Eliciting High Quality Feedback from Crowdsourced Tree Networks using Continuous Scoring Rules

Ratul Ray, Rohith D. Vallam, and Y. Narahari
Department of Computer Science and Automation,
Indian Institute of Science, Bengaluru, Karnataka, India
ratul@csa.iisc.ernet.in,rohithdv@csa.iisc.ernet.in,hari@csa.iisc.ernet.in

Abstract

Eliciting accurate information on any object (perhaps a new product or service or person) using the wisdom of a crowd of individuals utilizing web-based platforms such as social networks is an important and interesting problem. Peer-prediction method is one of the known efforts in this direction but is limited to a single level of participating nodes. We non-trivially generalize the peer-prediction mechanism to the setting of a tree network of participating nodes that would get formed when the query about the object originates at a root node and propagates to nodes in a social network through forwarding. The feedback provided by the participating nodes must be aggregated hierarchically to generate a high quality answer at the root level. In the proposed tree-based peer-prediction mechanism, we use proper scoring rules for continuous distributions and prove that honest reporting is a Nash Equilibrium when prior probabilities are common knowledge in the tree and the observations made by the sibling nodes are stochastically relevant. To compute payments, we explore the logarithmic, quadratic, and spherical scoring rules using techniques from complex analysis. Through detailed simulations, we obtain several insights including the relationship between the budget of the mechanism designer and the quality of answer generated at the root node.

Categories and Subject Descriptors

I.2.1 **[Artificial Intelligence]**: Applications and Expert Systems - Games

General Terms

Economics, Algorithms, Design, Experimentation

Keywords

Mechanism Design; Social Networks; Crowdsourcing; Peer-Prediction; Continuous Scoring Rules

1. INTRODUCTION

Eliciting honest feedback from experts as well as lay persons is an important problem in electronic markets and web-based platforms. Decision makers often depend on feedback given by multiple individuals while making decisions. Increasingly, online applications are trying to extract knowledge from a large group of users, which is called *wisdom of the crowds* [1]. The query posed to these individuals may be objective, the answer of which can be verified to be correct or wrong at a future point of time, for example: *which soccer team is going to win World Cup? Who is going to be the next president of a particular country? What will be the stock price of product X in next month?* etc. For these questions, it is easy to compare the feedback obtained with the *correct* answer. *Prediction markets* [2] precisely serve this purpose. The situation becomes more interesting when the question asked is hypothetical, that is, when the answer to the question is subjective rather than objective. Examples of such questions include: *How do you rate the restaurant X? Is it good for government of X to adopt policy Y?, What is the source of anomaly in this system?*, etc. These type of queries are especially relevant in online reputation systems, where users leave feedback about quality of some product or service. In our work, we focus on the latter type of questions. Our goal is to use the wisdom of the crowd or a social network to elicit a high quality answer to such a question.

The problem studied in this paper has a wide range of applications. For example if a social planner (for example, government) is interested in collecting opinion about a new bill/policy, the planner can use binary signals: *good* or *bad* and obtain collective feedback from experts and citizens by incentivizing them in some appropriate way. On the other hand it can be used to detect source of anomaly in a system by the administrator: suppose there are M possible sources of anomaly. Each subsystem can play the role of a node in the tree and report a probability distribution over M sources.

The success of such a system which tries to elicit opinions from individuals faces two challenges. Firstly it has to ensure that users put in enough effort to get the report, because reporting feedback does take time and effort. For example in order to provide a review about a product, one has to first buy and use the product or at least have enough knowledge about the product. Also the buyers have to understand the rating scale. Again some manual effort is involved in preparing and submitting the report. If no explicit reward is given to the agents, then it is possible that they provide feedback only when they have some ulterior motives, which results in a biased feedback [3]. The second challenge is to guarantee honesty among the agents. Rational agents try to maximize their utility and it may not always be a best response to report the truth. To overcome the first challenge, participating agents must be rewarded more than their cost of reporting for giving feedback, and to overcome second challenge an honest feedback must receive more reward than any distorted feedback.

The peer-prediction method [4] is one of the known efforts in this direction but is limited by its applicability to only situations involving agents at a single level. In this paper, we non-trivially generalize the peer-prediction mechanism to the setting of a tree network of participating nodes that gets formed when the query originates at a (root) node and propagates to different nodes in the social network or crowd in a hierarchical fashion through forwarding. The feedback to this query received from the nodes in the tree must be honest and accurate, and the feedback also must be aggregated in a hierarchical fashion to generate a high quality answer at the root level. In order to motivate the nodes to put in sufficient ef-

Appears in: *Proceedings of the 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2013), Ito, Jonker, Gini, and Shehory (eds.), May, 6–10, 2013, Saint Paul, Minnesota, USA.*

fort and report truthfully, we propose a tree-based peer-prediction framework using proper scoring rules for continuous probability distributions. In this framework, we prove that honest reporting is a Nash Equilibrium (NE) when prior probabilities are common knowledge of the nodes in the tree and the observations made by the sibling nodes are stochastically relevant. We investigate the application of this mechanism using three popular proper scoring rules namely logarithmic, quadratic, and spherical rules. We use techniques from complex analysis to compute payments derived from these proper scoring rules. We validate our findings through simulations and obtain several insights. In particular, we study the relationship between the budget of the mechanism designer and the quality of answer generated at the root node.

1.1 Related Work

There have been several recent studies on web-based crowdsourcing, where the objective is to elicit opinions from a population and incentivize them to contribute to the best of their abilities. Miller, Resnick and Zeckhauser [4] propose *peer-prediction rule* where a central body (information aggregator) collects feedback from a number of strategic agents on some particular product/object. Then each agent is scored based on her report and the report submitted by her *reference rater*. The authors prove the existence of an incentive-compatible payment scheme where honest reporting by all the agents forms a *Bayesian Nash Equilibrium*. Our work non-trivially generalizes this mechanism to a general hierarchical setting. In our setting an agent who is contacted by the central body can also make contacts to other agents who are known to him to get the answer. Our model resembles the *query incentive network* model (Kleinberg and Raghavan [5]) where request for information propagates along the paths in the network. This connects the agents having the required information to those seeking this information. The underlying idea is to use the contact-links present in an underlying social network to find the answer of a hypothetical question. Figure 1 shows the network structure in the case of the peer-prediction method and the network structure that we deal with in this paper.

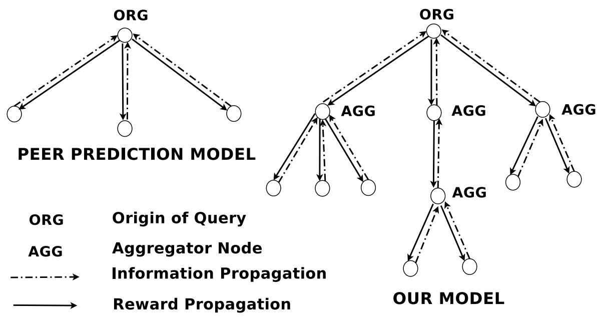


Figure 1: Generalization of the peer-prediction model

There are two major differences between our setting and the setting used in *peer prediction* or other methods designed for reputation systems. The first difference is that, in our model, the network will have multiple levels. In other words, the central information aggregator can be connected to a number of nodes each of which can be connected to another set of nodes and so on. The added advantage with this setting is that there may be users who are not directly reachable from the central body, but can contribute valuable feedback to the system. Also, some node directly connected to the central body may not have enough information about the query and may be willing to seek feedback from its own contacts. So, the network in *peer prediction* can be thought as a special case of our network with just one level.

Secondly, in the case of *peer prediction*, the role of the reputation mechanism is *signaling* and each agent reports a single signal about the product indicating his perception about the product. But there are situations where it is easier for a user to report a distribution on the signals. For example suppose the posed question is *will you vote in the next presidential election?*, and available signals are *definitely, probably, probably not, definitely not*. A user may prefer to report *definitely* with 50%,*probably* with 25%, *probably not* with 25% and *definitely not* with 0%, instead of choosing a single signal for answering. Clearly, reporting a single signal is a degenerate case of reporting a distribution. So, in our model we allow users to report a probability distribution over a set of available signals, making it much more powerful than the peer-prediction setting.

Another issue with the peer-prediction scheme is that the scaling of rewards can lead to arbitrarily high payments causing loss to the reputation mechanism. Jurca and Faltings [3] addressed this issue and proposed an automated mechanism design technique to compute the optimal payments that minimize the total budget required. It offsets both the cost of reporting and the external incentive an agent can achieve by reporting dishonestly. They also introduce the use of several *reference raters* instead of one. Another aspect of peer-prediction is that it makes the assumption that every rater has common knowledge of the prior probability distribution over the types of the product. Prelec [6] presents Bayesian Truth Serum (BTS) mechanism which relaxes this assumption. The work still assumes that there is a common prior but the mechanism designer need not know this. Here in addition to an *information report*, every agent also has to give a *prediction report*, which reflects his belief about the distribution of the information reports submitted by the entire population. In this mechanism, an agent gets paid more when its information report is more common than collectively predicted or in other words when his information report is "surprisingly common". On the other hand, when the prediction report matches with the true distribution of the population, his reward is maximized. One requirement for this mechanism to be incentive compatible is that the number of participating agents should be very large. In addition, the reward received by an agent may be negative and BTS is not numerically robust for all inputs.

Witkowski and Parkes [7] present a mechanism built on Bayesian Truth Serum (BTS) which relaxes the requirement of very large number of participating agents. Here, reporting scheme is same as BTS, but with the restriction that it applies to elicit only binary information. Using quadratic scoring rule, they propose a strictly incentive compatible mechanism for $n \geq 3$.

Jurca and Faltings [8] address reporting incentives for online opinion polls. Here, there are only two possible answers available for the raters, and current distribution of the submitted reports is published to the remaining agents. This mechanism requires the agents to report only the information report, but it is not incentive compatible. Jurca and Faltings [9] allow a small deviation in the prior probabilities known to the agents rather than it being common knowledge. They also prove that no reward mechanism can be strictly incentive compatible when the mechanism designer does not know the prior information of the participants. In [10], different scenarios, where some or all the agents can collude are analyzed using automated mechanism design techniques.

Kleinberg and Raghavan([5]) propose the branching process model for query incentive networks. In this setting, a query is originated at the root of an infinite d-ary tree where each node possesses the answer to the query with probability $1/n$, where n is the rarity of the answer. For any general node v, if r is the reward offered by its parent and $f_v(r)$ is the fraction of r which v offers to its children, then the payoff for node v is $(r - f_v(r) - 1)$. They analyzed the Nash Equilibrium in this model and also investigated the relation between the branching factor of the network and the reward required to find the answer with a constant probability.

Dikshit and Narahari [11] consider the issue of the quality of the answer in query incentive networks. They define a *quality conscious model* where the incentive is modulated based on the quality of the answer. They show the existence of a unique Nash Equilibrium and study the impact of quality of answer on the growth rate of the initial reward with respect to the branching factor of the tree network.

A fundamental difference between the model used in *query incentive networks* [11, 5] and our model is that in their model the query has a specific answer and whenever a node is found to be knowing the answer, searching for answer terminates and that answer only is propagated to the root, so there is no need for information aggregation. But in our model the objective is aggregate opinions from a population while ensuring the feedback is honest, so information aggregation plays an important role here.

Many of the studies above use proper scoring rules to set up the mechanisms. Such scoring rules have been used in a wide variety of ways, for example, see Boutilier [12], Bickel [13]. In our study here, we use proper scoring rules for continuous distributions, in particular the logarithmic, quadratic and spherical scoring rules.

1.2 Contribution and Outline

The various contributions of this paper are described below.

- In Section 2, we non-trivially generalize the peer-prediction method [4] to the case of a tree network. Further, in the proposed method, we use proper scoring rules for continuous distributions to enable the participating nodes to report probability distributions rather than discrete signal values. The key reason to consider tree networks is the fact that when a query originates at a (root) node and propagates to different nodes in the crowd or social network through forwarding, a hierarchy of participating nodes gets naturally formed. In this framework, we prove that honest reporting is a Nash Equilibrium when prior probabilities are common knowledge among the nodes in the tree and the observations made by sibling nodes are stochastically relevant.
- We pick three popular strictly proper scoring rules - logarithmic, quadratic, and spherical rules - to investigate the application of the proposed tree based peer prediction mechanism. In Section 3, we develop computational procedures to compute payments derived from these rules using techniques from complex analysis.
- In Section 4, we carry out detailed simulations using an underlying social network of 100 nodes and validate our findings. We obtain several insights, in particular, we study the relationship between the budget of the mechanism designer and the quality of answer generated at the root node.
literature:

2. MODEL AND PROBLEM FORMULATION

2.1 Peer Prediction Model: A Quick Review

We first outline the peer-prediction method proposed by Miller, Resnick and Zeckhauser [4]. Consider a product which a number of raters have experienced and they are able to provide their feedback about this product. Each product has a type associated with it and we assume the number of product types to be finite and be indexed by $t \in \{1, \ldots, T\}$. There is a prior probability assigned to type t which is denoted by $p(t)$ and it is assumed to be a common knowledge among the raters. We assume $p(t) > 0$ for all t. Let $S = \{s_1, \ldots, s_M\}$ denote the set of possible signals that a rater can observe. S^i denotes the random variable representing the signal for rater i, and s^i_m denotes the event $\{S^i = s_m\}$. Let $f(s_m|t) = \Pr(S^i = s_m|t) > 0$ for all s_m and t and $\sum_{m=1}^{M} f(s_m|t) = 1$ for all t. This $f(s_m|t)$ is assumed to be common knowledge. Assume that raters are risk neutral and seek to maximize their utility. Each rater has a perception about the product and it is assumed to be private information. Let $\tau_i(a)$ be the payment made to agent i where a is the vector representing all the reports made by all the raters of i. The report by any node will be a probability distribution on the set S. We denote the report by i-th node as $\bar{a}_i \in \Delta(S)$. Miller et al. [4] show that for any two distinct buyers i and j, S^i is *stochastically relevant* for S^j in their mechanism. That is, for any two distinct realizations s_i and $\hat{s}_i$ of S_i, there exists some realization s_j of S^j such that $P(S^j = s_j|S^i = s_i) \neq P(S^j = s_j|S^i = \hat{s}_i)$. If S^i is stochastically relevant for S^j, then rater i's signal provides some information about the distribution of rater j's signal. So, if it is known that rater j is reporting truthfully, then eliciting rater i's information is reduced to eliciting his belief about the distribution of j's signal. Finally they prove that if the payments are made according to the following rule then truthful reporting becomes a strict Nash equilibrium of the simultaneous reporting game:

$$\tau^*_i(\bar{a}_i, \bar{a}_{r(i)}) = R(\bar{a}_{r(i)}|\bar{a}_i)$$

where $r(i)$ denotes the reference rater of rater i and $r(i) \neq i$, R is a strictly proper scoring rule.

2.2 Need for Continuous Scoring Rules

In this setting, as the nodes are able to report probability vectors as their report, there is a need to use the continuous analogs of the strictly proper scoring rules for calculation of the payment for any node (i.e., for computing $\tau^*_i(\bar{a}_i, \bar{a}_{r(i)}), \forall i$). In this paper, we work with the continuous analogs of three popular and highly studied strictly proper scoring rules namely logarithmic, quadratic and spherical rules. The continuous logarithmic, quadratic and spherical scoring rules (Matheson and Winkler [14]) are given below:

$$S(q(x)) = ln(q(x)) \qquad \text{[Logarithmic scoring rule]}$$

$$S(q(x)) = 2q(x) - \int_{u \in X} q^2(u)du \qquad \text{[Quadratic scoring rule]}$$

$$S(q(x)) = \frac{q(x)}{\sqrt{\int_{u \in X} q^2(u)du}} \qquad \text{[Spherical scoring rule]}$$

where x is the revealed value of the variable of interest (in the context of this paper, this represents a probability vector over the signals) and $q(\cdot)$ is the corresponding density function that is assigned by the subject.

2.3 Problem Formulation

We have a mechanism designer who is interested to know the rating of a particular object (product, service, person etc.) using a network of social contacts who have experienced the product. As explained earlier, let the number of quality ratings be finite and be indexed by $t \in \{1, \ldots, T\}$. Let $p(t)$ be the prior probability for the object of being type t. We assume that the prior probabilities are common knowledge in the network. In this setting, it is a three-step process for eliciting the quality of the concerned object. We describe the three steps below.

2.4 Contract Signing

In the first step, the network is created by making contracts in the following way. It starts from the root node representing the entity interested to get feedback from the network. The root node agrees to offer a reward, A, to *each* of its children. Now they become leaf nodes. Then each of these children has two options: either it can report by itself or it can seek help from its own children by contacting them and propagating the query. In order to get help from any child, it has to promise a fraction of its own reward to that child. If that child agrees to acquire and report feedback with offered reward then it makes a contract with its parent. Initially, all the children of a parent are promised equal payment. The same process continues with current leaf nodes and they may in turn become parents of other nodes. The process stops when one of the two things occur: there is no node which wants to contact any of its children, or when none of the contacted children agrees to sign the contract with offered reward.

2.5 Information Propagation

After the first step, our network consists of only those nodes who have signed the contract with their parent and the root node itself. Propagation of reported signal starts from the leaf nodes. Each leaf node reports its *observation* to its parent, then the parent "appropriately" aggregates all the reports from its children to a single report which becomes its *observation* (belief about the object). The exact way of aggregation is discussed later. Then this parent also reports back to its parent. Finally the root node receives reports from its first level children and aggregates them to the final answer of the query. Our objective is to make this final result close to a distribution which reflects the true type of the object concerned.

2.6 Reward Propagation

In step 1, only contract signing was involved and there was no monetary transfer. Payment is done in a top-down manner starting from the root node. It is not possible for the root node to reach out to every node in the tree and make payment, so payment is done in a decentralized manner where a node gets paid by only its parent. When we make actual payments, we would like to do it in a way that honest reporting results in more reward than dishonest reporting. So, we have to ensure that for every parent node, all its children are reporting truthfully. The payments are computed by the parent using the scores received by the children based on their reports. This is discussed later in detail.

We make the assumption about the network that one node can be child of at most one parent, which prevents formation of any cycle in the network. Under the above setting, the only source of raw information is the leaf nodes, because parent nodes are only responsible for aggregating information received from their children. But this restriction can be relaxed by making the following small change to the setting. We assume every parent node also wants to contribute information, and it does so by having a *dummy* child which will report that information as perceived by the parent. The only difference between this dummy child and any other child is that a dummy child does not take part in contract signing and it does not have to be rewarded by the parent, though during aggregation of reports its report is taken into account.

3. TREE-BASED PEER PREDICTION MECHANISM

We now describe the proposed tree-based peer-prediction mechanism.

3.1 Reporting Scheme

Let $S = \{s_1, s_2, \ldots, s_M\}$ be the set of signals that a node can observe about the object's type. The report by any node will be a probability distribution on the set S. We denote the report of i-th node by $\bar{a}_i \in \Delta(S)$. Let $f(\bar{a}|t) > 0$ denote the probability density of a node reporting $\bar{a}$ given that true type of the object is t. So, we need following property to be satisfied: $\forall t \in \{1, 2, \ldots, T\}, \int_{\bar{a} \in \Delta(S)} f(\bar{a}|t) = 1$,

3.2 Payment Scheme

Let us consider any parent node having I children, indexed 1 to I. We employ the peer-prediction method at each parent to ensure truthful reporting by children. So, the score given to i-th child is

$$\tau_i(\bar{a}_i, \bar{a}_{r(i)}) = R(\bar{a}_{r(i)}|\bar{a}_i)$$

where $R(p|r)$ is a function derived from any strictly proper scoring rule (such as given in Section 2.2) and $r(i)$ is the reference function given by $r : \{1, 2, \ldots, I\} \to \{1, 2, \ldots, I\}$.

Theorem 1. *The above mentioned payment ensures that all the children reporting their true feedback is a strict Nash Equilibrium when, for every parent,*

- *For any child x of that parent $r(x) \neq x$.*
- *If i is a child of this parent, $f(\bar{a}_i|t)$ is a common knowledge among the siblings of i.*
- *If i and j are any two children of this parent, and their observations are $\bar{a}_i$ and $\bar{a}_j$, then $\bar{a}_i$ is* stochastically relevant *for $\bar{a}_j$.*

PROOF. Assume that, except the i-th child, all the other children are truthful, and for any j-th child, its observation (either obtained directly from the experience with the object or by aggregating observations of children) is given by $\bar{a}_j$. The i-th child reports $\bar{b}$. As $\bar{a}_i$ is stochastically relevant for $r(i)$'s report, the expected payoff for the i-th child will be

$$E_{\bar{a}_{r(i)}}[\tau(\bar{b}, \bar{a}_{r(i)})] = \int_{\bar{a}_{r(i)} \in \Delta(S)} R(\bar{a}_{r(i)}|\bar{b}) g(\bar{a}_{r(i)}|\bar{a}_i) \quad (1)$$

$g(\bar{a}_{r(i)}|\bar{a}_i)$ denotes the probability that $r(i)$ reports $\bar{a}_{r(i)}$ given that i has reported $\bar{a}_i$, and $g(\bar{a}_{r(i)}|\bar{a}_i)$ can be computed with the knowledge of $f(\bar{a}_{r(i)}|t), f(\bar{a}_i|t)$ and $p(t)$ for all $t \in \{1, 2, \ldots, T\}$. RHS of expression in Equation 1 can be written in the following form:

$$\int_{x \in X} S(q(x))h(x)dx \quad (2)$$

where $x = \bar{a}_{r(i)}, X = \Delta(S), q(\cdot) = g(\cdot|\bar{b}), h(\cdot) = g(\cdot|\bar{a}_i)$ and R is such that $S(q(x)) = R(\bar{a}_{r(i)}|\bar{b})$, and S is a strictly proper scoring rule in continuous probability distribution of x (such as given in Section 2.2). From the property of proper scoring rule, we get $\int_x S(q(x))h(x)dx$ is uniquely maximized when $q(x) = h(x), \forall x \in X$. Hence, the expected payoff will be uniquely maximized when $\bar{b} = \bar{a}_i$, i.e., when i-th child reports truthfully. $\square$

4. COMPUTATIONAL ISSUES

In this section, we discuss some computational issues and develop computational procedures for calculating payments based on the proposed tree-based peer-prediction mechanism. To recall, Table 1 presents the relevant notation used in the paper.

4.1 Computation of $g(\bar{a}_{r(i)}|\bar{a}_i)$

For this computation, we can assume the index does not matter i.e., $g(\bar{a}_i|\bar{a}_j)$ does not depend on i or j, it only depends on the values of $\bar{a}_i$ and $\bar{a}_j$.

We calculate $g(\bar{a}|\bar{b})$ as $g(\bar{a}|\bar{b}) = \sum_{t=1}^{T} f(\bar{a}|t)Pr(t|\bar{b})$ where $Pr(t|\bar{b}) = \frac{f(\bar{b}|t)p(t)}{f(\bar{b})}$ and $f(\bar{b}) = \sum_{t=1}^{T} f(\bar{b}|t)p(t)$

4.2 Computation of $f(\bar{a}|t)$

This $f(\bar{a}|t)$ is a common knowledge and it needs to satisfy $\int_{\bar{a} \in \Delta(S)} f(\bar{a}|t) = 1$ for all t because this quantity represents a

Symbol	Meaning
$\Delta(S)$	Set of all possible probability distributions over signal set S
$\bar{a}_i$	Report by agent i
$\bar{a}, \bar{b}$	Elements from the set $\Delta(S)$
$p(t)$	Prior probability for the object to be of type t
$f(\bar{a}\|t)$	Probability density of reporting $\bar{a}$ given that true type of object is t
$r(i)$	Reference Rater of i-th child

Table 1: Important Notations

probability density function. We assumed that it follows Dirichlet distribution. But the problem of directly using Dirichlet distribution is it assigns probability 0 to any event having 0 at some component of $\bar{a}$, but some rater may prefer to assign zero probability to some component(s) of their reported distribution. To avoid this problem we first transform the probability vector $\bar{a}$ to $\bar{c}$ where $\bar{c}^i = \frac{e^{\bar{a}^i}}{\sum_{j=1}^{M} e^{\bar{a}^J}}$ where $\bar{c}^i$ represents the ith component of $\bar{c}$. This mapping can easily be verified to be one-to one, and it ensures $\bar{c}$ is a probability distribution having all positive entries. The only place where we need to use $f(\bar{a}|t)$ is computation of $g(\bar{a}_{r(i)}|\bar{a}_i)$, so here we first transform both $\bar{a}_{r(i)}$ and $\bar{a}_i$, then use them for calculation.

4.3 Computation of Scoring Rules

As we can see from the definitions in Section 2.2, computing the strictly proper scoring rules involves two parts:

- Computation of $q(x)$
- Computation of $\int_{x\in X} q^2(x)dx$

We will examine how to compute these terms in the next section.

4.3.1 Computation of $q(x)$

From the proof of Theorem 1, we know that $q(x) = g(x|\bar{b})$.

$$q(x) = g(x|\bar{b}) = \sum_{i=1}^{T}\left(f(x|t_i)Pr(t_i|\bar{b})\right)$$
$$= \sum_{i=1}^{T}\Big[f(x|t_i)\underbrace{\Big(\frac{f(\bar{b}|t_i)p(t_i)}{\sum_{i=1}^{T} f(\bar{b}|t_i)p(t_i)}\Big)}_{\text{Constant (denote as } c(t_i))}\Big] = \sum_{i=1}^{T}\left(f(x|t_i)c(t_i)\right)$$

4.3.2 Computation of $\int\limits_{x\in X} q^2(x)dx$

$$\int\limits_{x\in X} q^2(x)dx = \int_{x\in X}\left[\sum_{i=1}^{T}\left(f(x|t_i)c(t_i)\right)\right]^2 dx$$
$$= \int_{x\in X}\sum_{i=1}^{T}\left(f(x|t_i)c(t_i)\right)^2 dx$$
$$+\, 2\int_{x\in X}\sum_{i<j}\left(f(x|t_i)f(x|t_j)c(t_i)c(t_j)\right)dx$$
$$= \sum_{i=1}^{T}\int_{x\in X}\left(f^2(x|t_i)c^2(t_i)\right)dx$$
$$+\, 2\sum_{i<j}\int_{x\in X}\left(f(x|t_i)f(x|t_j)c(t_i)c(t_j)\right)dx \tag{3}$$

From our assumption, $\forall i, f(x|t_i)$ follows a Dirichlet distribution. The expression for probability density function of a Dirichlet distribution is given by $f(x|t_i) = \frac{1}{B(\alpha^i)}\left(\prod_{i=1}^{K} x_i^{\alpha_j^i - 1}\right)$ where α^i is a vector of length K which represents the parameters of Dirichlet distribution $f(x|t_i)$ corresponding to type t_i. α_j^i represents the jth component of the vector α_i. By the definition of Dirichlet distribution, we need $\alpha_j^i > 0, \forall i, \forall j$. Note that K represents the length of the probability vector reported by a node in the network which is also equal to the number of possible signals that can be observed by the nodes in the network. $B(\alpha^i)$ is the normalizing constant which is defined to be the multinomial Beta function, which can be expressed as $B(\alpha^i) = \left(\frac{\prod_{j=1}^{K}\Gamma(\alpha_j^i)}{\Gamma(\sum_{j=1}^{K}\alpha_j^i)}\right)$ where $\Gamma(\cdot)$ is the Gamma function. n is an integer and z is a complex number with a positive real part. The (complete) gamma function $\Gamma(n)$ is defined to be an extension of the factorial to complex and real number arguments. Substituting in Equation 3,

$$\int_{x\in X} q^2(x)dx = \sum_{i=1}^{T}\left(\left(\frac{c^2(t_i)}{B^2(\alpha^i)}\right)\underbrace{\int_{x\in X}\left(\prod_{j=1}^{K} x_j^{\left(2\times(\alpha_j^i - 1)\right)}\right)dx}_{\text{(i)}}\right)$$
$$+\, 2\sum_{i<j}\left(\left(\frac{c(t_i)c(t_j)}{B(\alpha^i)B(\alpha^j)}\right)\underbrace{\int_{x\in X}\left(\prod_{k=1}^{K} x_k^{\left(\alpha_k^i + \alpha_k^j - 2\right)}\right)dx}_{\text{(ii)}}\right)$$

So, in order to compute the LHS, we need to evaluate (i) and (ii) which are basically integrals over a $(K-1)$ dimensional simplex. In order to do this, we will use some techniques from complex analysis to get a closed form expression for (i) and (ii). To keep the analysis simple, we will consider the case where $K = 3$ and evaluate the integral

$$\int\limits_{x_1}\int\limits_{x_2}\int\limits_{x_3} x_1^{\alpha_1 - 1}x_2^{\alpha_2 - 1}x_3^{\alpha_3}dx_1dx_2dx_3 \tag{4}$$

where $\alpha_1, \alpha_2, \alpha_3 > 0$ and $(x_1, x_2, x_3) \in X$ where X is the 2 dimensional simplex i.e., $x_1, x_2, x_3 \geq 0$ and $x_1 + x_2 + x_3 = 1$. It can been easily seen that (i) and (ii) can be basically reduced to the above form (i.e., when $K = 3$) and hence, evaluating the above integral will help us to evaluate $\int\limits_{x\in X} p^2(x)dx$ which, in turn, solves the problem of evaluating the quadratic and spherical scoring rules given in Section 2.2.

4.3.3 Computation of $\int\limits_{x_1}\int\limits_{x_2}\int\limits_{x_3} x_1^{\alpha_1 - 1}x_2^{\alpha_2 - 1}x_3^{\alpha_3}dx_1dx_2dx_3$

As noticed above, this computation requires an evaluation of an integral over a simplex. We use some techniques from complex analysis ([15]) to evaluate this integral. We define a Delta function as follows $\delta(1-x_1-x_2-x_3) = 1$ if $(1 - x_1 - x_2 - x_3 = 0)$, $\delta(1-x_1 - x_2 - x_3) = 0$ otherwise

$$(4) = \int\limits_{x_1=0}^{\infty}\int\limits_{x_2=0}^{\infty}\int\limits_{x_3=0}^{\infty} x_1^{\alpha_1 - 1}x_2^{\alpha_2 - 1}x_3^{\alpha_3}\delta(1-x_1-x_2-x_3)dx_1dx_2dx_3$$

We know that the delta function can be expressed as a Fourier transposition in the following way

$$\delta(x) = \frac{1}{2\pi}\int\limits_{-\infty}^{\infty} e^{iax}da \qquad \text{where } a \text{ is the Fourier variable}$$

Applying this and simplifying, we get

$$(4) = \frac{1}{2\pi}\int\limits_{-\infty}^{\infty} e^{ia}\int\limits_{x_1=0}^{\infty} x_1^{\alpha_1 - 1}e^{iax_1}dx_1$$
$$\int\limits_{x_2=0}^{\infty} x_2^{\alpha_2 - 1}e^{iax_2}dx_2\int\limits_{x_3=0}^{\infty} x_3^{\alpha_3}e^{iax_3}dx_3da$$

Making the substitution $ia = -\tau$ and simplifying we get

$$(4) = \frac{1}{2\pi i}\int\limits_{-i\infty}^{i\infty} e^{\tau t}\int\limits_{x_1=0}^{\infty} x_1^{\alpha_1 - 1}e^{iax_1}dx_1$$
$$\int\limits_{x_2=0}^{\infty} x_2^{\alpha_2 - 1}e^{iax_2}dx_2\int\limits_{x_3=0}^{\infty} x_3^{\alpha_3}e^{iax_3}dx_3d\tau\Bigg|_{t=1}$$

We know that Laplace Transform (LT) for a function $f(t)$ is

$$\mathbb{L}(f(t)) = \int_0^\infty f(t)e^{-st}dt = F(s) \quad \text{where } s \text{ is the L.T. variable}$$

By substituting $s = -ia$ where a is the Fourier variable, we get

$$(4) = \frac{1}{2\pi i}\int_{-i\infty}^{i\infty} e^{\tau t}\mathbb{L}\left(x_1^{(\alpha_1-1)}\right)\mathbb{L}\left(x_2^{(\alpha_2-1)}\right)\mathbb{L}\left(x_3^{(\alpha_3)}\right)d\tau\Bigg|_{t=1}$$

We know that $\mathbb{L}\left(p^\beta\right) = \frac{\Gamma(\beta+1)}{s^{\beta+1}}$ where s is the L.T. variable. Substituting in the above expression and using the property of Gamma functions namely $\Gamma(\beta+1) = (\beta \times \Gamma(\beta))$, we get

$$(4) \quad = \quad \alpha_3\prod_{i=1}^3\Gamma(\alpha_i)\underbrace{\frac{1}{2\pi i}\int_{-i\infty}^{i\infty} e^{\tau t}\frac{1}{\tau^{\alpha_1+\alpha_2+\alpha_3+1}}d\tau\Bigg|_{t=1}}_{(a)}$$

(a) can be identified as the inverse Laplace Transform of $F(\tau) = \frac{1}{\tau^{(\alpha_1+\alpha_2+\alpha_3+1)}}$. We know that

$$\mathbb{L}^{-1}(F(\tau)) = \frac{1}{2\pi i}\int_{-i\infty}^{i\infty} e^{\tau t}F(\tau)\,d\tau$$

By using properties of Laplace Transforms, it can be shown that

$$\mathbb{L}^{-1}\left(\frac{1}{\tau^{(\alpha_1+\alpha_2+\alpha_3+1)}}\right) = \frac{1}{\Gamma\left(1+\sum_{i=1}^3\alpha_i\right)}$$

Substituting the above, we get the final expression for (4) as below

$$(4) = \left(\left(\frac{\alpha_3}{\sum_{i=1}^3\alpha_i}\right)\times\left(\frac{\prod_{i=1}^3\Gamma(\alpha_i)}{\Gamma\left(\sum_{i=1}^3\alpha_i\right)}\right)\right)$$

Note that the above result can be extended to any finite value of K.

4.4 Rank Order Aggregation of Answers

Our method ensures that every parent will truthfully report its belief which is formed after aggregating reports from children. So, a parent is free to use any aggregation method as long as it correctly captures its true belief. The aggregation method used in our implementation is the following: While evaluating every child, the parent assigns a score based on a continuous scoring rule to every child. If this score is high it in some sense means that this current child's report is helping to make consensus among reports (since the reference of this child is selected randomly from other children). So, we assign higher weightage to a report with higher score than a report with a low score. One approach is to simply take the weighted average based on the score values of all the reports. But there is an issue that needs to be carefully handled. Some of the standard scoring rules like the logarithmic and quadratic scoring rules can provide scores which are negative. To circumvent this problem, we generate a rank order among the children which is induced by the scores generated. If there are m children whose reports need to be aggregated, then the highest ranked child gets the highest rank count of m, the next highest child gets a rank count of $m-1$ and so on. Now, we take a weighted average of the reports generated by the children based on the squares of the rank counts of the children. We do the squaring of the rank count in order to induce a non-linear importance to the children. In general, any such monotone function can be used for computing the rank counts of the children. Thus, the resulting weighted average vector is guaranteed to be a probability distribution as this vector is essentially obtained by take linear combinations of probability vectors from a $M-1$ dimensional simplex where M is the number of possible signals that a node can observe. Note that the assumption of every node (including parent nodes) contributing information and every parent node having a dummy child ensures that every parent node has at least two children. Hence it is always possible to find a *reference rater* for every rater.

5. EXPERIMENTAL RESULTS

5.1 Setup

We assume a social network of size n which is modeled as a random network with edge density γ. For all our simulations, we fix $n = 100, \gamma = 0.2$. This means that there is a social network of 100 nodes which has roughly 20% of the possible edges in it. We designate a root node which acts as the mechanism designer interested in an accurate answer to a query. The root has a budget amount which it is willing to pay to its children. This is the total amount of money distributed throughout the entire network. We call this amount *initialReward*. As explained earlier, a tree sub-network is generated from the social network starting with the root during the contract signing phase. Basically, this tree consists of all the nodes who are interested in providing an answer to the query of the root in return for some payment promised by their corresponding parents.

We also define a quantity called *confidence* for every node except the root node. This *confidence* denotes how certain a node is about its answer for the query. We assume *confidence* is normally distributed with mean μ_c and variance σ_c^2. There is a threshold for confidence denoted by t_c, such that if a node has *confidence* greater than or equal to t_c then it prefers to answer by itself instead of forwarding the query to its children, in which case it does not have to share any reward with children. In order to simulate contract signing, we assume every node has an expectation from its parent which is modeled as a normal random variable with mean equal to a parameter *basicThrFee* and variance which is inversely related to the confidence of the node. We set *basicThrFee* $= 50$ in our simulations. This ensures that the expectation amount of a node monotonically increases with its *confidence*. When the amount offered to a child node is greater than or equal to its expected amount it signs the contract. The excess amount that a node has after subtracting expected reward from reward promised by its parent node, is, in turn, offered to its children. A node becomes a leaf node when either it decides not to forward the query further or it does not have enough excess reward to hire any children.

5.2 Parameters and Metrics

In all our simulations, we used $T = 2$ i.e., there are two types for the object say *High* and *Low*, and $M = 4$ i.e $S = \{s_1, s_2, s_3, s_4\}$. As explained earlier, S represents the possible signals that a node can observe. So, every report $\bar{a}$ is an element from $\Delta(S)$ (simplex of dimension 3). We set $p(\mathit{High}) = 0.7$ and $p(\mathit{Low}) = 0.3$ which denotes the prior probabilities on the product type which is assumed to be common knowledge. We assume there is a mapping from set S to the set $\{\mathit{High, Low}\}$, such that corresponding to every element in S there is a unique element in $\{\mathit{High, Low}\}$. We assume, under this mapping, s_1 and s_2 correspond to *High*, and s_3 and s_4 correspond to *Low*. So, if the true type of the object is *High* then the correct answer at the root would be a vector whose first two components sum up to 1 and other two components are zero. Similarly, for true type *Low*, the correct outcome would be a vector whose last two components sum up to 1 and other two components are zero.

With respect to report generation by the individual nodes, we note that nodes in the tree network perceive the object with some noise and this noise is assumed to follow multivariate normal distribution with mean $[0, 0, 0, 0]$ and covariance matrix $K * n_c * I_4$, where I_4 is the 4×4 identity matrix, K is a constant denoting the noise level and n_c depends on the confidence of the node (the more confident a node is the smaller will be the value of its n_c).

We define a metric, *prediction Accuracy* $\in [0, 1]$, as the sum of the components of the final report received by the root node corresponding to the true type. For example, if the true type is *High* and the final answer received at the root is the distribution $[0.3, 0.4, 0.2, 0.1]$, then accuracy of this answer is 0.7.

In all our simulations, initialReward is increased from 100 to 1000 in steps of 100, next it is increased from 1000 to 10000 in steps of 1000 and finally, 10000 to 100000 in steps of 10000. Due to space constraints, we will provide only a selected portion of the results. The rest of the results follow in fact exhibit similar trends.

5.3 Contract Signing

We vary *initialReward* to simulate the process of generation of a tree keeping other parameters fixed. We set $\mu_c = 0.5, \sigma_c = 0.5, t_c = 0.8$. We generate 1000 tree samples for every value of *initialReward* and measure the average of these three quantities: total number of nodes, number of leaf nodes and height of the tree. We provide 95% confidence interval (CI) guarantees on the plotted values. The details are shown in Figure 2. As per intuition, we can observe that all these three parameters are monotonically increasing with *initialReward*. This is due to the fact that if the root node has more budget then it can (i) hire more children and/or (ii) give higher amount of money to each child so that the children in turn can form larger subtrees. This relationship between the size of the network and budget (also termed as *initialReward* in the rest of the paper) can help the mechanism designer (i.e., the root node) to get an estimate of the total number of participants who might be willing to undertake the task. We now examine the important simulation parameters and metrics which are relevant to the information generation and aggregation process on the generated tree network of participating nodes.

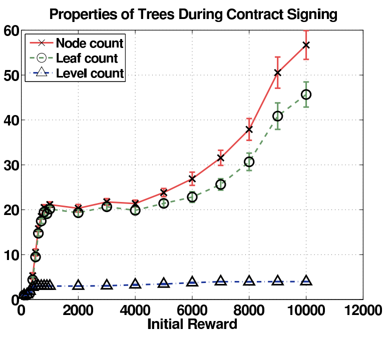


Figure 2: Growth of the tree with *initialReward*

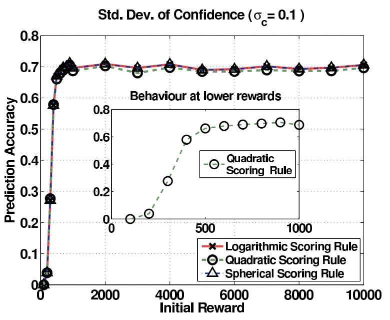
5.4 Information and Reward Propagation

First, we increase *initialReward* keeping other parameters fixed. We set $\mu_c = 0.5, K = 1$. We run our simulations for various values of *initialReward* as explained earlier. For every value of *initialReward*, we simulate the three steps mentioned in Section 2.3. We run the tree generation (or contract signing), information propagation and reward propagation 1000 times and take the average of the *predictionAccuracy* at each run. As we are dealing with random processes, we need to compute the confidence interval for each *(initialReward, predictionAccuracy)* pair. Figure 3(a) plots the *predictionAccuracy* obtained for different values of initialReward (or budget) for ($\sigma_c = 0.1$). Note that this figure is a double axis plot with a main outer axis and an inner inset axis. The outer axis plots values of *predictionAccuracy* for initialReward from 100 to 10000. However, as the behavior at lower values is not clearly observable from this plot, we use the inner inset axis to zoom in to the range of low initialReward values from 100 to 1000. Figure 3(b) is also plotted in a similar way but changing the variance ($\sigma_c = 0.2$) parameter of the *confidence* random variable. We now make the following some observations based on these figures.

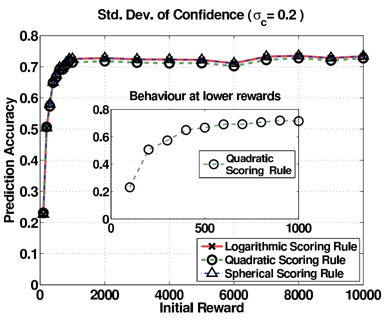
- As observed from the outer axis of the two figures, the *predictionAccuracy* initially increases with *initialReward* and saturates for higher values of *initialReward*. The saturation can be attributed due to the noise perceived by the nodes of the network which is representative of the inherent 'wisdom' of the social network of the mechanism designer.
- The *predictionAccuracy* level is higher for higher value of σ_c for some fixed value of *initialReward*. This is observed clearly for lower *initialReward* values as given in the inset figures of Figure 3(a) and Figure 3(b). This observation can be explained as follows. We keep the threshold for not hiring children t_c fixed. So, as σ_c increases, there will be more number of nodes not having any children by choice. These nodes will report all by themselves and the noise present in their report is less as their confidence high. The *predictionAccuracy* of the answer is higher when number of such nodes is higher. Thus, when the crowd is heterogeneous i.e., there are both experts and laypersons in the crowd, increasing *initialReward* causes higher increase in *predictionAccuracy* than in case of a homogeneous crowd.
- The rate of increase of *predictionAccuracy* with *initialReward* is high when σ_c is high for lower *initialReward* values as shown in the inset figures of Figure 3(a) and Figure 3(b). We have seen that number of nodes in the tree is monotonic in *initialReward* (Figure 2). So, when the root has low *initialReward*, the total number of nodes in the tree is low. Thus, there is an increased risk for the root and there is a good chance that a fair percentage of nodes are reporting with high noise and the quality of the final answer is low. This risk will increase when σ_c is high. But when total number of nodes is larger effect of higher σ_c diminishes resulting in reduced risk and *predictionAccuracy* increases.
- It requires more *initialReward* to saturate *predictionAccuracy* when σ_c is higher. This indicates that more investment is required on the part of root node to get optimal solution when the heterogeneity in the crowd is more.
- All the three scoring rules behave similarly in terms of *predictionAccuracy*. As explained earlier, the aggregation by the parent node depends on the rank orders induced by the scoring rules and the similarity in performance can be attributed to obtaining similar rank orders during the aggregation by the parent nodes after the children send their reports to the parent. The similarity of rank order observation has also be seen in several empirical studies (For eg:,Bickel [13]) where it has been reported that all three scoring rules yielded similar rankings when averaged over several assessment tasks.

We now provide detailed results for two *initialReward* values namely 200 (low budget) and 10000 (high budget) in Table 2. We can make the following observations from the table.

- Lower *initialReward* value results in lower *predictionAccuracy* as there is not enough participation from the nodes.
- Another interesting observation is that for a fixed *initialReward* and fixed σ_c, it can be observed that the three rules behave similarly in terms of *predictionAccuracy*. However, on a closer observation, we see that the *predictionAccuracy* of logarithmic and spherical rules are higher than that of the quadratic rule. Hence, as a design choice, it may be better to choose spherical scoring rule since spherical rule has the property that the scores generated are guaranteed to be non-negative unlike the logarithmic scoring where the scores can be unbounded below.



(a)



(b)

Figure 3: Change of *predictive accuracy* with *initialReward* with varying degree of skills.

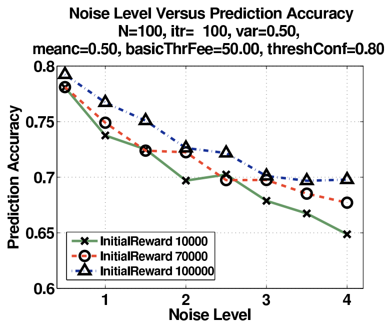


Figure 4: Change of predictive accuracy with noise level

Initial Reward	Continuous Scoring Rule	Confidence (Std. Dev.)	Pred Acc. (Mean)	Pred. Acc. (Std. Dev.)	95% CI
200	Logarithmic	0.10	0.0501	0.1824	[0.0388, 0.0614]
		0.20	0.4748	0.3244	[0.4547, 0.4949]
		0.30	0.6415	0.1968	[0.6293, 0.6537]
	Quadratic	0.10	0.0503	0.1831	[0.0390, 0.0617]
		0.20	0.4731	0.3237	[0.4531, 0.4932]
		0.30	0.6370	0.1996	[0.6246, 0.6494]
	Spherical	0.10	0.0501	0.1824	[0.0388, 0.0614]
		0.20	0.4747	0.3245	[0.4546, 0.4948]
		0.30	0.6414	0.1966	[0.6292, 0.6536]
10000	Logarithmic	0.10	0.7004	0.0555	[0.6970, 0.7039]
		0.20	0.7181	0.0696	[0.7138, 0.7225]
		0.30	0.7498	0.0899	[0.7442, 0.7553]
	Quadratic	0.10	0.6923	0.0594	[0.6886, 0.6960]
		0.20	0.7118	0.0725	[0.7073, 0.7163]
		0.30	0.7407	0.0948	[0.7348, 0.7466]
	Spherical	0.10	0.7003	0.0556	[0.6968, 0.7037]
		0.20	0.7180	0.0697	[0.7137, 0.7223]
		0.30	0.7496	0.0899	[0.7441, 0.7552]

Table 2: Comparison among different scoring rules [Number of Samples: 1000]

- We also observe that the standard deviation of the *predictionAccuracy* is higher for low values of *initialReward*. At low budgets, there may be many runs of the simulations where no nodes will be willing to participate and in such runs, the *predictionAccuracy* is given a value of zero. However, due the the underlying random nature of the expectation amount of the nodes, the root node may be able to hire some nodes even at a low budget in some runs of the simulations. In such runs, we will get a much better value of *predictionAccuracy* which explains the high variance of the samples. At higher budget value of 10000 (which represents the saturated region as shown in Figure 3), the standard deviation is low as the budget is enough to hire enough nodes to achieve a very good *predictionAccuracy* in all runs of the simulation.

5.5 Tolerance to Noise

Next, we vary the noise level by changing the value of K, and study the effect on the *predictionAccuracy* for fixed value of *initialReward*. We vary K from 1 to 4 in steps of 0.5 keeping all other parameters fixed, and we repeat the same experiment with three different values of *initialReward*. The result is shown in Figure 4. We observe that increase in the noise level of the environment results in reducing the *predictionAccuracy* which is on expected lines.

6. DISCUSSION

The Nash equilibrium discussed in Section 3 is not a unique equilibrium. Also, in our analysis, the effect of collusion among nodes was not considered. If the scenario is such that one child does not know about the identities of other children or their communication is limited then the chance of collusion is reduced. But in order to ensure there is no collusion we can either: (a) design a payment mechanism where honest reporting is the unique Nash equilibrium, or (b) design a payment scheme where honest reporting is *Pareto-optimal* Nash equilibrium, so that some of the colluding agents will receive less payoff than what he would have received in honest reporting equilibrium. So, designing collusion resistant mechanism is an interesting task. It would also be interesting to study the proposed mechanism using real-world crowdsourcing networks.

7. ACKNOWLEDGEMENTS

This research work was partially supported by an IBM Research doctoral fellowship grant to the second author.

8. REFERENCES

[1] Radu Jurca and Boi Faltings. Mechanisms for making crowds truthful. *Journal of Artificial Intelligence Research*, 34(1):209–253, March 2009.

[2] Emile Servan-Schreiber, Justin Wolfers, David M. Pennock, and Brian Galebach. Prediction markets: Does money matter? *Electronic Markets*, 14(3):243–251, 2004.

[3] Radu Jurca and Boi Faltings. Minimum payments that reward honest reputation feedback. In *Proceedings of the 7th ACM conference on Electronic commerce*, EC '06, pages 190–199, 2006.

[4] Nolan Miller, Paul Resnick, and Richard Zeckhauser. Eliciting informative feedback: The peer-prediction method. *Management Science*, 51(9):pp. 1359–1373, 2005.

[5] Jon Kleinberg and Prabhakar Raghavan. Query incentive networks. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '05, pages 132–141, 2005.

[6] D. Prelec. A bayesian truth serum for subjective data. *Science*, 306(5695):462–6, 2004.

[7] Jens Witkowski and David C. Parkes. A Robust Bayesian Truth Serum for Small Populations. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI '12)*, 2012.

[8] Radu Jurca and Boi Faltings. Incentives for expressing opinions in online polls. In *Proceedings of the 9th ACM conference on Electronic commerce*, EC '08, pages 119–128, 2008.

[9] Radu Jurca and Boi Faltings. Robust incentive-compatible feedback payments. In M. Fasli and O. Shehory, editors, *Agent-Mediated Electronic Commerce*, volume LNAI 4452, pages 204–218. Springer-Verlag, Berlin Heidelberg, 2007.

[10] Radu Jurca and Boi Faltings. Truthful opinions from the crowds. *SIGecom Exchanges*, 7(2), 2008.

[11] Devansh Dikshit and Narahari Yadati. Truthful and quality conscious query incentive networks. In *Proceedings of the 5th International Workshop on Internet and Network Economics*, WINE '09, pages 386–397, 2009.

[12] Craig Boutilier. Eliciting forecasts from self-interested experts: scoring rules for decision makers. In *Proceedings of 11th Intl. Conference on Autonomous Agents and Multiagent Systems*, AAMAS '12, pages 737–744, 2012.

[13] J. Eric Bickel. Some comparisons among quadratic, spherical, and logarithmic scoring rules. *Decision Analysis*, 4(2):49–65, June 2007.

[14] James E. Matheson and Robert L. Winkler. Scoring rules for continuous probability distributions. *Management Science*, 22(10):pp. 1087–1096, 1976.

[15] Leo Alekseyev. Integrating Dirichlet distributions. http://dnquark.com/blog/2012/05/integrating-dirichlet-distributions/, 2012. [Online; accessed May 29, 2012].