Resource Burning for Permissionless Systems

Jared Saia

Joint with Diksha Gupta and Maxwell Young

Resource Burning for Permissionless Systems

Jared Saia

Joint with Diksha Gupta and Maxwell Young

Permissionless System:

Participants are virtual IDs

Join and depart without scrutiny

Permissionless System:

Participants are virtual IDs

Join and depart without scrutiny

Resource Burning:

Verifiable consumption of a resource

↑ Permissionless Systems

Blockchains

Peer-to-peer

↑ Resource Burning

Proof of work

CAPTCHAs

Positions

Positions

1. Resource burning is fundamental

Positions

1. Resource burning is fundamental

2. Resource burning must be optimized

Positions

1. Resource burning is fundamental
2. Resource burning must be optimized
3. Resource burned shouldn't matter

Positions

1. Resource burning is fundamental
2. Resource burning must be optimized
3. Resource burned shouldn’t matter
4. Need Permissionless → Permissioned reduction

Positions

1. Resource burning is fundamental

2. Resource burning must be optimized

3. Resource burned shouldn’t matter

4. Need Permissionless → Permissioned reduction

5. Need domain specific **and** general results

Positions

1. **Resource burning is fundamental**
2. Resource burning must be optimized
3. Resource burned shouldn't matter
4. Need Permissionless → Permissioned reduction
5. Need domain specific **and** general results

Resource burning is fundamental

Cybersecurity:

[Dwork and Naor ’92] combat spam

Blockchains, DDoS attacks, review spam, DHTs

Resource burning is fundamental

Cybersecurity:

[Dwork and Naor '92] combat spam

Blockchains, DDoS attacks, review spam, DHTs

Biology

Economics/Game theory

Biology: Costly Signaling

Sexual selection: peacock tail, antlers

Predator/Prey signaling: stotting

Biology: Costly Signaling

Sexual selection: peacock tail, antlers

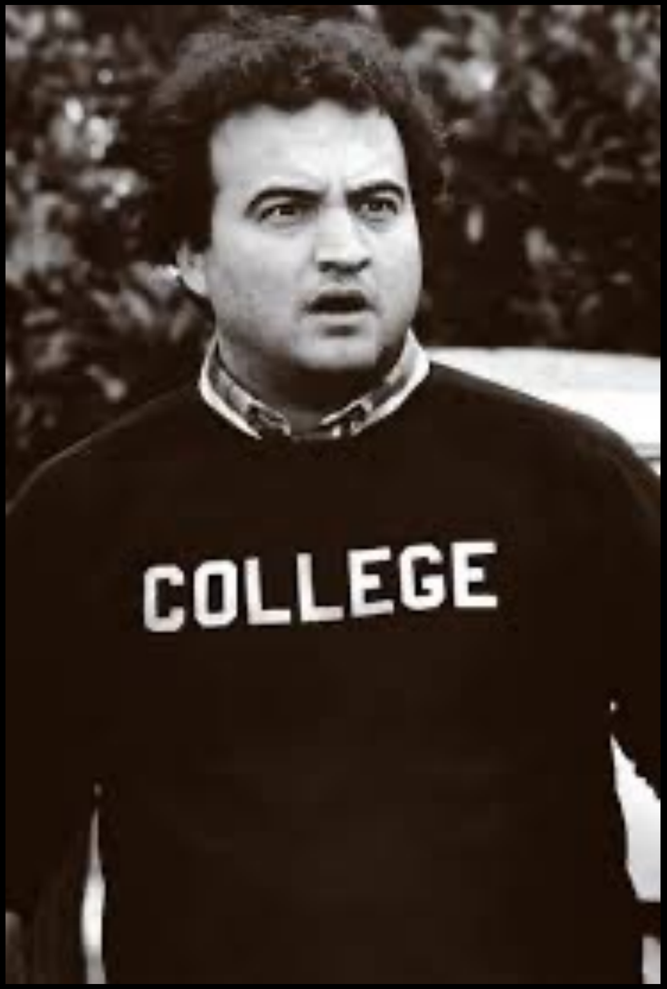
Predator/Prey signaling: stotting

Game Theory: Money Burning

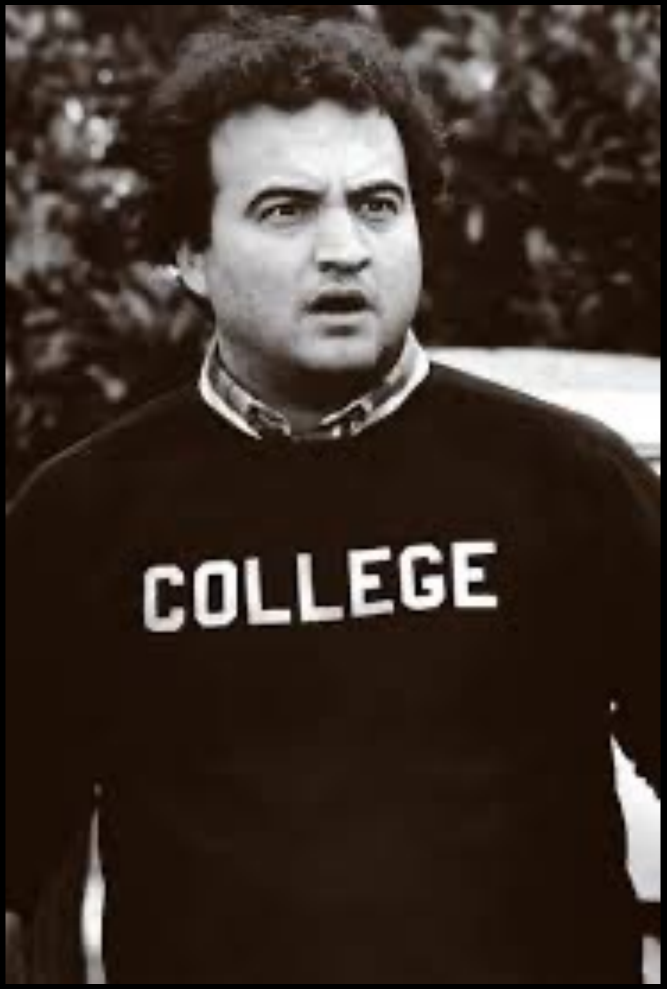
Purpose is to signal:

- Type of a player
- Commitment to an action

Signaling Type: College Game



Signaling Type: College Game



“Great! Seven years of college down the toilet.”

College?

	Smart	Daft
Attend	-1	-3

College?

	Smart	Daft
Attend	-1	-3

Hire?

	Smart	Daft
Hire	2	-2

Student: Payoff of 2 if hired; else 0

College?

	Smart	Daft
Attend	-1	-3

Hire?

	Smart	Daft
Hire	2	-2

Student: Payoff of 2 if hired; else 0

Nash equilibrium: (1) Only smart students attend college; (2) Employer hires only college attendees.

Positions

1. Resource burning is fundamental
2. Resource burning must be optimized
3. Resource burned shouldn’t matter
4. Need Permissionless → Permissioned reduction
5. Need domain specific **and** general results

Positions

1. Resource burning is fundamental
2. Resource burning must be optimized
3. Resource burned shouldn’t matter
4. Need Permissionless → Permissioned reduction
5. Need domain specific **and** general results

Resource burning must be optimized

Resource burning must be optimized

Bitcoin uses 58 TWh/year; ≈ Bangladesh

Resource burning must be optimized

Bitcoin uses 58 TWh/year; ≈ Bangladesh

Humans spend 150,000 hours/day solving CAPTCHAs

Resource burning must be optimized

Bitcoin uses 58 TWh/year; ≈ Bangladesh

Humans spend 150,000 hours/day solving CAPTCHAs

Theoretical results suggest significant improvements possible

Can optimize RB like any other resource

T = Adversary's resource burning (RB) rate

$f(T)$ = Algorithm's resource burning rate

↓ RB = ↑ Security

Reduced Resource Burning cost can improve security

Can analyze using game theory

Zero-sum game between adversary and algorithm

Zero-sum Game

T = cost to attack

$f(T)$ = cost to defend

D = Cost of defeat

	Attack	¬Attack
Defend	$T - f(T)$	$-f(0)$
¬Defend	$-D$	0

T = cost to attack; $f(T)$ = cost to defend;

D = cost of defeat; p = probability to defend

To solve, set

$$p(T - f(T)) - (1 - p)D = p(-f(0))$$

	Attack	¬Attack
Defend	$T - f(T)$	$-f(0)$
¬Defend	$-D$	0

T = cost to attack; $f(T)$ = cost to defend;

D = cost of defeat; p = probability to defend

To solve, set
$p(T - f(T)) - (1 - p)D = p(-f(0))$

$$p = \frac{D}{T - f(T) + f(0) + D}$$

	Attack	¬Attack
Defend	$T - f(T)$	$-f(0)$
¬Defend	$-D$	0

T = cost to attack; $f(T)$ = cost to defend;

D = cost of defeat; p = probability to defend

To solve, set

$$p(T - f(T)) - (1 - p)D = p(-f(0))$$

$$p = \frac{D}{T - f(T) + f(0) + D}$$

Payoff:

$$\frac{-f(0)D}{T - f(T) + f(0) + D}$$

	Attack	¬Attack
Defend	$T - f(T)$	$-f(0)$
¬Defend	$-D$	0

Domain	Primary Resource Consumed	Mechanism	Enabled Functionality	Conjectured Cost
Blockchains	CPU	CPU Puzzles	Distributed Ledger	$O(\sqrt{TJ_G} + J_G)$
DHTs	CPU	CPU Puzzles	Decentralized storage and search	$\tilde{O}(\sqrt{TJ_G} + J_G)$
DDoS Attacks	Bandwidth / CPU	Messages / CPU Puzzles	Fair allocation of server resources	No Conjecture
Review Spam	Human Time	CAPTCHAS	Trusted consumer recommendations	$\tilde{O}(T^{2/3} + P_G)$

T = attacker's RB rate

J_G = good ID join rate

P_G = good ID posting rate

Domain	Primary Resource Consumed	Mechanism	Enabled Functionality	Conjectured Cost
Blockchains	CPU	CPU Puzzles	Distributed Ledger	$O(\sqrt{TJ_G} + J_G)$
DHTs	CPU	CPU Puzzles	Decentralized storage and search	$\tilde{O}(\sqrt{TJ_G} + J_G)$
DDoS Attacks	Bandwidth / CPU	Messages / CPU Puzzles	Fair allocation of server resources	No Conjecture
Review Spam	Human Time	CAPTCHAS	Trusted consumer recommendations	$\tilde{O}(T^{2/3} + P_G)$

T = attacker's RB rate
J_G = good ID join rate
P_G = good ID posting rate

T = cost to attack; $f(T)$ = cost to defend;

D = cost of defeat; p = probability to defend

Payoff:

$$\frac{-Df(0)}{T + f(0) - f(T) + D}$$

T = cost to attack; $f(T)$ = cost to defend;

D = cost of defeat; p = probability to defend

Payoff:

$$\frac{-Df(0)}{T + f(0) - f(T) + D}$$

Algorithm Cost | Game Payoff

$$f(T) = f(0) + o(T) \quad \rightarrow \quad O(-f(0))$$

$$f(T) = f(0) + \sqrt{Tf(0)} \quad \rightarrow \quad O\left(\frac{-f(0)D}{f(0) + D}\right)$$

Positions

1. Resource burning is fundamental
2. Resource burning must be optimized
3. Resource burned shouldn't matter
4. Need Permissionless → Permissioned reduction
5. Need domain specific **and** general results

Positions

1. Resource burning is fundamental
2. Resource burning must be optimized
3. Resource burned shouldn't matter
4. Need Permissionless → Permissioned reduction
5. Need domain specific **and** general results

Resource Burned Shouldn’t Matter

Resource Burned Shouldn't Matter

Resource burning must be

- Verifiable
- *Non-amortizable*
  - Solving x challenges of difficulty d requires $\approx xd$ resource consumption

RB Common Examples

Proof of work via SHA hashing

Proof of space & space-time

CAPTCHAs

Radio resource-testing (wireless networks)

RB can also do useful work

[Ball et al. ’18]: “Proof of Useful Work”

[Von Anh et al. ’08]: RECAPTCHA

RB can also do useful work

[Ball et al. ’18]: “Proof of Useful Work”

[Von Anh et al. ’08]: RECAPTCHA

For Blockchains:

PoX: Matrix Multiplication

PrimeCoin: Finding primes

Permacoin: Maintaining blockchain

Piecework: Spam deterrence

Not RB: Proof of Stake

Used in: Algorand, Ouroboros, Ethereum

Proof of stake is a measurement

ID's stake must be known

Not RB: Proof of Stake

Used in: Algorand, Ouroboros, Ethereum

Proof of stake is a measurement

ID’s stake must be known

I think proof of stake is fundamentally vulnerable… In my opinion, it’s giving power to people who have lots of money - Dahlia Malkhi

Positions

1. Resource burning is fundamental
2. Resource burning must be optimized
3. Resource burned shouldn't matter
4. Need Permissionless → Permissioned reduction
5. Need domain specific **and** general results

Positions

1. Resource burning is fundamental
2. Resource burning must be optimized
3. Resource burned shouldn’t matter
4. Need Permissionless → Permissioned reduction
5. Need domain specific **and** general results

Permissionless → Permissioned

Permissionless → Permissioned

Five decades of research on designing secure permissioned systems

Permissionless → Permissioned

Five decades of research on designing secure permissioned systems

Permissioned = bounded bad fraction

Permissionless → Permissioned

Five decades of research on designing secure permissioned systems

Permissioned = bounded bad fraction

Can leverage permissioned results if we bound fraction of bad IDs in permissionless

Bounding fraction of bad IDs

GenID Problem

n good, synchronized IDs; n unknown

Byzantine adversary has κ fraction of the RB resource for "sufficiently small" κ

Goal: All IDs have same set S that contains

- All good IDs
- At most $O(\kappa)$ fraction of bad IDs

GenID Problem

n good, synchronized IDs; n unknown

Byzantine adversary has κ fraction of the RB resource for "sufficiently small" κ

Goal: All IDs have same set S that contains

- All good IDs
- At most $O(\kappa)$ fraction of bad IDs

Adversary sees all messages, can inject any message into network, etc.

GenID Results

GenID Results

n good IDs; Adversary controls κ fraction of RB

GenID Results

n good IDs; Adversary controls κ fraction of RB

[Aspnes et al. '05] Defined problem; but inconsistent views of bad

GenID Results

n good IDs; Adversary controls κ fraction of RB

[Aspnes et al. '05] Defined problem; but inconsistent views of bad

[Andrychowicz et al. '15] Requires $\Theta(n)$ rounds

GenID Results

n good IDs; Adversary controls κ fraction of RB

[Aspnes et al. '05] Defined problem; but inconsistent views of bad

[Andrychowicz et al. '15] Requires $\Theta(n)$ rounds

[Hou et al. '18] Requires $\Theta\left(\frac{\ln n}{\ln \ln n}\right)$ rounds

GenID Results

n good IDs; Adversary controls κ fraction of RB

[Aspnes et al. ’05] Defined problem; but inconsistent views of bad

[Andrychowicz et al. ’15] Requires $\Theta(n)$ rounds

[Hou et al. ’18] Requires $\Theta\left(\frac{\ln n}{\ln \ln n}\right)$ rounds

All rely on SHA-style PoW
Open problem: Adapt these for arbitrary RB

What about Churn?

DefID

Goal: IDs **always** have same set S that contains

All good IDs

At most $O(\kappa)$ fraction of bad IDs

Our Result

DefID [Gupta et al. '20]

Theorem: Let **T** be adversarial spend rate and J_G be good join rate. Then can solve DefID with

$O(J_G + \sqrt{J_G T})$ algorithm spend rate

DefID [Gupta et al. '20]

Theorem: Let **T** be adversarial spend rate and $\mathbf{J_G}$ be good join rate. Then can solve DefID with

$$O(J_G + \sqrt{J_G T}) \text{ algorithm spend rate}$$

These results assume α, β churn for $\alpha, \beta = \Theta(1)$;
Still allows for exponential change in system size.

Assumptions

There is α, β churn for, $\alpha, \beta = \Theta(1)$

Adversary can't target specific good IDs

System size is always "sufficiently large"

Epoch

Define *epoch* to be time till set of good IDs (G_t) changes by constant fraction, e.g.
$|G_t - G_{t'}| \geq 3/4 |G_t|$

For some t and $t' > t$

α, β Churn

α, β Churn

ρ_j is good ID join rate in epoch j

Good join rate changes by at most α between epochs:

$$\frac{\rho_{j-1}}{\alpha} \leq \rho_j \leq \alpha\rho_{j-1}$$

α, β Churn

ρ_j is good ID join rate in epoch j

Good join rate changes by at most α between epochs:

$$\frac{\rho_{j-1}}{\alpha} \leq \rho_j \leq \alpha \rho_{j-1}$$

Let n_ℓ be # good IDs joining in ℓ seconds in epoch j. Then n_ℓ differs by at most β from expected value:

$$\left\lfloor \frac{\ell \rho_j}{\beta} \right\rfloor \leq n_\ell \leq \lceil \beta \ell \rho_j \rceil$$

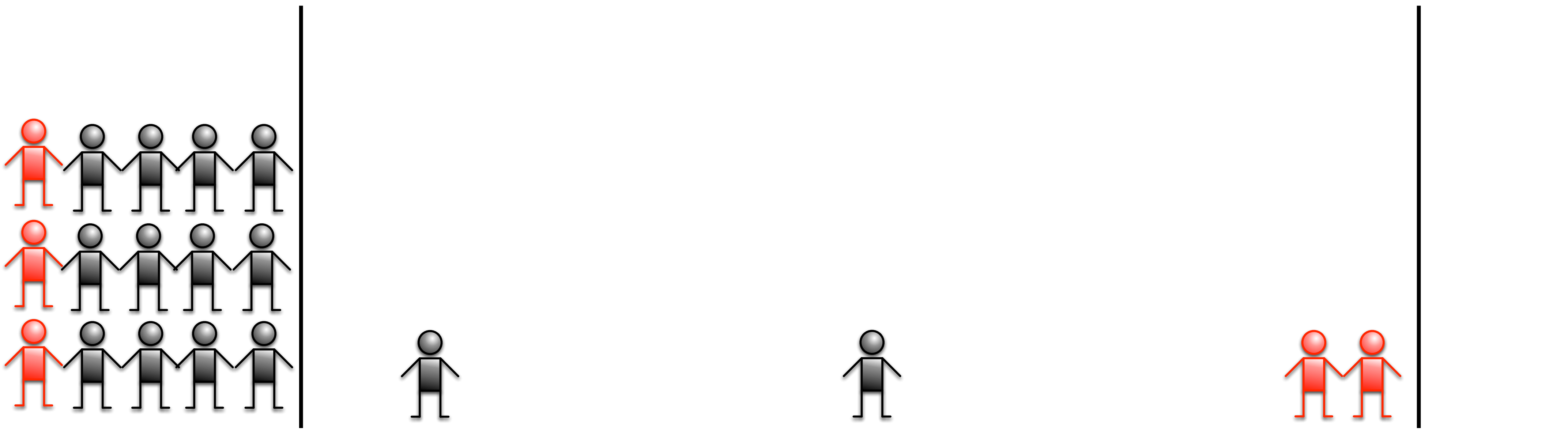
Idea behind result

Idea behind result

“Small” Committee runs algorithm

Maintenance/Coordination of Committee: in paper

Naive Algorithm



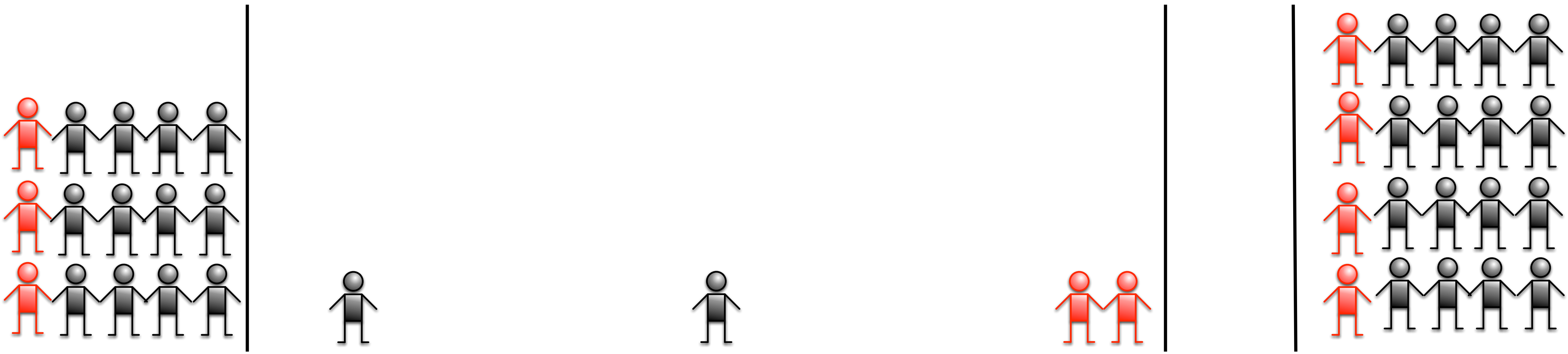
New IDs solve
Entrance Puzzle

All IDs solve **Purge Puzzle**
after constant fraction of churn

Purge Puzzle: Cost of 1

Entrance Puzzle: Cost of 1

Naive Algorithm



New IDs solve **Entrance Puzzle**

All IDs solve **Purge Puzzle** after constant fraction of churn

Purge Puzzle: Cost of 1

Entrance Puzzle: Cost of 1

Naive Result

Both Entrance and Purge puzzles cost 1

Algorithm spend rate is $O(T + J_G)$

T is adversarial spend rate; J_G is good join rate

Naive Result

Both Entrance and Purge puzzles cost 1

Algorithm spend rate is $O(T + J_G)$

T is adversarial spend rate; J_G is good join rate

Can we do better?

Best Entrance Cost

Fix an iteration

T = adversarial spending rate

J = join rate for all IDs

$\mathbf{J_G}$ = join rate for good IDs

ξ = entrance cost

Solving for ξ (Entrance Cost)

Assume: $\mathbf{T} = \xi \mathbf{J}$

Solving for ξ (Entrance Cost)

Assume: $\mathbf{T} = \xi \mathbf{J}$

Good spend rate for entrance: $\xi \mathbf{J_G}$

Solving for ξ (Entrance Cost)

Assume: $\mathbf{T} = \xi \mathbf{J}$

Good spend rate for entrance: $\xi \mathbf{J_G}$

Good spend rate for purges: $\mathbf{J}$

Solving for ξ (Entrance Cost)

Assume: $\mathbf{T} = \xi \mathbf{J}$

Good spend rate for entrance: $\xi \mathbf{J_G}$

Good spend rate for purges: $\mathbf{J}$

To Balance:

$$\xi = \frac{\mathbf{J}}{\mathbf{J_G}}$$

Solving for ξ (Entrance Cost)

Assume: $\mathbf{T} = \xi \mathbf{J}$

Good spend rate for entrance: $\xi \mathbf{J_G}$

Good spend rate for purges: $\mathbf{J}$

To Balance:

$$\xi = \frac{\mathbf{J}}{\mathbf{J_G}}$$

$$\mathbf{J} = \sqrt{\mathbf{J}^2} = \sqrt{\mathbf{J_G} \xi \mathbf{J}} = \sqrt{\mathbf{J_G} \mathbf{T}}$$

Solving for ξ (Entrance Cost)

Assume: $\mathbf{T} = \xi \mathbf{J}$

Good spend rate for entrance: $\xi \mathbf{J_G}$

Good spend rate for purges: $\mathbf{J}$

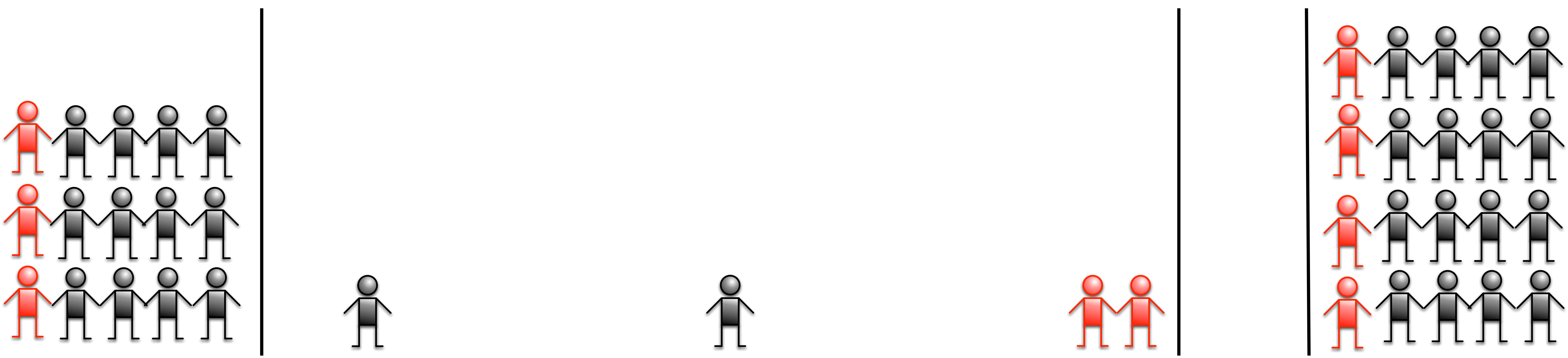
To Balance:

$$\xi = \frac{\mathbf{J}}{\mathbf{J_G}}$$

$$\mathbf{J} = \sqrt{\mathbf{J}^2} = \sqrt{\mathbf{J_G} \xi \mathbf{J}} = \sqrt{\mathbf{J_G} \mathbf{T}}$$

So good spend rate: $\mathbf{J_G} + \sqrt{\mathbf{J_G} \mathbf{T}}$

Our Algorithm: ERGO



New IDs solve
Entrance Puzzle

All IDs solve
Purge Puzzle

Purge Puzzles: Require 1 unit of computation

Entrance Puzzles: Require $\frac{\mathbf{J}}{\tilde{\mathbf{J}}_{\mathbf{G}}}$ units of computation

How to estimate J_G?

Problem: Don't know in advance which IDs are good or bad

We developed an algorithm that maintains a constant factor estimate of J_G assuming α, β-churn for $\alpha, \beta = \Theta(1)$

How to estimate J_G?

Problem: Don't know in advance which IDs are good or bad

We developed an algorithm that maintains a constant factor estimate of J_G assuming α, β-churn for $\alpha, \beta = \Theta(1)$

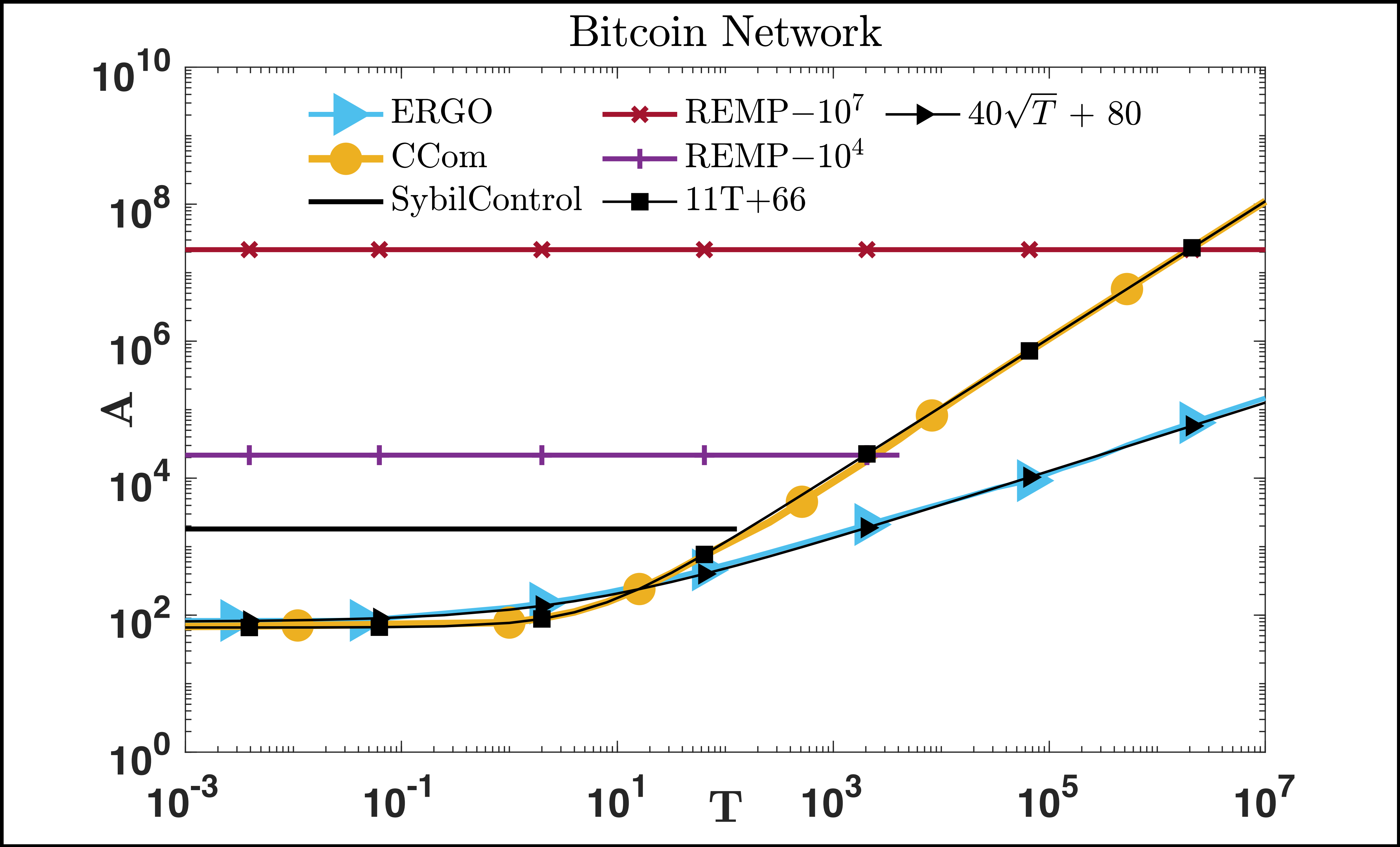
This algorithm for estimating J_G is key technical challenge of our work

Empirical Results

Four data sets: Bitcoin, Ethereum, Gnutella, Bittorrent

Tested **ERGO** vs

- **CCom**: ERGO-light: entrance cost is 1
- SybilControl: Puzzle every 5 seconds
- REMP: Puzzle every x seconds, where x is based on upper bound of adversary power



Bitcoin Network
ERGO
CCom
SybilControl
REMP−10^7
REMP−10^4
11T+66
40√T + 80
A
T
10^0
10^2
10^4
10^6
10^8
10^10
10^-3
10^-1
10^1
10^3
10^5
10^7

Positions

1. Resource burning is fundamental
2. Resource burning must be optimized
3. Resource burned shouldn’t matter
4. Need Permissionless → Permissioned reduction
5. Need domain specific **and** general results

Positions

1. Resource burning is fundamental
2. Resource burning must be optimized
3. Resource burned shouldn't matter
4. Need Permissionless → Permissioned reduction
5. Need domain specific **and** general results

Domain	Primary Resource Consumed	Mechanism	Enabled Functionality	Conjectured Cost
Blockchains	CPU	CPU Puzzles	Distributed Ledger	$O(\sqrt{TJ_G} + J_G)$
DHTs	CPU	CPU Puzzles	Decentralized storage and search	$\tilde{O}(\sqrt{TJ_G} + J_G)$
DDoS Attacks	Bandwidth / CPU	Messages / CPU Puzzles	Fair allocation of server resources	No Conjecture
Review Spam	Human Time	CAPTCHAS	Trusted consumer recommendations	$\tilde{O}(T^{2/3} + P_G)$

T = attacker's RB rate

J_G = good ID join rate

P_G = good ID posting rate

Spam and DDoS

Spam and DDoS

Review Spam:

Weak Learner detects spam with accuracy $> 1/2$

Spam has social cost of 1; P_G is good posting rate

Recent Conjecture: Can achieve cost of $\mathrm{O}(\mathrm{T}^{2/3} + \mathrm{P}_\mathrm{G})$

Spam and DDoS

Review Spam:

Weak Learner detects spam with accuracy $> 1/2$

Spam has social cost of 1; P_G is good posting rate

Recent Conjecture: Can achieve cost of $\mathrm{O}(\mathrm{T}^{2/3} + \mathrm{P}_\mathrm{G})$

Application-layer DDoS Attack:

Goal: Good IDs obtain a $1 - \mathrm{O}(\kappa)$ fraction of service

Cost per service request set by server

Weak Conjecture: Can achieve cost of $\mathrm{o}(\mathrm{T})$

Conclusion

Conclusion

1. Resource burning is fundamental

2. Resource burning must be optimized

3. Resource burned shouldn't matter

4. Need Permissionless $\rightarrow$ Permissioned reduction

5. Need domain specific **and** general results

Future Work

Burn, baby burn, Resource Inferno!

Burn, baby burn, Resource Inferno!

Other application domains? (besides Blockchains, DDoS, Spam, DHTs)

Burn, baby burn, Resource Inferno!

Other application domains? (besides Blockchains, DDoS, Spam, DHTs)

Lower bounds for resource burning

Burn, baby burn, Resource Inferno!

Other application domains? (besides Blockchains, DDoS, Spam, DHTs)

Lower bounds for resource burning

Better integration with game theory

Burn, baby burn, Resource Inferno!

Other application domains? (besides Blockchains, DDoS, Spam, DHTs)

Lower bounds for resource burning

Better integration with game theory

RB cost $\longleftrightarrow$ Payoff for security game

Burn, baby burn, Resource Inferno!

Other application domains? (besides Blockchains, DDoS, Spam, DHTs)

Lower bounds for resource burning

Better integration with game theory

- RB cost $\longleftrightarrow$ Payoff for security game
- Rational agents

Questions?

Backup Slides

Communication

Communication

Diffuse:

Sends a message to all IDs

Communication time is negligible compared RB time

Messages signed with digital signatures

PoW

PoW

Random Oracle Assumption**:** We have a function, h, and h(x) is uniformly random on (0, 1) the first time bit string x is input to h

Computation Cost: Computational cost is number of times h is called

Committee

Logarithmic size

Use state-machine replication to get committee to act in concert

After every purge, old committee elects a new committee from set of current IDs, using Byzantine-resilient coin-flipping

RB can also do useful work

[Ball et al. ’18]: “Proof of Useful Work”

SETH → Hardness of challenge

Can use RB challenges for conjectured hard problems

[Von Anh et al. ’08]: RECAPTCHA

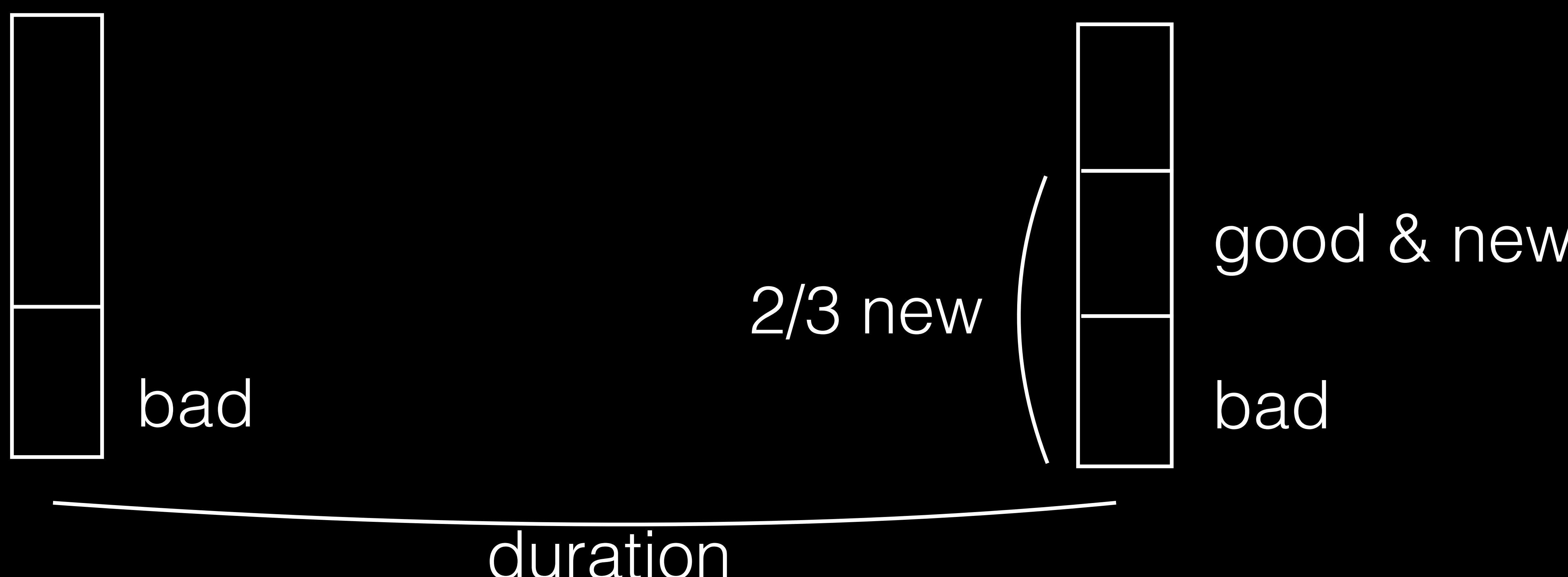
CAPTCHAs used to decipher scanned words

Digitized New York Times archive

$\tilde{J}_G$: Estimate of J_G

Duration: Length of time for set of all IDs to change by 2/3 factor

$$\tilde{J}_G = \frac{\text{number of IDs at start of last duration}}{\text{length of last duration}}$$



$\tilde{J}_G$: Estimate of J_G

Duration: Length of time for set of all IDs to change by 2/3 factor

$$\tilde{J}_G = \frac{\text{number of IDs at start of last duration}}{\text{length of last duration}}$$

$$\tilde{J}_G = \Theta(J_G)$$