A Decentralised Prediction Market

CS907 Presentation

Thomas Archbold

April 21, 2020

Department of Computer Science, University of Warwick

Introduction

Prediction Markets allow you to trade on the outcomes of future events

Prediction Markets

Prediction Markets allow you to trade on the outcomes of future events

Market prices indicate how its traders feel about the outcome of an event

Prediction Markets

Prediction Markets allow you to trade on the outcomes of future events

Market prices indicate how its traders feel about the outcome of an event

- traders have different beliefs/knowledge informing their decisions

Prediction Markets

Prediction Markets allow you to trade on the outcomes of future events

Market prices indicate how its traders feel about the outcome of an event

- traders have different beliefs/knowledge informing their decisions
- provides means of aggregating information on topics of interest via crowdsourcing ("wisdom of the crowd")

Prediction Markets allow you to trade on the outcomes of future events

Market prices indicate how its traders feel about the outcome of an event

- traders have different beliefs/knowledge informing their decisions
- provides means of aggregating information on topics of interest via crowdsourcing ("wisdom of the crowd")

Securities are traded between 0 and 100%, and the payoff is contingent upon the outcome

Prediction Markets

Prediction Markets allow you to trade on the outcomes of future events

Market prices indicate how its traders feel about the outcome of an event

- traders have different beliefs/knowledge informing their decisions
- provides means of aggregating information on topics of interest via crowdsourcing ("wisdom of the crowd")

Securities are traded between 0 and 100%, and the payoff is contingent upon the outcome

- typically, one share pays out $1 in the event the outcome occurs, otherwise $0

Traditional prediction markets are **centralised**

- trusted centre creates and closes market, manages transactions
- determines the outcome of the event and pays out accordingly

Centralised Prediction Markets

Traditional prediction markets are **centralised**

- trusted centre creates and closes market, manages transactions
- determines the outcome of the event and pays out accordingly

This restricts the types of bets that can be made

- must be explicitly offered by market maker/system
- high workload on trusted centre

Decentralised markets

A **decentralised market** allows anyone to create markets for events

- event outcome is determined by consensus among users (*arbiters*)

Decentralised markets

A **decentralised market** allows anyone to create markets for events

- event outcome is determined by consensus among users (*arbiters*)

Removes the need for a trusted centre BUT:

Decentralised markets

A **decentralised market** allows anyone to create markets for events

- event outcome is determined by consensus among users (*arbiters*)

Removes the need for a trusted centre BUT:

- bets may become ambiguous, since there is no central moderator

Decentralised markets

A **decentralised market** allows anyone to create markets for events

- event outcome is determined by consensus among users (*arbiters*)

Removes the need for a trusted centre BUT:

- bets may become ambiguous, since there is no central moderator
- arbiters may manipulate the outcome of the market

Decentralised markets

A **decentralised market** allows anyone to create markets for events

- event outcome is determined by consensus among users (*arbiters*)

Removes the need for a trusted centre BUT:

- bets may become ambiguous, since there is no central moderator
- arbiters may manipulate the outcome of the market
- users still need to bet based on their true beliefs

Decentralised markets

A **decentralised market** allows anyone to create markets for events

- event outcome is determined by consensus among users (*arbiters*)

Removes the need for a trusted centre BUT:

- bets may become ambiguous, since there is no central moderator
- arbiters may manipulate the outcome of the market
- users still need to bet based on their true beliefs

Freeman, Lahaie, and Pennock [1] introduced a mechanism that tackles these issues

Project Aims

Goal: implement a decentralised prediction market

Project Aims

Goal: implement a decentralised prediction market

In particular:

Project Aims

Goal: implement a decentralised prediction market

In particular:

- create a web-application where users may make and trade bets with virtual money

Project Aims

Goal: implement a decentralised prediction market

In particular:

- create a web-application where users may make and trade bets with virtual money
- place no restrictions on the bets that can be made

Project Aims

Goal: implement a decentralised prediction market

In particular:

- create a web-application where users may make and trade bets with virtual money
- place no restrictions on the bets that can be made
- outcome is determined by a group of arbiters (who may hold a stake in the market)

Project Aims

Goal: implement a decentralised prediction market

In particular:

- create a web-application where users may make and trade bets with virtual money
- place no restrictions on the bets that can be made
- outcome is determined by a group of arbiters (who may hold a stake in the market)

Why is this worthwhile?

- less restrictive way to crowdsource public opinion
- possibility for manipulation requires game-theoretic approach

Decentralised prediction markets based on cryptocurrencies:

Related Work

Decentralised prediction markets based on cryptocurrencies:

- *Augur*
    - reptutation system to protect against manipulation

Related Work

Decentralised prediction markets based on cryptocurrencies:

- *Augur*
  - reptutation system to protect against manipulation
- *Gnosis*
  - a distributed exchange for “tokens”
  - uses “trading rings” to trade the large numbers of illiquid bets

Related Work

Decentralised prediction markets based on cryptocurrencies:

- *Augur*
    - reptutation system to protect against manipulation
- *Gnosis*
    - a distributed exchange for “tokens”
    - uses “trading rings” to trade the large numbers of illiquid bets
- *Hivemind*

Related Work

Decentralised prediction markets based on cryptocurrencies:

- *Augur*
  - reptutation system to protect against manipulation
- *Gnosis*
  - a distributed exchange for “tokens”
  - uses “trading rings” to trade the large numbers of illiquid bets
- *Hivemind*

Predictalot was a centralised **combinatorial** prediction market

- combinatorial bids
- combinatorial outcomes

Design Overview

Mechanism

Market stage

- set up market for event X using a **market scoring rule** (MSR)
- agents trade in the market
- the market closes and trading stops

Arbitration stage

- each arbiter receives signal (e.g. reading the news) and reports the outcome
- arbiters are paired randomly and paid according to 1/prior mechanism
- outcome of the market is the fraction of arbiters reporting $X = 1$

Tools

Language: Lisp (SBCL)

Packages:

- Hunchentoot
- CL-WHO
- Parenscript
- Mito

Current Progress

- Environment setup
- Skeleton code for mechanism
- Logarithmic Market Scoring Rule
- 1/prior payment mechanism

Web server

The web server is run on Hunchentoot

```
(setf *web-server* (make-instance 'hunchentoot:easy-acceptor 8080))
(hunchentoot:start *web-server*)
```

HTML is generated using macros supplied by CL-WHO

```
(cl-who:with-html-output-to-string ...)
```

Macros are used define page templates

```
(defmacro define-url-fn ((name) &body body)
    ...)
```

We have two options to implement a prediction market:

Market stage implementation – Continuous Double Auctions

We have two options to implement a prediction market:

- Continuous Double Auction (CDA)

Market stage implementation – Continuous Double Auctions

We have two options to implement a prediction market:

- Continuous Double Auction (CDA)
- Market Scoring Rule (MSR)

Market stage implementation – Continuous Double Auctions

We have two options to implement a prediction market:

- Continuous Double Auction (CDA)
- Market Scoring Rule (MSR)

In CDAs, an order book tracks bids and asks

- if a bid matches an ask, the trade is executed
- used in traditional stock markets

Problem: low liquidity

- with few participants, there is no guarantee any bid will match an ask

CDA issues

Problem: low liquidity

- with few participants, there is no guarantee any bid will match an ask

Solution: an *automated market maker* which always assumes the opposite side of a trade

- participants are always able to trade
- BUT the “house” is likely to lose money

Scoring rules measure the accuracy of probabilistic predictions

- we use the **Logarithmic Market Scoring Rule** [2]

$$C_b(q) = b \cdot \log(1 + e^{q/b}) \tag{1}$$

where:

- q is the quantity (number of shares)
- b is an arbitrary constant ("liquidity" parameter)
  - smaller value $\Rightarrow$ higher price responsiveness

Logarithmic Market Scoring Rule

An agent wishing to buy $q' - q$ shares of a security pays $C_b(q') - C_b(q)$

- suppose $b = 10$ and there are 10 outstanding shares
- Alice wishes to buy 7 shares $\Rightarrow q' = 17$
- she pays $C_{10}(17) - C_{10}(10) = \$5.54$

An agent wishing to buy $q' - q$ shares of a security pays $C_b(q') - C_b(q)$

- suppose $b = 10$ and there are 10 outstanding shares
- Alice wishes to buy 7 shares $\Rightarrow q' = 17$
- she pays $C_{10}(17) - C_{10}(10) = \$5.54$

Share price is calculated with:

$$p(q) = \frac{e^{q/b}}{1 + e^{q/b}} \tag{2}$$

Logarithmic Market Scoring Rule

An agent wishing to buy $q' - q$ shares of a security pays $C_b(q') - C_b(q)$

- suppose $b = 10$ and there are 10 outstanding shares
- Alice wishes to buy 7 shares $\Rightarrow q' = 17$
- she pays $C_{10}(17) - C_{10}(10) = \$5.54$

Share price is calculated with:

$$p(q) = \frac{e^{q/b}}{1 + e^{q/b}} \tag{2}$$

$p(q)$ is not used to charge users, only to calculate closing price of the market

Market stage – event outcomes

The outcome of an event is not binary – rather, it takes value $\hat{X} \in [0, 1]$

- corresponds to the proportion that reported that the event occurred

Market stage – event outcomes

The outcome of an event is not binary – rather, it takes value $\hat{X} \in [0, 1]$

- corresponds to the proportion that reported that the event occurred

Two nice properties:

Market stage – event outcomes

The outcome of an event is not binary – rather, it takes value $\hat{X} \in [0, 1]$

- corresponds to the proportion that reported that the event occurred

Two nice properties:

1. each question has well-defined, unambiguous outcome

Market stage – event outcomes

The outcome of an event is not binary – rather, it takes value $\hat{X} \in [0, 1]$

- corresponds to the proportion that reported that the event occurred

Two nice properties:

1. each question has well-defined, unambiguous outcome
2. limits the influence a single arbiter can have on the value of a security

Market stage – trading fees

Trading fees imposed on the worst–case loss incurred by an agent

1. raises funds to pay the arbiters
2. bounds security prices away from 0 and 1

Market stage – trading fees

Trading fees imposed on the worst–case loss incurred by an agent

1. raises funds to pay the arbiters
2. bounds security prices away from 0 and 1

For *liquidation* transactions, no fees are imposed

- users are selling shares they own, or buying back those they have sold

Market stage – trading fees

Trading fees imposed on the worst–case loss incurred by an agent

1. raises funds to pay the arbiters
2. bounds security prices away from 0 and 1

For *liquidation* transactions, no fees are imposed

- users are selling shares they own, or buying back those they have sold

Otherwise, the user is charged

- fp for a security bought at price p
- $f(1-p)$ for security sold at price p

Arbitration stage – 1/prior mechanism

Arbiters need to be incentivised to act truthfully, since they may hold a stake in the market

This is achieved using (a modified) 1/prior mechanism:

$$u(\hat{x}_i, \hat{x}_j) = \begin{cases} k\mu & \text{if } \hat{x}_i = \hat{x}_j = 1 \\ k(1-\mu) & \text{if } \hat{x}_i = \hat{x}_j = 0 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where:

- k is an arbitrary constant
- μ is the prior probability that an agent receives a positive signal

Parameter tuning

The mechanism uses the following parameters:

- b – liquidity
- k – to pay the arbiters
- f – trading fee
- δ – "update strength", measures how strongly correlated signals are between arbiters

Once the system is running we will need to tune these to suit the market:

- number of participants, ambiguity of question, maximum budget of a user, etc.

Project Management

Change of focus

Project has changed slightly from that of the research proposal

- still looking to incorporate some aspects of it

Change of focus

Project has changed slightly from that of the research proposal

- still looking to incorporate some aspects of it

Much of the coming term will be spent revising for exams

- basic functionality not far off working, but needs to work in a web application
- bulk of the project to be done over the summer

Tasks

Once the skeleton code is finished it will need integrating into a *website*

The next major tasks will be to:

- design and implement the database to store information on all users and markets
- creating the pages through which users will interact with the system
- tuning the parameters of the mechanism

Beyond this is the (stretch) goal of computing relationships between markets

- how does a bet placed on one security affect the odds of another?

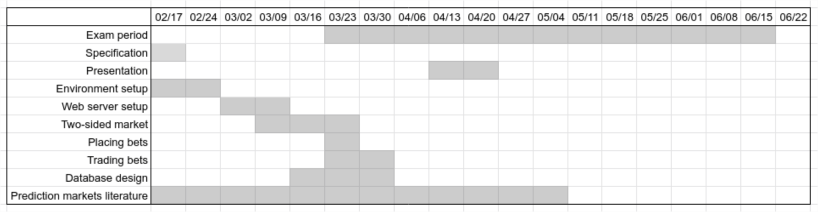

Figure 1: Timetable as of the research proposal submission

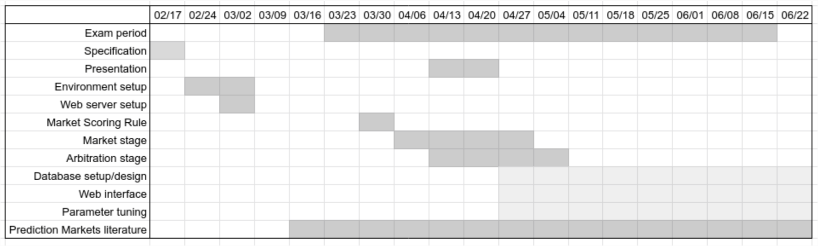

Figure 2: Revised timetable

Demonstration

Thank you for listening!

References

[1] Rupert Freeman, Sebastien Lahaie, and David M. Pennock. Crowdsourced Outcome Determination in Prediction Markets. *arXiv:1612.04885 [cs]*, December 2016. URL `http://arxiv.org/abs/1612.04885`. arXiv: 1612.04885.

[2] Robin Hanson. Logarithmic Market Scoring Rules for Modular Combinatorial Information Aggregation. page 12.