Simple Communication-Optimal Agreement Protocols

Seth Gilbert
EPFL

Dariusz R. Kowalski
University of Liverpool
<table>
<thead>
<tr>
<th>Time Complexity</th>
<th>Communication Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>seconds, rounds, throughput</td>
<td>messages, packets, bits of data</td>
</tr>
</tbody>
</table>
Preliminaries

Consensus

• Agreement
• Validity
• Termination, eventually, with probability 1
Preliminaries

Basic network

• $n$ nodes
• crash failures, majority correct
• synchronous network
Preliminaries

Randomized algorithms

- Oblivious adversary: fix in advance who fails when.
- Safety: guaranteed
- Termination: eventually guaranteed
- Efficiency: with high probability, i.e.,
  \((1-1/n)^c\)
## Prior Work

<table>
<thead>
<tr>
<th>Prior Work</th>
<th>Message/Bit Complexity</th>
<th>Round Complexity</th>
<th>Random?</th>
</tr>
</thead>
<tbody>
<tr>
<td>FloodSet</td>
<td>(O(n^3))</td>
<td>(O(n))</td>
<td>No</td>
</tr>
<tr>
<td>GMY’95</td>
<td>(O(n))</td>
<td>(O(n^{1+\epsilon}))</td>
<td>No</td>
</tr>
<tr>
<td>CK’02, CK’06</td>
<td>(O(n \log^{O(1)} n))</td>
<td>(O(n))</td>
<td>No</td>
</tr>
<tr>
<td>CKS’09</td>
<td>(O(n \log^{O(1)} n))</td>
<td>(O(n))</td>
<td>No</td>
</tr>
<tr>
<td>CMS’89</td>
<td>(O(n^2 \log n))</td>
<td>(O(\log n))</td>
<td>Yes</td>
</tr>
<tr>
<td>CK’09</td>
<td>(O(n \log n))</td>
<td>(O(\log n))</td>
<td>Yes</td>
</tr>
<tr>
<td>Today</td>
<td>(O(n))</td>
<td>(O(\log n))</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Key Technique

Universe Reduction

1. Choose a small set of coordinators
2. Coordinators run (small) consensus protocol
3. Coordinators disseminate the decision
Universe Reduction

Piotr Berman, Juan A. Garay
• Asymptotically Optimal Distributed Consensus

Vinod Vaikuntanathan
• Randomized Algorithms for Reliable Broadcast

Ben-Or, Pavlov, Vaikuntanathan
• Byzantine Agreement in the Full-Information Model in $O(\log n)$ rounds.
Universe Reduction

Kapron, Kempe, King, Saia, Sanwalani

• Fast Asynchronous Byzantine Agreement and Leader Election with Full Information

King, Saia

• Fast, Scalable Byzantine Agreement in the Full Information Model with a Nonadaptive Adversary
Simple Communication-Optimal Agreement

Protocol Presentation

 Universe Reduction

1. Choose a small set of coordinators
2. Coordinators run (small) consensus protocol
3. Coordinators disseminate the decision
4. Fallback protocol
Protocol Presentation

Choosing Coordinators

1. Elect self coordinator with probability: $\Theta\left(\frac{\log n}{n}\right)$

2. If coordinator: choose $\Theta(\sqrt{n \log n})$ intermediaries uniformly at random. Send each a message.

3. Each intermediary sends a response containing a list of coordinators.
Choosing Coordinators

- **Claim**: There are $\Theta(\log n)$ correct coordinators, with high probability.
  - There are $n/2$ correct nodes.
  - There are $(n/2)(c\log n/n)$ correct coordinators, in expectation.
  - Chernoff bound...

$$Pr(X \leq \mu/2) \leq e^{-\mu/4}$$
Protocol Presentation

Choosing Coordinators

- **Claim**: All non-failed coordinators know about all other non-failed coordinators.
Choosing Coordinators

- **Claim**: WHP, there exists a subset $S$ such that:
  
  i. Every process in $S$ is a coordinator.
  
  ii. Every non-failed coordinator is in $S$.
  
  iii. For each non-failed coordinator, its list of coordinators is a subset of $S$.
  
  iv. If ($p \in S$), and ($p \not\in$ a coordinator list), then:
      $p$ fails by the end of the protocol.
Protocol Presentation

Choosing Coordinators

• **Claim:** All non-failed coordinators know about all other non-failed coordinators.
  - Birthday paradox: any two coordinators share an intermediary, with high probability.

\[
\left(1 - \frac{|I|}{n}\right)^{2c\sqrt{n}\log n} \leq \left(1 - \frac{2c\sqrt{n}\log n}{n}\right)^{2c\sqrt{n}\log n} \\
\leq \left(\frac{1}{2}\right)^{4c^2\log^2 n} \leq \left(\frac{1}{n}\right)^{c+2}
\]
Choosing Coordinators

- Communication cost: $O(\sqrt{n} \log^4 n)$
  - # coordinators: $O(\log n)$
  - msgs / coordinator: $O(\sqrt{n} \log n)$
  - max message size: $O(\log^2 n)$

- Time: $O(1)$
Protocol Presentation

Universe Reduction

1. Choose a small set of coordinators
2. Coordinators run (small) consensus protocol
3. Coordinators disseminate the decision
4. Fallback protocol
Protocol Presentation

Limited Universe Consensus

• Each coordinators repeats $\Theta(\log n)$ rounds:
  - Send estimate to other coordinators.
  - Adopt minimum estimate received.

• Output estimate.
Protocol Presentation

Limited Universe Consensus

- **Claim**: With high probability, every coordinators outputs the same value.
  - Each coordinator has a complete list of other coordinators, with high probability.
  - In some round, no coordinator fails (by the pigeon-hole principle).
  - Ergo all coordinators adopt same estimate.
Limited Universe Consensus

• Guarantees:
  - Probabilistic agreement
  - Validity
  - Termination

• Communication Cost: $O(\log^3 n)$

• Time: $O(n)$
Protocol Presentation

Universe Reduction

1. Choose a small set of coordinators
2. Coordinators run (small) consensus protocol
3. Coordinators disseminate the decision
4. Fallback protocol
Protocol Presentation

Disseminate Decision

- Work sharing paradigm:
- Coordinators evenly divide up the work of notifying processes.
- Check for unlikely problems.
- Related to Do-All: Chlebus, Kowalski '06

“Randomization helps to perform independent tasks reliably.”
Protocol Presentation

Disseminate Sub-Protocol

• Inputs:
  - Value $v$ to disseminate
  - List of coordinators

• Outputs:
  - Set of values $V$ received
  - Flag $ds$ indicating success/failure
Protocol Presentation

Disseminate Sub-Protocol

- **Dissemination**: The initial value of every non-failed coordinator is sent to every process.
- **Validity**: Every value received was some coordinators initial value.
- **Consistency**: If \( p \) and \( q \) both output success (\( ds = \text{true} \)), then both had the same initial value.
- **Termination**
Protocol Presentation

Disseminate Sub-Protocol

- Partition processes into $\log n \log^* n$ (disjoint) groups.

- Maintain:
  - List of unnotified groups
  - Count (lower bound) of responded processes
Protocol Presentation

Disseminate Sub-Protocol

• Repeat $\Theta(\log^* n)$ times:
  
  (a) Each coordinator chooses a group at random, sends it the value to disseminate.
  
  (b) Each node sends a response if it has received no other values.
  
  (c) Coordinators count responses, update list, and exchange information.
Protocol Presentation

Disseminate Sub-Protocol

- Final steps:
  - If list not empty:
    - Coordinator sends value directly to everyone.
    - Collects responses.
  - If \((\text{count} > \frac{n}{2})\) then return \text{true}, else \text{false}. 
Protocol Presentation

Disseminate Sub-Protocol

- **Claim**: Dissemination
  - If a coordinator's list is empty, then the value has been sent to everyone. Otherwise, it sends the value directly.

- **Claim**: Validity

- **Claim**: Termination
Protocol Presentation

Disseminate Sub-Protocol

- **Claim**: Consistency
  - The *count* is a lower bound on the number of processes that received value first.
  - If *(count > n/2)* then a majority received value first. Only possible for one value!
Protocol Presentation

Disseminate Sub-Protocol

- **Claim**: Efficient
  - By the end of $\Theta(\log^* n)$ rounds, every group has been selected at least once by a non-failed coordinator.
Detour: Balls & Bins

Bin clearing (review)

- A player has:
  - $b$ balls
  - $b$ bins

- In each round:
  - Throw balls at random into bins.
  - If bin has >0 balls, then remove bin.
Detour: Balls & Bins

Bin clearing

• **Claim**: All the bins are cleared within $\Theta(\log^* n)$ rounds, with high probability.
Detour: Balls & Bins

Bin clearing

- **Claim:** All the bins are cleared within $\Theta(\log^* n)$ rounds, with high probability.

  - Round 1: $b$ balls, $b$ bins

    Expected # remaining bins:

    $$ b \left( 1 - \frac{1}{b} \right)^b \approx \frac{b}{2} $$
Detour: Balls & Bins

Bin clearing

- **Claim**: All the bins are cleared within \( \Theta(\log^* n) \) rounds, with high probability.
  - Round 2: \( b \) balls, \( b/2 \) bins

Expected \# remaining bins:

\[
b \left(1 - \frac{2}{b}\right)^b \approx \frac{b}{2^2}
\]
Detour: Balls & Bins

**Bin clearing**

- **Claim**: All the bins are cleared within $\Theta(\log^* n)$ rounds, with high probability.
  - Round 3: $b$ balls, $b/2^2$ bins

  Expected # remaining bins:

  $$b \left(1 - \frac{2^2}{b}\right)^b \approx \frac{b}{2^{2^2}}$$
Detour: Balls & Bins

Bin clearing

- **Claim**: All the bins are cleared within $\Theta(\log^* n)$ rounds, with high probability.
  - Round $\log^* b$:
    
    Expected # remaining bins:
    
    $$b \left( 1 - \frac{2^{2\cdots2}}{b} \right)^b \approx \frac{b}{2^{2\cdots2}} \approx 1$$
Detour: Balls & Bins

Bin clearing

- **Claim**: All the bins are cleared within $\Theta(\log^* n)$ rounds, with high probability.
  - Round $\log^* b$:
    
    Expected # remaining bins:

    $$\frac{b \left(1 - \frac{2^{2\ldots2}}{b}\right)^b}{2^{2\ldots2}} \approx 1$$
Detour: Balls & Bins

Bin clearing

• Claim: All the bins are cleared within $\Theta(\log^* n)$ rounds, with high probability.
Protocol Presentation

Disseminate Sub-Protocol

• **Claim**: Efficient
  
  - By the end of $\Theta(\log^* n)$ rounds, every group has been selected at least once by a non-failed coordinator.
Protocol Presentation

Disseminate Sub-Protocol

- **Claim**: Efficient
  - Within $\Theta(\log^* n)$ rounds, at most $O(n)$ groups remain un-notified.
  - While ($>2\log n$) unnotified groups: each coordinator picks an un-notified group with probability $> 1/2$.
  - With high probability, # unnotified groups is reduced by $\Theta(\log n)$. 

Protocol Presentation

Disseminate Sub-Protocol

- **Claim**: Efficient
  - Bin clearing:
    - Number of groups: $O(n)$
    - Number of coordinators: $\Theta(\log n)$
  - Conclusion:
    - Within $\Theta(\log^* n)$ rounds, every group has been notified, with high probability.
Protocol Presentation

Disseminate Sub-Protocol

- **Claim**: Efficient
  - Total complexity: $O(n)$
  - Rounds: $\Theta(\log^* n)$
  - Coordinators: $O(n)$
  - Messages: $O(n / \log n \log^* n)$ of size $O(1)$
  - Inter-coordinator message size: $O(n)$
  - Inter-coordinator messages: $O(\log^2 n \log^* n)$
Protocol Presentation

Complete Protocol

1. Choose coordinators
2. Limited universe consensus -> v
3. Disseminate(v) -> true/false (+ v)
   • If false, then stop.
4. Disseminate(v) -> true/false + v
   • Adopt estimate v.
5. Disseminate(v) -> (true/false) + v
   • If v is estimate, decide(v)
Protocol Presentation

Complete Protocol

6. If undecided, send “FALLBACK” message to all.
7. If undecided or receive “FALLBACK” message, then execute classical consensus protocol.
Protocol Presentation

**Complete Protocol**

- **Claim**: Agreement
  - Only one value possible after Step 3, due to *consistency* property.
**Protocol Presentation**

**Complete Protocol**

1. Choose coordinators
2. Limited universe consensus $\rightarrow v$
3. Disseminate$(v) \rightarrow true/false (+ v)$
   - If false, then stop.
4. Disseminate$(v) \rightarrow true/false + v$
   - Adopt estimate $v$.
5. Disseminate$(v) \rightarrow (true/false) + v$
   - If $v$ is estimate, $\text{decide}(v)$
Protocol Presentation

CompleteProtocol

- **Claim**: Agreement
  - Only one value possible after Step 3, due to *consistency* property.
  - Only one decision possible in Step 5...
  - Only one decision possible in FALLBACK protocol...
Protocol Presentation

**CompleteProtocol**

- **Claim**: Agreement
  - Only one value possible after Step 3, due to **consistency** property.
  - Only one decision possible in Step 5...
  - Only one decision possible in FALLBACK protocol...
  - If decision in Step 5, then all processes received value in Step 4, so all processes start FALLBACK with the same value.
Simple Communication-Optimal Agreement

Protocol Presentation

Complete Protocol

1. Choose coordinators
2. Limited universe consensus → v
3. Disseminate(v) → true/false (+ v)
   • If false, then stop.
4. Disseminate(v) → true/false + v
   • Adopt estimate v.
5. Disseminate(v) → (true/false) + v
   • If v is estimate, decide(v)
Protocol Presentation

CompleteProtocol

• **Claim**: Agreement
• **Claim**: Validity
• **Claim**: Termination
• **Claim**: Efficiency
  - With high probability, no process reaches the FALLBACK protocol.
Protocol Presentation

Complete Protocol

1. Choose coordinators
2. Limited universe consensus $\rightarrow v$
3. Disseminate($v$) $\rightarrow \text{true/false} (+ v)$
   - If false, then stop.
4. Disseminate($v$) $\rightarrow \text{true/false} + v$
   - Adopt estimate $v$.
5. Disseminate($v$) $\rightarrow (\text{true/false}) + v$
   - If $v$ is estimate, $\text{decide}(v)$
Protocol Presentation

Universe Reduction
1. Choose a small set of coordinators
2. Coordinators run (small) consensus protocol
3. Coordinators disseminate the decision
4. Fallback protocol

Complexity:
- Time: $O(n)$ w.h.p.
- Communication: $O(n)$ w.h.p.
Partially Synchrony

What if...

• Some executions are synchronous
• Some executions are asynchronous

Goal:

• Efficiency in synchronous executions
• Correctness in all executions
Partial Synchrony

Model (in brief; see DLS)

- Processes have clocks.
- In synchronous executions:
  - clock skew is bounded
  - message delay is bounded
- Skew/delay bounds are known.
Partial Synchrony

Modifications

• Simulate synchronous rounds
  - Wait long enough to ensure that, if the execution is synchronous, every round $r$ message is received before starting round $r+1$.
  - Start round $r$ at time (according to local clock):
    \[
    d \left( \frac{1}{1 - \delta} \right)^{\delta^{-1}} \left( \frac{1 + \delta^\mu}{1 - \delta} \right)^j_{j=0}
    \]
Partial Synchrony

Modifications

• Fallback:
  1. Attach estimate to “FALLBACK” request.
  2. Abort immediately on “FALLBACK” request.
  3. Adopt value received in “FALLBACK” request.
  4. Send “FALLBACK” request to all.
  5. Wait for a majority of “FALLBACK” messages before beginning fallback protocol.
  6. Use asynchronous fallback protocol.
Partial Synchrony

Re-analysis

• In asynchronous executions, no guarantee of good coordinators or good agreement!
• Dissemination is still ok!
  - Consistency/Dissemination do not depend on synchrony.
Protocol Presentation

Disseminate Sub-Protocol

- Repeat $\Theta(\log^* n)$ times:
  
  (a) Each coordinator chooses a group at random, sends it the value to disseminate.
  
  (b) Each node sends a response if it has received no other values.
  
  (c) Coordinators count responses, update list, and exchange information.

- If not done, send value directly to all.
Partial Synchrony

Re-analysis

• In asynchronous executions, no guarantee of good coordinators or good agreement!
• Dissemination is still ok!
  - Consistency/Dissemination do not depend on synchrony.
• Only one decision value possible, even in asynchronous executions.
Extensions
Extensions

Fault-tolerant Gossip

• Each process begins with initial rumor
• Goal: distribute every rumor to every process

Typical algorithm:

• Repeat:
  - Choose target at random.
  - Send it all rumors.
Extensions

Typical algorithm:

- Repeat:
  - Choose target at random.
  - Send it all rumors.

Complexity:

- Rounds: $O(\log n)$
- Message complexity: $O(n \log n)$
Extensions

Typical algorithm:

- Repeat:
  - Choose target at random.
  - Send it all rumors.

Complexity:

- Rounds: $O(\log n)$
- Message complexity: $O(n \log n)$
Extensions

Coordinator Gossip:

1. Choose coordinators
2. Collect rumors
3. Disseminate rumors
4. Disseminate “DONE”.
5. If not “DONE”, then send “FALLBACK” request.
6. Send rumor directly to all processes.
Extensions

Collect rumors:

1. Run Disseminate protocol
2. When a coordinator sends messages to a group, each process attaches its rumor to its response.
3. Coordinators exchange (and aggregate) rumors.
Extensions

**Complexity:**

- Rounds: $\Theta(\log^* n)$
- Messages: $O(n)$

- Communication depends on rumor size...
Extensions?

Local Algorithms

- No process sends too many messages
- Work is “evenly” shared.

Coordinator-based algorithms are not local!
Extensions?

Coordinator-based algorithms are not local!??

- Problem: coordinator sends too many messages during Disseminate sub-protocol.
- Solution: coordinator initiates gossip in a group...

- Problem: coordinator discovery
- Solution: careful flooding
Extensions?

Upper / Lower Bound Gap

- Rounds: \( O(\log n) \)
- Lower bound: \( \Omega\left(\frac{\log n}{\log \log n}\right) \)
Extensions?

Expected running time gap

- Expected rounds: $O(\log n)$

- Easy (?) improvement: $\Theta(\log^* n)$

- Lower bound: $O(1)$
Hard Open Question

Deterministic Algorithms

- Possible or impossible:
  - Running time: $O(n)$
  - Communication complexity: $O(n)$
Hard Open Question

Deterministic Algorithms

- Conjecture: Impossible!
Hard Open Question

Deterministic Algorithms

- Conjecture: Impossible!
  - Yoram Moses says:
    “For simultaneous consensus, easy to see via `knowledge-based’ analysis.
  - Dan Alistarh / Petr Kouznetsov say:
    “Maybe topology implies you need more connectivity than is possible with so little communication.”
**Hard Open Question**

**Deterministic Algorithms**
- Conjecture: Impossible!

  Intuition:
  - Each process sends only $O(1)$ messages!
  - Imagine a communication graph with (average) degree $O(1)$.
  - No such graphs exist (?) that are $(n/2)$ node-connected!
  - Ergo, partitioning argument...
Hard Open Question

Adaptive Randomized Algorithms

• Possible or impossible:
  - Running time: $O(n)$
  - Communication complexity: $O(n)$
Conclusions

Universe reduction is simple...

1. Choose a small set of coordinators
2. Coordinators run (smaller) protocol
3. Coordinators disseminate the decision

Universe reduction is efficient...

• Time: $O(n)$ with high probability
• Communication: $O(n)$ with high probability