Truth, Lies, and Random Bits

Jared Saia

University of New Mexico, Albuquerque, NM, USA
Westerns

Wide-open spaces

Epic struggles

Borrow from many sources
Westerns

Wide-open spaces

Epic struggles

Borrow from many sources
Westerns
Research

Wide-open spaces

Epic struggles

Borrow from many sources
Truth, Lies, and Random Bits
Overview

Coin Game

Spectral Approach

Analysis

Motivation
Game Definition
Difficulty
Byzantine Agreement

Each node starts with a bit

Goals: 1) all good nodes output same bit; 2) this bit equals an input bit of a good node

\[ t = \# \text{ bad nodes controlled by an adversary} \]
Group Decisions

Periodically, components unite in a decision

Idea: components vote. Problem: Who counts votes?
Taking Majority is Fragile
Byzantine Agreement fixes this
Recent Applications

Bitcoin

“Bitcoin is based on a novel Byzantine agreement protocol in which cryptographic puzzles keep a computationally bounded adversary from gaining too much influence”

Secure Multiparty Computation

“Such protocols strongly rely on the extensive use of a broadcast channel, which is in turn realized using authenticated Byzantine Agreement.”

Game Theory (Mediators)

“… deep connections between implementing mediators and various agreement problems, such as Byzantine agreement”
Previous Work

Leslie Lamport ‘13

Barbara Liskov ‘08

Two Turing Awards

Tens of thousands of papers
Classic Model

**Full Information**: Adversary knows state of all nodes

**Adaptive Adversary**: takes over nodes at any time up to $t$ total

**Asynchronous**: Adversary schedules message delivery
Previous Work - Classic Model

[Ben-Or ’83] gave first randomized algorithm to solve BA in this model

[FLP ’85] showed BA impossible for deterministic algorithms even when t=1

Ben-Or’s algorithm is exponential expected communication time

Communication Time = maximum length of any chain of messages
Recent Work [KS ’13,’14]

Valerie King
University of Victoria


Recent Work [KS ’13,’14]

Las Vegas algorithm that solves Byzantine agreement in the classic model

We tolerate $t = \Theta(n)$

Expected communication time is $O(n^3)$

Computation time and bits sent are polynomial in expectation
Ben-Or’s algorithm

Consists of rounds

Uses private random bits to create a global coin with probability $1/2^n$ in each round

For each round there is a correct direction

If there is a global coin and it is in this direction, agreement is reached

**Our goal**: Get a good global coin after polynomial rounds using private random bits
“Easy” Problems

Ignore in this talk
“Easy” Problems

Equivocation: Bad nodes send different coins to different nodes

Bracha’s Reliable Broadcast: If a good node receives a message from a bad node, q, all other good nodes that receive a message from q will eventually receive the same message

Missing messages: Adversary delays messages so that different nodes receive different coins

Common coins: coins known to most nodes

No more than 2t coins from good nodes, no more than 2 per node that are not common.

Common coins are known to n−4t good nodes.
Hard Problem

Bad nodes create biased bits
Byzantine Agreement
Algorithms use a Global Coin

Global coin is generated from random bits of individual nodes

In each round, there is a correct direction

If global coin is in that direction, algorithm succeeds
Nodes, Server, and Adversary

Good **nodes** generate random bits

**Server** wants to generate a random bit (global coin) but can’t generate randomness itself

**Adversary** can take over nodes

These nodes will generate adversarial bits

Adversary wants to thwart goal of server
Coin Game

$n$ nodes; 1 server

every round:

- each node sends a random bit

- server receives bits and outputs global coin

**Goal:** Global coin is in correct direction
Coin Game

Adversary takes over up to $t = \Theta(n)$ nodes every round:

- each node sends a random bit
- bad nodes send adversarial bits
- server receives bits and outputs global coin

**Goal:** Global coin is in correct direction
Single Round Coin Games

“Boolean functions always have small dominant sets of variables” [KKL ’88]

Let $f$ be a boolean monotone function over $n$ variables, where $\Pr(f=1)$ is not $o(1)$

Then, almost surely, there are $o(n)$ variables that can make $f$ equal 1

Result uses harmonic analysis

Spawned work on influence
Multiround Global Coin

Goal: In all but X rounds, global coin has constant probability of correct outcome

Want small X
Summing Bits

With constant probability, sum of bits of good nodes will be in correct direction

Bad nodes must generate bad deviation in opposite direction to foil this good event

If the few bad nodes generate large deviation repeatedly, we can find them
Bad Deviation
Bad Deviation
The Server
Unable to generate randomness on its own. Uses bits received from good and bad nodes to output global coin.

The Good
Generate and send truly random bits.

The Bad
Generate adversarial bits. Want to bias the global coin. Constant fraction of nodes.
Overview

Coin Game

Spectral Approach

Analysis

Problem
Related Work
Our Algorithm
Terminology

**epoch** is $m = \Theta(n)$ rounds

**deviation** of a set of nodes in an epoch is absolute value of sum of all nodes’ bits

**direction** of a set of nodes in a round is sign of the sum of the nodes’ bits
Matrix

After every epoch, there is a matrix $M$

$M$ is a m by n matrix

$M(i,j) = \text{for round } i, \text{ node } j\text{'s bit}$

Use $M$ to detect “suspicious” behavior
Good rounds

In each epoch, expect a constant fraction of rounds to be **good**: deviation of good nodes is $\sqrt{n}$ in correct direction.

Bad nodes have deviation $\geq \sqrt{n}$ in a good round.
Bad deviation

In every epoch, there is a constant fraction of rounds, $R$, and at most $t$ nodes, $B$, such that:

\[
\text{The sum over all rounds in } R, \text{ of the deviation of all nodes in } B \text{ is } \Omega(n^{1.5})
\]
Matrix as a graph

Sum of edge weights is $\geq \sqrt{n}$

(nodes) B --- R (rounds)
Prior Work

Yojimbo, 1961
Prior Work - Spectral

Page Rank
Eigentrust
Hidden Clique
Page Rank

Google’s $300 billion “secret sauce”

M is a stochastic matrix (giving a random walk over the web graph)

r is top right eigenvector of M (and stationary distribution of M’s walk)

For a web page, i, r[i] = “authority” of i
Eigentrust [KSG '03]

$M(i,j)$ represents amount party $i$ trusts party $j$

$r$ is top right eigenvector of $M$

$r[i] = \text{"trustworthiness"}$ of party $i$

Intuitively, party $i$ is trustworthy if it is trusted by parties that are themselves trustworthy
Differences with Coin Game

Eigentrust and PageRank: Want to identify good nodes based on feedback from other nodes

Coin Game: Want to identify bad nodes based on deviation from random behavior
A random $G(n, 1/2)$ graph is chosen.

A $k$-clique is randomly placed in $G$.

[AKS ’98] give an algorithm for $k = \sqrt{n}$

1. $\mathbf{v}$ is second eigenvector of adj. matrix of $G$.
2. $W$ is top $k$ vertices sorted by abs. value in $\mathbf{v}$.
3. Returns all nodes with $3k/4$ neighbors in $W$. 

Hidden Clique [AKS ’98]
Differences with Hidden Clique

Hidden Clique:
Want to find sub-matrix that is all 1’s

Coin Game:
Want to find sub-matrix where sum of each row has high absolute value
Our Algorithm
Distrust
Distrust

Each node starts with a **distrust** value of 0.

After each epoch, server increases the distrust value of each node by the square of its entry in the top right eigenvector.

When distrust value of a node is 1, that node is blacklisted - subsequent messages from it are ignored.
Algorithm

1. Run an epoch; Let M be the epoch’s matrix

2. If \(|M|\) is “sufficiently large”
   
   I. Compute the top right eigenvector, r, of M
   
   II. Increase distrust value of node i by \(r[i]^2\)

3. Blacklist a node if its distrust value reaches 1
Overview

Coin Game

Spectral Approach

Analysis

$M_g \ vs \ M_b$

$r_g \ vs \ r_b$

Distrust & Blacklisting
$M_b$ and $M_g$

$M$ is the $m$ by $n$ epoch matrix

$M_b$ is bad columns of $M$

$M_g$ is good columns of $M$

Assume $M = [M_b M_g]$
Fact 1: $|M_g| = O(\sqrt{n})$ (whp)

Proof:

Each entry of $M_g$ is an independent random variable with expectation 0; range $[-1,+1]$; and $\sigma = O(1)$.

Fact 1 then follows from classic results on stochastic matrices.
Fact 2: $|M_b| = \Omega(\sqrt{n})$

Proof:

$x$ is a unit vector with entries 0 for good nodes and entries $1/\sqrt{t}$ for bad nodes

$y$ is a unit vector with entries 0 for bad rounds and entries $\pm 1/\sqrt{(cm)}$ for good rounds (sign is direction of bad deviation)

Then $y^T M_b x = \Omega(\sqrt{n})$
Lemma 1: $|M_b| \geq C |M_g|$ for any constant $C$

Proof:

Fact 1: $|M_g| = O(\sqrt{n})$ (independence)

Fact 2: $|M_b| = \Omega(\sqrt{n})$ (to bias good rounds)
$r_b$ and $r_g$

$r$ : top right eigenvector of $M$

$r_b$ : entries for bad nodes

\[ r_b[i] = r[i] \text{ for } 1 \leq i \leq t; \text{ all other entries are 0} \]

$r_g$ : entries for good nodes

\[ r_g[i] = r[i] \text{ for } t+1 \leq i \leq n; \text{ all other entries are 0} \]
$|r_b|$ is large
Lemma 2: $|r_g|^2 < |r_b|^2 / 2$

**Proof:** Assume not. Then $|r_b|^2 \leq 2/3$

\[
|M_b| \leq \ell^T (Mr)
\]

\[
\leq |\ell| |Mr|
\]

\[
\leq |M_b| |r_b| + |M_g| |r_g|
\]

\[
\leq |M_b| (|r_b| + 1/C |r_g|)
\]

\[
\leq |M_b| (\sqrt{2/3} + 1/C)
\]

\[
< |M_b|
\]

Last line holds if $C \geq 5.45$ (i.e. $t \leq .004n$)
Algorithm

1. Run an epoch; Let M be the epoch’s matrix

2. If |M| is “sufficiently large”
   I. Compute the top right eigenvector, r, of M
   II. Increase distrust value of node i by r[i]^2

3. Blacklist a node if its distrust value reaches 1
Distrust reveals bad nodes

Distrust values for bad nodes increase at twice the rate as distrust values for good nodes (by Lemma 2)

Thus we blacklist no more than $t$ good nodes

Distrust of all nodes increases by 1 in any epoch where adversary foils the good rounds

Thus have at most $O(n)$ such epochs before all bad nodes are blacklisted
Summary

First expected polynomial time algorithm for classic Byzantine agreement

Previous best algorithm (Ben-Or’s) was expected exponential time

**New technique:** coin game - forces attackers into statistically deviant and detectable behavior
Future Work

True Grit, 2010
Coin Game

No Country for Old Men, 2007
Coin Game

Adversary takes over up to $t = \Theta(n)$ nodes every round:

- each node sends a random bit
- bad nodes send adversarial bits
- server receives bits and outputs global coin

Goal: Global coin is in correct direction
Coin Game

1) Reduce X

2) Other Applications

Adversary must engage in statistically deviant behavior to attack system

Secure Multiparty Computation, Threshold cryptography, Wisdom of crowds, Page rank
Research

Epic struggles

Borrow from many sources

Wide-open spaces
Epic Struggles

Focus on least understood problems
Modern life contrives against this
Takes effort
Borrow From Many Sources
Wide-Open Spaces

CS is young

Easy to find new problems

Shouldn’t forget the old ones!
Two Classics
“Sake. I’ll think while I drink.”
Two Classics
Noisy Channel

Coding Theorems for a Discrete Source
With a Fidelity Criterion*

Claude E. Shannon**

Abstract

Consider a discrete source producing a sequence of message letters from a finite alphabet. A single-letter distortion measure is given by a non-negative matrix \(d_{ij}\). The entry \(d_{ij}\) measures the "cost" or "distortion" if letter \(i\) is reproduced at the receiver as letter \(j\). The average distortion of a communications system (source-coder-noisy channel-decoder) is taken to be \(d = \sum_{i,j} P_{ij}d_{ij}\) where \(P_{ij}\) is the probability of \(i\) being reproduced as \(j\). It is shown that there is a function \(R(d)\) that measures the "equivalent rate" of the source for a given level of distortion. For coding purposes where a level \(d\) of distortion can be tolerated, the source acts like one with information rate \(R(d)\). Methods are given for calculating \(R(d)\), and various properties discussed. Finally, generalizations to ergodic sources, to continuous sources, and to distortion measures involving blocks of letters are developed.
Noisy Channel

How can we *compute* over a noisy channel? [S ‘96]

Coding Theory fails

[H ’14] gives conjectured optimal communication rate w/ known noise rate

What about unknown noise rate?
PROBABILISTIC LOGICS AND THE SYNTHESIS OF RELIABLE
ORGANISMS FROM UNRELIABLE COMPONENTS

J. von Neumann

1. INTRODUCTION

The paper that follows is based on notes taken by Dr. R. S. Pierce on five lectures given by the author at the California Institute of Technology in January 1952. They have been revised by the author but they reflect, apart from minor changes, the lectures as they were delivered.
Noisy Gates

Ideal gates: never fail

Noisy gates: flip output independently with some small probability

Takes $n$ ideal gates to compute a function $f$

How many noisy gates does it take to compute $f$ with probability approaching 1?
Noisy Gates

$\theta(n \log n)$ noisy gates are required

Problem: $\log n$ multiplicative blowup even if no gates fail

Q: Can we tune the cost overhead to depend on the number of gates that fail?
Collaborators

Varsha Dani (UNM), Mahnush Mohavedi (UNM), Mahdi Zamani (UNM), Maxwell Young (Drexel University)
Questions?
Extra Slides
Ben-Or’s algorithm

Consists of rounds

Uses private random bits to create a global coin with probability $1/2^n$ in each round

For each round there is a correct direction

If there is a global coin and it is in this direction, agreement is reached

Our goal: Get a good global coin after polynomial rounds using private random bits
“Easy” Problems

Ignore in this talk
“Easy” Problems

Equivocation: Bad nodes send different coins to different nodes

Bracha’s Reliable Broadcast: If a good node receives a message from a bad node, q, all other good nodes that receive a message from q will eventually receive the same message.

Missing messages: Adversary delays messages so that different nodes receive different coins

Common coins: coins known to most nodes

No more than 2t coins from good nodes, no more than 2 per node that are not common.

Common coins are known to n-4t good nodes.
Hard Problem

Bad nodes create biased bits
Reliable Broadcast (Bracha)

All bits sent using reliable broadcast

Ensures if a message is “received” by a good node, same message is eventually “received” by all nodes

Prevents equivocation

Doesn’t solve BA

If a bad player reliably broadcasts, may be case that no good player “receives” the message
When to update distrust

Some good nodes may not receive the coinflips of the bad nodes in a given epoch

If $|M| \leq (mn)^{1/2}/(2c_1)$ then don’t do distrust updates ($t = c_1n$)

If there is no agreement, a linear number of good nodes will perform updates
Motivation: Wisdom of crowds

Average estimate is quite accurate

Why? People have independent “noise” [S ’04]

Idea: Coin game can create a robust means to harness wisdom of crowds
Motivation: Threshold Cryptography

A group of nodes want to generate a public key

Requires creation of string of random bits

Group may contain malicious nodes

Idea: Coin game robustly generates key