Blockchains & Cryptocurrencies

Applications of Blockchains - II

Instructor: Abhishek Jain
Johns Hopkins University - Spring 2021

*Some slides based on NBFMG

Last Time

What can we build on top of Bitcoin/Blockchains?

Recap: Why Applications from Blockchains?

Decentralization: Many applications easy to realize with a central trusted authority. Bitcoin/ Blockchains can often help in removing central trust

Recap: Applications

- Timestamping
- Token tracking
- Public randomness
- Prediction markets
- Fair protocols: Multiparty lotteries, MPC
- One-time Programs
- Non-Interactive Zero Knowledge
- …

Prediction markets & real-world data feeds

Assertions about the outside world

- **Idea:** add a mechanism to assert facts
  - election outcomes
  - sports results
  - commodity prices
- Bet or hedge results using smart contracts
- Forwards, futures, options...

General formulation: **prediction market**

Prediction markets

- Idea: trade shares in a potential future event
- Shares worth X if the event happens, 0 if not
- Current price / x = estimated probability

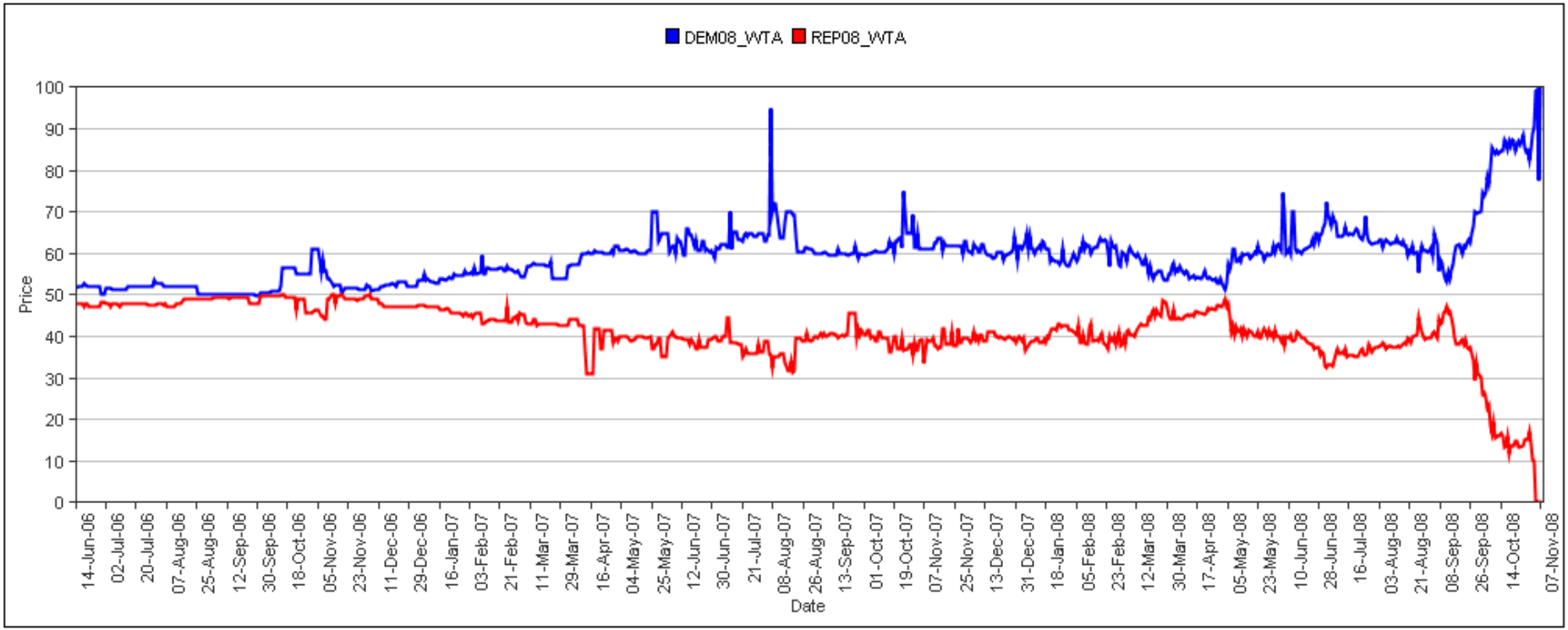
Example: World Cup 2014

					
pre-tournament	0.12	0.09	0.22	0.01	0.05
after group stage	0.18	0.15	0.31	0.06	0.00
before semis	0.26	0.21	0.45	0.00	0.00
before finals	0.64	0.36	0.00	0.00	0.00
final	1	0	0	0	0

Can immediately profit!

Should have shorted

Example: 2008 US Presidential election



source: Iowa Electronic Markets

Prediction markets

- Economists love them
  - reveal all knowledge about the future (under a number of assumptions)
  - allows profit from accurate predictions
  - “a tax on BS”
- Often beat polls and expert opinions
- Significant regulatory hurdles
  - InTrade shut down in 2013

Decentralized prediction markets?

- Decentralized payment & enforcement
- Decentralized arbitration
- Decentralized order book

Decentralized payment & settlement

- Simple solution: Bitcoin + trusted arbiters
- Better solution: altcoin with built-in support

Payment & settlement - FutureCoin

- BuyPortfolio(event e)
  - one share in every outcome for $1
- TradeShares(...)
  - exchange shares for each other or currency
  - one way of profiting
- SellPortfolio(event e)
  - redeem one share in every outcome for $1

Clark et al. 2014

Arbitration models

- Trusted arbiters
  - allow anybody to define & open a market
  - risk of incorrect arbitration, absconding
- Users vote
  - requires incentives, bonds, reputation
- Miners vote
  - may be disinterested or not know

Order books

- Goal: match best bid and ask offers

Predictious.com

Centralized order books

- Traditional model
- Promise to split surplus between buyer, seller
- Front-running is considered a serious crime!
  - require regulation, auditing, monitoring

Decentralized order books

- Idea: Submit orders to miners, let them match any possible trade
- Spread is retained as a transaction fee
- Front-running now not profitable!
- May be less efficient
  - Higher fees
  - Slower trades to avoid higher fees

What can be built on Bitcoin?

payment	✓
settlement	no trades
arbitration	trusted arbiter only
order books	must be external

Bitcoin isn't enough

Cryptocurrencies

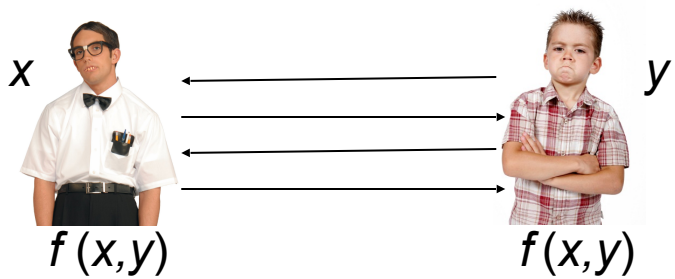
How to Use ~~Bitcoin~~ to Design Fair Protocols

Iddo Bentov, Ranjit Kumaresan

CRYPTO 2014

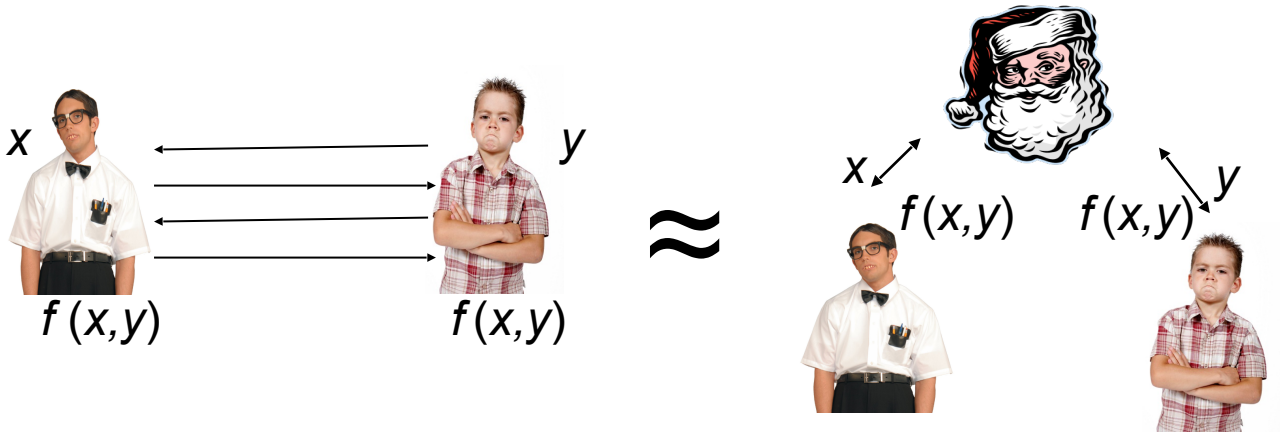
Slides based on Ranjit's talk

Secure Computation



- Most general problem in cryptography
- Feasibility results [Yao86,GMW87,…]
- Moving fast from theory to practice

Secure Computation



Protocol for secure computation emulates a trusted party

Ideally…

- Privacy
- Correctness
- Fairness

Goal for Today

- **Fairness** in Secure Computation
- We will not cover privacy, and simply assume that secure computation protocols with (input) privacy exist
- If you are interested in privacy, attend Modern Crypto course

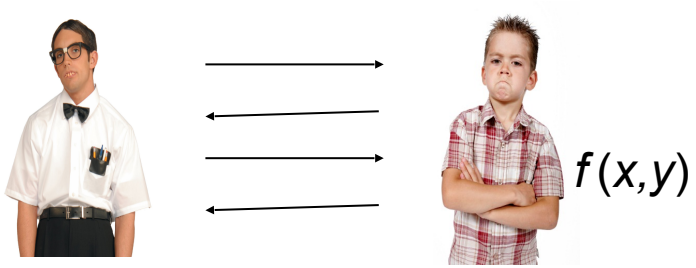
What is Fairness (in cryptography)?

- **Coin Tossing:** Outcome is random (adversary cannot “bias" the coin-toss)
- **Secure Computation:** Either all parties learn the function output or no one does

Why is Fairness hard to achieve?

- **Premature "abort":** In typical cryptographic protocols (say involving two parties), one party can learn the output before the other. That party can simply abort, denying the other party from learning the output.
- **Similar idea in coin-tossing:** Abort if outcome does not seem beneficial (e.g., in determining lottery winner) and possibly re-start protocol

Can we achieve Fairness?



- 2-party fair coin tossing impossible [Cle86]
  - (Last time: Workaround using Bitcoin)
- Fair secure computation possible only in restricted settings
  - For restricted class of functions [GHKL08,Ash14]
  - If majority of parties are honest [BGW88,RB89]

Fair Exchange

[Rab81,BGMR85,ASW97,ASW98,BN00,….]

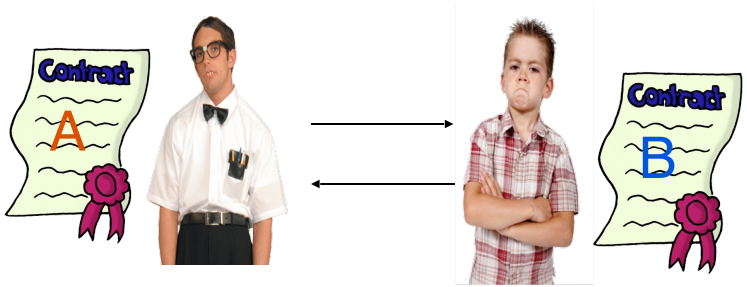
- E.g., contract signing, digital media
- Special case of secure computation

Contract signing:

- Two parties want to sign a contract
- Neither wants to sign first
  - The other signer could be malicious…
- **Want**: Atomicity

Fair Exchange

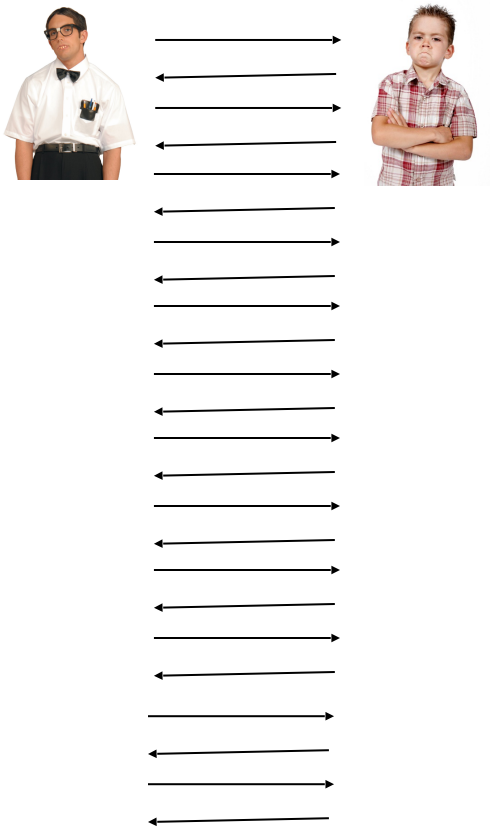
[Rab81,BGMR85,ASW97,ASW98,BN00,….]



Fair exchange is **impossible** [Cle86,PG99,BN00]

Workarounds I

- **Gradual release mechanisms**
  [BG89,GL91,BN00,GJ02,GP03,Pin03,GMPY06,…]



Control adversary's advantage in learning the output first

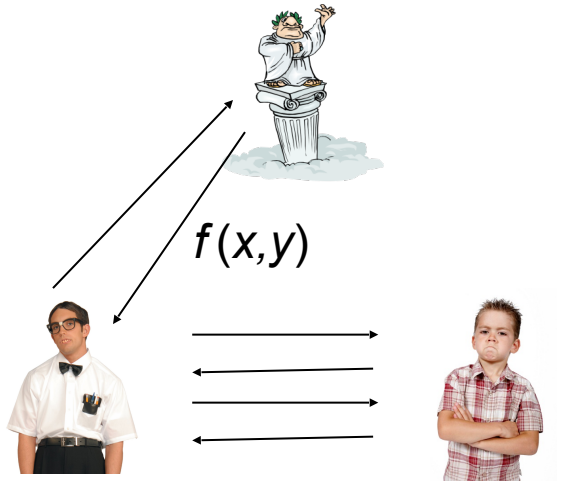
Requires lots of rounds

Workarounds II

- **Optimistic model**
  [Rab81,BGMR85,ASW97,GJM99,Mic03,DR04,KL10…]
  - Trusted arbiter restores fairness
  - Contacted only when required

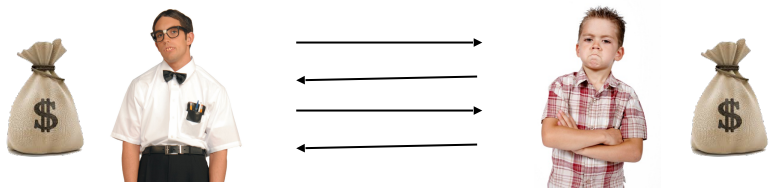
- Requires trusting a third party
- Potentially need to pay "subscription fee" to arbiter

Bad guys get away with cheating



Workarounds III

- Penalty model [ASW00,MS01,CLM07,Lin08,KL10]
  - Deviating party pays monetary penalty to honest party



- **Bad guys ~~get away with cheating~~ lose money!**

"Secure computation with penalties"

Penalty Model

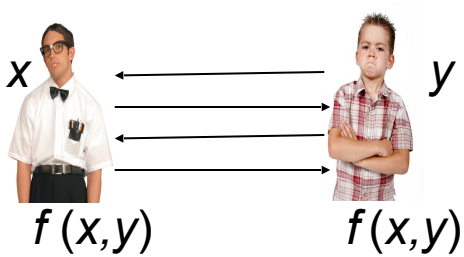
- "Legally enforceable fairness" [Lin08]
  - Requires central trusted bank
- "Usable optimistic exchange" [ASW00,MS01,CLM07,KL10]
  - Requires trusted arbiter (+ e-cash)

Ideally…

- **Decentralized system; no trusted third party**
- **Widely adopted**

Secure computation with penalties

Can Bitcoin replace a trusted bank/arbiter?



Strategy

Reduce fair secure computation to fair reconstruction

- Functionality *F'*
  - Computes output of *f*, say z
  - Secret share z into n additive shares $sh_1,\ldots,sh_n$
  - Computes commitments on shares
    - $c_i = \mathrm{com}(sh_i; r_i)$ for every *i*
  - Delivers output: $(\{c_1,\ldots,c_n\}, T_i = (sh_i, r_i))$ to party P_i
- Parties run an unfair secure computation protocol for computing F'
- Then run a "fair reconstruction" (with penalties) protocol

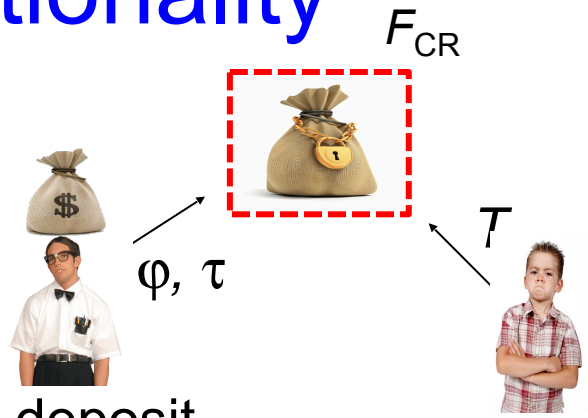
High-Level Idea (for two-party case)

- Each party deposits some money into a smart contract
- Other party can claim that money from smart contract by depositing its output share (**thus making it public**)
- Will use time-lock mechanisms to allow for refunds
- **Main Challenge:** Implement this in a way such that if a party aborts after learning the output, then other party gets money as compensation

Defining Coins

- Atomic entities
- Indistinguishable from one another
- (Unique) owner possesses given coin
- Ownership changes upon transfer
- Notation: coins(x)

Claim-or-Refund Functionality



- Accepts from “sender”
  - Deposit: coins(x)
  - Time bound: τ
  - “Checking predicate”: φ
- Designated “receiver” can claim this deposit
  - Produce witness T that satisfies φ
  - Within time τ
- If claimed, then witness revealed to ALL parties
- Else coins(x) returned to sender

Efficient realization via Bitcoin scripts

How to use Claim-or-Refund "Smart-Contract"

- Checking predicate φ: Commitment verification algorithm
- Witness: (One or more) Output share and decommitment information

Goal of fair reconstruction (with penalty)

- Honest parties must never lose money
- If a party aborts after learning output, then all honest parties must be compensated

Two Party Fair Reconstruction

$$P_1 \xrightarrow[q,\tau]{T_2} P_2$$

$$P_2 \xrightarrow[q,\tau]{T_1} P_1$$

$$P_1 \xrightarrow[q,\tau]{T} P_2$$

denotes

P_2 must reveal witness $T = (sh,r)$ within time τ to claim coins(q) from P_1

"Abort" Attack

- P_2 aborts without making its deposit but claims P_1's deposit
- Honest P_1 loses money (although it learns output)

Secure computation with penalties

- Honest parties must never lose coins
- If a party aborts after learning the output then every honest party is compensated

Two Party Fair Reconstruction

$\tau_1 < \tau_2$

$$P_1 \xrightarrow[q,\tau_2]{T_1 \wedge T_2} P_2$$

$$P_2 \xrightarrow[q,\tau_1]{T_1} P_1$$

$$P_1 \xrightarrow[q,\tau]{T} P_2$$

denotes

P_2 must reveal witness $T = (sh,r)$ within time τ to claim coins(q) from P_1

- Deposits made top to bottom
- Claims made in reverse direction
- If P_1 claims the 2nd deposit, then P_2 can always claim the 1st

Secure computation with penalties

- Honest parties must never lose coins
- If a party aborts after learning the output then every honest party is compensated

Three Party Fair Reconstruction

$\tau_1 < \tau_2 < \tau_3$

$$P_1 \xrightarrow[q,\tau_3]{T_1 \wedge T_2 \wedge T_3} P_2$$

$$P_2 \xrightarrow[q,\tau_2]{T_1 \wedge T_3} P_3$$

$$P_3 \xrightarrow[q,\tau_1]{T_1} P_1$$

$$P_1 \xrightarrow[q,\tau]{T} P_2$$

denotes

P_2 must reveal witness $T = (sh,r)$ within time τ to claim coins(q) from P_1

Malicious Coalitions

- Coalition of P_1 and P_2 obtain T_3 from P_3
- Then P_2 does not claim 1st transaction
- P_1, P_2 learn output but P_3 is not compensated

Secure computation with penalties

- Honest parties must never lose coins
- If a party aborts after learning the output then every honest party is compensated

Three Party Fair Reconstruction

$\tau_1 < \tau_2 < \tau_3$

$$P_1 \xrightarrow[q,\tau_3]{T_1 \wedge T_2 \wedge T_3} P_3$$

$$P_2 \xrightarrow[q,\tau_3]{T_1 \wedge T_2 \wedge T_3} P_3$$

$$P_3 \xrightarrow[2q,\tau_2]{T_1 \wedge T_2} P_2$$

$$P_2 \xrightarrow[q,\tau_1]{T_1} P_1$$

- P_2 claims *twice* the penalty amount
  - Sufficient to deal with malicious coalition of P_1, P_3

$$P_1 \xrightarrow[q,\tau]{T} P_2$$

denotes

P_2 must reveal witness $T = (sh,r)$ within time τ to claim coins(q) from P_1

Secure computation with penalties

- Honest parties must never lose coins
- If a party aborts after learning the output then every honest party is compensated

Multiparty “Ladder” Protocol

Order of deposits/claims

- Roof deposits made simultaneously
- Ladder deposits made one after the other
- Ladder claims in reverse
- Roof claims at the end

Roof

$$P_1 \xrightarrow[q,\tau_n]{T_1 \wedge \cdots \wedge T_n} P_n$$

$$P_2 \xrightarrow[q,\tau_n]{T_1 \wedge \cdots \wedge T_n} P_n$$

$$\vdots$$

$$P_{n-2} \xrightarrow[q,\tau_n]{T_1 \wedge \cdots \wedge T_n} P_n$$

$$P_{n-1} \xrightarrow[q,\tau_n]{T_1 \wedge \cdots \wedge T_n} P_n$$

High-level Intuition

- At the end of ladder claims, all parties except P_n have “evened out”
- If P_n does not make roof claims then honest parties get coins(q) via roof refunds
- Else P_n “evens out”

Ladder

$$P_n \xrightarrow[(n-1)q,\tau_{n-1}]{T_1 \wedge \cdots \wedge T_{n-1}} P_{n-1}$$

$$P_{n-1} \xrightarrow[(n-2)q,\tau_{n-2}]{T_1 \wedge \cdots \wedge T_{n-2}} P_{n-2}$$

$$\vdots$$

$$P_3 \xrightarrow[2q,\tau_2]{T_1 \wedge T_2} P_2$$

$$P_2 \xrightarrow[q,\tau_1]{T_1} P_1$$