Upshot One: A Question & Answer Protocol

DRAFT v0.2

Nicholas Emmons, Kenneth Peluso

Upshot

August 2020

Abstract

Reaching consensus on answers to subjective questions is, today, practically achievable via trusted sources or expensive mechanisms. This greatly limits the potential of many applications (e.g. dapps, content moderation, insurance, fact-checking). We propose a protocol for solving this problem that combines cryptographic sortition, mutual information games, and subjectivocracy. In our protocol, agents of a decentralized network stake capital to answer questions. A random subset of these agents is scored based on a recently invented peer prediction mechanism called Determinant based Mutual Information. This mechanism is dominantly truthful for a constant number of questions and agents, so it can efficiently yield a plausible estimation of correct answers. Controversial answers may incite forks in a native token and history of answers to reflect differing views of reality. Application-specific fork-choice rules may then be used to determine the correct reality. This paper explicates the above protocol and analyzes its robustness and security.

1 Introduction

Many applications rely on accurate information to function. When this information is subjective in nature or when its veracity is difficult to ascertain, assessing its quality is difficult or impossible. Settings where this is so include decentralized applications that rely on external information (i.e. the oracle problem [Hog]), non-parametric insurance claims processing, and identifying misinformation on social media. Existing solutions to this problem are either expensive[1], rely on trusted actors[2], have high latency[3], focus exclusively on continuous outcome spaces[4], or require questions to already possess a "ground truth"[5].

Our main contribution is a protocol – Upshot One – that can, without assuming a preexisting ground truth, efficiently supply accurate answers to questions with discrete answer spaces. In contrast to existing approaches to onboarding information, our protocol achieves a dominant strategy of informed truth telling by using efficient mutual information games rather than inefficient majority-rule schemes, which require encumbering economic games to achieve the same truth telling equilibrium (e.g. escalation games [Pet+], voting [Cléa]). After outlining this protocol's structure (Section 2), we rebuild it from scratch to best motivate each component's existence (Sections 3-11), provide an analysis of its robustness and security (Section 12), discuss outstanding risks (Section 13), and list some potential applications (Section 14).

2 Structure

At a high-level, our protocol operates as follows:

1. **Asking:** Anyone submits a question (Section 3) they would like to have answered.

[1] For example, democratic elections, which ask the subjective question, "Who should fill X office?" are expensive and cumbersome.
[2] See Provable [Pro] or Facebook's Forecast [Blo]. The latter relies on Facebook's ability to honestly prequalify "experts."
[3] This includes many voting schemes.
[4] This includes decentralized price oracles such as UMA [Mec], Uniswap [Unia], and Chainlink [Chaa].
[5] This includes *objective-consensus mechanisms* such as Nakamoto Consensus [Nak].

2. **Categorization:** Questions are categorized into groups (Section 5).
3. **Staking:** Agents stake capital (Section 6) toward answering categorized questions.
4. **Answering:** Agents answer questions.
5. **Committee:** A committee of these staked agents is randomly selected (Section 6).
6. **Scoring:** Agents' answers are scored using a recently invented peer prediction mechanism that scores agents according to the informativeness of their answers (Section 4).
7. **Resolution:** If the protocol is confident in its elicited answers, they are associated to their respective questions. Selected agents are paid according to their scores (Section 4).
8. **Forking:** If an agent disagrees with a resolution, they can fork it by sending native tokens (Section 11) to those they disagree with in exchange for newly created, forked native tokens (Section 9).
9. **Fork-Choice:** Applications choose the fork that they think represents reality (Section 10).

This full sequence may not be fully realized for a number of reasons (e.g. conditions are not met, some steps are optional). To explain such nuances and motivate each component's existence, we endeavour to iteratively rebuild our protocol from scratch. Haunting the entirety of this construction is the setting in which our protocol exists – a trustless, permissionless environment – so we cannot assume any relationship between agents aside from those endogenous to the protocol. We begin by defining the objects of central concern whose accurate resolution necessitates our entire protocol: questions.

3 Questions

Our journey begins whenever a user (henceforth, *asker*) seeks information. To this end, they create a question, which is the full set of information necessary for others to provide answers. This set includes:

- The **domain** of answers to the question
- The **reward** tied to the question
- Any additional **context** that is useful for answering the question
- The minimum **quorum** that must be met for the question to be considered answered
- The **category** this question belongs to
- An optional **filter** specifying agents that are eligible to answer the question

The domain for the question specifies the possible answers for the question. The reward is the amount of capital sent with the question to compensate agents. Any additional context required to answer the question is included in the "context" field, a string whose interpretation is left open-ended. Quorum specifies a minimum amount of capital that must be staked to a question group (see Section 6) for the question's *resolution* (i.e. its association to an answer) to be valid. The category is set by the protocol, not by the asker, prior to the question's resolution (see Section 5).

In creating a question, askers may desire granular control over which agents can answer it. To do this, the asker specifies logic that determines an agent's eligibility to answer a question in a `checkEligibility(agentId)` $\rightarrow [0, 1]$ function held in an external contract. The address where this eligibility logic exists is then stored in the question object as its filter. This function returns a value greater than 0 for each *eligible agent*, or an agent whose answers (and stake) is considered. Filters enable a number of powerful extensions to the protocol. For example, one could maintain a graph of agents' responses and find its Personalized PageRank to incorporate a notion of reputation into resolutions[6]. Alternatively, an asker conducting user research for a new product may wish to whitelist a private cohort of "beta testers." When no filter is provided, every agent is eligible.

[6]Nodes may represent agents and weighted edges may represent the amount of shared capital that agents have earned from answering questions together. An off-chain PageRank instance may be fully personalized [Pet] to a temporary node representing the question, and directed edges pointing from this question node to existing agent nodes may be a function of each agent's respective stake toward the question. In such a setup, PageRank would be more likely to settle in nodes with established histories of well-received work alongside peers. The output of `checkEligibility` may be informed by these PageRank values.

4 Scoring Rounds

Our ultimate task is to promote the association of questions to their accurate answer. Anonymous *agents* can submit answers at will, however, the need to distinguish between true and false answers remains unmet. What we need is a way to coordinate these agents such that the true answers among their submissions are efficiently identifiable. *How do we incentivize researched, truthful reporting while resisting exploitation and spam?*

Fortunately, there exists an untapped, nascent academic field that is directly relevant to the problem at hand. *Peer prediction* studies methods promoting truthful responses from agents where there is no way to verify the quality of said responses. These mechanisms ensure that answering questions honestly is the most profitable choice in equilibrium – more profitable than answering maliciously or apathetically. Peer prediction mechanisms apply regardless of whether a ground truth is too costly to query or is simply nonexistent, hence why they are powerful tools for answering fundamentally subjective questions. [Man+][Kon]

The specific peer prediction mechanism Upshot One employs is the *Determinant based Mutual Information Mechanism (DMI-Mechanism)*[7]. DMI-Mechanism exploits the fact that every agent's answer is related to their peers' answers and therefore answers can be used to check themselves. In particular, it asks every agent the same set of questions and pays them according to the mutual information between their answers and their peers' answers. Payment is an unbiased estimator of an information-monotone measure of mutual information that generalizes Shannon's mutual information and ensures that any "data processing" of random variables decreases their mutual information[8]. As a result, DMI-Mechanism achieves some desirable properties: [Kon]

1. **Dominantly-truthful:** Truth-telling is the dominant strategy.
2. **Informed-truthful:** Truth-telling is strictly better than uninformative strategies.
3. **Minimal:** Agents are just required to submit reports of their beliefs and not said reports plus predictions regarding other agents' beliefs.
4. **Detail-free:** It does not require any prior knowledge to be inputted by the mechanism designer.
5. **Constant:** Only a constant number of questions and agents are required for it to be both dominantly-truthful and informed-truthful.

We reserve the exact mechanics of DMI-Mechanism to the paper that created it, [Kon]. Insofar as we must know now, ≥ 3 agents are tasked with answering $\geq 2C$ of the same questions (where C is the question domain's cardinality) that are mutually a priori similar (see Section 5), and each agent receives a normalized score based on the information gained from their answers relative to others. Scores can be likened to a judgement of an agent's informativeness or merit because DMI-Mechanism guarantees that informed truth-telling is the most profitable strategy.

Abundant value is extracted from scores. First, they are used to pay agents. Any linear map can be applied to agents' scores to yield their payments, and this secures DMI-Mechanism's properties [Kon]. For example, if R is the sum of question rewards, n is the number of agents who answered, the average score is $\bar{s}$, agent i scored s_i, and if scores are always bounded within $[a, b]$, then agent i's payment is $p_i := \frac{R(s_i - \bar{s} + b - a)}{n(b-a)}$. $\sum_i p_i = R$, $p_i \geq 0$, and p_i is linear in s_i, so the incentives of DMI-Mechanism remain intact. Second, scores can be used to determine an amount of assurance in each possible answer. In other words, any aggregation of submitted answers can be weighted by their respective agents' scores at an onlooker's discretion (e.g. simply the answer submitted by the agent with the highest score)[9].[10]

[7] This component, like much of the others comprising our protocol, is modular such that it can be swapped for another scoring mechanism at one's discretion. The mechanism described here is simply our preference.

[8] In other words, the more information you add, the more you are paid, and if you use fancy statistics to guess other agents' answers, then your score may be lower.

[9] Any such finality gadget for resolutions or calculation of p_i can be included in within Upshot One or in a higher layer by anything that reads Upshot One. This feature exemplifies the general modularity and architectural fluidity allowed by Upshot One.

[10] Note that a question's resolution and payment to participating agents occur only if the total amount of stake among all agents exceeds the question's quorum. This provides askers and onlookers a rough assurance of confidence among agents; that the confidence agents had in their submitted answers, as proxied by their stakes, was beyond the question's quorum.

5 Categorizing Questions

The promise of DMI-Mechanism is (temporarily) marred by a lingering question: *How do we deal with mutually a priori similar questions?* DMI-Mechanism requires that questions "look similar" to one another i.e. are *mutually a priori similar*. We must focus special attention to this constraint, for it presents a significant limiting factor: Not only do agents need to answer the same questions, but these $\geq 2C$ questions must be of a particular kind – they must be similar to one another[11]. Our task is then to categorize questions such that question groups could be formed of only mutually *a priori* similar questions.

The naïve solution – conscripting askers into knowing a question's category in advance – is error-prone. Considering the wealth of possible categories in existence, assigning questions to categories is likely a cognitively arduous task. Even if askers had efficient awareness of each category, their categorizations could still be wrong, tarnishing the resolutions of other questions found in the same group. Regardless of how it occurs, miscategorizing a question breaks a fundamental assumption of DMI-Mechanism and thus jeopardizes its many guarantees.

There exists an alternative, elegant solution for determining categories that requires no new infrastructure. When a question is first created, it is automatically placed in a dedicated *category-free group*. Agents resolve this group with DMI-Mechanism as usual, however, the question every agent answers is *"Is uncategorized question X similar to a different, categorized question Y?"* The trick is in noticing that questions of whether or not two preexisting questions are a priori similar are themselves mutually a priori similar. The prior probability of "Yes, these questions are similar" is the marginal probability of "Yes, these questions are similar" over the support of all possible pairings of questions. This is constant for all such "pairing questions," therefore pairing questions are mutually a priori similar. The results of these scoring rounds assign questions to their appropriate categories, bringing them one step closer to their resolution. Thus, DMI-Mechanism allows us to satisfy its own prerequisite conditions.[12]

6 Groups and Committees

After mutually a priori similar questions are identified, they are resolved with DMI-Mechanism. To this end, we introduce *question groups*, or the sets of questions agents answer[13]. A question group is fully, uniquely defined by a category and a filter; if questions have the same category and filter, they belong to the same question group. Eligible agents are required to stake to question groups before they submit answers to questions therein. Those that do stake are called *participating agents*. If they fail to submit enough valid answers, their stake may be revoked and redistributed to other agents. Otherwise, their stake is returned after a scoring round. Agents can unstake from a group at any time except when they are in a committee or when a previous scoring round they participated in is forked (see Section 9). After an agent stakes, they submit salted, hashed answers that they later reveal in a scoring round. This ensures that agents' answers to unresolved questions do not influence one another. These hashes are signed off-chain and submitted in bulk on-chain to minimize transaction costs.

At any time, an aggregator (see below) can trigger a random sampling of three participating agents, weighted by their stakes to the question group, to form a *committee*. The committee's answers are scored by DMI-Mechanism and may resolve a scoring group. We sample agents *with replacement* because it is computationally cheaper and devalues sybil attacks, because an attacker gains nothing when they distribute capital across multiple sybils.

We have questions, answers, and agents but they are not yet organized into a *scoring group*, or a data set compatible with DMI-Mechanism. We say that agents *overlap* if they have submitted answers toward the same $\geq 2C$ questions, so we task a new actor, *aggregators*, with randomly finding the largest set of questions on which the committee mutually overlaps. They search randomly to ensure that the composition of scoring groups, and thus the outcome of a scoring round, is unpredictable. If a scoring group cannot be found, (e.g. committee members have not answered the same $\geq 2C$ questions), the scoring round fails and reverts. Additional measures may be taken to limit the frequency of failures (e.g. committee members' stakes may be revoked and the aggregator may be punished[14]). Analysis of these additional measures is left to a forthcoming paper.

[11] This limitation can be qualified as a favorable trade. DMI-Mechanism does not assume that agents hold a common prior on a question's answer, so instead of assuming that "every participating agent thinks the *same thing* about a question," it asserts a weaker assumption, that "every participating agent is thinking something about the *same kind (category)* of question."

[12] Applications atop Upshot One have full liberty to seed categorizations (which questions to compare first), maneuver through category space (which questions to compare next), and define stopping criteria (when to stop categorization and start resolution).

[13] Note that a category-free group is a type of question group.

[14] These measures would further incentivize agents to answer questions and aggregators to find scoring groups.

These innovations are desirable for a number of reasons. Agents need not submit an on-chain transaction per answer; they can submit one transaction per group, limiting their transaction costs from $O(C)$ to $O(1)$. Aggregators need only parse a single, limited domain of questions – just those in a question group answered by committee members – instead of an unwieldy universal cross set of questions and agents. Aggregators also cannot censor answers because they are provably associated to questions on-chain. Limiting scoring to a committee of three agents limits the computational cost of the scoring round. Finally, committee selection by sortition is secure. Without sortition, an attacker can be certain that if they stake enough, their answers will be scored. With sortition, an attacker must afford an even larger stake (and its associated risk) such that they can guarantee their dominance of a committee with high probability (see Section 12).

7 Non-Overlapping Questions

Whereas *overlapping agents* submitted answers to the same $\geq 2C$ questions, *overlapping questions* received ≥ 3 answers. Note that we need not mandate that all questions be overlapping to run DMI-Mechanism. Recall that participating agents must mutually admit $\geq 2C$ overlapping questions for the guarantees of DMI-Mechanism to hold. Beyond that, any number of *non-overlapping questions* answered by committee members can be simultaneously resolved, improving the latency of resolutions. Their inclusion is at the aggregator's discretion.[15]

8 Agent Records

Answers from all participating agents are stored in an *agent record*, which enables past answers to be batched with new answers in current scoring rounds. An answer can be accepted and logged at any time, but its agent is qualified for rewards only if they are an eligible agent and if the associated question is unresolved. The agent record lowers the amortized work of agents per scoring round and makes attacks more costly. An attacker must now consider not only anticipated answers for unresolved questions, but also an agent's entire record of answers, because some of them may be appended to a scoring group (see Section 12.2). Agent records also aid in Upshot One's security measure of last resort: forks.

9 Subjectivocracy

What is still lacking is protection against actualized existential threats, namely a protocol-compromising attack or a genuine disagreement about a question's resolution. We use subjectivocracy to mediate such situations. *Subjectivocracy* ("forking-as-governance") is a noncoercive form of governance that allows people to choose which *reality* (series of resolutions) they believe is true [Bute]. Our design is largely inspired by [Bute].

In Upshot One, a distinct instance of a native protocol token exists for each distinct reality. For example, there may be a token for which the question "Did 'Smart Contract X' get hacked?" is resolved "yes" and another other in which it is resolved "no." We say that these are *resolutions in dispute*. The tokens are mutually distinguished via an ID field whose values uniquely represent distinct realities. One may wish to switch their balance between different forks, which we call *(fork) migration*. Migration is important because its possibility, in part, empowers the Lamarckian aspects of subjectivocracy that help make subjectivocracy attractive; forks experiencing the most demand for their use (the "fittest") should thrive the most. Hence why we elect the following scheme.

Say a native token holder (a *migrant*) wishes to migrate from fork A to fork B. If they were not a committee member of any resolution in dispute and own Z A tokens, then they already own Z B tokens[16]. Conversely, if the migrant was part of a resolution in dispute where they received X A tokens, then they do not have any B tokens, but by sending $X' \leq X$ tokens to those committee members they disagree with, they receive $Y > X'$

[15] The extent to which non-overlapping questions can be used is a function of obscurity. If an attacking agent knows that a question has received few answers and from low-stake agents, they may accurately posit that the question will either not be in an upcoming scoring group or be included as a non-overlapping question. In either case, they can provide an uninformed answer without consequence. This risk can be mitigated with cryptographic infrastructure (e.g. [app]; which blurs the actions of agents and aggregators), high asking and answering rates (which complicate the identification of non-overlapping questions), and well-designed aggregator incentives (which encourage secure scoring group formation).

[16] All queries of past information are pulled from the agent record (Section 8) and the record of questions (Section 3).

B tokens[17], where the difference $Y - X'$ is taken from the would-be B token balances of the remaining A holders. This ensures that the migrant is rewarded for migrating earlier than other holders in the event that B is ultimately correct. Simultaneously, this further rewards those other committee members if A is ultimately correct. Overall, this scheme instigates a profit motive for early-adopters of the ultimately most in-demand fork, which is exactly what we sought to establish – market forces determine the value of forked native protocol tokens. Importantly, this scheme does not require all native token holders to participate in the forking process.[18]

10 Fork-Choice Rule

Subjectivocracy bolsters our protocol's adaptability in face of otherwise irreconcilable disputes, but forking alone offers little finality. Simply put: If we allow our protocol to fork, which fork do we choose? This decision is ultimately dealt with on the application-layer, above Upshot One. There, applications employ fork-choice rules that align with their respective perspectives on reality.

Fork-choice rules are used in forkable-environments (e.g. blockchain and git protocols) to determine which reality among many is *canonical* ("the most generally accepted version")[19]. This definition deliberately excludes those intra-protocol mechanisms – such as the "ultimate appeals" of Kleros and Augur [Cléb] – that coerce entire arbitration protocols into endorsing a *single* reality. Such forced endorsement is not desired in arbitration protocols – such as Upshot One – that resolve subjective questions, where there may be multiple, equally valid realities. Hence, non-coercive fork-choice rules should be adopted.

We recommend the adoption of the **Time-weighted average Fork-choice rule (TF)**. TF is simple: *Choose the fork of greatest time-weighted average price (TWAP).* It is similar to the "longest chain rule," where the reality formed with the greatest amount of provable work is considered canonical [Wal][Bute]. In TF, which we now detail, the reality with the most demand is considered canonical.

Recall that every forked, distinct reality in Upshot One uniquely defines a new instance of a native protocol token (henceforth, ONE). Any such instance can be put on an exchange or, more generally, on a service that enables its price's discovery. All instances' prices should be expressed in a common unit (e.g. USD). Define S as the set of ONE instances whose prices have been discovered, are denominated in a common unit, and do not have descendants (are leaves in the DAG of realities). After an arbitrary interval of time, we query a reliable price oracle to fetch the prices of each ONE instance $a \in S$. This updates its TWAP T^a:

$$T^a = \frac{P^a_{t_i} - P^a_{t_{i-1}}}{t_i - t_{i-1}}$$

[Unib] where t_i is the ith timestamp for $i \in \mathbb{N}$ and P^a_t is the price of a at time t. t_i is strictly increasing in i and P^a_t arbitrarily fluctuates on t. The canonical fork is associated with ONE instance $a^* = \arg\max_{a \in S} T^a | (t_i)_{i \in \mathbb{N}}$. We analyze the robustness and security of TF in Section 12.4.

The cadence of price oracle queries (the sequence $(t_i)_{i \in \mathbb{N}}$) depends on an application's *desired liveness*, or how often they want to be sure that the most recently selected fork is the canonical fork. An application with a high desired liveness may query ONE instance prices every few minutes whereas one with a lower desired liveness may query every few days. Note that there is a trade-off between economic costs and liveness.

Importantly, all forks still persist so long as there is interest in their respective views of reality. Any application that reads Upshot One can ignore TF in favor of their own rule, which may choose an alternative fork.

[17] The exact calculation of Y(X) is a modular, opinionated component left open-ended. It should be concave and increasing in the cumulative amount of migrated stake to ensure that the earliest adopters exact the most benefit if their fork becomes canonical.

[18] The migration mechanism developed here is modular. Others can be used instead, such as minting/burning forked tokens as ones migrates to/from a fork or creating Uniswap markets at each resolution to facilitate the exchange of all possible forked tokens.

[19] We only concern ourselves with *intra-protocol forks*, or those that do not change the protocol. All public protocols can be "hard forked," or forked into different protocols, but this is beyond the scope of a paper defining one such protocol.

11 The ONE Token

As mentioned above, subjectivocracy requires a native protocol token that is amenable to forking. This motivates the construction of a custom native token that can be used as a ubiquitous "fuel" for Upshot One. ONE can be...

- spent by askers to pay for agents' services. Askers must purchase and send ONE alongside question object data. This induces a familiar supply-demand interaction where an increasing demand for answers (or asking rate), paired with a weakly decreasing supply of answers (or answering rate), increases the minimum reward necessary to resolve a question, in turn increasing the price of ONE.
- staked by participating agents. ONE appreciates as more agents attempt to profit on Upshot One.
- forked to represent multiple realities. The value of each forked token is determined by the demand for its associated reality.
- used to make governance decisions. Upshot One is a Nomic-style protocol in which all of its aspects can be changed by its participants [Sub] – ONE holders. ONE holders vote to change protocol parameters, administrative rights, etc. Forthcoming work will specify the exact governance mechanism of Upshot One.

Such a fourfold employment of ONE – as will be pursued by Upshot – necessitates an intelligent token distribution that efficiently secures network effects and, in turn, Upshot One's longevity. When Upshot One launches, a distribution schedule will mint and disburse ONE to key active participants, coalescing a network of motivated actors whose funds are equitably diffused. At some time thereafter, control over the distribution schedule itself will be given to ONE holders to facilitate the complete transfer of protocol ownership from Upshot to ONE holders.

12 Analysis

In this section we analyze the robustness and security of Upshot One. We first show that there exists a profit motive among agents, which implies that we should expect a healthy supply of answers. We then illustrate that attacks involving established agents are ruinous because of the agent record. This motivates why there is only one feasible type of attack, and we conclude our analysis by showing that it is costly.

12.1 A Profit Motive Exists

We have explicated opposing incentives that affect agents, namely the allure of question rewards and, in their worst-case, the threat of stake confiscation (which we henceforth assume is implemented). Whether or not these forces encourage agent participation remains to be shown and is now our focus. We adopt a simple agent-based model where we only consider a single question group wherein $a \geq 3$ independent, identical agents answer new questions at most once with probability p. Between timesteps, q new questions enter the question group, and any among them may be included in a scoring group alongside previously answered overlapping questions. Let X_t be a random variable counting the number of new overlapping questions (with ≥ 3 answers from different agents) in the tth scoring group. Clearly, X_t are mutually independent across t, and if $Y_t \coloneqq \sum_{i=1}^{t} X_i$, then the expected number of overlapping questions at scoring group t is $\mathbb{E}[Y_t] = tq\big(1 - \sum_{i=0,1,2} f(i; a, p)\big)$, where $f(y; a, p)$ is the binomial PMF with a trials and p probability of a success.

Let x, s, r, t be an agent's stake to a question group, the sum of all other agents' stakes to the same question group, the (uniform, for simplicity) reward per agent per question, and a scoring group index, respectively. Let $S(x, s)$ (or S) be a random variable that is one if the agent is selected as a committee member and zero otherwise. Let $L_C(x, s, r, t)$ (or L_C) be a random variable that is qr if there are at least $2C$ overlapping questions in the potential scoring group and $-x$ otherwise. Then the agent's expected reward is:

$rq > 0$		p		
$0 < s < yrq$		.2	.5	.9
tq	10	8.277e-7	.0282	173.038
	100	.0088	1070.55	1.585e50
	1000	22.9188	1.984e52	-1

Table 1: We list the values of y such that, for the given values of p and tq, if $rq > 0$ and $0 < s < yrq$, then $\mathbb{E}[SL_C] > 0$. Clearly, for a sufficiently high rate answering rate p and total number of questions tq, there are generously wide latitudes in which an agent's expected profit is positive.

$$\begin{aligned}
\mathbb{E}[SL_C] &= \mathbb{E}[S]\,\mathbb{E}[L_C] \\
&= \mathbb{P}(S=1)\Big(\mathbb{P}(L_C = qr)qr - \mathbb{P}(L_C = -x)x\Big) \\
&= \frac{x}{s+x}\Big(\mathbb{P}(Y_t \geq 2C)qr - \mathbb{P}(Y_t < 2C)x\Big) \\
&= \frac{x}{s+x}\Big((1-\mathbb{P}(Y_t \leq 2C-1))qr - \mathbb{P}(Y_t \leq 2C-1)x\Big) \\
&= \frac{x}{s+x}\Big(qr - F(2C-1; tq, p^3)(qr+x)\Big)
\end{aligned}$$

where $F(y;\cdot,\cdot)$ is the binomial CDF. Our task is to show that there is a practical domain on which $\mathbb{E}[SL_C] > 0$. In Table 1, we list parameter intervals for which $\mathbb{E}[SL_C] > 0$, and these ranges are large enough to make the practicality of this domain self-evident.

12.2 History Implies Ruinous Attacks

It is entirely up to the aggregators to determine which questions are included in a scoring group. Once a committee is selected, the aggregator may find that committee members exhibit some number of overlapping questions, and it is up to them to decide how many and which of these overlaps to include in the scoring group. An attacking agent can attempt to guess the sets of selected agents and overlapping questions to avoid effort (and still be rewarded) or to ensure that they are scored well even if they untruthfully answer a targeted question (minimally impacting their reputation and suspicion of their answer among external onlookers), but this is ill-fated the more established agents tend to be.

We assume that the agent record is public and that an attacker is capable of thoroughly querying it to identify all historical, overlapping questions. Apart from knowing everything the aggregator knows[20], their most optimistic case is knowing the number of past questions that will be included n (perhaps a fixed n is routinely used). Then the probability of correctly choosing the right set of previously resolved questions is $1/\binom{N}{n}$ where N is the total number of overlapping questions between selected agents. It is easy to show that their probability of success converges Q-linearly to zero as $N \to \infty$, and since $\mathbb{E}[Y_t] \in O(tq)$, we expect this to occur quickly in practice.

An attacker may instead not have access to n, in which case, they must guess it and then guess which specific overlapping questions will be included. The success of the latter guess follows a hypergeometric distribution with PMF $h(N, K, n, k)$ where the number of successes k equals the number of "items with the trait" K. If the attacker uniformly and independently of all else chooses $n = \hat{n}$, then their probability of success is:

$$\mathbb{P}(n=\hat{n})h(N,k,\hat{n},k) = \frac{1}{N}\frac{\binom{k}{k}\binom{N-k}{\hat{n}-k}}{\binom{N}{\hat{n}}} = \frac{1}{N}\frac{\hat{n}!(N-k)!}{N!(\hat{n}-k)!} = \frac{1}{N}\frac{{}_{\hat{n}}P_k}{{}_{N}P_k}$$

which similarly converges to zero Q-linearly. Note that the degree to which an attacker foresees a committee selection (one of their remaining vectors) is the degree to which they can lower N, however, overcoming this barrier is a function of overcoming a strong source of randomness, which we take for granted (see Section 13.2).

[20] With well-designed aggregator incentives, as developed in a forthcoming paper, aggregators lack a motive to leak information.

α	$x^*(s;\alpha)$	$r^* \ni \mathbb{E}[SL_C] = 0$
0.10	86.62	354709
0.33	223.67	915929
0.50	384.73	1.57547e6
0.70	792.09	3.24361e6
0.80	1295.05	5.30323e6
0.90	2797.66	1.14564e7
0.95	5798.86	2.37463e7
0.99	29799.78	1.2203e8

Table 2: For select α, we give a lower bound on cost of attack $x^*(s;\alpha)$ where the sum of all non-attacker stakes in the group is $s = 100$. $x^*(s;\alpha)$ is linear in s, so these values can be interpreted as "the percent of any s an attacker must stake to be successful with probability α." We also approximate the reward r^* necessary for the attacker to churn a profit from their attack when $p = 0.5, C = 2, tq = 2C = 4$. tq is set to the smallest number of questions a scoring group can have so an attacker averts having a history.

12.3 Attacks Are Costly

It is helpful to think of agents and their behaviors as proxies for intent. If many agents stake to a question group and report the same answer, then, because we select committee members with replacement, the probability of selecting a committee that resolves with that answer is unaffected by whether their stake is sourced from several colluding agents or if it is sourced from a single whale. This implies that, within Upshot One, sybil attacks are irrelevant and bribery (including "$P+\epsilon$ attacks" [Butd]), collusion, and whale attacks that solely involve agents are equivalent vectors. With this equivalence, the nonzero cost of communication, and the aforementioned impact of agent records, we can simplify our attack analysis to a single wealthy agent with no history. Under one assumption, we calculate the cost of a successful attack given the sum stake of all non-attacking agents, the attacker's stake, and the attacker's desired certainty in their success. We assume that *all non-attacking agents report informed-truthfully*, which assures that an attack would fail if the attacker is not alone on a committee. It is a weak assumption because of the profit motive invoked by DMI-Mechanism.

Recall that committee members are selected by sortition from a pool of participating agents with replacement. Agents are sampled thrice according to their stake, so an attacker need only ensure that they are selected thrice. Let s be the sum of stakes from all non-attacking agents in a question group, α be the desired certainty in the attacker's success, and x be the attacker's stake. The attacker stakes x such that:

$$\big(\mathbb{P}(\text{attacker selected})\big)^3 = \frac{x^3}{(s+x)^3} \geq \alpha$$

$$\Rightarrow \quad g(x) \coloneqq (1-\alpha)x^3 - 3\alpha s x^2 - 3\alpha s^2 x - \alpha s^3 \geq 0 \tag{1}$$

Appendix A shows that Formula 1 admits a single real, positive root $\forall(\alpha, s) \in \mathbb{D} \coloneqq (0,1) \times \mathbb{R}_{++}$. This root is the minimum stake any agent must place to be the only selected agent with probability α. If we find a functional relationship, $x^*(s;\alpha)$, between these roots and s, then we know the minimum cost of an attack for any given α. This is accomplished in Appendix B and some results are published in the second column of Table 2. $x^*(s;\alpha)$ is merely a minimum cost because there is a risk that the attacker is selected alongside other agents and the resultant committee fails to elicit a valid scoring group, in which case their stake is completely lost. This risk is incorporated into the third column of Table 2, which gives the "break-even" reward an attacker earns for a successful attack $r^* \geq qr$ that may include rewards exogenous to the protocol:

$$0 = \mathbb{E}[SL_C] = \alpha\big(r^* - \mathbb{P}(Y_t \leq 2C-1)(r^* + x^*(s;\alpha))\big)$$
$$\Rightarrow r^* = x^*(s;\alpha)\frac{\mathbb{P}(Y_t \leq 2C-1)}{1-\mathbb{P}(Y_t \leq 2C-1)}$$

12.4 Analysis of TF

Recall S and T^a from Section 10. TF's robustness is its simplicity. Its runtime and storage are both $O(|S|)$, which is not necessarily increasing in time – it may level off as the influx of ONE instances into S equals their efflux. Both a decreasing influx and increasing efflux through S (until equilibrium) is achieved by the inherent security of TF.

T^a's cost of manipulation[21] increases nearly linearly with a's liquidity [Unib] and is equal to the amount lost to arbitrage (by artificially propping up an ONE instance's value) and the revenue not accrued by resolving questions (by selling truly canonical ONE, their opportunity to answer questions asked in the truly canonical reality). This makes it increasingly risky to frivolously create forks (decreasing S's influx) and increasingly expensive to maintain valueless ONE instances (increasing S's efflux). Thus, both TF's security and robustness scale with a canonical reality's demand.

13 Risks and Mitigations

In this section, we discuss outstanding substantive risks facing Upshot One and their mitigation.

13.1 Further Curbing Attacks

Though we have shown that successful attacks are costly (Section 12), they are surmountable. Forking, the final defense, protects against all attacks, but its severity begs the use of more intermediary defenses. Advanced cryptographic machinery such as MACI [app] can be used to obfuscate the identities of agents and their submissions without invalidating the agent record, thereby rendering proofs of cooperation among attacking agents (nearly[22]) impossible. Additional measures can be taken adjacent to the protocol to increase the cost of an attack including escalation games, such as Augur's dispute bonds [Pet+], and dynamic fees (as in Section 14.1).

13.2 Poor Source of Randomness

Committee sortition assumes access to on-chain entropy. If this randomness is predictable or corruptible, the cost of attack can be substantially lowered. If the entropy is predictable, the attacker can wait for an entropy value in which they are selected. If the entropy is corruptible, the attacker can simply supply such a value. This risk's mitigation is simple: We leverage a strong, third-party source of randomness such as Chainlink's VRF [Chab] or Keep's Random Beacon [Net].

13.3 Fractured Liquidity

Categorization arbitrarily fractures the liquidity of question groups, thus risking the efficacy of Upshot One. Askers can tune question rewards and filters to, respectively, draw more answers to more unified question groups, but the budgets and interests of askers limit their ability to mitigate liquidity fracturing.

We can exploit a weak, empirically-informed assumption to mitigate this risk. Many dynamic networks "in the wild" follow a power-law distribution, where characteristics of a majority of nodes regularly, at all scales, mirror a minority of all types of nodes [New]. Therefore, we can reasonably assume that most questions are of just a few categories. If we bootstrap question throughput in these few categories, then we cost-effectively have spurred the self-sustained scaling of our protocol, where, for most cases, the failure of a question to reach resolution is a function of genuine disinterest rather than fractured liquidity. Our mitigation is, then, soon after our protocol's launch, a fourfold approach: supply many simple a priori similar questions to groups we deem poised to naturally grow the most, subsidize both asker and agent participation, maximize question throughput as our protocol's first aggregator, and establish incentives for future aggregators to follow suit.

[21] Such manipulation is easily attainable if the majority of ONE's demand endorses a different reality, however this begs the questions "Maybe this is reality after all?" and "How representative is the demand?" Both may be impossible to answer.

[22] Currently, aggregators ("coordinators" in MACI-lingo) could broadcast any and all agents' secrets, but this issue and the broader development of MACI are both actively being advanced.

13.4 Malicious Aggregators

A malicious aggregator could manipulate a resolution by only including questions that, when scored, result in a specific outcome. Initially, Upshot One will mitigate this risk by only considering aggregators that are on a governance-populated whitelist. This may include an albeit costly but simple, automated, on-chain aggregator that, upon an external "poke," triggers a scoring round and randomly hunts for overlapping questions. While not maximally scalable or decentralized, a whitelist assures a young, vulnerable Upshot One that its aggregators are non-malicious. Our protocol will progress to a more decentralized aggregator design as it matures. This decentralization and aggregator incentives are left to a forthcoming paper.

14 Applications

There are a plethora of applications that benefit from or require consensus on information whose quality is not easily accessible. We briefly discuss some such applications and how they might leverage Upshot One.

14.1 Prediction Markets

Prediction markets (PMs) offer financial derivatives on any future event. They are a useful means of querying the "wisdom of the crowd" to gain accurate insights on the likelihoods of future events. However, for PMs to function, they require access to a ground truth – they must eventually reference what event actually occurred to determine who deserves what payment. Upshot One can be used as a source of ground truth:

1. The PM creator deploys an oracle contract. The oracle has administrative rights to say who in the PM wins what, specifies the execution logic qualifying and following a finalized decision, and interacts with Upshot One, where it asks questions, manages their categorization, and interprets their resolution.
2. After the PM's "trading period," its question is categorized and then resolved by Upshot One.
3. Application-specific dispute logic can occur (e.g. Augur's dispute bonds [Pet+]) that may result in a fork.
4. If there is a fork, the PM contract can track the forked tokens' prices and finalize itself according to the canonical fork selected by the oracle's fork-choice rule. Otherwise, the market is finalized and its rewards are distributed accordingly to traders.

Extra care must be taken in applying Upshot One to PMs. A PM's traders have an explicit interest in their side winning over another. This means that traders have an incentive to subvert the resolution by bribing agents or being an agent themselves. While the existing design of Upshot One alone makes this attack expensive, it can be made costlier through *dynamic trading fees*. Just as Augur changes trading fees in accordance with the market cap of REP [Pet+], PMs leveraging Upshot One can change their trading fees according to the market cap of ONE. Implementing this is trivial, requiring no more machinery than a reliable price oracle for ONE. Now, when the market creator deploys their oracle contract, they simply configure a price oracle and a functional relationship between trading fees and the price oracle's output.

14.2 Insurance

Insurance is an increasingly necessary utility in the world as surmounting risks (e.g. the plethora of known technological vulnerabilities) and "unknown unknowns" (e.g. name your favorite hack) both become all but unavoidable. However, the contemporary insurance industry struggles to efficiently resolve claims involving discretionary (nonparametric) assessment. Upshot One offers a refreshingly practical solution to this problem.

Say we wish to insure smart contracts against technical risk. Assume the insurance provider has created a way to assess and underwrite these technical risks (PMs may be used for this). Similar to how the PM creator had criteria for "finality," the insurance provider deploys a contract allowing policyholders to receive payouts for their claims as a function of results from Upshot One. An insurance provider may also want all claims to be asked to a whitelisted set of agents (set in a question filter) who specialize in verifying claims of policies on

smart contract risk. As a policyholder, if I have lost money in a smart contract, I would create a claim that gets sent to a dedicated question group in Upshot One. My claim would then be resolved by those whitelisted agents and I would receive my deserved payout from the insurance protocol.

14.3 Curation

In contrast with the previous two applications, which required the onboarding of difficult to access though not necessarily subjective information, curation is subjective by definition. Furthermore, curation tends to not be as "high-stakes" a task as resolving PMs or insurance claims. Therefore, in curation applications, not much machinery must be added atop Upshot One to maintain its existing incentive alignment.

Some instances of curation, such as digital content curation, may require very low latency. To this end, we can follow [Butb] and build a service that creates low-stakes PMs[23] for (all or just popular or controversial) social media posts. This service can be embedded within a social media network or live adjacent to one. Upshot One may then be used to resolve some subset of these PMs. If a trader participates in a PM that is resolved by Upshot One, they receive capital from the PM if they accurately wagered whether or not a post was true, funny, relevant to people that like X, etc. If a threshold amount of time passes without resolution, the traders are refunded. A byproduct of this system is public good – a stream of content whose elements have been categorized as true, funny, relevant to people that like X, etc.

Note that although the accessible, ubiquitous nature of digital content may lend "content curation" to an immediate, fathomable application of Upshot One, "physical" forms of curation (e.g. judging the fashion value of outfits in user-submitted images) are far from inconceivable.

14.4 Governance

Even where "code is law," the maintenance and prosperity of a constitution is a function of consent among the governed. The expediency of consent's revelation, however, may overrule consent by stifling its expression, and, in turn, deprive a constitution and its governed body of empowerment. Self-interested communities of all kinds charge themselves with enduring despite this reality, yet some among them elect governance mechanisms whose destiny is dysfunction. In particular, many have adopted simple voting mechanisms (e.g. [Cro], [Fou], [Lab]) even though these schemes fail to satisfy a set of widely-desired criteria [Arr50].

Upshot One offers an alternative. Governance decisions can be mapped to questions that are to be answered by governance token holders. From there, Upshot One continues its usual operation. This is an efficient governance mechanism because, instead of a census, a minimal amount of agents are necessary. Efficiency is also bolstered by a built-in delegation capacity. By default, this solution offers a liquid democracy [Sch] with explicit delegation by token transfer and implicit delegation by abstinence. This is a robust governance mechanism by the high cost of Upshot One's coercion (see Section 12)[24]. This mechanism empowers participation in governance because participation is no longer an act that, at best, is worth hundredths of a penny [Butc] – one's answers can, with non-trivial odds, determine entire decisions (while earning agents question rewards). Finally, subjectivocracy enables esoteric, powerful mechanisms to coalesce. Forking a decision amounts to placing a "futarchy-like bet" [Buta] on the efficacy of policies. Forking also allows an ecosystem of disparate, specialized governments to coexist. The possibilities arising from Upshot One's application to governance are bountiful and we hope, at minimum, to inspire further research in applications of peer prediction to governance.

14.5 Forecasting

PMs support *Bayesian actors*, or actors who update their beliefs based on the revealed beliefs of other actors. This means that PMs enable an online view of a crowd's likely time-dependent forecast, whereas peer prediction mechanisms capture only snapshots of said forecast at runtime. However, by simply repeatedly asking the same question to Upshot One, we can leverage the "wisdom of the crowd" for forecasting without, unlike PMs, the need for an external source of truth or an end date – Upshot One could be re-asked in perpetuity. This qualifies Upshot One as an alternative to PMs – it can be used as the "predictive" component of the insurance and

[23] e.g. small-cap PMs or markets whose currency is some social media-recognized "clout coin," such as Reddit Donuts [uRi].
[24] We assume the distribution of governance tokens is not already a product of coercion.

curation applications above. Future research will compare the costs of PMs (that last a certain time and have enough depth and volume to support online forecasting) to the costs of repeatedly querying Upshot One.

15 Conclusion

Upshot One is a protocol built to arbitrate difficult to ascertain or subjective information. In this paper, we have reconstructed Upshot One from scratch by successively motivating each component's involvement. Together, these components form a gestalt, outlined in Section 2, that accepts questions and returns truth.

Our protocol is implemented as a set of smart contracts on the Ethereum blockchain.

References

[Arr50] Kenneth J. Arrow. "A Difficulty in the Concept of Social Welfare". In: *Journal of Political Economy* 58.4 (1950), pp. 328–346. ISSN: 00223808, 1537534X. URL: http://www.jstor.org/stable/1828886.

[app] appliedzkp. *maci/specs/*. Available at https://github.com/appliedzkp/maci/tree/master/specs.

[Blo] NPE Blog. *A community for crowdsourced predictions*. Available at https://www.forecastapp.net/.

[Buta] Vitalik Buterin. *An Introduction to Futarchy*. Available at https://blog.ethereum.org/2014/08/21/introduction-futarchy/.

[Butb] Vitalik Buterin. *Prediction markets for content curation DAOs*. Available at https://ethresear.ch/t/prediction-markets-for-content-curation-daos/1312.

[Butc] Vitalik Buterin. *Quadratic voting with sortition*. Available at https://ethresear.ch/t/quadratic-voting-with-sortition/6065.

[Butd] Vitalik Buterin. *The P + epsilon Attack*. Available at https://blog.ethereum.org/2015/01/28/p-epsilon-attack/.

[Bute] Vitalik Buterin. *The Subjectivity / Exploitability Tradeoff*. Available at https://blog.ethereum.org/2015/02/14/subjectivity-exploitability-tradeoff/.

[Chaa] Chainlink. *Chainlink*. Available at https://chain.link/.

[Chab] Chainlink. *Chainlink VRF: On-chain Verifiable Randomness*. Available at https://blog.chain.link/verifiable-random-functions-vrf-random-number-generation-rng-feature/.

[Cléa] Federico Ast Clément Lesaege William George. *Kleros*. Available at https://kleros.io/assets/whitepaper.pdf.

[Cléb] William George Clément Lesaege. *Kleros and Augur – Keeping people honest on the blockchain through game theory*. Available at https://medium.com/kleros/kleros-and-augur-keeping-people-honest-on-ethereum-through-game-theory-56210457649c.

[Cro] Andre Cronje. *yEarn*. Available at https://yearn.finance/.

[Fou] Maker Foundation. *MakerDAO*. Available at https://makerdao.com/en/team.

[Hog] Victor Hogrefe. *The Oracle Problem: Why decentralizing everything is more difficult than it sounds*. Available at https://medium.com/@victorhogrefe/the-oracle-problem-5611bed763ba.

[Kon] Yuqing Kong. *Dominantly Truthful Multi-Task Peer Prediction with a Constant Number of Tasks*. Available at https://arxiv.org/pdf/1911.00272.pdf.

[Lab] Balancer Labs. *Balancer*. Available at https://balancer.finance/.

[Man+] Debmalya Mandall et al. *Peer Prediction with Heterogeneous Tasks*. Available at https://arxiv.org/pdf/1612.00928.pdf.

[Mec] UMA's Data Verification Mechanism. *Hart Lambur*. Available at: https://medium.com/uma-project/umas-data-verification-mechanism-3c5342759eb8.

[Nak] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. Available at https://bitcoin.org/bitcoin.pdf.

[Net] Keep Network. *The Keep Random Beacon: An Implementation of a Threshold Relay*. Available at http://docs.keep.network/random-beacon/.

[New] M. E. J. Newman. *Power laws, Pareto distributions, and Zipf's Law*. Available at https://arxiv.org/pdf/cond-mat/0412004.pdf.

[Pet] Ashish Goel Peter Lofgren. *Personalized PageRank to a Target Node*. Available at https://arxiv.org/pdf/1304.4658.pdf.

[Pet+] Jack Peterson et al. *Augur: a Decentralized Oracle and Prediction Market Platform*. Available at https://www.augur.net/whitepaper.pdf.

[Pro] Provable. *Provable*. Available at https://provable.xyz/.

[Sch] Dominik Schiener. *Liquid Democracy: True Democracy for the 21st Century*. Available at https://medium.com/organizer-sandbox/liquid-democracy-true-democracy-for-the-21st-century-7c66f5e53b6f.

[Sub] Peter Suber. *Nomic: A Game of Self-Amendment*. Available at https://legacy.earlham.edu/~peters/nomic.htm.

[uRi] u/Rickbumctious. *DONUT - Donuts, Reddit backed superstar*. Available at https://www.reddit.com/r/CryptoMoonShots/comments/hny3l9/donut_donuts_reddit_backed_superstar/.

[Unia] Uniswap. *Uniswap*. Available at https://uniswap.org/.

[Unib] Uniswap. *Uniswap v2 Core Concepts: Oracles*. Available at https://uniswap.org/docs/v2/core-concepts/oracles/.

[Wal] Greg Walker. *Longest Chain: The chain of blocks that nodes adopt as their blockchain.* Available at https://learnmeabitcoin.com/technical/longest-chain.

[Wika] Wikipedia. *Cubic equation*. Available at https://en.wikipedia.org/wiki/Cubic_equation#General_cubic_formula.

[Wikb] Wikipedia. *Discriminant*. Available at https://en.wikipedia.org/wiki/Discriminant#Degree_3.

Appendices

A Formula 1 has One Positive Real Root

Proof. The the general cubic formula f and its discriminant $\mathscr{D}(f)$ are given by: [Wikb]

$$\begin{aligned} f(x) &= ax^3 + bx^2 + cx + d \\ \mathscr{D}(f) &= b^2c^2 - 4b^3d - 4ac^3 - 27a^2d^2 + 18abcd \end{aligned} \tag{2}$$

Then the discriminant of Formula 1 is:

$$\begin{aligned} \mathscr{D}(g) &= (3\alpha s)^2(3\alpha s^2)^2 - 4(3\alpha s)^3(\alpha s^3) + 4(1-\alpha)(3\alpha s^2)^3 - 27(1-\alpha)^2(\alpha s^3)^2 - 18(1-\alpha)(3\alpha s)(3\alpha s^2)(\alpha s^3) \\ &= s^6(81\alpha^4 108\alpha^3 - 108\alpha^4 - 108\alpha^4 - 27\alpha^2 + 54\alpha^3 - 27\alpha^4 - 162\alpha^3 + 162\alpha^4) \\ &= -27\alpha^2 s^6 \end{aligned}$$

Both $\mathscr{D}(g)$ and $d = -\alpha s^6$ are negative on $\mathbb{D}$. Therefore, Formula 1 has one real, positive root on $\mathbb{D}$. ■

B $x^*(s;\alpha)$ is Linear in s on $\mathbb{D}$

Proof. We first state the equation for the roots x_k^* of a general cubic polynomial $f(x)$ [Wika]:

$$f(x) = ax^3 + bx^2 + cx + d$$

$$\Delta_0 = b^2 - 3ac, \quad \Delta_1 = 2b^3 - 9abc + 27a^2d, \quad C = \sqrt[3]{\frac{\Delta_1 \pm \sqrt{\Delta_1^2 - 4\Delta_0^3}}{2}}$$

$$x_k^* = \frac{-1}{3a}\left(b + \xi^k C + \frac{\Delta_0}{\xi^k C}\right)$$

where $\xi = \frac{-1+\sqrt{3}}{2}, k = 0, 1, 2$. Appendix A shows that there is just one real, positive root x^* on $\mathbb{D}$, but what is x^*'s dependency on s, or what is $x^*(s;\alpha)$? Plugging Formula 1 into the above equations yields:

$$\Delta_0 = 9\alpha s^2, = c_1 s^2, \quad \Delta_1 = -27(\alpha + \alpha^2)s^3 = c_2 s^3$$

$$\Rightarrow C = s\sqrt[3]{\frac{c_1 + \sqrt{c_1^2 - 4c_0^3}}{2}} = c_3 s$$

$$\Rightarrow x^*(s;\alpha) = \frac{-1}{3(1-\alpha)}\left(- 3\alpha s + \xi^k c_3 s + \frac{c_1 s^2}{\xi^k c_3 s}\right) = c_4 s$$

for constants $c_1, c_2, c_3, c_4 \in \mathbb{R}$. We know that there is only one k that yields a real, positive root on $\mathbb{D}$, so we reject the other k and conclude that $x^*(s;\alpha)$ is linear in s on $\mathbb{D}$. ■