A Taxonomy of Blockchain Oracles: The Truth Depends on the Question

Michael Bartholic
Georgetown University
Washington, DC USA
ORCID 0000-0002-8788-3170

Aron Laszka
NTT Research, Inc.
Sunnyvale, CA USA
ORCID 0000-0001-7400-2357

Go Yamamoto
NTT Research, Inc.
Sunnyvale, CA USA
ORCID 0000-0003-3610-1009

Eric W. Burger
Georgetown University
Washington, DC USA
ORCID 0000-0002-2143-9368

2022 IEEE International Conference on Blockchain and Cryptocurrency (ICBC) | 978-1-6654-9538-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICBC54727.2022.9805555

***Abstract*—Blockchains benefit from guarantees of immutability and reliability due to their high redundancy and distributed nature. They show their value especially when operating between untrusted parties. Their functionality can be extended programmatically by smart contracts, but are limited by high costs of on-chain computation and only being able to truly trust data which is directly included on-chain. To attempt to bridge this limitation, blockchain oracles are introduced as a conceptual solution to act as a trusted source of information within the blockchain. The Oracle Problem emerges as we consider how one can introduce trusted information into a trust-free environment without compromising the validity of the blockchain. Many promising designs for oracle mechanisms have been proposed, but it is not readily apparent how one should assess the applicability of a given mechanism, nor the strengths and features between mechanisms. To be equipped to assess and categorize oracles, we must consider not just the possible answers, but the questions to which these oracles are trying to speak. Categorizing questions by their possible answering populations, we propose a framework for considering oracle questions and the context with which they are posed. We observe that there are limitations to what an oracle can hope to achieve, depending on the nature of the question, while noting the context in which a question exists can change what is viewed as true.**

***Index Terms*—Blockchain, Oracle, Cryptocurrency, Peer-to-peer Computing, Distributed Information Systems, Real-time Systems, Feeds**

I. Introduction

Blockchains are a robust, distributed data structure that provide immutability and trustworthy computation in a decentralized manner. Blockchain functionality can be extended by Smart Contracts to provide capabilities for implementing programs and business logic, but are limited in computational resources and can directly rely only on information inside the blockchain.

Many compelling use cases for blockchains would benefit from trusted sources of external information. Blockchain oracles are introduced as a conceptual solution to introduce information into the blockchain by some means. Thus far, numerous oracle designs have been introduced. However, it is not clear exactly what features they have, how to directly compare them to each other, and which oracles are the right ones for a given query.

One of the reasons we may not be fully equipped to assess the problem is that most existing work on blockchain oracles focuses primarily on the different answering mechanisms that particular oracles provide. Instead, we believe we have to consider what the questions are and how these questions are framed. To adequately assess the truth of information being introduced to the blockchain, we must have a framework to consider the way questions are posed and the context in which they exist, as the truth can change depending on the context of the question.

While we are not striving to give answers to all questions, we set out to at least categorize what can be realistically expected to have an oracle-provided answer. When seeking answers from an oracle, one must keep in mind not only the context of the question at hand, but also the expected mechanism for receiving an answer. Different oracle mechanisms necessitate different expectations for the quality, type, and confidence of the answer. Certain mechanisms may only be able to convey, or are only intended to vouch for, the integrity / authenticity of information that is already trusted. Other mechanisms may be able to effectively crowd-source answers with high confidence in an untrusted environment. These differences are determined by the nature of the question being asked as well as the context as for what, how, for when, and by what means an answer is expected. Ultimately, we find that the method for answering oracle inquiries should be approached with different models based on the population of parties who may be able to answer such a question.

A. Background

A blockchain is an immutable, shared, distributed ledger of database transactions in the form of linked blocks that is perpetuated by participants "mining" new blocks. Blockchains are unique in that their internal state is trusted due to the shared, distributed computation that perpetuates them. This leads blockchains to be unable to trust outside information that is not part of the network of blockchain nodes itself.

This problem of incorporating external information into the blockchain is known as "the Oracle Problem" [1]. **The Oracle Problem is that–in an untrusted environment–how does**

This work was supported in part by NTT Research, Inc. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NTT Research.

one bring information from outside the trusted blockchain environment onto the blockchain in a trusted manner? The problem particularly manifests itself when there is a single source of information: How does one trust that single source? If one trusted single sources, one might not bother with the computational costs of a blockchain solution in the first place.

A blockchain oracle is a conceptual mechanism for transferring information between the isolated blockchain and the "outside world" i.e. anywhere else. This isolation creates significant difficulties for many of the aspiring blockchain use cases that would naturally involve integrating with any sort of external data, including other blockchains. In particular, this impacts business process management applications and any circumstance where an operation is meant to occur in relation to an external state [2], [3].

Oracles are often discussed solely as a mechanism to bring data into the blockchain. However, there are numerous variations upon this common pattern. Oracles may convey information to or from the blockchain in either direction. Similarly, oracles may differ in mechanism of answering, direction of information flow, direction of initiation (agency), and more [4]–[7].

B. Motivation

We chose to pursue constructing a taxonomy of blockchain oracle questions while surveying the current state of proposed solutions to the oracle problem. We recognize that there are numerous existing methods for implementing oracle mechanisms, but little description of what is actually possible to answer, what confidences these answers have, or what methods might speak to certain categories of questions. Specifically, we find that there is no unifying sense for what types of assumptions are necessary to provide answers for given sets of questions.

We have also found difficulties in existing works, namely that they tend to be quite loose about specifically what their proposal / implementation is able to answer, and require significant assumptions about the parseability of general user queries as well as query responses. We find that specifying a more strict model for questions and answers would benefit the feasibility of these mechanisms and provide a basis for determining what categories of questions may realistically expect (reliable) answers.

Furthermore, while we find there to be a strong need to identify the assumptions required to answer given oracle queries, we also embrace the belief that all computational systems involve trust at many levels. This trust may be at the hardware level, through the full network stack, at the application level, in the interactions of any network's participants, or otherwise. One must be able to assess the trust assumptions that are made and verify wherever possible that these assumptions are sound. We find that study of the oracle problem would benefit from a framework to consider these factors for given sets of questions.

C. Prior Oracle Surveys

Existing work in overarching studies of blockchain oracles are largely surveys that choose to focus primarily on aspects of the implementation and categorize the varieties of design patterns, data flow, operating principles, and agency patterns present in oracles [4]–[6]. Notably, [1] has several valuable analyses discussing the practical behavior, usage, and vulnerabilities of extant oracle mechanisms. We specifically find [8] valuable in its attempt to devise a reliable way to consider the attributes of existing blockchain oracle implementations where less academic literature is available. Likewise, as in [1], we also emphasize the need to understand the oracle problem both from a design and implementation perspective, as well as from the point of view of how to assess truth in varying contexts.

D. Our Contribution

This work focuses on how to categorize possible questions that an oracle may be tasked to answer. Our novel approach explains the problem with most extant blockchain oracle systems that fail to provide reliable information. We consider how to go about identifying possible solutions for these categories based on properties of each question type. We evaluate how the framework fits with existing conclusions and implementations. Then, we provide an assessment of certain implementations, through the lens of the framework, to demonstrate shortcomings and their possible reconciliations.

We begin by introducing the foundation of reasoning and terminology behind the framework. We then discuss basic definitions and building blocks for composing what we call Oracle Events, followed by their categorization. Finally, we consider how the framework suits and describes empirical patterns in existing oracle work, as well as what it suggests about advantages and disadvantages of existing oracles. We conclude with notes regarding how we hope the framework can be of use to future study of the oracle problem and oracle mechanism design.

We also provide two appendices that contain several examples of unreasonable and reasonable oracle queries, respectively. These appendices are intended to be used as references for considering the context and characteristics of questions that may be posed to an oracle in practice. Some readers may prefer to begin with the appendices as an introduction to reasoning about what types of questions an oracle may answer and how these questions ought to be presented. While the appendices make use of some terminology that is established throughout this work, they should be approachable without completing a full reading and more enriching afterward.

II. Framework for Questions

A. Conceptual foundation

We want to consider, from a fundamental level, what the purpose of a blockchain oracle is and how they act to provide information by asserting certain truths and falsities. Most simply, oracles convey information in a way that satisfies a posed unknown: that is, they answer questions. While not every fact may be immediately considered in the form of a question, we claim that all oracle queries can be posed as questions with a truth value. In this way, the utility of an

oracle is that of the utility of questions in general: that is, information gain.

Information gain occurs any time that the space of possible answers to a given question is reduced, leaving less uncertainty in the ultimate outcome. In principle, this reduction could be any size so long as repeated questions are allowed to enable approaching a sufficiently useful uncertainty. However, any practical implementation of an oracle will need to pay mind to answering questions efficiently and asserting the answer as directly as possible.

B. Terminology

1) Event: A logical proposition, question, or claim that may include measurements or other observations about what we call "objective reality", that can be decided as either True, False, or Unknown for any particular answer party.

This may initially seem like a somewhat narrow definition, but it is quite broad and only explicitly omits open-ended questions with scarcely defined spaces of possible answers. While many works seem to recognize the utility in questions which are in a propositional format and utilize them, we find it valuable to consider the particular benefits of doing so. When considering all possible questions, many (or even most) are open-ended. These open-ended questions are likely to have broad or even undefined answer spaces. By categorically avoiding such questions, we can greatly simplify the task of an oracle.

Although we choose to avoid open-ended questions, many open-ended questions can be re-framed in a compatible way by simply being more specific. Instead of asking: "What color is the sky?" ask "Is the sky blue?/Is the sky gray?" or better "Is the sky blue today in New York City?". In this way, previously incompatible questions can become a proposition when a specific outcome from its answer space is proposed as a possibility that can be either affirmed or denied.

We find that reliable answers necessitate Event structures which are logical statements, to be as clear as possible, and be able to include embedded context to sufficiently disambiguate the question. As we will see, it is key to employ syntactic structures that allow the embedding of rich and sufficient context to avoid ambiguity.

Note that for brevity and common use of language, throughout this text we often phrase Events simply as questions. It is to be understood that to properly be considered as an Event, corresponding to our specified syntax, such questions would need to be formulated as a logical statement (possessing a truth value), and conveyed with sufficient context.

2) Event Creating Party: The individual or group represented by an individual which poses an Event to an oracle. Events should be posed with sufficient context such that it is immaterial to know who asked, as it does not change the answer.

3) Event Answering Party/Parties: The individual or group of individuals that are able to assess the truthfulness of a given Event. The size of this population of entities may (and likely will) change with respect to the context associated with the question. To have confidence in the correctness and accuracy of an oracle's ability to answer any given question, the Event Answering Parties must be knowledgeable of both the subject matter and context of the question.

4) Objective reality: The measurable outcome that corresponds to the truthfulness of the Event in question. Usually, objective reality is assessed as "what really happened" as might be observed by many individuals. In order to avoid circular reasoning and circumstances where the reality might not be broadly observed, we let "objective reality" correspond to the outcome that the majority of honest entities provide as the answer. This converts the problem into a task of assessing how honest entities answer questions, instead of making any statement about how to precisely measure reality. The distinction may seem subtle, but this also allows us to avoid some senses of measurement error and limitations in precision.

5) Answer space: The collection of possible answers to a question or proposition that has been posed. In general, answers may be categorical, numerical (scalar), or truth values. Here, we most directly consider the answer space comprised of the answers $\{True, False\}$. "Unknown" is not included in an answer space as it is merely a statement of unknowing by some entity, not an assertion of the overall answer.

6) Information Gain: Information gain occurs when, in some answer space, the subset of answers that are still possible is reduced. This can be considered in terms of the likelihood of any individual answer being the final outcome. As we are principally evaluating propositions as either True or False, either answer provides maximum information gain as the assessed outcome is then certain.

7) Context: Possibly even more important than the structure of the Event itself is the inclusion of implicit, or preferably, explicit context as to what, where, when, from whom, and/or by what means the Event corresponds. This context serves to precisely disambiguate the meaning of an Event and make it well-defined enough to permit consistent interpretation and answer distributions. Not all facets of context may be open for a given mechanism, as some may be restricted by the way the mechanism is defined or operates. Facets of context can include what, where, when, from whom, by what means (how). Generally, the base query itself will include "what" the query is about, and the remaining facets will need to be included to ensure the query is answerable.

8) Ambiguous Event: We consider an Ambiguous Event to be an Event that may be answered differently by different parties, who are themselves answering truthfully. This divergence in their view of reality depends on their individual understanding of the Event's context and verbiage. We need to avoid unclear antecedents, unclear time/location/settings, and unclear/unspecified answering mechanisms. To create a reliable and trustworthy oracle, it is key to avoid instances of Ambiguous Events. Someone who honestly believes a given answer for a particular context cannot be faulted for acting without deceit; yet, allowing such ambiguity would undermine the efficacy of an oracle mechanism.

9) Events vs. Non-Events (Syntax): As previously described, we choose to frame Events as logical statements which can be true, false, or unknown. This syntactic structure allows our questions to be framed as a decision problem, with the additional choice that a given party may not themselves know the answer. We can form Base Queries by devising a proposition and specifying how any variables should be quantified. Base Queries are then extended with additional clauses to specify contextual information to indicate each of the open facets of context.

Not all facets may be open for a given oracle mechanism, for example a given oracle mechanism may only provide answers from a specific location, say the last trade price of a security on the New York Stock Exchange. In this case, specifying the exchange in each query is unnecessary and irrelevant for the oracle to know because the mechanism is definitionally associated with that context. Such compound logical statements, with the Base Query and context together, are then fully contextually specific and unambiguous, and are considered to be Complete Queries for a given oracle mechanism.

Furthermore, this framing of oracle inquiries as decision problems also goes far to suggest that general solutions to the Oracle Problem are impossible. If this framing is more than an analogy, we would expect that there are questions that can not be answered as well as it being non-trivial to determine the full set of answerable questions.

An example base query may be:

Example Query: The temperature in Washington, DC is greater than 20 degrees celsius.

This includes "where" and "what." However, it lacks the facets when, from whom, and by what means, which may be interpreted in several different ways as currently posed. Accordingly, the complete query could be:

Example Query: (The temperature in Washington, DC is greater than 20 degrees celsius) and (time is 2021 Oct 10 at 16:00 pm UTC) and (from any measurement source) and (by any xyz voting mechanism).

Alternatively, one could have specified a particular measurement source. However, this could drastically change the possible answering mechanisms as the population of answering parties is reduced. For example:

Example Query: (The temperature in Washington, DC is greater than 20 degrees celsius) and (time is 2021 Oct 10 at 16:00 pm UTC) and (from the KDCA meterological station).

There may be many variations upon this syntax to allow for ranges of measurement values or ranges of times ($t_1 <$ t_measurement $< t_2$), multiple measurement locations (x_1 or x_2 or x_3), and so on.[1] The central emphasis is that complete queries contain clauses for all open facets of context and that the entire query, with all clauses, must be affirmed or denied as a unit. In this way, queries require each answering party to be able to satisfy all facets of the context.

[1]Ranges would likely be required when posing questions involving scalar values, as they provide a way to account for volatility and uncertainties in measurement precision.

III. Framework Fundamentals

We begin by introducing more complete definitions of sets and concepts that will be utilized when reasoning about categories of Events. The following is the list of the discrete random variables for the Events. We define $|X|$ by $|\{x|\Pr[X = x] \neq 0\}|$ for discrete random variable X.

Questions Q (as logical statements): Questions are useful for seeking information gain by reducing the possible space of answers. Events in our framework are questions that are posed with an answer such that there exists a truth value when paired with sufficient information (i.e. context) that is included with the question. Answering questions in this syntax provides maximum information gain because a single answer is asserted to be true or false. Question context is strictly required and so open-ended questions are not permitted. Logical statements are equivalent to questions paired with an answer such that the statement can have a truth value.

Answers A (from a given answer space for a given question): Question syntax requires a logical statement to be an Event, so we must have answers that are possible truth values. Therefore, answers may include: True, False, Unknown. These represent that space of all answers to a question. Generally, an "unknown" answer could be a circumstance where a given individual does not know for lack of context, or that there is something fundamentally unknowable about the inquiry. However, an individual claiming "unknown" should not be understood as an overall claim claim of unknowablity. Instead, such a response simply means that one's answer would not be included in the distribution.

State of the "reality" R: The external information in the observable world that may or may not impact the answer to a given question. This sense of external information is important to keep in mind as most questions are not stateless. Any question whose answer depends on external information is liable to change as a result of changes in context.

Entities E (answering parties): The set of Answering Entities is the set of all honest entities who could possibly provide an answer to the question. This set varies in cardinality depending on the nature of the question and an individual's relationship to the context. To assess the truth of queries, we seek to measure how "honest entities" provide answers. We care little about how a question came to be, as all the information needed to answer the question should be included in the Complete Query and answer set, regardless of who posed the question. Stated another way, questions should always have sufficient context such that it does not matter to know who answered them or who asked.

Additionally, our framework utilizes certain properties of question types which we define as follows:

Definition of Objective: For all possible answering entities, a given question has a single unimodal answer distribution.

$$\forall a, q, r, e: \Pr[A = a|\ Q = q, R = r] = \Pr[A = a|\ Q = q, R = r, E = e].$$

Therefore, the answer does not depend on the entity that provides the answer. A corollary to this is that objectivity implies that answers to such questions will form a unimodal distribution of one answer.

Definition of Subjective: For all possible answering entities, a given question has a non-unimodal answer distribution.

$$\exists a, q, r, e: \Pr[A = a|\ Q = q, R = r] \neq \Pr[A = a|\ Q = q, R = r, E = e].$$

In contrast to the previous definition, we see that an answer to such a question does depend on the answering entity. Likewise, we can conclude that many of such answers would form a non-unimodal distribution across all possible answers.

Definition of Deterministic: For all computations of a given question q, the probability of a given answer a, from an honest entity, is equal to one. There is only one answer to this question.

$$\exists a: \Pr[A = a|\ Q = q, R = r, E = e] = 1.$$

Definition of Non-deterministic/stochastic: For all computations of a given question q, the probability of a given answer a, from an honest entity, is less than one. That is, there is variability inherent to the answer itself.

$$\forall a: \Pr[A = a|\ Q = q, R = r, E = e] < 1.$$

This captures the uncertainty in answers which will result in a broader spread in the answering distribution.

Definition of a Computational Problem: Computational problems have the general form:

$$\forall a, q, r, e: \Pr[A = a|\ Q = q] = \Pr[A = a|\ Q = q, R = r, E = e].$$

Since the computational problem depends only on the question, a deterministic computational problem appears as

$$\forall q: \Pr[A = f(q)|\ Q = q] = 1$$

where $f(q)$ is the algorithm for deriving the answer $a \in A$.

IV. Categories of Events

Over all possible Events, we define four main categories of questions where we can assert the existence of an answer. We also identify a fifth category, as an outlier, for questions for which the existence of a particular answer cannot be asserted. We will observe that, aside from a few special cases, questions within a category are far more similar than they are between categories.

Additionally, we will observe that Recondite and Sanctioned Events (see below) are more similar to each other than Computational and Discernible Events, which themselves are related. This similarity lends to solution methods for such questions that are similar as well.

These categories are generated as disjoint subsets of all possible questions by a partition on the cardinality of E for such questions. The Non-Event category is also disjoint from the others in that it only contains questions for which we cannot assert an answer, while in all other categories we can.

As we will observe, the cardinality of these sets implies additional properties of the questions that are included, along with mechanisms to go about answering them.

A. Event Types

1) Recondite Events: Answers that only one party knows. This category is special in that we are unable to tell the difference between Subjective and Objective questions, as we have no way to assess answers beyond the single answering entity.

2) Sanctioned Events: Answers that may or may not be broadly known, but only a small collection of parties can answer authoritatively. This authority may be due to legal regulations governing the ability to convey proof, or merely due to a special relationship being required with the information in question.

3) Discernible Events: Answers that can be broadly observed and assessed by a large collection of parties, with some limitation on who has access to the answer. Such limitations may be due to the spatial / temporal relationship of a entity to the answer, the entity's background / expertise, or otherwise.

4) Computational Events: Answers that can be computationally reduced given a predefined algorithm, without practical limitations on who computes the answer.

5) Non-Event: Questions that cannot be framed in such a way that information gain makes sense or are without an answering population. Questions which are proved to be unanswerable or which lack any answering entities *at a given time* are non-Events because an oracle cannot be expected to produce an answer for such inquiries.

An example of a Non-Event is a query against random information. This can be seen by the fact that the answer to such a query for random data cannot be formulated by the party posing the inquiry without producing a contradiction in motives. Consider the query: "Is the string 'frWKWwMD' 8 random ASCII letters and numerical digits?" It may appear random if there is no easily discernible pattern, but this depends on if an answering party knows an algorithm to produce such an output. Yet, knowing the pattern immediately reveals that the information is not random.

The core of the contradiction is that posing a random information query in this syntax cannot be done without implicitly or explicitly asking is "x data from a random source". However, such a question is not answerable without knowing the process by which the supposedly random information was generated. If it was generated from a sufficiently random method, then the generator is the one best equipped to answer this question and onlookers can do little to assess this. Accessing a given piece of random information amounts to assessing the generation source and this can only be done by observing if many samples from it are uniformly distributed. As such, individual oracle queries of this type have little utility.

The relationships between these sets of questions also suggests that there is a so-called "Hierarchy of Knowability" based on the question category (and therefore the size $|E|$), and

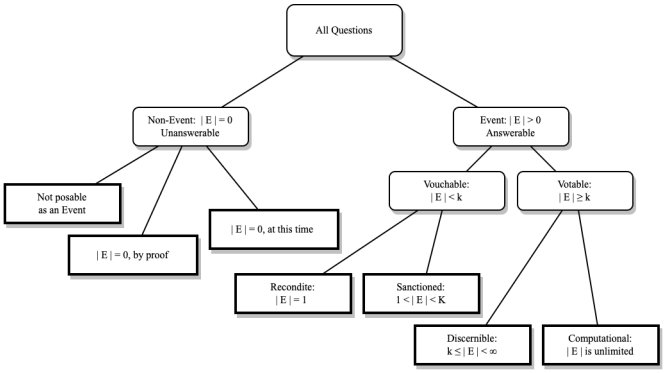

Fig. 1. A tree depiction of the question categories.

the corresponding assumptions required to answer each type of question. This hierarchy establishes a ranking of the strength to which a truth may be known. Conversely, the hierarchy also demonstrates limitations of what can be known based on the available level of trust.

We can consider:

1) Fundamentally unknowable or unanswerable questions. (For example, a question that is undecidable, open-ended, philosophical, or possesses zero answering entities).
2) Knowable without any trust. (This level of knowability seems unattainable).
3) Knowable only if you trust the origin and the mechanism. (Vouchable).
4) Knowable only if you trust the mechanism. The mechanism can produce a trusted answer from untrusted parties in a decentralized manner. (Votable).

In a practical sense, oracle mechanisms can only hope to answer questions that fall into knowability levels #3 and #4. These levels allow for the introduction of trusted sources and trusted mechanisms to mediate in the process of ascertaining certain information. Between these levels, the nature of the question determines the level of trust required. Items in level #1 are perhaps possible to philosophize about, but by being unknowable or unanswerable, we have little ability to pursue a rigorous answer with popular agreement. Likewise, level #2 offers little traction for practical methods. How could one come to know something without either trusting where it comes from, or trusting a mechanism that produces the answer? As we discussed early on, every step of computation and every network operation involves trust of some kind, whether it be in the hardware, the network, or elsewhere.

B. Type Definitions

1) Recondite: $|E| = 1$. The answering entity is $e \in E$, with $|E| = 1$. Here, by definition, the cardinality of E is only one. Then, for some $q \in Q$, we consider that q may be either objective or subjective given the current state of reality $r \in R$.

In the objective case, $\forall a, r \Pr[A = a|\ Q = q, R = r] = \Pr[A = a|\ Q = q, R = r, E = e]$ s.t. $|E| = 1$. In the subjective case, $\exists e, a, r \Pr[A = a|\ Q = q, R = r] \neq \Pr[A = a|Q = q, R = r, E = e]$ s.t. $|E| = 1$.

Yet, since there is only one entity in E, there is no other entity besides the one answering entity. The distribution of answers is whatever the single entity responds. In essence, there is no one to compare answers between, so all answers must be accepted as truth if $|E| = 1$. In possessing only a single answering entity, the distributions of answers will always be a single measurement and there can be no way to distinguish what type of distribution it represents, subjective or objective.

We can see that this category lends itself to answers obtained by vouching mechanisms as we do not have a population sufficient to collect a distribution of answers. In vouching for an answer, our mechanism can only attest to the authenticity or integrity of the data being conveyed, and must assume that what is being conveyed is true.

2) Sanctioned: $|E|$ is small.[2] Next, consider the case where more than one entity is able to answer a given question, but that the information is not broadly known. Instead of a single party, $|E|$ is some small positive integer $\leq k$. In this case, we consider the question to be a Sanctioned Event. These are called Sanctioned not because any particular party necessarily has direct control over the answers, but because some particular group of individuals are those who are knowledgeable about the answer and have authority to answer it in the given context.

In this category, we have in the objective case, $\forall a, r\ \Pr[A = a|\ Q = q, R = r] = \Pr[A = a|\ Q = q, R = r, E = e]$ s.t. $|E| \leq k$. In the subjective case, $\exists e, a, r \Pr[A = a|\ Q = q, R = r] \neq \Pr[A = a|\ Q = q, R = r, E = e]$ s.t. $|E| \leq k$.

Similarly as to Recondite Events, we do not have a population sufficient to collect a reliable distribution of independent answers. The small population of answering entities again limits our ability to seek a broad collection of independent measurements. However, we may be able to observe the difference between objective and subjective questions as there are parties to compare between. While not a possibility in the Recondite Event case, with Sanctioned Events we may begin to measure the variability in answers and use this to assess the confidence of the outcome, though still not without assuming trust in the answering entities. Thus, again our best option is to vouch for some assumed truth.

[2]Regarding "smallness": The choice of k and the particular meaning of "small" in the above definitions is a qualitative description that the cardinality of the set of answering entities, E, is small enough that the distribution of answers cannot be readily accessed to draw useful conclusions. In this way, "small" means that the possible answering parties are not numerous enough to form a meaningful distribution. The sense of "large" is then the opposite of this, that we can draw conclusions by assessing the distribution of answers. Importantly, the threshold between large and small is not singularly defined and may differ by application and confidence level. In general, it needs to be sufficiently large to allow the distribution of votes to produce a clear mode or modes. This number is not strictly fixed and it has the potential to be used as a control mechanism by oracle implementers. For example, choosing to increase this threshold can require that problems in the given context be answered only via consensus mechanisms and that vouching mechanisms (which involve a higher assumption of trust) are not applicable there.

3) Discernible: $|E|$ is large. Again, we can consider the next largest case of answering population size. If $|E| > k$ for some positive integer k, then we begin to be able to ask / measure from a broad population of answering entities. Instead of seeking specialized, contextually-relevant sources, we start to have room for some level of generality as answering mechanisms will be able to find a much broader collection of sufficiently knowledgeable parties.

In the objective case, $\forall a, r \Pr[A = a|\ Q = q, R = r] = \Pr[A = a|\ Q = q, R = r, E = e]$ s.t. $|E| > k$. In the subjective case, $\exists e, a, r \Pr[A = a|\ Q = q, R = r] \neq \Pr[A = a|\ Q = q, R = r, E = e]$ s.t. $|E| > k$.

At this scale, we begin to be able to assess enough independent answers to observe a useful distribution of answers. If we imagine varying the question such that the groups of individuals who pick a particular answer (affirming or denying the truth of the Event) changes, the distribution should shift.

Such Events are considered Discernible because ordinary individuals have a stronger likelihood of being able to discern then answer without specialized knowledge, and because we now have an answering population that is large enough to discern a meaningful answering distribution.

4) Computational: $|E|$ is unlimited; practically, it is some very large finite size, but there is no limit or knowledge requirement for who can be in the set. Computation is generalizable and attainable by any party with the means to evaluate an algorithm. So, $|E|$ can be any size as anyone may perform the computation. Then, for some computational question $q \in Q$, $\forall e, \Pr[A = \text{none}|\ Q = q, E = e] = 0$. This tells us that no one who is an answering entity for this question should provide a "none" answer.

Now, q may be either deterministic or nondeterministic. We assume that q is computable to find an answer 'a' in A in finite time. We say that f is our algorithm to compute q with $f(q) = a$ as our resulting answer. Given the current state of reality, r in R, r for a given computation is simply the set of inputs to the computation. If a computational question q is defined including its inputs, then r may be the empty set.

So $f(q, r)$ is the algorithm applied to the question along with its possible inputs. We note that reality comes into play if this computation has stateful inputs, but not always. Also, a question that is posed in terms of stateful inputs may not take inputs in as obvious a way. This implies that $\forall r, e \Pr[A = a|\ Q = q] = \Pr[A = a|\ Q = q, R = r, E = e]$.

In the deterministic case, $\exists a \Pr[A = a|\ Q = q, R = r, E = e] = 1$. And then, $\Pr[A = a = f(q, r)|\ Q = q, R = r] = 1$, or r is empty, which simplifies to $\Pr[A = a = f(q)|\ Q = q] = 1$. So, we can observe that anyone with the algorithm and its inputs can perform the computation to get the same result.

In the stochastic case, $\forall a \Pr[A = a|\ Q = q, R = r, E = e] < 1$, hence we have $\Pr[A = a = f(q, r)|Q = q] < 1$, or r is empty, which simplifies to $\Pr[A = a = f(q)|Q = q] < 1$. So, we can observe that even with having the algorithm and its inputs, one cannot necessarily expect to compute a single answer a. This implies that special attention will need to be paid to the distribution of answers and the level of variability expected in the result.

We again have a case where collecting votes (measurements) to establish a truth is a viable option. Uniquely for Computational Events, anyone may be able to answer as the cardinality of the set of answering entities has no specific limit. Anyone with computational resources and the ability to implement the specified algorithm will be able to produce an answer.

5) Non-Event: $\forall q \in Q, \forall e \in E$, there does not exist an answer. Certainly, if an answer to the question does not exist from any party, then there is no way an oracle mechanism could satisfy such a question. Stated another way, $|E| = 0$ for any reason implies that the question is unanswerable.

C. Examples

1) Recondite: These Events tend to be those that occur in such a manner that only a single individual has access to the required information. This can be either due to the proximity of the information or definitionally based on where the information is expected to come from.

- *What did John, who lives alone, have for breakfast?* (objective for a single individual)
- *How is John feeling today?* (subjective for one individual)
- *What is the temperature in Washington, DC, USA according to the weather station at KDCA airport?* (specific single measurement source)
- *What is John's citizenship?* (only the sovereign can authoritatively attest to an individual's citizenship)

2) Sanctioned: These Events tend to be where a small party has some monopoly or sovereignty over the truth itself or the verification of the truth.

- *Bilateral agreement:* May be commonly known and acceptable, but usually only authoritatively answerable by the two parties.
- *Price feeds:* While many parties may be willing to tell you a price at a given point in time, the only authoritative answer is the market (market-maker) itself.

Price feeds or price oracles are a highly popularly type of oracle mechanism due to the desire to use token prices within smart contracts. While numerous examples exist, they all function as attempts to address the same fundamental problem: token prices are inaccessible from within the blockchain itself. Token prices in terms of other currencies, whether cryptocurrency or fiat, exist only on markets where these tokens are being traded. As such, an oracle is needed to be able to access their prices within on-chain computations. Implementations are often platform specific and may differ in aggregation techniques, but all exist to collect market prices and convey them as an accepted truth. A price oracle that is directly associated with the market of interest would be optimal in being able to convey prices directly from the information source. Example implementations include Acala [9], Bluzelle [10], MakerDAO [11], Tellor [12], and Uniswap [13].

3) Discernible: These Events tend to be broadly observable at a global scale due to the information being highly accessible and easily obtainable, such as

- sports scores / outcomes,
- encyclopedia-type facts,
- world events,
- weather in a particular region.[3]

One can observe that many Discernible questions have answers that are directly accessible via some defined means, such as an Internet API, a specific text, etc. Such questions could conceivably be answered using a mechanism to vouch for the truth from the given entity, but would require predefined data models and an assumption of trust in that entity to utilize. These cases provide examples of questions where being able to receive answers from a broader population may be an advantage by decentralizing the answering mechanism.

4) Computational - Deterministic:

- Bitcoin, as an algorithm that repeatedly answers the same question in a distributed manner.
- $\text{fib}(n) = \text{fib}(n-1) + \text{fib}(n-2)$ and $\text{fib}(50) = 12586269025$.
- Anything computationally reducible given a known algorithm.

5) Computational - Non-deterministic:

- Algorithms with pseudo-random choices.
- Algorithms that requires approximating a result with limited precision.

6) Non-Event:

- Random oracles (unable to be posed as Events).
- Questions with zero answering entities. $|E| = 0$ at a given time.
- Questions proven to be undecidable. It has been proved that $|E| = 0$.

V. Validation of the Framework

In this section, we consider how our framework fits empirically with the traits of existing work, and in particular, existing embodiments of oracle mechanisms that have been proposed or implemented. In doing so, we validate that the model that our framework provides fits a broad collection of evidence and aligns with common assumptions. We also show that making use of the framework can provide new capabilities in the design and analysis of oracle mechanisms.

At the end of the section, we include Table 1 as a summary of the results discussed herein. Major outcomes include the level of trust required and implicitly associated with a given question type (and therefore oracles that answer the type), as well as how the method of conveyance for an oracle mechanism relates to trust level that is required for its use. This trust falls in two locations: the origin and the mechanism. Trust in "the origin" of information for an oracle is trust in the fidelity of the original source of that information. Trust in "the mechanism" is trust that the oracle mechanism is designed and operated in such a way that it can be trusted to produce the expected quality of results. We emphasize that the given oracle examples are not necessarily comprehensive, but meant to characterize the types of oracles which can be found to answer their corresponding categories. As new oracle designs are created, existing examples may be analogized to inform the categorization of the new ones.

[3]The subjective interpretation of weather brings up the issue that qualitative descriptions can have different meanings to different people. Even at the same time in the same place, different individuals may answer differently to the statement "The weather in X location is sunny" if their experience is overcast. Such an example reveals a further difficulty in wanting to formulate queries that are semantically less subjective in meaning.

A. New Capabilities

In existing work, there is not a strong emphasis on the types of questions that can be answered, what information or external context is needed to answer them, or the types of mechanisms that might more directly work for this question. Our framework introduces the ability to match classes of questions with classes of solution types. Additionally, it provides a method to consider the contents and syntax of questions beyond a specific oracle mechanism.

B. Unanswerability

While it may appear obvious from some perspectives, we find it important to explicitly acknowledge that not all possible questions are answerable. Such questions exist in the Non-Event category. These can be either questions that are unable to be posed as Events (i.e. randomness queries) or questions with $|E| = 0$. Questions can have $|E| = 0$ by representing a proven undecidable problem, by not conforming to Event syntax in another way (such as being open-ended), or by being well-formed but simply lacking any answering parties. In each of these cases, one cannot expect an answer by any oracle mechanism. It may not be readily clear if a question is unanswerable at a given time or at all times.

C. Context

We can observe that context can be the differentiating factor between different honest parties' answers. As such, complete context is necessary to expect any consistency of answers. However, even significant context is not necessarily sufficient to guarantee consistent answers due to individual perspectives and interpretations of language, but it is likely the best we can do within the limitations of natural language. We can also easily devise examples where a slight shift in context changes the answer of the question.

D. Variations on Questions

An important facet of our framework is the categorization that produces equivalence classes by methods to answer said questions. For example, we can demonstrate how seemingly unrelated domains can have a similar oracle solution in this model, and conversely that seemingly similar questions can require different oracle solutions in this model.

Similar subject matter can require different types of answering mechanisms. A question about the weather broadly

in Washington, DC vs. weather according to the station at KDCA have very different methods of answering: the first is something that voting methods are likely able to answer, whereas the second likely requires a vouching method due to the specificity of the location.

In another case, even differing subject matter may be answerable by similar mechanisms. For example, weather broadly and sports scores of publicly held matches have similar answering methods: both are likely something that can be voted on. Notably, both categories here are Discernible Events.

Our framework captures these distinctions by grouping questions by answering population and provides a method for determining what possible solutions may apply to a given question. With this paradigm, we can provide a means for categorizing similar oracle mechanism methods in a way that is likely to allow for transfer of solutions between problems in the same category.

E. Vouching Mechanisms

A major category of oracle mechanisms are embodiments of what we describe as "vouching mechanisms". These are mechanisms that take information from a given source as trusted, and then use cryptography to convey the authenticity and/or the integrity of the information to a destination.

Such mechanisms are inherently situational and are used to convey information from a single source which is assumed to be trusted, for if it was not trusted for some use, then there would be no purpose in ascertaining the authenticity or integrity of it. We can observe that these vouching based mechanisms speak directly to Event types that have a small or single answering entity and are clearly designed around this purpose. Notable vouching mechanisms include DECO [14], TLSNotary [15], PADVA [16], Town Crier [17], and Chainlink [18].

The proposals for DECO, TLSNotary, PADVA, and Town Crier are all attempts to design a system that augments the existing Transport Layer Security (TLS) implementations that are already popular through the use of the web. They each are designed to answer web based queries that would contain personal information corresponding to a single circumstance. In these cases, the mechanisms act to convey information in a transaction from a trusted third-party to another, thereby vouching for it. Proposed use cases include transactions such as confirming date of birth or citizenship. These mechanisms speak directly to Recondite and Sanctioned Events as strong candidate solutions for both Event types due to the highly limited populations that are able to answer such questions authoritatively.

Chainlink is a popular and complex implementation of such a vouching scheme where there are integrations with numerous data sources that receive ratings based on their reliability. Users pay the Chainlink network for access to data from this collection of integrations. The network acts as a data consolidator and vouches for truth of integrated data sources. The integrations are incentivized to behave honestly in order to maintain their ratings and inclusion in the network. While it possesses a more complex incentive structure, Chainlink is fundamentally a scheme for vouching for information that is assumed to be true as well.

While there are several mechanisms proposed for the purpose of "vouching", the commonality between them is that the population of entities that can answer to a given query is quite small. This is precisely the defining characteristic of our Recondite and Sanctioned categories.

F. Voting Mechanisms

A second major category of oracle mechanisms are what we describe as "voting mechanisms". These are mechanisms that serve to "crowd-source" their sense of truth from a decentralized population of participants. Such mechanisms tend to operate with direct votes from individual participants, or staked voting using a token based measure of value, but the voting style can vary greatly while still relying on the central principle of a large answering population. For example, in [19] the authors describe a hybrid voting-based mechanism, focused on interoperability, that requires sufficient validators to assess the truth of a result that is then collected by the aggregator. While structured differently than direct voting, the mechanism relies on similar scale requirements with respect to the types of questions that other voting mechanisms can serve.

There are two significant types of voting mechanisms that appear. One type are prediction markets where an oracle query is created and then there is a market where participants can trade outcomes to bet on the truth. Examples of such mechanisms include Augur [20], Hivemind [21], and Delphi [22].

Augur is a platform for creating prediction markets for questions in any form. The market life cycle has many phases that begin with Market Creators putting up multiple bonds to pose a question, and ends with a resolution on the truth of the question and a settlement for market participants. Between market creation and market settlement, anyone can participate in trading of tokens representing the possible question outcomes. If the designated reporter fails to report on the outcome or there are outcome disputes, the true outcome of the market is determined by the holders of a native Reputation token (REP). The system encourages a consensus outcome by rewarding those who stake their REP on the outcome they belief is true and requiring the forfeiture of REP on non-consensus outcomes. As a last resort, Augur also uses the threat of highly disruptive forks to strongly encourage users to reach a consensus.

Similarly, Hivemind and Delphi are presented as frameworks for user created prediction markets with flexible question formats. Hivemind allows for Boolean and scalar questions, with outcomes that are decided by a limited set of voters who are obligated to use their "VoteCoin" to select what they believe is true. Delphi considers categorical and scalar questions and describes a structure for weighted multi-signature outcome collection, with weights that can be assigned flexibly. While less detailed in their proposals than Augur, they do demonstrate the popular interest of implementers in creating

networks that take advantage of the strong economic incentives created by prediction markets.

Prediction markets do possess certain risks, as there tends to be no cap or limit on how much can be bet on a particular outcome. Large enough bets can, in some circumstances, act to incentivize participants to influence the result toward particular (possibly socially / morally undesirable outcomes) outcomes.

The second significant type of voting mechanism are direct-voting games that are run in such a way that participants have economic incentive to vote "true" only for propositions which they honestly believe to be true. Some examples of such games are Astraea [23] and Shintaku [24].

Astraea is a decentralized voting game where there are three categories of participants. "Submitters" pose propositions to the system. "Voters" maintain a low-risk, low-reward role where they put in a small monetary stake to be given a random proposition to vote on. "Certifiers" have a high-risk, high-reward role where they put in a comparatively large monetary stake to certify with high confidence that they believe a given outcome is correct. Both Voters and Certifiers are rewarded when they vote along with the majority / consensus result after the proposition is settled. Stake submitted with votes on non-consensus outcomes is redistributed to fund the rewards for those who answered the consensus outcome.

Shintaku is a proposal for minor modifications to Astraea in order to avoid degenerate voting, but otherwise relies heavily on the analysis put forth for Astraea. Degenerate voting can occur in voting games when there is a method of collective behavior that results in rewards from the system without needing to uphold honest behavior. One simple way that this can occur is if a sufficiently large population of answering parties colludes to always vote the same way. Voting games that simply reward players that voted in the same way as the majority would still pay out in such a circumstance, despite there being no reason to believe the selected outcome is the one that reflects reality. Shintaku avoids this by presenting voters with pairs of propositions at a time, and only paying out rewards when users vote differently between their two propositions.

We consider both major voting mechanism types equivalently here. Regardless of the specific details of how votes are tabulated, the core assumption for these mechanisms is that there is a large number of answering entities that all have sufficient information to answer accurately. Notably, [20] goes as far as to call out this detail in asserting that the mechanism should only be used for queries which are generally answerable by the public. This assumption is precisely what permits such voting schemes to function, as without a sufficiently large answering population the economic incentives that ensure the mechanisms produce valid answers can break down.

We note that these are precisely the circumstances that define Discernable and Computational Events. Both of these categories are constituted by questions which have a large enough population of answering entities to be able to assess the distribution of the answers and assert the outcome.

TABLE I
ORACLE QUESTION TAXONOMY RESULT SUMMARY

Size of $\lvert E\rvert$	Question Properties			
	Event Type	***Method***	***Trust Required***[a]	***Examples***
0	Non-Event	N/A	N/A	Section V.B
1	Recondite	Vouching	O & M	Section V.E
$< k$	Sanctioned	Vouching	O & M	Section V.E
$\geq k$	Discernible	Voting	M	Section V.F
∞	Computational	Voting	M	Section V.F

[a]"O" denotes "the Origin" and "M" denotes "the Mechanism".

VI. EVALUATION OF EXISTING WORK

In this section, we evaluate how existing works can be considered in the context of our framework, and what they are doing correctly or could improve as implied by the framework. Several of these points could be identified as strengths or weaknesses otherwise, but this framework is particularly helpful in giving a model to systematically consider how well a mechanism suits the traits of the types of questions it is trying to answer.

A. Context is Key

Mechanisms need to focus on ways to embed sufficient context, either through being domain specific and only requiring a few details to determine a fully specified query, or by creating methods that allow the specification of all facets of context to reduce the chance of ambiguity disrupting their operation.

B. Prefer a Limited Focus

Several existing attempts at oracle mechanisms try to solve many categories at once. However, mechanisms that solve Discernible and Computational problems are unlikely to be functional for any questions that have small answering populations and vice versa. Mechanisms that solve problems where voting is applicable do not readily apply to circumstances where you have to trust a party to have an answer.

Instead, mechanisms should be designed with clear assumptions about the trust involved. They should be designed with specific, well defined use cases in mind instead of trying to cover many possible avenues and doing so less well. Such mechanisms make it easier to include the sufficient context, either by being explicitly about it or by having clearer expectation. Domain specific mechanisms have a greater chance of having access to sufficient populations of answering entities to ensure questions can be answered. This can be due to direct association with a certain information source, or by attracting relevant answering parties by domain.

C. Integration, Adoption, and Fragmentation

Mechanisms that attempt to augment TLS such as the aforementioned DECO [14], TLSNotary [15], PADVA [16], and Town Crier [17] have promise for Recondite queries due to their ability to integrate with existing technologies that have already achieved widespread adoption. These also have the benefit of being directly associated with much of the context

required to query unambiguously. Such vouching mechanisms would typically be used for satisfying a particular circumstantial requirement at a point in time. As seen in examples such as those described in [14], an instance where there is a need for proof of age tends to be inherently associated with a specific online transaction that itself would specify who, what, when, and by what means the query applies.

A notable weakness of voting mechanisms that currently exist is an assumption that question and answer data can be easily parsed by participants of the system. For example, Augur [20] attempts to solve a large number of query types from plain form questions to API queries. Without a precise data model, most or all of these queries will involve the engagement of a person. Without a clear structure of the data to be considered, it may prove difficult for a mechanism to ensure that there are sufficient answering parties to reliably service a voting question, without which the integrity of the answer can come into question.

D. Under-Addressed Categories

From one perspective, we can observe an apparent lack of oracle mechanisms that are specifically designed to speak to Computational Event inquiries. While there are some existing examples of oracle mechanisms that support such questions, there has been substantially less attention in this domain than vouching or voting mechanisms designed to answer the other three categories. Existing implementations include Chainlink [18] and Provable [25]. In addition to its previously described vouching features, the Chainlink Keepers functionality provides off-chain smart contract computation. Similarly, Provable offers an interface for calls to off-chain computational resources for arbitrary computation, specified via a Dockerfile to be run in a virtual machine.[4] In both of these cases, the user still has to trust the entity performing the computation. Thus, there is room for more decentralized implementations that address Computational Events.

Yet, from another perspective, we can observe that computational oracles exist in case specific forms for distributed consensus algorithms. For example, in a sense, proof of work algorithms such as those in Bitcoin [26] and Ethereum [27] are a computational oracle which simply answers the same algorithmic question repeatedly. The difference being that these mechanisms seek primarily to perpetuate the blockchain itself, rather than answer any externally useful inquiry. However, if a Computational Event oracle mechanism was designed to support perpetuating the blockchain through answering queries, it could potentially make for a robust oracle mechanism with built-in incentive to perpetuate itself. This could pose a direction for inquiry into one of the long standing questions about Bitcoin: how to ensure incentives surrounding its perpetuation are stable in the long term.

[4]Dockerfiles are the specification format for images in the popular containerization system Docker. They are a sequence of commands that are executed to build the image, with optional startup commands that run when the image is instantiated as a container. See: https://docs.docker.com/engine/reference/builder/

E. Price Oracles

Fundamentally, prices are a feature of markets. They are established as the opposing "forces" of supply and demand reach a dynamic equilibrium, but are constantly in flux. Accordingly, the location to look for answers to price inquiries is in the market itself and only directly answerable by the market-maker who is settling transactions. Prices of a single item can and often do vary between distinct markets, but these are two or more distinct values unless some form of aggregation is specified.

Arbitrageurs take advantage of this divergence in pricing between two or more markets. In the long-term, the prices for Bitcoin in Dollars should converge across exchanges because of arbitrage actors, not because there is an inherent, fixed price of Bitcoin. As such, a party inquiring the price of something would be required to specify what (the item of interest), where (the market), and when that information should correspond to (the time). In fact, the base query, "What is the price of Bitcoin versus the US Dollar?" is a non-answerable query. This is because the price of Bitcoin versus the US Dollar only has meaning on a given exchange at a given time. With this in mind, we find that the question "What is the price of Bitcoin versus the US Dollar on the Coinbase exchange at 5pm Eastern Time on January 4, 2022?" is a meaningful query. It is the last trade price at or before the specified time on the specified exchange.

From the perspective of our framework for Events, the proliferation of distinct price oracles is quite curious. If there is only one place to look for prices directly, then an effective oracle would be directly associated with the market itself. Relying on a third-party to collect this information and then vouch for it unnecessarily extends the chain of trust involved.

Most of the differentiating facets of these price (feed) oracles would seem to be not in the *information* that they convey, but in some other feature of how it is aggregated, processed, or conveyed. This is a behavior that can be observed by comparing various price feed oracle implementations. Each price feed oracle is effectively performing the same function with varying assumptions about how to process the information and where to look. This fragmentation is not only unnecessary, but also harmful to the adoption of the technologies.

We are often reminded of the latent risk associated with oracle failures by the manipulation of price oracles.[5] While the consequences of lapses in oracle integrity are not always obvious in general, vulnerabilities in price oracles are a distinct reminder of the financial impacts of unreliable oracles.

This framework suggests that an efficient and effective price oracle for a given market would be a vouching mechanism that conveys values for Recondite or Sanctioned Events (depending on the precise answering population size at hand). In such an implementation, the mechanism and location would both be fixed context. A user would simply need to convey the

[5]See the DEUS Finance DAO Price Oracle manipulation of 2022 March 15, which resulted in the loss of at least 3 million USD: https://lafayettetabor.medium.com/deus-post-mortem-3c65df12927f

remaining open context of what they desire the price of and when.

VII. Conclusion

A. Summary

We have proposed a framework for thinking about oracle queries in terms of their answer population sizes (the cardinality of their answer entity sets, $|E|$). We have established a model that covers all categories of questions disjointly and groups questions with similar oracle mechanisms together.

From this categorization, we can derive commonly accepted conclusions and justify the operation of existing oracle mechanisms. Importantly, we can also observe that the framework demonstrates that assumptions about the degree with which we trust a given information source are embedded in solution types based on which category they are addressing.

We observe that these disjoint categories also relate to the level of assumptions that are required to answer such questions. Our model suggests some questions are fundamentally unanswerable without certain assumptions, and similarly, the level of trust that can be assumed allows some oracle mechanism types but will prohibit others.

The only assumptions that we invoke are that questions can be framed as logical statements, which we find to be reasonable, but most importantly a useful method for posing and reasoning about questions. The syntax excels by allowing us to perform logical manipulations and to frame all possible questions in terms of three answers, namely true, false, and unknown.

B. Future Work

We see this work as a foundation for considering the effectiveness of existing oracle implementations, as well as informing where to look when designing new implementations for a specific question type. Instead of trying to span question types, further investigation should focus on solutions that work well for one type or transfer within the same question type.

We recognize a lack of strong solutions to the Computational and Discernible categories and believe that this framework can inform methods for improving upon existing mechanisms. We hope that this way of thinking can be used as a step towards the following goals of:

- considering mechanisms to embed the necessary context in oracle question-answering mechanisms;
- considering how we can observe and assess the honesty of answering entities by the distribution of their answers;
- devising mechanisms to address unaddressed or less strongly addressed categories of questions;
- further evaluating existing oracle mechanisms in terms of our framework to determine where improvements can be made to the quality and robustness of their answers;
- designing new oracle implementations with query context as a first class priority.

Appendix A
Examples of Unreasonable Queries

This appendix is a non-exhaustive demonstration of example oracle queries that are for some reason lacking in formulation. It should be noted that some of the given queries are unreasonable due to insufficient context as *explicitly* detailed; however, similar queries may appear in practice due to some implicit context associated with the oracle mechanism in use. In these cases, further investigation would be necessary to determine if the oracle mechanism itself satisfies the seemingly underspecified context.

Additionally, these examples can be used as a model thought process for evaluating whether a given query or query making process is sufficiently precise to avoid ambiguity. We provide a brief explanation as to why such a query is not properly specified. This way of thinking can then inform similar reasoning for disparate subject matter in other possible questions.

A. Token Price

Example Query: What is the price of Bitcoin?

This is insufficiently specified for many reasons, but most significantly due to being open-ended with merely an implied answer space of "some currency value".

Example Query: What is the price of Bitcoin in USD?

This is an improvement in that now the question's answer space is at least in terms of some specific scalar value. It may still be difficult to collect reliable answers for without being in the form of a proposition. The query still does not specify where to look for the price.

Example Query: The price of Bitcoin in USD is greater than $41,000.00.

This is a dramatic improvement and now represents an specific propositional claim that can either affirmed or denied. However, the query still lacks specificity as for when, where, and how to answer it. While still ambiguous as written, depending on the implicit contextual assumptions of a given oracle mechanism, this may be answerable in practice.

Example Query: (The price of Bitcoin in USD is greater than $41,000.00.) and (time is 2022 March 14 at 18:28 UTC) and (market of interest is Coinbase).

This query now has a base query with a truth value and possesses precise context for both when and where the query applies. However, the query itself does not specify by what means to acquire an answer. It could conceivably be posed to a variety of mechanisms and receive differing answers. Since the only direct information about market prices lies with the market-maker itself, a maximally precise and minimally trusting mechanism would access a price oracle directly associated with the market. (See Appendix B.A for the complete query).

B. Environmental Sensing

Example Query: What is the air quality like today?

This query is insufficiently specified in many ways, but principally because it asks an open-ended question. Such a question does not immediately have any (even implicit) answer space other than plain language.

Example Query: The air quality today is good.

This query is an improvement as it makes a particular claim which can be either affirmed or denied by answering parties. However, the subject matter itself is highly subjective. An answering parties' response could vary due to their age, physical fitness, chronic health conditions, and more. Even aside from lacking sufficient context as for where, when, and how to answer the query, it is unlikely that an oracle mechanism could collect consistent answers for such a question in most circumstances.

Example Query: (The Air Quality Index in Washington, DC is orange) and (time is 2022 March 20).

This query is significantly better as it uses the Air Quality Index[6] metric which has a standard definition and summarizes air quality based on multiple factors. While the time context for this is less precise than other examples and greater precision should generally be preferred, it is conceivable that a one day window is a desirable interval that can occur in practice. Of greater concern is the loose sense of location (where) context. With a data source (such as AQI, or any other) that can vary considerably throughout a geographic region, it would be wise to specify a geographic center to reduce the variability in answers. Geo-coordinates could likewise be specified for apparently maximum precision, but, for many queries, a landmark based geographic center may be sufficient to disambiguate and produce consistent answers. In either case, the emphasis here is that the reduction in the number of possible information sources decreases ambiguity.

Example Query: (The Air Quality Index near the Lincoln Memorial in Washington, DC is orange) and (time is 2022 March 20).

This query has a base query that includes both the subject matter of the question (what) as well as the location for the query (where). It also includes additional context for when the query should correspond. As written, such a query lacks information about by what means to acquire the answer; however, in practice this query may be sufficient to be presented to an oracle that is implicitly associated with a specific mechanism for receiving the answer. Crucially, as it is explicitly written, this query may be answered by different mechanism types that have different means of acquiring the answer. Different mechanism types may produce different answers to seemingly similar questions, and they also have differing trust requirements for their use. See items B, C, and D in Appendix B for complete queries demonstrating this distinction.

APPENDIX B
EXAMPLES OF REASONABLE QUERIES

This appendix is a non-exhaustive collection of example oracle queries that can be considered to be reasonably complete and adequately specified. Our intent is to offer easy to understand examples with characteristics that parallel a variety of practical oracle questions. We provide a brief explanation as to the circumstances where such a query may appear.

A. Token Price (Recondite)

Example Query: (The price of Bitcoin in USD is greater than $41,000.00.) and (time is 2022 March 14 at 18:28 UTC) and (market of interest is Coinbase) and (price oracle is Coinbase).

We see that this query adequately specifies a base query that possess a truth value, as well as context for when, where, and by what means to receive an answer. For price oracles, particular emphasis should be placed on the last facet, as the market itself it the only direct source of price information. This model query analogizes with other market based queries for token prices, option prices, or other financial derivatives.

B. Environmental Sensing (Discernible)

Example Query: (The Air Quality Index near the Lincoln Memorial in Washington, DC is orange) and (time is 2022 March 20 at 16:00 UTC) and (source is anyone).

This query asks anyone with ownership or purview of the corresponding air quality information to answer. Such a query is a Discernible Event because it has a large answering population, that may rely on a variety of information sources. Some answering parties may possess their own low-cost air quality monitors, others may look to governmental sources, and some may simply check the Internet. For this type of query, one can expect variability in information sources. However, such a query can also utilize the lowest assumptions of trust as an oracle mechanism for Discernible Events may crowd-source a consensus from disparate sources.

C. Environmental Sensing (Recondite)

Example Query: (The Air Quality Index in Washington, DC is orange) and (time is 2022 March 20 at 16:00 UTC) and (source is www.airnow.gov).

Such a query seeks to answer a very similar base query as to the previous example, but instead specifies a specific information source. While there may be answering parties other than the one specified, this is a Recondite Event because a single answering entity is defined. This kind of query is liable to represent a more localized (and perhaps less representative) observation than a corresponding Discernible Event, but permits a far simpler vouching based oracle mechanism. Note that posing a query for this subject matter as a Recondite Event instead of a Discernible Event is possible only if the circumstances allow one to trust the information from that specific source.

D. Environmental Sensing (Sanctioned)

Example Query: (The Air Quality Index near the Lincoln Memorial in Washington, DC is orange) and (time is 2022 March 20 at 16:00 UTC) and (source is EPA-certified monitoring stations).

This query asks for the same fundamental information as the previous environmental sensing queries, but requires that

[6]The Air Quality Index (AQI) is a United States Environmental Protection Agency (EPA) combined metric for conveying the impacts of five preeminent environmental pollutants that are associated with health risks. See: https://www.airnow.gov/aqi/aqi-basics/

the information be sourced from EPA-certified monitoring stations only. Such a query is unique in that many parties may theoretically be able to answer to the subject matter of the question, but the information source specification gives a special authority to particular measurements which are known to be of a certain quality. This query is a Sanctioned Event because there is special monopoly over its truth possessed by the particular parties qualified to answer it.

E. Sports Betting (Recondite)

Example Query: (The Boston Red Sox win their game on 2021 October 5) and (source is ESPN.com) and (time is 2021 October 6 at 12:00 UTC).

Oracle Events may correspond to real-world, temporally specific events that not only specify when the event takes place, but also a finalization time. This finalization time is the time (when) context that the oracle needs to be able to know when to answer the question. In contrast, the other date actually acts to specify the subject matter. This query is a Recondite Event because it specifies a particular single source from which to retrieve the answer. Even if this Event may be something that many independent parties observed, if one can assume trust in a particular data source, an oracle mechanism may be considerably simpler by merely vouching for the trusted answer. Without access to such an assumption of trust, this type of query would need to be targeted to a much broader voting based mechanism instead.

F. Sports Betting (Discernible)

Example Query: (The Boston Red Sox win more than 90 games in the 2022 season) and (source is anyone) and (time is 2022 October 6 at 04:00 UTC).

This query represents a "futures bet" and does not correspond to a single temporal real-world event. Such a bet is made before the season starts and has an answer that is only known after the conclusion of the season. This query is a Discernible Event because it is presented in such a way that any knowledgeable party may answer. This kind of long-lived query is also unique amongst the other examples here: it is desirable for an oracle to be able to answer questions, conveyed at present, not only in the near term, but arbitrarily far into the future.

G. Proof of Citizenship (Recondite)

Example Query: (Alice is a United States citizen) and (source is United States Department of State) and (transaction is a web-session at 2022 April 1 at 16:47 UTC).

This query is an example of someone trying to assert their citizenship to a third-party over the Internet. A query for proof of citizenship is a direct example of a Recondite Event that clearly requires an assumption of trust in the information source to be answerable: only the sovereign can authoritatively attest to an individual's citizenship. Even for an attribute like citizenship which does not change frequently (if ever), it is key for a reliable oracle to associate such a query with specific time context. Without context for when the query is meant to correspond, an oracle could also be at risk of compromising its future integrity should the authority of truth for this information ever claim otherwise.

H. Specific Algorithms (Computational)

Example Query: The SHA256 hash of the ASCII string "Oracl3sAr3Gr3at!" is "DB4B0AC7 9BC1EBA6 8529C258 CC41FC7C 56413061 7CF9F41C 2F45BE97 FF0C2D3D".

This query is a Computational Event that specifies an algorithm to use along with its input and expected output. Although the query itself does not provide a definition for the algorithm, such a query would often be able to rely on the standardized definition and well-known meaning of the SHA256 algorithm. Of course, greater interoperability with automated systems could be offered by providing a specification of the algorithm to be used in a standardized form. However, this is not necessarily required, even for a Computational Event, when the meaning of the query would be well understood by parties of the system.

I. Result Confirmation (Computational)

Example Query: The greatest common divisor of 123456789 and 38 is 1.

While this query does not specify a particular algorithm to use, there are a number of well-known algorithms that could be used to compute the answer. In this way, it is possible to pose Computational Events without necessarily specifying how to calculate the result. Instead, this type of query relies on the common mathematical definitions of these terms, and can still be evaluated by any party using any algorithm. One can observe that, despite its short form, this query is posed unambiguously as no further context is needed to receive consistent answers.

REFERENCES

[1] Giulio Caldarelli. Understanding the blockchain oracle problem: A call for action. *Information*, 11(11):509, 2020.

[2] Jan Mendling, Ingo Weber, Wil Van Der Aalst, Jan Vom Brocke, Cristina Cabanillas, Florian Daniel, Søren Debois, Claudio Di Ciccio, Marlon Dumas, Schahram Dustdar, Avigdor Gal, Luciano García-Bañuelos, Guido Governatori, Richard Hull, Marcello La Rosa, Henrik Leopold, Frank Leymann, Jan Recker, Manfred Reichert, Hajo A. Reijers, Stefanie Rinderle-Ma, Andreas Solti, Michael Rosemann, Stefan Schulte, Munindar P. Singh, Tijs Slaats, Mark Staples, Barbara Weber, Matthias Weidlich, Mathias Weske, Xiwei Xu, and Liming Zhu. Blockchains for business process management - challenges and opportunities. *ACM Transactions on Management Information Systems (TMIS)*, 9(1):1–16, 2018.

[3] Ingo Weber, Xiwei Xu, Régis Riveret, Guido Governatori, Alexander Ponomarev, and Jan Mendling. Untrusted business process monitoring and execution using blockchain. In *Proceedings of the 14th International Conference on Business Process Management (BPM), Rio de Janeiro, Brazil, September 18-22, 2016*, page 329–347, 2016.

[4] Roman Mühlberger, Stefan Bachhofner, Eduardo Castelló Ferrer, Claudio Di Ciccio, Ingo Weber, Maximilian Wöhrer, and Uwe Zdun. Foundational oracle patterns: Connecting blockchain to the off-chain world. *arXiv*, page 35–51, 2020.

[5] Abdeljalil Beniiche. A study of blockchain oracles. *arXiv preprint arXiv:2004.07140*, 2020.

[6] Jonathan Heiss, Jacob Eberhardt, and Stefan Tai. From oracles to trustworthy data on-chaining systems. In *Proceedings of the 2019 IEEE International Conference on Blockchain (Blockchain)*, page 496–503, 2019.

[7] Xiwei Xu, Cesare Pautasso, Liming Zhu, Qinghua Lu, and Ingo Weber. A pattern collection for blockchain-based applications. In *Proceedings of the 23rd European Conference on Pattern Languages of Programs (EuroPLoP'18)*, pages 3:1–20, 2018.

[8] Giulio Caldarelli and Joshua Ellul. The blockchain oracle problem in decentralized finance—a multivocal approach. *Applied Sciences*, 11(16):7572, 2021.

[9] Acala. Oracles. Available at: https://wiki.acala.network/karura/defi-hub/kusd-stablecoin/stability-and-liquidation/oracles, 2021. Accessed: 2021-12-18.

[10] Bluzelle. Decentralized oracles powered by Bluzelle. Available at: https://bluzelle.com/oracles, 2021. Accessed: 2021-12-18.

[11] MakerDAO. Oracle security module (OSM) - detailed documentation. Available at: https://docs.makerdao.com/smart-contract-modules/oracle-module/oracle-security-module-osm-detailed-documentation/, 2021. Accessed: 2021-12-18.

[12] Tellor. Tellor: Whitepaper. Available at: https://docs.tellor.io/tellor/whitepaper/introduction, 2021. Accessed: 2021-12-18.

[13] Uniswap. Oracles. Available at: https://docs.uniswap.org/protocol/concepts/V3-overview/oracle, 2021. Accessed: 2021-12-18.

[14] Jay Ligatti, Xinming Ou, Jonathan Katz, Giovanni Vigna, Fan Zhang, Deepak Maram, Harjasleen Malvai, Steven Goldfeder, and Ari Juels. DECO: Liberating web data using decentralized oracles for TLS. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS '20)*, page 1919–1938, 2020.

[15] TLSNotary. TLSNotary - a mechanism for independently audited HTTPS sessions. Available at: https://tlsnotary.org/TLSNotary.pdf, 2014. Accessed: 2021-08-31.

[16] Pawel Szalachowski. Padva: A blockchain-based tls notary service. In *Proceedings of the 2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)*, page 836–843, 2019.

[17] Edgar Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew Myers, Shai Halevi, Fan Zhang, Ethan Cecchetti, Kyle Croman, Ari Juels, and Elaine Shi. Town Crier: An authenticated data feed for smart contracts. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, page 270–282, 2016.

[18] Lorenz Breidenbach, Christian Cachin, Benedict Chan, Alex Coventry, Steve Ellis, Ari Juels, Farinaz Koushanfar, Andrew Miller, Brendan Magauran, Daniel Moroz, Sergey Nazarov, Alexandru Topliceanu, Florian Tramèr, and Fan Zhang. Chainlink 2.0: Next steps in the evolution of decentralized oracle networks. Available at: https://research.chain.link/whitepaper-v2.pdf, 04 2021. Accessed: 2021-08-31.

[19] Michael Sober, Giulia Scaffino, Christof Spanring, and Stefan Schulte. A voting-based blockchain interoperability oracle. *arXiv preprint arXiv:2111.10091*, 2021.

[20] Jack Peterson and Joseph Krug. Augur: a decentralized, open-source platform for prediction markets. *CoRR*, abs/1501.01042, 2015.

[21] Paul Sztorc. Truthcoin. Available at: https://bitcoinhivemind.com/papers/truthcoin-whitepaper.pdf, 12 2015. Accessed: 2021-08-31.

[22] Delphi. Delphi whitepaper. Available at: https://delphi.systems/whitepaper.pdf, 2021. Accessed: 2021-12-18.

[23] John Adler, Ryan Berryhill, Andreas Veneris, Zissis Poulos, Neil Veira, and Anastasia Kastania. Astraea: A decentralized blockchain oracle. In *Proceedings of the 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, page 1145–1152, 2018.

[24] Ryuuji Kamiya. Shintaku: An end-to-end-decentralized general-purpose blockchain oracle system. Available at: https://gitlab.com/shintaku-group/paper/raw/master/shintaku.pdf. Accessed: 2021-08-31.

[25] Provable. The provable blockchain oracle for modern DApps. Available at: https://provable.xyz/. Accessed: 2021-08-31.

[26] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Cryptography Mailing list at https://metzdowd.com*, March 2009.

[27] Gavin Wood et al. Ethereum: A secure decentralized generalised transaction ledger. Available at: https://github.com/ethereum/yellowpaper/, 2021. Accessed: 2021-12-18.