Fast Byzantine Agreement

Nicolas Braud-Santoni
Rachid Guerraoui   **Florian Huc**

MMTDC'13, Bremen

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Failure model

Byzantine fault

- Arbitrary behaviour
- Full information

Failure model

Byzantine fault

- Arbitrary behaviour
- Full information

- **Non-adaptive** adversary

Failure model

Byzantine fault

- Arbitrary behaviour
- Full information

- **Non-adaptive** adversary
- **Rushing** adversary

Failure model

Byzantine fault

- Arbitrary behaviour
- Full information

- **Non-adaptive** adversary
- **Rushing** adversary
- No “cryptographic” hypotheses

Consensus

Consensus

- **Propose** values
- Agreement
- Validity
- Termination

Consensus

Consensus

- **Propose** values
- Agreement
- Validity
- Termination

- FLP
- Lower bounds

Consensus

Consensus

- **Propose** values
- Agreement
- Validity
- Termination

Randomised Agreement

- Random values
- Agreement
- **Non-biased**
- Termination

- FLP
- Lower bounds

Contribution

Randomised agreement

	[BPV06]	[KS09]	**BA**
Model	Sync	Sync	Sync
Time	$O(\log n)$	Polylog	Polylog
Bits	$n^{O(\log n)}$	$\tilde{O}(\sqrt{n})$	Polylog
n	$4t+1$	$3t+1$	$3t+1$

Contribution

Randomised agreement

	[BPV06]	[KS09]	**BA**	[PR10]	[KS13]
Model	Sync	Sync	Sync	Async+PC	Async
Time	$O(\log n)$	Polylog	Polylog	n^3	$\tilde{O}(n^{2.5})$
Bits	$n^{O(\log n)}$	$\tilde{O}(\sqrt{n})$	Polylog	$\Omega(n^2 \log n)$	?
n	$4t+1$	$3t+1$	$3t+1$	$4t+1$	$500t$

Almost-everywhere Agreement

Randomised Agreement

- **Random** values
- Agreement
- **Non-biased**
- Termination

Almost-everywhere Agreement

Randomised Agreement

- **Random** values
- Agreement
- **Non-biased**
- Termination

Almost-everywhere Agreement

- **Random** values
- Agreement **over most nodes**
- **Non-biased**
- Termination

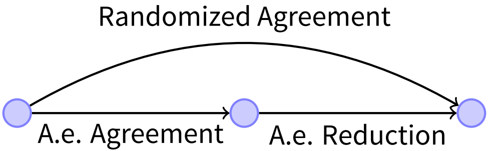
Almost-everywhere Agreement

Randomised Agreement

- Random values
- Agreement
- Non-biased
- Termination

Almost-everywhere Agreement

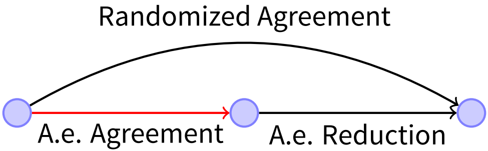
- Random values
- Agreement over most nodes
- Non-biased
- Termination



Almost-everywhere Agreement

State of the art

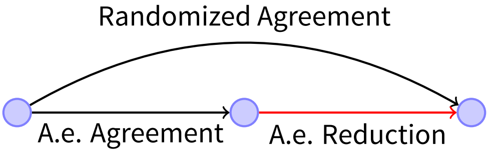
- [Kin+06]: Synchronous almost-everywhere agreement
  Poly-log in time and communication



Almost-everywhere Agreement

State of the art

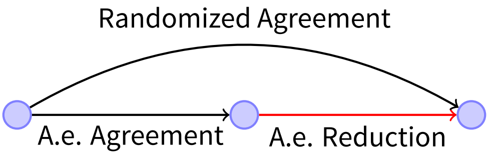
- [Kin+06]: Synchronous almost-everywhere agreement
  Poly-log in time and communication
- [KS09]: Synchronous almost-everywhere reduction
  Poly-log in time, but $O\left(\sqrt{n}\right)$ messages



Contribution

Almost-everywhere Reduction

	AER	**AER**	[KS09]
Model	Sync	Async	Sync
Time	$O(1)$	$O\left(\frac{\log n}{\log \log n}\right)$	Polylog
Bits	$O\left(\log^3 n\right)$	$O\left(\log^3 n\right)$	$\tilde{O}\left(\sqrt{n}\right)$

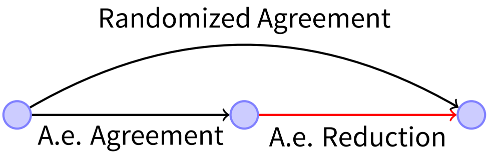


Contribution

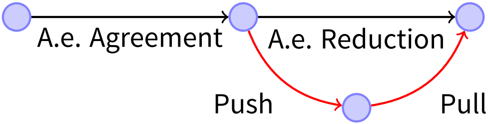
Almost-everywhere Reduction

	AER	**AER**	[KS09]
Model	Sync	Async	Sync
Time	$O(1)$	$O\left(\frac{\log n}{\log \log n}\right)$	Polylog
Bits	$O\left(\log^3 n\right)$	$O\left(\log^3 n\right)$	$\tilde{O}\left(\sqrt{n}\right)$

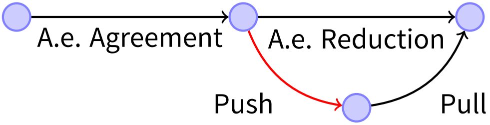
Not load-balanced



Push-pull protocols



Push-pull protocols

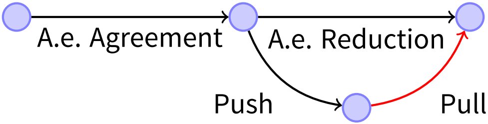


Push

Each process:

1. spreads its **candidate value**;
2. builds a **candidate set** from received proposals.

Push-pull protocols



Push

Each process:

1. spreads its **candidate value**;
2. builds a **candidate set** from received proposals.

Pull

Validate values from the candidate set.

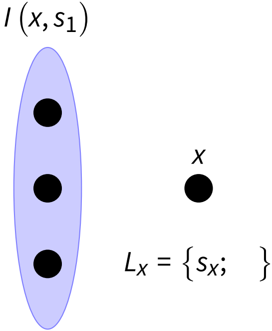
Push phase

Phase 1: Push to node x

$$L_x = \left\{ s_x; \quad \right\}$$

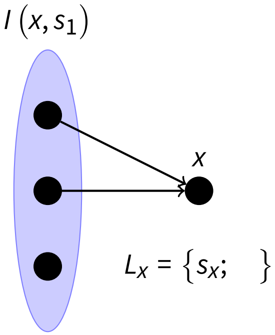
Push phase

Phase 1: Push to node x



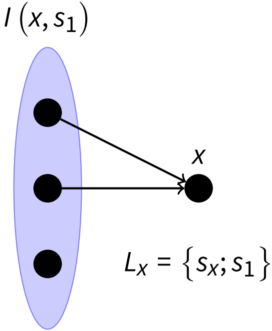
Push phase

Phase 1: Push to node x



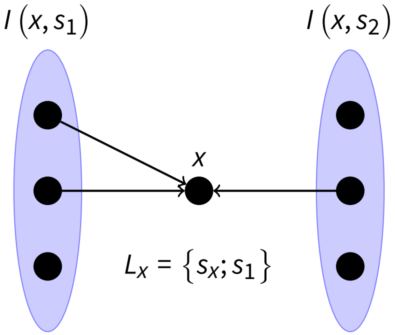
Push phase

Phase 1: Push to node x



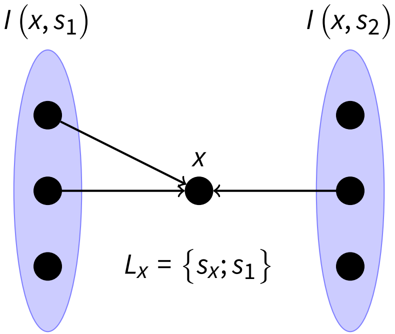
Push phase

Phase 1: Push to node x



Push phase

Phase 1: Push to node x



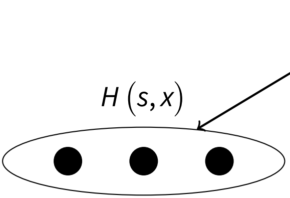
Push $O(n)$ strings

Pull phase

Phase 2: Pull request from node x for value s

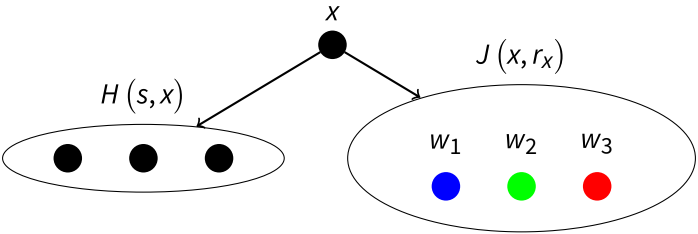
Pull phase

Phase 2: Pull request from node x for value s



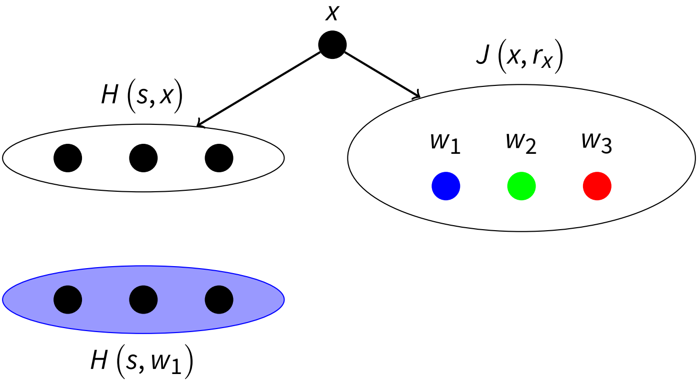
Pull phase

Phase 2: Pull request from node x for value s



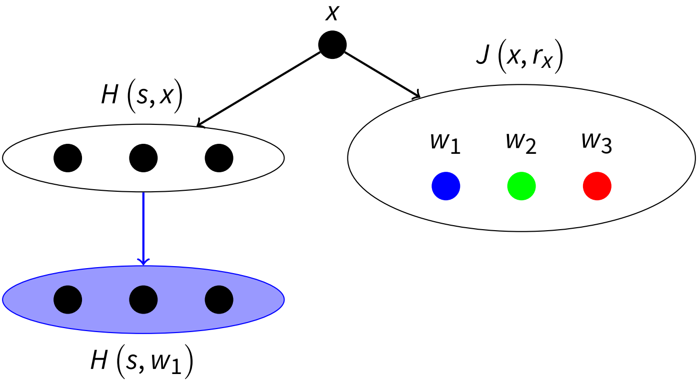
Pull phase

Phase 2: Pull request from node x for value s



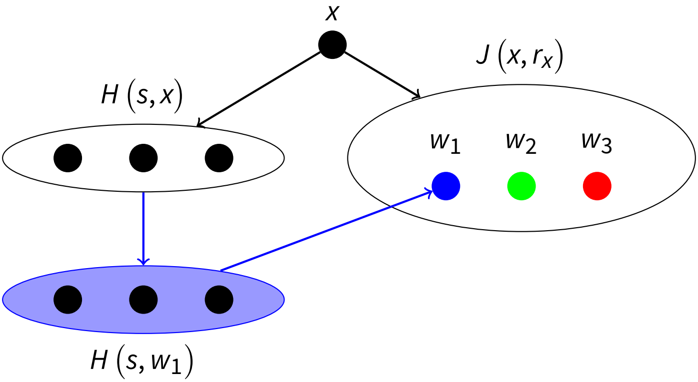
Pull phase

Phase 2: Pull request from node x for value s



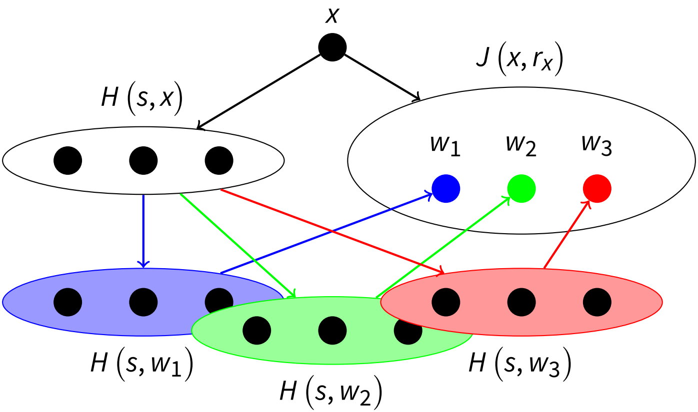
Pull phase

Phase 2: Pull request from node x for value s



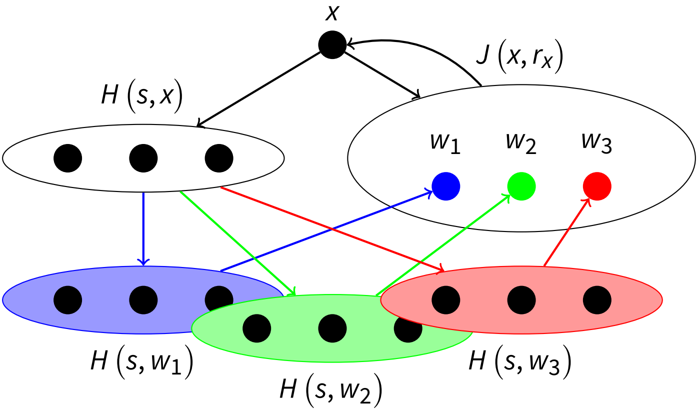
Pull phase

Phase 2: Pull request from node x for value s



Pull phase

Phase 2: Pull request from node x for value s



How explicit can you be ?

- Seems simple

How explicit can you be ?

- Seems simple
- Constructive ?

How explicit can you be ?

- Seems simple
- Constructive ?

NO !

Why not ?

“Magic” selection function

- $\mathbb{S} : X \to 2^Y$ is a (Θ, δ) sampler if for $S \subseteq Y$, fraction δ of inputs have

$$\frac{\left|\mathbb{S}(x) \cap S\right|}{\left|S\right|} > \frac{\left|S\right|}{\left|Y\right|} + \Theta$$

Why not ?

"Magic" selection function

- $\mathbb{S} : X \to 2^Y$ is a (Θ, δ) sampler if for $S \subseteq Y$, fraction δ of inputs have

$$\frac{|\mathbb{S}(x) \cap S|}{|S|} > \frac{|S|}{|Y|} + \Theta$$

- Additional properties: overloading, edge expansion

Why not ?

“Magic” selection function

- $\mathbb{S} : X \to 2^Y$ is a (Θ, δ) sampler if for $S \subseteq Y$, fraction δ of inputs have

$$\frac{|\mathbb{S}(x) \cap S|}{|S|} > \frac{|S|}{|Y|} + \Theta$$

- Additional properties: overloading, edge expansion

Proof techniques

- **Composable** properties

Why not ?

“Magic” selection function

- $\mathbb{S} : X \to 2^Y$ is a (Θ, δ) sampler if for $S \subseteq Y$, fraction δ of inputs have

$$\frac{\left|\mathbb{S}(x) \cap S\right|}{\left|S\right|} > \frac{\left|S\right|}{\left|Y\right|} + \Theta$$

- Additional properties: overloading, edge expansion

Proof techniques

- **Composable** properties
- If $\mathbb{P}\left[P(x)\right] > 0$, such an x exists

Why not ?

"Magic" selection function

- $\mathbb{S} : X \to 2^Y$ is a (Θ, δ) sampler if for $S \subseteq Y$, fraction δ of inputs have

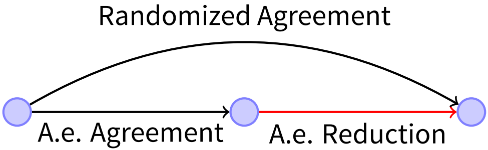
$$\frac{\left|\mathbb{S}(x) \cap S\right|}{\left|S\right|} > \frac{\left|S\right|}{\left|Y\right|} + \Theta$$

- Additional properties: overloading, edge expansion

Proof techniques

- **Composable** properties
- If $\mathbb{P}\left[P(x)\right] > 0$, such an x exists
- If $\mathbb{P}\left[P(x)\right] + \mathbb{P}\left[Q(x)\right] > 1$ then $\mathbb{P}\left[P(x) \wedge Q(x)\right] > 0$

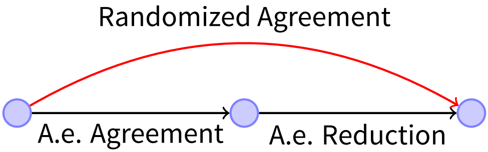
Recap



Almost-everywhere Reduction

	[KS09]	**AER**	**AER**
Model	SNR	SR	Async
Time	$O\left(\log^2 n\right)$	$O\left(1\right)$	$O\left(\frac{\log n}{\log\log n}\right)$
Bits	$\tilde{O}\left(\sqrt{n}\right)$	$O\left(\log^3 n\right)$	$O\left(\log^3 n\right)$

Recap



Randomised agreement = A.e. agreement + reduction

	[BPV06]	[KS09]	**BA**	[PR10]	[KS13]
Model	Sync	Sync	Sync	Async+PC	Async
Time	$O(\log n)$	Polylog	Polylog	n^3	$\tilde{O}(n^{2.5})$
Bits	$n^{O(\log n)}$	$\tilde{O}(\sqrt{n})$	Polylog	$\Omega(n^2 \log n)$	?
n	$4t+1$	$3t+1$	$3t+1$	$4t+1$	$500t$

Conclusion

Result

- A new, highly efficient algorithm
- yields a poly-log Byzantine Agreement protocol.

Conclusion

Result

- A new, highly efficient algorithm
- yields a poly-log Byzantine Agreement protocol.

Almost-everywhere agreement

- High (relative) complexity
- Synchronous

Perspectives

Future work

- Faster, asynchronous almost-everywhere agreement
- Load balancing ?

Perspectives

Future work

- Faster, asynchronous almost-everywhere agreement
- Load balancing ?
- Power of adaptive adversary
- Models for randomization ?

Perspectives

Future work

- Faster, asynchronous almost-everywhere agreement
- Load balancing ?
- Power of adaptive adversary
- Models for randomization ?

Thank you for your attention !