BROWN

ON INFORMATION AGGREGATION IN PREDICTION MARKETS

Luke E. Camery

Supervised by Amy Greenwald

Submitted to satisfy the requirements for the degree of Master of Science in Computer Science

May 15, 2017

1 Abstract

A prediction market is a mechanism for aggregating information. Specifically, agents' private beliefs about the likelihood of some future event. If the agents truthfully reveal their beliefs to the market, which they are incentivized to do by strictly proper scoring rules, then the market price can eventually converge to their aggregate belief. One such strictly proper scoring rule, namely the Logarithmic Market Scoring Rule (LMSR), successfully aggregates agents' beliefs to discover the rational expectations equilibrium (REE), and incurs bounded loss in so doing. The goal of this work is to calculate settings for the LMSR liquidity parameter that minimize cumulative loss, while still being constrained to discover the REE. First, we provide an intuitive heuristic for adjusting the liquidity parameter over time. Then we prove that this heuristic is optimal, which means that it minimizes cumulative loss; moreover, it is also unique.

2 Motivation

Prediction markets are exchanges that aggregate beliefs about the outcome of an event by incentivizing agents to offer their private information. Traditional mechanisms for trading prediction market securities such as the Continuous Double Auction fail to aggregate beliefs into a single price and do not provide sufficient liquidity at the typically small scale that these markets operate at. Several automated market makers have been proposed to solve this problem. The most popular is the Logarithmic Market Scoring Rule (LMSR) [1].

The Logarithmic Market Scoring Rule, while popular in the theoretical literature, faces several critical barriers to adoption. The first is that an LMSR market maker cannot earn a profit. LMSR has bounded loss, which means that the market maker knows the most that they can lose. Guaranteed loss is a non-starter for most real world applications The second is that there is no definitive algorithm for setting the liquidity parameter b. LMSR was designed to only have a single liquidity parameter during the life of a market, but other research has introduced the notion of updating b over time. The third obstacle is that there is little work seeking to understand how well LMSR aggregates beliefs according to a notion of ground truth. This work seeks to further the field on all three questions. This paper will select a notion of ground truth from the literature, the Rational Expectations Equilibrium, and attempt to devise an algorithm for setting b that recovers the Rational Expectations Equilibrium. In addition, since the liquidity parameter b, or the sequence of b's, determines the market maker's loss, this paper will find an algorithm that chooses the sequence of b's such that the selection is the least costly selection.

This work seeks an algorithm for determining the liquidity parameter in LMSR that minimizes cost while still recovering the aggregate belief according to the Rational Expectations Equilibrium.

3 Model

We denote **time** by $\tau \in \mathbb{R}_+$.

An **event** e has an outcome at time $e_t \in \mathbb{R}_+$. We restrict our attention to binary outcomes. The outcome is equal to YES in case the event occurs, and NO otherwise. We denote the YES outcome with a 1, and the NO outcome with a 0. We assume there is a way to unambiguously determine the outcome of an event.

R is an **oracle** that maps an event to an outcome. We denote the outcome of event e under R as $R(e) \in \{0, 1\}$.

Given an event e, an **option** $o = \langle o_e, o_d \rangle$ is a security that yields a return depending on the outcome of an event o_e. Each option has a direction $o_d \in \{0, 1\}$. The option will convert to \$1 at time e_t if $R(o_e)$ equals o_d. Otherwise, it converts to \$0. Two options are said to be **complementary** if they represent opposite directions on the same event.

Given an option o, an **agent** $a \in A$ has a **private belief** $a_v \in [0, 1]$ about event e, and a **budget** $a_b \in \mathbb{R}_+$. The agent's private belief a_v is the subjective probability that the agent assigns to the outcome o_e being direction o_d at strike time e_t.

A **prediction market** M is a forum for trading complementary options. Formally, a prediction market is a tuple $M = \langle o_0, o_1, A, q_0, q_1 \rangle$ where each **agent** $a \in A$ purchases either some number of o_0 or o_1 options paying the price quoted by the market maker K. We denote the total quantity purchased of o_0 and o_1 as q_0 and q_1 respectively. These quantities are typically initialized to zero.

A **market maker** K provides liquidity in a market M by offering to trade either o_0 or o_1 at a price p that it determines. Formally, K is a function that maps an option o, and a quantity $q \in \mathbb{R}_+$ at time $\tau \in [0, t)$ to a price $p_\tau(o, q) \in \mathbb{R}_+$. By definition, $p_\tau(o, q) = R(o_e)$, if $\tau \geq t$.

The **equilibrium price** P is the aggregate belief according to the Rational Expectations Equilibrium, which is defined below. P is the market price that takes all of the agents' private beliefs a_v into account. The goal of the market maker K is to recover P such that $p_\tau(o, q) = P$, if $\tau > \max_{a \in A} a_\tau$.

3.1 Assumptions

In our model we assume without loss of generality that agent $a \in A$ has an arrival time $a_\tau \in \mathbb{R}_+$ where $a_\tau < t$, and execute its strategy only once at time a_τ. We will refer to the price after all agents have participated - $p_\tau(o, q)$ where $\tau > \max_{a \in A} a_\tau$ - as the final market price.

We assume that all agents have a budget equal to 1.

We assume agents observe the current price of both options o_0 and o_1.

We assume that the market maker K and all its parameters are common knowledge.

4 Rational Expectations Equilibrium

In order to constrain our algorithm for setting the liquidity parameter b such that the final market price is equal to the aggregate belief, we first must define the aggregate belief. We will use a standard solution concept from the literature.

The **Rational Expectations Equilibrium** (REE) hypothesizes that all agents act as if they knew the private belief a_v of all other agents [2]. This fully revealing hypothesis implies that all agents would operate as if their belief were the aggregate belief constructed from the pooled information. As a result, the equilibrium price should be the aggregate belief regardless of the market mechanism.

Since REE is a perfectly competitive equilibrium concept, the aggregate belief is the budget weighted average of each of the agents private beliefs. Formally, $\frac{\sum_{i=1}^{|A|}(a_v^i a_b^i)}{\sum_{j=1}^{|A|} a_b^j}$. When agents have equal budgets, the REE reduces to: $\frac{\sum_{i=1}^{|A|} a_v^i}{|A|}$.

5 Logarithmic Market Scoring Rule

LMSR is a strictly proper scoring rule proposed by Robin Hanson [1]. A scoring rule charges agents depending on the revealed accuracy of their beliefs. A scoring rule is strictly proper if an agent has no incentive to report anything but their true belief.

LMSR is defined by a logarithmic cost function: $C(q_0, q_1) = b \ln\left(e^{\frac{q_0}{b}} + e^{\frac{q_1}{b}}\right)$. The instantaneous price of market M, which is also the market's prediction for the option $o_{\{0,1\}}$, is quoted with: $p_{\{0,1\}} = \frac{e^{\frac{q_{\{0,1\}}}{b}}}{e^{\frac{q_0}{b}} + e^{\frac{q_1}{b}}}$. The cost charged to a trader wanting to buy q_a shares of o_0 and q_b shares of o_1 is: $C(q_0 + q_a, q_1 + q_b) - C(q_1, q_1)$. Agents are charged to move the market prediction. The liquidity parameter b is set *ex ante* by the market maker. It controls how much the market maker can lose and also adjusts how easily a trader can change the market price. The market maker never loses more than $b \ln(2)$. A large b means that it would cost a lot to move the market price while a small b makes large swings relatively inexpensive.

Although b was designed to be a static parameter, recent papers discuss updating b as the state of the market changes [3]. By updating b, the market maker can change how quickly the price fluctuates and how much loss they are willing to accept. The goal of this work is to find a novel approach to setting a sequence of b's that minimizes loss and recovers the aggregate belief.

Advantages

1. Path Independence - any way the market moves from one state to another state yields the same payment or cost to the traders in aggregate [1].
2. Translation Invariance - all prices sum to unity. This ensures that the price of each outcome state maps to a probability.

Disadvantages

1. Liquidity Insensitive - for a fixed b, the market cannot adjust to periods with low or high activity. The market maker must set the liquidity parameter based on their prior belief, but has little to no guidance on how to set it [3].
2. Guaranteed Loss - the market maker cannot profit, but has a bounded loss.

5.1 LMSR Algorithms

Algorithm: cost

input : state $\langle q_0, q_1 \rangle$, liquidity parameter b, to trade q, direction $\in \{\text{true}, \text{false}\}$
output: cost
oldScore $= b\left(\exp\left(\frac{q_0}{b}\right) + \exp\left(\frac{q_1}{b}\right)\right)$
if *direction* **then**
  newScore $= b\left(\exp\left(\frac{q_0+q}{b}\right) + \exp\left(\frac{q_1}{b}\right)\right)$
else
  newScore $= b\left(\exp\left(\frac{q_0}{b}\right) + \exp\left(\frac{q_1+q}{b}\right)\right)$
return newScore - oldScore

Algorithm: price

input : state $\langle q_0, q_1 \rangle$, liquidity parameter b
output: price
quantity, price = getQuantityPrice(state, b, null, true)
return price

Algorithm: quantity

input : state $\langle q_0, q_1 \rangle$, liquidity parameter b, belief a_v, direction $\in \{\text{true}, \text{false}\}$
output: quantity q
quantity, price = getQuantityPrice(state, b, belief, direction)
return quantity

```
Algorithm: getQuantityPrice
input  : state ⟨q0, q1⟩, liquidity parameter b, belief a_v, direction ∈ {true, false}
output: quantity q, price p
if belief == null then
 |  quantity = 0
else
 |  if direction then
 |   |  price = belief
 |   |  side = q0
 |   |  top = q1
 |  else
 |   |  price = 1 - belief
 |   |  side = q1
 |   |  top = q0
 |  quantity = b log( exp(top/b) price / (1−price) ) − side
if direction then
 |  newPrice = price(q0 + quantity, q1, b)
else
 |  newPrice = price(q0, q1 + quantity, b)
return (quantity, newPrice)
```

$$\text{quantity} = b\log\left(\frac{\exp(\frac{\text{top}}{b})\text{price}}{1-\text{price}}\right) - \text{side}$$

```
Algorithm: capitalToShares
input  : market maker K = ⟨q0, q1, b⟩, money m, direction ∈ {true, false}
output: quantity
if direction then
 |  side = q0 top = q1
else
 |  side = q1 top = q0
return b log( exp( m/b + log( exp(M_q0/b) + exp(M_q1/b) ) ) − exp(top/b) ) − side
```

$$\text{return } b\log\left(\exp\left(\frac{m}{b} + \log\left(\exp\left(\frac{M_{q_0}}{b}\right) + \exp\left(\frac{M_{q_1}}{b}\right)\right)\right) - \exp\left(\frac{\text{top}}{b}\right)\right) - \text{side}$$

5.2 Agent Strategy

The Logarithmic Market Scoring rule is known to be myopically incentive compatible, which means that myopic agents bid truthfully. There is much work demonstrating that this assumption fails when agents can use **bluffing** and **reticence** to mislead other agents and profit off of that deception [4]. **Bluffing** occurs when an agent moves the market price away from their belief, and **reticence** occurs when the agent moves the market price towards its belief, but does not move it as close to their belief as possible. The following lemma shows that in our model where each agent participates only once, bidding myopically is dominant.

Lemma 5.1. *Given that agents participate only once, the myopic strategy is dominant.*

PROOF. Consider a non-myopic strategy that is dominant for an agent in a market scoring rule based market. The non-myopic strategy must either move the price p towards their belief a_v, but less than their budget a_b permits, or away from it. In the case where the non-myopic strategy moves the price away from their belief, the agent will profit based on how much their price movement improves the market price. In order for the agent to make positive profit, the market's price needs to move closer to the underlying true probability. If the agent thinks that this non-myopic strategy is improving the price then they cannot hold belief a_v, which is a contradiction. Similarly, if the agent holds belief a_v then it would be strictly preferable to move the price as close as possible to a_v rather than simply closer. This means that the myopic strategy strictly dominates the non-myopic strategy. Since in both cases the myopic strategy strictly dominates therefore the myopic strategy is dominant under this model. ■

Therefore, it is safe to assume that all agents will behave myopically.

6 Problem Statement

The liquidity parameter b both controls how quickly that agents can change the market price and bounds the loss that the market maker can incur. Our algorithm will deal with this trade-off by minimizing the market maker's loss while constraining the market's final price to equal the aggregate belief within a small margin of error.

6.1 Markov Decision Process

The following Markov Decision Process (MDP) finds the optimal policy given the aforementioned trade-offs. Formally, the MDP will find a loss minimizing market maker whose final market price is equal to the REE aggregate belief.

6.1.1 Definitions

An MDP $M = \langle \mathcal{S}, \mathcal{A}, T, R \rangle$ is defined by a set of states, an action space $\mathcal{A}$, a deterministic transition probability function $T : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$, and a reward function $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$.

6.1.2 Construction

We construct an MDP to model the cost of the paths of b's through the MDP that result in the final price equaling (close to) the equilibrium price P, which is derived from the Rational Expectations Equilibrium. We assume the order in which agents arrive to trade is known and fixed as $\langle a_0, a_1, ..., a_n \rangle$, where $a_i \in A$ and a_0 is a dummy agent with belief .5. We assume agents are truthfully reporting their beliefs a_v as LMSR incentivizes them to do.

Given an LMSR market maker, we construct the MDP as follows.

The set of states is $S = \{s_0, s_1, ..., s_n, \text{YES}, \text{NO}, \text{END}\}$, where $s_i = \langle q_0, q_1 \rangle$ for $q_0, q_1 \in \mathbb{R}_+$, respectively, the total quantity of YES securities and total quantity of NO securities sold by the market maker. There are two special states YES and NO. There is also an absorbing state END.

The set of actions is $\mathcal{A}(s_i) = [b_0, b_1]$, where b_0 and b_1 represent the range of valid liquidity parameter values, determined by b. The set of actions $\mathcal{A}(\text{YES}) = \mathcal{A}(\text{NO}) = \mathcal{A}(\text{END}) = \emptyset$.

We define $q_0' = q_0$ + getQuantity($\langle q_0, q_1\rangle$, b, a_i, YES), and $q_1' = q_1$ + getQuantity($\langle q_0, q_1\rangle$, b, a_i, NO).

Transitions are deterministic. When $i = 0, \ldots, n-1, T(s_i, b) = \langle q_0', q_1'\rangle$, where $s_i = \langle q_0, q_1\rangle$. When $i = n$,

$$T(s_n) = \begin{cases} \text{YES} & \text{if R}(o) = 1 \\ \text{NO} & \text{if R}(o) = 0 \end{cases}$$

$T(s_{n+1}, \text{YES}) = T(s_{n+1}, \text{NO}) = \text{END}$. As T is absorbing, $T(s_i, \text{END}) = \text{END}$.

Rewards are defined as follows: when $i = 0, \ldots, n$, $R(s_i, b) = \text{cost}(q_0', q_1') - \text{cost}(q_0, q_1)$, where $s_i = \langle q_0, q_1\rangle$, and q_0', q_1' are defined above. In addition,

$$R(s_n, \text{YES}) = \begin{cases} -q_0 & \text{if getPrice}(\langle q_0, q_1\rangle, b) = P \\ -\infty & \text{otherwise} \end{cases}$$

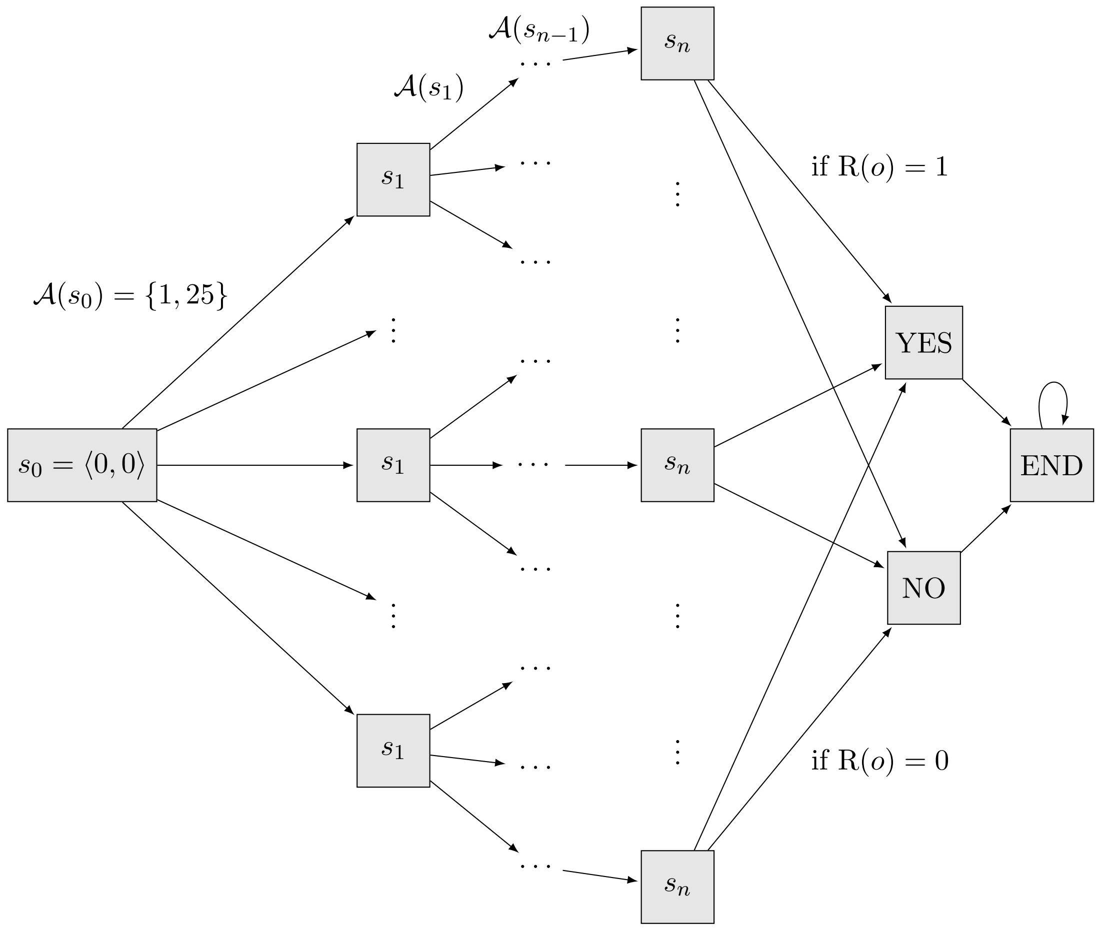
$$R(s_n, \text{NO}) = \begin{cases} -q_1 & \text{if getPrice}(\langle q_0, q_1\rangle, b) = P \\ -\infty & \text{otherwise} \end{cases}$$

$$R(s_{n+1}, \text{END}) = 0 \ .$$

6.1.3 Diagram

The following is a graphical representation of our Markov Decision Process. The number of intermediate stages n corresponds to the cardinality of the agent set $|A|$. The state s_0 represents the initial quantities chosen by the market maker before any agents arrive.

The parameter that our solution can vary is the sequence of b's used. A parameter that we will hold as fixed for now is the initial state s_0.



6.2 Assumptions

We will use a discretization factor of $\delta = \{.25, .5, 1\}$ for our action space of b values. Since b is undefined for $b \leq 0$, we restrict our action space to $\mathcal{A} = \{\delta, ...\}$, but we will only search over $\mathcal{A} = \{\delta, 100\}$ for efficiency.

Due to finite precision, we will permit a tolerance of ϵ for the final market price such that $|P - \text{getPrice}(\langle q_0, q_1 \rangle, b)| < \epsilon$.

7 Online Learner

One algorithm is straightforward: for meeting belief aggregation, ensure that after each agent has participated, the market price reflects the aggregate belief of all the agents that have participated so far.

7.1 Background

Since b controls how quickly agents can move the market price, a minuscule b value permits an agent to set the market price to their belief at almost no cost while a large value of b makes moving the market price to their belief almost infinitely expensive. Agents with unit budgets, per our model, can move the market price to their belief without exhausting their budgets when $b \leq 1$. Similarly, agents with unit budgets cannot move the price more than a rounding error when $b \geq 100$. Since the LMSR cost function is both continuous and differentiable on that interval, the Mean Value Theorem implies that b can be set to recover any value on the interval $[p_t, a_v]$. Agents will always exhaust their budgets moving the market price as close as possible to their beliefs.

When operating on the real line, the definition of an arithmetic mean of arbitrary values a and b constrains the result to the interval $[a, b]$. We can utilize this fact to ensure that the market price is always equal to the arithmetic mean of the agent beliefs that the market maker has seen so far. Formally: $p_t = \frac{\sum_{i=1}^{t} a_v^i}{t}$, $\forall t \in \{1, |A|\}$.

7.2 Algorithm

The Online Learner operates by selecting a minuscule b for the first agent that arrives. After the first agent has participated, the price will be equal to first agent's belief. For all periods afterwards, the algorithm will receive the new agent's belief, update its aggregate belief by computing the new average accounting for that agent, and solve for b such that the new agent will exhaust its budget by moving the market price to the new aggregate belief. A trivial way to solve for b is by conducting a binary search on the interval $(0, 100]$. If multiple b values solve the system then the Online Learner selects the smallest in order to minimize cost.

The following is an implementation of the Online Learner that for clarity does not use binary search:

Algorithm: setB

input : state $\langle q_0, q_1 \rangle$, price p, new belief a_v, old average a, time τ
output: liquidity parameter b
newAverage = $\frac{a\tau + a_v}{\tau + 1}$
newB = ϵ
while *true* **do**
    direction = newBelief > p
    desiredShareNumber = quantity($\langle q_0, q_1 \rangle$, newB, a_v, direction)
    if *direction* **then**
        desiredCost = cost(desiredShareNumber, 0)
    else
        desiredCost = cost(0, desiredShareNumber)
    if *desiredCost* > a_b **then**
        desiredShareNumber = capitalToShares($\langle q_0, q_1 \rangle$, newB, a_b, direction)
    newState = $\langle q_0, q_1 \rangle$ + desiredShareNumber
    newPrice = price(newState, newB)
    if *newAverage* $\neq$ *newPrice* **then**
        newB = newB + ϵ
    else
        return newB

8 Results

The outcome of Value Iteration on this Markov Decision Process reveals that the cost minimizing policy converges to the Online Learner algorithm. This result comes from experiments on five different instantiations of the MDP. Three of the MDPs contained two agent beliefs and the other two contained four agent beliefs. All agent beliefs were randomly sampled. Each MDP was then evaluated using $\delta = \{.25, .5, 1\}$. For every combination, the b path selected by Value Iteration was within δ from the solution that the Online Learner selected.

This outcome was contrary to intuition, which suggested that the greedy policy of the Online Learner would be overly costly in order to meet the constraint every round when it only needs to meet the constraint after the last round. In actuality, however, the running average turns out to be the cost minimizing selection after each time step in expectation. Empirically, the Online Learner loses between 50% and 80% of the loss bound for LMSR depending on our parameters.

8.1 Cost Minimizing Policy

The fact that the Online Learner algorithm is the cost minimizing policy becomes apparent when examining all possible paths that result in the final market price equaling the equilibrium price. In order to arrive at the equilibrium price, the equilibrium price must be between the penultimate price and the final agent's belief a_v^τ. This is a trivial result of the arithmetic mean and myopic agent behavior. The range of valid b settings holds using that logic via backward induction for each time step from the end time t to the origin $\tau = 0$.

Conjecture 8.1. *The Online Learner finds the minimum cost path where the final market price equals the equilibrium price.*

Consider another algorithm that satisfies the validity condition, which would also ensure that the penultimate price is the farthest price from the equilibrium price. Similarly, another valid algorithm would ensure that the penultimate price is already equal to the equilibrium price. The trade-off between these two extreme algorithms is that the more an agent is able to move the market price, the more shares that they buy and therefore the more liability that the market maker must take on. In order to limit how far an agent can move the market price, however, the market maker must increase b which also increases the market maker's liability. The mean of the beliefs aggregated at time τ is the optimal point between those two extreme strategies that minimizes the market maker's cost of ensuring that the final price will equal the equilibrium price.

8.2 Unique Solution

In addition to being the minimum cost policy when the agents beliefs and order are known *a priori*, the Online Learner is the only algorithm which guarantees that the final market price equals the equilibrium price when the agents' beliefs are not known *a priori.*

Theorem 8.2. *The Online Learner is the unique solution where the final market price equals the equilibrium price when the agents' beliefs are not known* a priori.

PROOF. Consider the penultimate market price p, the average that is known to the market maker $a = \frac{\sum_{i=1}^{|A|-1} a_v^i}{|A|-1}$ before the final agent arrives, and the final agent's belief $v = a_v^{|A|}$, which is unknown to the market maker but is not equal to a. The equilibrium price P would therefore be $v > P > a$ or $v < P < a$.

Consider three cases for the penultimate market price p where $p < a$, $p > a$, or $p = a$. If the market maker utilized a strategy such that $p < a$ then $\exists v$ such that $v < p < P < a$, which would result in there being no choice of b that would result in the final market price equaling P. If the market maker utilized a strategy such that $p > a$ then $\exists v$ such that $v > p > P > a$, which would result in there being no choice of b that would result in the final market price equaling P. If the market maker utilized a strategy such that $p = a$ then $\forall v$, $v > P > p = a$ or $p = a > P > v$, which would result in there existing a choice of b that would result in the final market price equaling P.

Therefore the only strategy that is guaranteed to succeed is where $p = a$, which is the Online Learner's algorithm.■

8.3 Optimal Strategy

Hanson demonstrates that a market maker using LMSR loses the difference between the score of the initial distribution, which is set by the market maker using the initial quantities s_0, and the score of the final distribution [1]. This is due to Path Independence [3]. Under the assumption that the aggregate belief is known *a priori*, we can show that the Online Learner is cost less provided that the market maker initializes the price to the aggregate belief. The market maker can do this by altering s_0 from zero quantities to some other $\langle \mathbb{R}_{0+}, \mathbb{R}_{0+} \rangle$.

Theorem 8.3. *When the equilibrium price is known* a priori*, the Online Learner is a cost less solution where the final market price equals the equilibrium price if the equilibrium price is set as the initial price.*

PROOF. Theorem 9.2 proves that the Online Learner recovers the equilibrium price even if the agents' individual beliefs and order are unknown. Hanson proves that the total cost of aggregating $|A|$ reports is equal to the cost of moving the initial price to the final price [1]. Since the Online Learner guarantees that the final price is equal to the equilibrium price and can set the initial price to the equilibrium price, the initial price is therefore equal to the final price. Therefore the Online Learner is cost less since the initial price equals the final price. ■

9 Conclusion

This paper presented a novel algorithm for setting the liquidity parameter in LMSR that achieves several desirable outcomes, namely aggregating beliefs to a defined equilibrium and minimizing the market maker's cost. In addition, it contributed to the literature about LMSR's ability to aggregate information by showing properties of the minimum cost path given an equilibrium solution concept. Furthermore, it demonstrated that an optimal algorithm is achievable without knowing the agents' beliefs and order. This is an important result given that in practice market makers have little information about agents' beliefs and arrival times.

Future research could attempt to extend the Online Learner so that it earns positive profit since the bounded loss condition is a barrier to the practical adoption of LMSR. Additional research could extend this work to situations where agents' beliefs are not drawn from a singular distribution. The most immediate solution would be to charge a variable commission for each transaction that would be some percentage of each agent's cost. The Liquidity Sensitive Automated Market Maker provides a natural example since it extracts a commission from the liquidity that it provides to agents [3].

10 Acknowledgements

I wish to thank Professor Amy Greenwald for the years of mentorship and advice that lead to this work. I also want to thank Enrique Areyan Viqueira for all of his help working through problems with the model, with latex, and with our friend, Java.

References

[1] R. Hanson, "Combinatorial information market design," *Information Systems Frontiers*, vol. 5, no. 1, pp. 107–119, 2003.

[2] C. R. Plott and S. Sunder, "Rational expectations and the aggregation of diverse information in laboratory security markets," *Econometrica*, vol. 56, no. 5, pp. 1085–1118, 1988.

[3] A. Othman, D. M. Pennock, D. M. Reeves, and T. Sandholm, "A practical liquidity-sensitive automated market maker," *ACM Trans. Econ. Comput.*, vol. 1, pp. 14:1–14:25, Sept. 2013.

[4] Y. Chen, D. M. Reeves, D. M. Pennock, R. D. Hanson, L. Fortnow, and R. Gonen, *Bluffing and Strategic Reticence in Prediction Markets*, pp. 70–81. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.