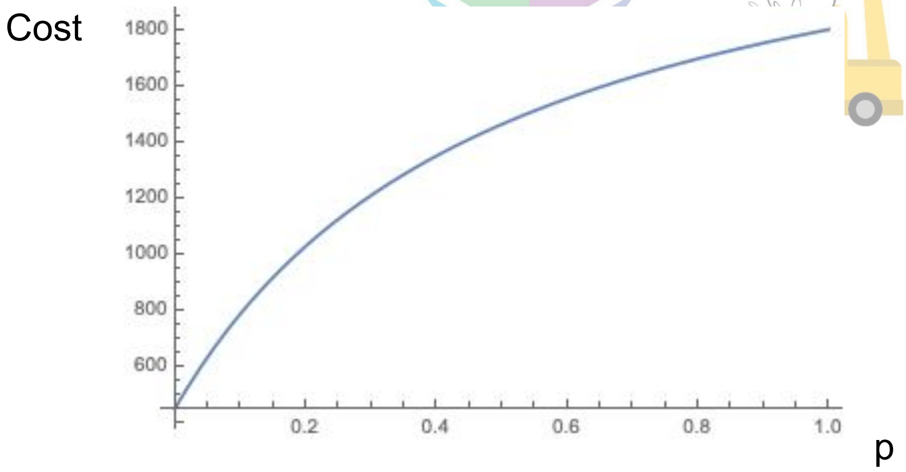
- The derivative of *C* w.r.t. *p* is: $$\frac{dC}{dp} = \frac{r \cdot R \cdot (1-q)^2}{4(p \cdot q + r)^2} \geq 0$$

- Therefore, the cost of attack is maximum when *p* = 1

AND-Gate Game 3 - Attack

- The cost of attack is: $$C = \frac{(1-q)\cdot R}{2} + \frac{p(1-q)^2 \cdot R}{4(p \cdot q + r)}$$

- We plot for q = 0.1, r = 0.05, R = 1000



AND-Gate Game 4 - Iterated Version

The iterated AND game proceeds as follows:

1. Alice and Bob place a deposit of D_A and D_B respectively.
2. They play *n* rounds of AND-Gate Game 3 with:
    a. Reward *(R/n)*
    b. Penalty *p*
    c. Chance of failure *q*

AND-Gate Game 4 - Analysis

- Let *X ~ Binomial(q, n)* denote the number of times the AND gate failed (due to the chance of failure)
- **Payoff from rewards** $\mathbf{E}[P_{R,A}]$

$$E[P_{R,A}] = E[(n - X) \cdot \frac{R \cdot D_A}{n \cdot (D_A + D_B)}]$$

$$E[P_{R,A}] = (n - n \cdot q) \cdot \frac{R \cdot D_A}{n \cdot (D_A + D_B)}$$

$$E[P_{R,A}] = (1 - q) \cdot \frac{R \cdot D_A}{D_A + D_B}$$

AND-Gate Game 4 - Analysis

- **Payoff from expected failures E[F]**

  Each time the AND gate fails, the players' deposits become $(1-p)$ times their current value. If the original deposit is D_A, then after first failure, the deposit becomes $(1-p) \cdot D_A$. After second failure, the deposit becomes $(1-p)^2 \cdot D_A$.

$$E[F] = E[-\{1-(1-p)^X\} \cdot D_A]$$

$$E[F] = E[-D_A + (1-p)^X \cdot D_A]$$

$$E[F] = -D_A + D_A \cdot E[(1-p)^X]$$

AND-Gate Game 4 - Analysis

- Calculating $E[(1-p)^X]$, using MGF:

$$E[e^{t \cdot X}] = (1 - q + q \cdot e^t)^n$$
$$\text{Subsititing } t = log(1-p)$$
$$E[e^{log(1-p) \cdot X}] = (1 - q + q \cdot e^{log(1-p)})^n$$
$$E[(1-p)^X] = (1 - q + q \cdot (1-p))^n$$
$$E[(1-p)^X] = (1 - q + q - q \cdot p)^n$$
$$E[(1-p)^X] = (1 - q \cdot p)^n$$

AND-Gate Game 4 - Analysis

- Resuming our calculation for $E[F]$:

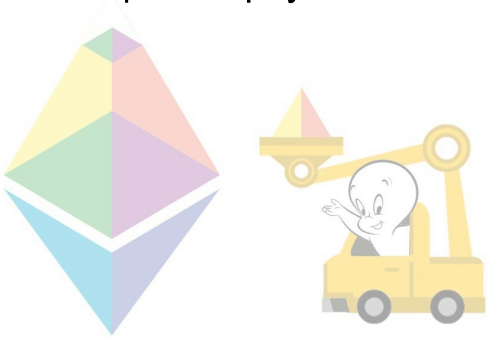
$$E[F] = -D_A + D_A \cdot E[(1-p)^X]$$
$$E[F] = -D_A + D_A \cdot (1 - q \cdot p)^n$$

- **Total expected payoff** $E[P_A]$

$$E[P_A] = E[P_{R,A}] + E[F] - C_{capital}$$
$$E[P_A] = (1-q) \cdot \frac{R \cdot D_A}{D_A + D_B} - D_A + D_A \cdot (1 - q \cdot p)^n - r \cdot D_A$$

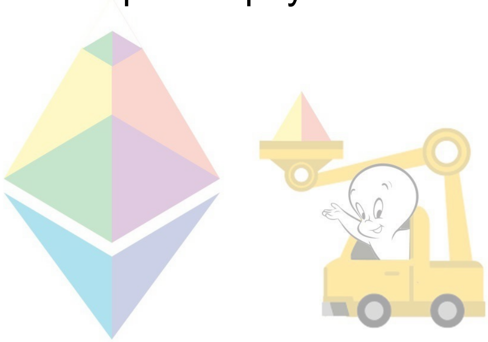
AND-Gate Game 4 - Analysis

- Find the equilibrium values for D_A and D_B
  - Set $\frac{dE[P_A]}{dD_A} = 0$ to maximize expected payoff



AND-Gate Game 4 - Analysis

- Find the equilibrium values for D_A and D_B
  - Set $\frac{dE[P_A]}{dD_A} = 0$ to maximize expected payoff



$$D_A = D_B = (1-q) \cdot \frac{R}{4 \cdot [(1+r) - (1-q \cdot p)^n]}$$

AND-Gate Game 4 - Attack

- The attacker wants to make all *n* rounds fail, and bribes one of the players
- The player's payoff from the mechanism in the case of attack (all rounds result in penalties) is:

$$A_P = -\{1 - (1-p)^n\} \cdot D$$

$$A_P = -D + D \cdot (1-p)^n$$

- The cost of attack $C = E[P] - (A_P - C_{capital})$

$$C = (1-q) \cdot \frac{R}{2} + (1-q) \cdot \frac{R}{4 \cdot [(1+r) - (1 - q \cdot p)^n]} \cdot [(1 - q \cdot p)^n - (1-p)^n]$$

AND-Gate Game 4 - Attack

- To maximize cost of attack,

$$\frac{dC}{dp} = \frac{d\left\{ \frac{(1-q)\cdot R\cdot[(1-q\cdot p)^n - (1-p)^n]}{4\cdot[(1+r)-(1-q\cdot p)^n]} \right\}}{dp} = 0$$

Imperfect Attribution Games

Strategy Profile : S

S represents all the strategy profiles that are playable by the set of validators. An element in S contains the a strategy choice for each of the validators.

Information accessible to incentive mechanism : I

The incentive mechanism may not be able to see the strategy choice of validators. I is the set of *attributable* information that the incentive mechanism can view given the strategy profile of all validators.

Imperfect Attribution Games

Information accessible to incentive mechanism : I

The incentive mechanism may not be able to see the strategy choice of validators. I is the set of *attributable* information that the incentive mechanism can view given the strategy profile of all validators.

Attributes : $F : S \to I$

F represents the accessible information that the incentive mechanism can view given a strategy profile in S. Note that in the general case, F may be non-invertible, giving rise to imperfect attribution.

Imperfect Attribution Games

Incentive Mechanism Payout : $M : I \to \mathbb{R}^n$

M represents the payout to each validator given the information accessible to the incentive mechanism. Note that this is not the final payoff of the game to validators.

Censorship Game

- We now consider the case of censorship in blockchain consensus protocols
- Basic concept - If majority censors the minority, then the minority is invisible to the mechanism

Censorship Game

- $V = \{v_1, v_2, ..., v_n\}$
- $W(v_i) = w_i \ , \forall 1 \leq i \leq n$
- $S = \{online, censoring, offline\}^n$
- $I = \{online, offline\}^n$

Censorship Game

- $num : S \times \{online, censoring, offline\} \rightarrow \mathbb{N}$
  $num((s_1, s_2, ..., s_n), strategy) = \sum_{i \in [1,n], s_v = strategy} W(v_i)$

- $F(s) = (F_1(s), F_2(s), ..., F_n(S))$
  $$F_i(s) = \begin{cases} online & \text{if } s_i = censoring \\ online & \text{if } s_i = online \wedge num(s, censoring) < num(s, online) \\ online & \text{if } s_i = online \wedge num(s, censoring) \geq num(s, online) \\ offline & \text{if } s_i = offline \end{cases}$$

Censorship Game 1

$$M(F(s)) = \{M_1(F(s)), M_2(F(s)), ..., M_n(F(s))\}$$

$$M_i(F(s)) = \begin{cases} R & \text{if } F_i(s) = online \\ 0 & \text{if } F_i(s) = offline \end{cases}$$

- What is the Nash Equilibrium in this case?

Censorship Game 2

- Collective Penalties

$$M(F(s)) = \{M_1(F(s)), M_2(F(s)), ..., M_n(F(s))\}$$

$$M_i(F(s)) = \begin{cases} R \cdot \frac{num(F(s), online)}{n} & \text{if } F_i(s) = online \\ 0 & \text{if } F_i(s) = offline \end{cases}$$

- What is the equilibrium in this case?

Thank You!