Noname manuscript No.
(will be inserted by the editor)

Automated Agents for Reward Determination for Human Work in Crowdsourcing Applications

Amos Azaria · Yonatan Aumann · Sarit Kraus

Received: date / Accepted: date

Abstract Crowdsourcing applications frequently employ many individual workers, each performing a small amount of work. In such settings, individually determining the reward for each assignment and worker may seem economically beneficial, but is inapplicable if manually performed. We thus consider the problem of designing automated agents for automatic reward determination and negotiation in such settings. We formally describe this problem and show that it is NP-hard. We therefore present two automated agents for the problem, based on two different models of human behavior. The first, the Reservation Price Based Agent (RPBA), is based on the concept of a reservation price, and the second, the No Bargaining Agent (NBA) which tries to avoid any negotiation.The performance of the agents is tested in extensive experiments with real human subjects, where both NBA and RPBA outperform strategies developed by human experts.

1 Introduction

Determining the proper reward to pay a worker is a complicated task, involving the consideration of many elements and often requiring considerable expertise. When the work in question, and associated reward, are substantial, investing the time and effort in determining the exact reward is justifiable. However, the emergence and proliferation of crowdsourcing in general [1], and microwork in particular [2], has created situations in which such individual reward consideration is prohibitive. Crowdsourcing solutions often employ thousands of workers, each performing small amounts of work, for a relatively low pay. The work is frequently further divided into smaller units of *microwork*, which "refers to the fact that tasks

Amos Azaria
Department of Computer Science, Bar-Ilan University, Ramat Gan 52900, Israel
E-mail: azariaa1@cs.biu.ac.il

Yonatan Aumann
Department of Computer Science, Bar-Ilan University, Ramat Gan 52900, Israel
E-mail: aumann@cs.biu.ac.il

Sarit Kraus
Department of Computer Science, Bar-Ilan University, Ramat Gan 52900, Israel
Institute for Advanced Computer Studies, University of Maryland, MD 20742
E-mail: sarit@cs.biu.ac.il

are cut into small pieces and their execution is paid for"[2]. Although each such microwork unit may earn its worker only a few cents, in many cases, tens of thousands of such tasks must be performed resulting in high total costs. In such settings, manually and individually determining the reward for each worker and assignment is prohibitive. Rather, it is necessary to develop automated strategies, and agents, that can interact with the different workers and automatically determine the reward for each worker and assignment. In this paper we take the first steps in building such reward determining agents, and, more importantly, in understanding the necessary ingredients of such agents. We note that to date, rewards for crowdsourcing and microwork are most commonly determined ahead of time, and are fixed. However, there seems to be no inherent economic justification for such uniformity. In this paper, we thus explore the possibility of using automated agents for the reward determination and negotiation, and its effects on the final cost.

The Setting. The core setting we consider is the following. An employer has a large set $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}$ of essentially identical small assignments that must be completed.[1] Each such assignment can be completed by anyone of a set of available *human workers*.The reward to be paid for each assignment is negotiated and determined separately with each worker by an automated agent, which we call the *requester*. The negotiation process works as follows. There is a set $O = \{o_1, \ldots, o_m\}$ of possible rewards for the task. For each assignment and human worker, in order to determine the reward, the requester makes an offer in $o \in O$ to the worker, who can either accept or reject the offer. If the worker accepts the offer, he performs the assignment, gets the reward, and the assignment is deemed completed. If the offer is rejected by the worker, the requester may either make another offer to the same worker or abandon this worker and start interacting with a different one. In addition to the payments made to the workers for their work, a cost of C_c is associated with bringing-in a new worker, and a cost, C_o, is associated with making each offer.

The set of human workers is assumed to be composed of different types $\mathcal{T} = \{t_1, t_2, ..., t_\ell\}$ with a given distribution π over the types. Each type t is associated with a decision policy, which, given the history of previous offers, h, and the current offer o, determines whether the worker of this type accepts or rejects the new offer. Technically, each type is characterized by a *decision function* $d_t(h, o) \rightarrow \{accept, reject\}$[2], which, for any history of offers to the worker, h, and current offer o, determines if the worker accepts the offer or rejects it. Note that according to the negotiation process, the history is consisted solely of offers rejected by the worker.

Given a set of tasks $\mathcal{A}$ and type distribution π, the objective of the requester is to minimize the total cost for completing all assignments in $\mathcal{A}$. We call this the *Task Completion Problem (TCP)*. A solution for a TCP is a *policy* for the requester, which, given a history of offers, acceptances and rejections, determines the next move: either the next offer to the same worker, or a new worker to a new worker.

The following are assumed with regards to the TCP:

– The requester *must* find a worker that will complete each of the assignments (i.e. he will encounters a cost of ∞ if he fails to do so).
– The set of human workers is sufficiently large so that the requester does not need to take into account a scenario in which no more workers are available.

[1] In most cases the assignments are not completely identical, but they are sufficiently similar so that the associated reward is identical (e.g. checking if a given link is stale).

[2] We consider only types where d_t can be represented by a description which is polynomial in the size of the game

- The distribution on the types of future workers does not change as the process goes on, regardless of the worker types that the requester observes.
- The amount of money available to the requester is not a limiting factor. Its goal is to minimize the expected expenditure to complete all tasks.
- The worker strategy is not necessarily determined by some fixed game theoretic solution concept, e.g. Nash equilibrium, Nash Bargaining, etc. We believe that in real-life scenarios, limiting the worker strategies to such solution concepts is too restrictive, as workers are not necessarily familiar with the costs and benefits for the requester or even the negotiation protocol. Furthermore, full rationality cannot be assumed, and even the exact choice of solution concept is not always clear. Therefore, our model is more general and captures any type of strategy the workers may use.
- Any cost that might exist to the worker (such as the cost to complete the task, to consider the task in the first place or to consider an offer) is assumed to be embedded in his decision function.

In addition to the basic TCP setting, we also consider a setting where each assignment is further divided into sub-assignments, which we call *milestones*. All milestones of a given assignment must be performed by the same worker, but are individually rewarded. Specifically, in the *Milestones Task Completion Problem (M-TCP)*, each assignment A_i is composed of a sequence of k *milestones* $A_i = \{m_1^i, m_2^i, ..., m_k^i\}$. In order to successfully complete the assignment, all milestones of the assignment must be completed *by the same worker*, but the worker is rewarded for performing each milestone separately and has the right to leave at any time, even after getting a reward for some but not all of the milestones. In the M-TCP setting, the history, h, is assumed to hold the details of all previous offers, including those for previous milestones (which may have been accepted by the worker). The decision policy, d_t, is extended accordingly. For this version, we consider two possible processes for determining the rewards. In the *stepwise process* the reward for each milestone is determined separately before performing the specific milestone. In the *upfront process* the individual rewards for all milestones are determined in advance (before starting the first milestone). As mentioned, in both cases, the worker may leave at any time, keeping whatever rewards he has accumulated so far.

Thus, in total we consider three settings: the basic TCP (with indivisible assignments), and two versions of the M-TCP - stepwise reward determination and upfront reward determination.

Methods and Results. We show that solving the Task Completion Problem is NP-Hard even for the basic-TCP, and therefore consider restricted versions. We consider two restrictions, each corresponding to a different, plausible, restriction on the workers' possible decision policies. For each such restriction, we develop an automated agent for the corresponding Task Completion Problem and the variants thereof. We thus obtain two agents: RPBA (*Reservation Price Based Agent*) and NBA (*No Bargaining Agent*). We performed extensive experiments with an actual TCP carried out on Amazon's Mechanical Turk service with 1117 people. We measured the performance of our agents in the three different settings. We also compared the performance of these automatically generated agents to the performance of strategies manually determined by human experts. The results are described in detail in the following, but the overall outcome is that NBA outperforms the human experts. Thus, proper modeling of human behavior is essential in such settings and can significantly increase the performance.

2 Related Work

Negotiation has been studied extensively [3,4]. Most research has considered the archetypal form of negotiation, wherein the sides alternate in making offers (see [5,6,7]). This model was extended by [8] who specify commitments that agents make to each other when engaging in persuasive negotiations using rewards. In practice, however, this is often not the actual dynamic. In many real world situations it is only one side that makes the offers, and the other side merely accepts or rejects them. This is most commonly the case, for example, in the job market. Employers make offers to job candidates, and may improve the offers if rejected by the candidates. Job candidates, on the other hand, most commonly simply accept or reject the offers and are much less likely to make counter offers.

Assignment of tasks to automated agents has been studied extensively in the multi-agent systems community. They studied both cooperative games where the goal of the negotiation was to exchange information (e.g., [9]) and environments of self- interested agents that are more similar to ours. Most of the research develops protocols and strategies for agents that reallocate tasks between themselves (e.g.[10]). A few works consider negotiations between a controller and automated agents that are being paid to perform tasks. Sarne and Kraus [11] studied the process of allocating tasks to self-interested automated agents in uncertain changing open environments. The allocator in that model is responsible for the performance of dynamically arriving tasks using a second price reverse auction as the allocation protocol. The paper defined the problem as a game and focuses on the problem of identifying a set of equilibrium strategies. The main difference between their work and ours is that in [11], although the set of tasks arrive dynamically, the automated workers know the distribution of costs associated with possible tasks as well as the probability of the arrival of new agents. Therefore, they can act strategically. In our case however, it is assumed that the workers do not act strategically. Although one of our proposed methods uses the Vickery auction as well, it is used for estimating the distribution of the reservation prices of the workers and not for the actual allocation as in Sarne and Kraus. Furthermore, they do not face the challenge of negotiating with people and therefore their assumptions about agents' behavior are much simpler to handle. Negotiation has also been used for resource allocation (see [12] for a survey). Some works consider negotiations as a distributed mechanism for resource allocation between multiple agents focusing on properties such as social welfare and fairness of the resulting allocation (e.g., [13]). Our work is much different in the sense that we have one agent that needs to negotiate repeatedly with different human workers who lack knowledge about the negotiation protocol.

The *Secretary Problem* (see [14]) is a stopping point problem, where in its simplest form there is a secretary opening with n candidates. The candidates are interviewed in random order and a rejected candidate cannot be recalled. The interviewer is interested in maximizing his probability for choosing the best candidate. It has been shown that the best policy is to reject the first $1/e$ applicants and then choose any applicant better than all applicants seen so far. Using this policy, the interviewer will find the best candidate with a probability of $1/e$. Our problem differs from the secretary problem by allowing bargaining with each candidate. We are not interested in the best (or cheapest) candidate rather minimizing the overall expected cost.

Sandholm and Gilpin [15] study sequences of take-it-or-leave-it offers, where a seller proposes a sequence of offers to a set of buyers and each buyer in turn either accepts the offer, pays that amount, obtains the good and ends the game, or rejects the offer and the next buyer receives his own offer. However, in their study the seller must announce the full sequence in advance, leading to a completely different strategy than we intend to study.

The ultimatum game (see [16]) is a well known game with two players in which a proposer suggest a way to divide a fixed sum, and a responder may either accept the proposal and each player will get its share, or reject it in which case both the proposer and the responder receive nothing. Although the sub-game perfect equilibrium (SPE) in the ultimatum game is that the proposer retains nearly the entire amount, it has been shown that people do not follow this SPE (see [17, 18]).

Greezy et al. [19] study the reverse ultimatum game, which is identical to the ultimatum game with one exception: any time the responder rejects an offer, the proposer may suggest another offer, as long as the new offer is strictly higher than the previous one. This change inverts the SPE, leaving the responder with nearly the entire sum and the proposer with nearly nothing. Once again, Greezy et al. show that people do not follow this SPE, but do not suggest any strategy for the proposer.

Woolley et al. [20] study "collective intelligence for groups of people. They define a collective intelligence factor and state that this factor appears to depend both on the composition of the group and on factors that emerge from the way group members interact when they are assembled. Although we intend to design a system that interacts with many workers, we concentrate on settings where there is a single task that has to be performed many times by different humans independently, and do not consider collaborative tasks.

Galuscak et al. [21] provide a survey on criteria which are used to determine the wages of newly hired employees in Europe. In [22] Walque et al. provide a formal model of a labor market, including methods for payment and wage negotiation. However, these papers merely study the nature of wage determination and do not provide employers with any strategies for determining appropriate wages.

Horton and Zeckhauser [23] study negotiation over tasks payment in crowdsorcing, using the alternating offers protocol. They present an agent for negotiation, which accepts any offer below some threshold, rejects and ends the negotiation if the offer is above some threshold. In any other case it makes a counter offer which is a weighted average between the subject's offer and the acceptance threshold. They tested their algorithm using Amazon's Mechnical Turk ([24]). They split their subjects into two groups, where one group was initially offered 1 cent, and the second 5 cents. However, most subjects (68%) avoided any negotiation, and accepted the initial offer. We examine an additional environment where the task is composed of many milestones, and in general, use a more appropriate protocol where only the requester makes the offers.

Faridani et al. [25] consider the question of what reward to offer in order to complete all tasks in a certain time using Amazon's Mechanical Turk. They analyze the survival time of a task (i.e. the expected time until all tasks are completed) given its reward and attributes such as task difficulty, number of tasks and day of week it was posted on. Using this data they propose an algorithm, that when given a desired completion time and the task's attribute, calculates the optimal reward.

Riley and Zeckhauser [26] have shown that a seller receives the highest utility when he does not allow any bargaining at all. In [27] we tested an effect which we termed "the bargaining effect". This effect claims that if a worker is offered a price which is too low and rejects it, he is more likely to reject also future offers than a worker who did not receive the initial low offer. More formally, suppose a worker's minimum offer that he will accept as a first offer is y. The bargaining effect claims, that the worker is likely to reject this offer (y) if it follows a lower offer already rejected by the same worker.

We recruited 60 subjects and split them into two groups - the "No Bargaining Group" and the "Bargaining Group" (30 workers in each). Workers in the "Bargaining Group" were first offered 1 cent for performing a task, and, if rejected, were offered 4 cents for the same task. Workers in the "No Bargaining Group" were offered 4 cents for performing the same task with no other offer. Neither of the two groups knew if they would receive any sequential offers upon rejection. All subjects must have intentionally accept or reject each offer received in order to complete the task.

Table 1 lists the results of the experiment. In the "No Bargaining Group" 24 subjects accepted the 4 cent offer. In the "Bargaining Group" 13 subjects accepted the first offer (of 1 cent), however, among those who rejected the first offer, only a single subject accepted the second offer (of 4 cents). Thus, a total of only 14 subjects accepted the 4 cent offer in this group substantially less than in the No Bargain Group. These results differ significantly using Fisher's exact test ($p < 0.01$).

There are several possible psychological explanations for such a behavior, and the exact cause remains an interesting question. One possible explanation was already noted by [28] and later by [29] saying (p. 121): "A 'No' response is a most difficult handicap to overcome. When you have said 'No' all your pride of personality demands that you remain consistent with yourself."

Table 1 Bargaining Effect Experiment

		subjects accepting offer		
group	participants	1 cent	4 cents	total
No Bargaining	30	–	24	24
Bargaining	30	13	1	14

In [30] Ariely et al. demonstrate an effect called anchoring. In their study, subjects were asked if they would be willing to pay as much in dollars as the last two digits in their Social Security Number (SSN) for an item. They were then asked how much they would be willing to pay for that same item. Ariely et al. show a high correlation between the last two digits of the SSN and the amount the subjects were willing to pay. We would like to note that in the bargaining averse experiment above, the second group, who first received an offer of only 1 cent, were actually anchored to a lower offer than the first, therefore according to the anchoring effect, they should have been willing to accept a lower final offer. However, as shown by the results, it seems that (under the settings we used) the bargaining effect is stronger than the anchoring effect.

3 TCP NP-Harness

In this section we show that even the Basic-TCP is NP-Hard. The proof is by reduction from Set-Cover - a classical NP-Complete problem [31], defined as follows.

SET-COVER:
Given: set of elements $U = \{e_1, e_2, ..., e_N\}$, collection of subsets $\mathcal{X} = \{S_1, S_2, ...S_M\}$ ($S_i \subseteq U$), and integer κ,
Decide: is there a collection $\mathcal{Y} \subset \mathcal{X}$, of size κ, the union of which is U (i.e. $\cup_{S_i \in \mathcal{Y}} S_i = U$)?

For proving NP-hardness we also define the decision version of the TCP:

TASK-COMPLETION-PROBLEM:
Given: task set $\mathcal{A} = \{A_1, \ldots, A_n\}$, set of ℓ worker decision policies, $d_1, \ldots, d_\ell$,

distribution π over the ℓ worker types, set of possible offers $O = \{o_1, \ldots, o_m\}$, costs C_c and C_o, and target cost x,
Decide: is there a policy for the requester that completes all tasks with expected cost $\leq x$?

Theorem 1 TASK-COMPLETION-PROBLEM *is NP-hard.*

Proof Given an instance of $(U, \mathcal{X}, \kappa)$ for Set-Cover, we construct an instance of TCP, as follows. We set $\mathcal{A} = \{A_1\}$ (i.e. only a single assignment), and $O = \{1, 2, ..., M, M+1\}$ (the set of possible offers). For each element $e_i \in U$ we construct a type t_i, with the following decision function:

$$d_{t_i}(h, o) = \begin{cases} accept & \text{if } o = M+1 \text{ and } M+1 \notin h \text{ and } e_i \in \bigcup_{j \in h} S_j \\ reject & \text{else} \end{cases}$$

Thus, all types may only accept the offer $M+1$, and that also only if this is the first time it is offered. Additionally, type t_i accepts $M+1$ iff (the index of) one of the sets containing e_i was previously offered, i.e. iff $e_i \in S_j$, for some $j \in h$. We set π to be the uniform distribution, $C_o = 1$, $C_c = (\kappa + M + 2) \cdot N$, and $x = \kappa + M + 2 + C_c$. We now show that there exists a solution to TCP with expected cost $\leq x$ iff there is a solution to the set cover problem with at most κ sets.

Let $\mathcal{Y} = \{S_{i_1}, \ldots, S_{i_\kappa}\}$ be a solution to the set-cover problem. Then the requester can use the following policy: first make the offers $i_1, \ldots, i_\kappa$ (in any order), and then offer $M+1$. Since $\cup_{j=1}^{\kappa} S_{i_j} = U$, this policy guarantees acceptance by any type. The total number of offers is $\kappa + 1$, and the payment to the worker is $M + 1$. Thus, the task is completed with a fixed cost $C_c + \kappa + M + 2 = x$.

Conversely, suppose that there is requester policy with expected cost $\leq x$. First note that it must be the case that the policy guarantees acceptance by the first worker, regardless of its type. In contradiction, suppose that there is a type t_{i_0} that rejects with this policy. Then, if t_{i_0} happens to be the first worker, the total cost is $> 2C_c$. Thus, the expected total cost is

$$> C_c + \frac{C_c}{N} = C_c + \kappa + M + 2 = x$$

(since t_{i_0} has probability $1/N$). Thus, the requester policy must guarantee completion of the task with the first worker, regardless of its type. Clearly, the policy makes the offer $M+1$, and does so last. Let $i_1, \ldots, i_{\kappa'}$ be the set of offers it makes before offering $M+1$. Since acceptance is guaranteed for all types it must be that for every type t_i, $e_i \in \cup_{j=1}^{\kappa'} S_{i_j}$. Thus $S_{i_1}, \ldots, S_{u_{\kappa'}}$ covers U. The total cost of the policy is $C_c + \kappa' + M + 2$, and hence $\kappa' \leq \kappa$. Thus, $S_{i_1}, \ldots, S_{u_{\kappa'}}$ is a solution for the set-cover problem.

4 Agents

Since solving the TCP is intractable, we turn to solve restricted versions of the problem. The restrictions we consider correspond to possible restrictions on the structure worker decision policies. We consider two such restrictions, each of which is based on human behavior models proposed in the literature, and may thus provide a reasonable approximation of the true behavior of humans. For each behavior model we develop an agent that solves the corresponding TCP and interacts with the human workers. Note that we make no claim that either of these agents perfectly models true human workers. Rather, we hope that they are based on a good enough approximation and evaluate their performance experimentally.

4.1 Reservation Price Agent

Our first restricted model presumes that worker responses do not depend on previous offers, but solely on the current offer. We further assume that the policy is monotone: if x is accepted, then so is any offer $y > x$. This model is based on the concept of *reservation price (RP)*, which assumes that non strategical behavior implies that people buy (or sell) any item that has a cost lower (or higher) than their reservation price. This assumption is very common in the economy literature (e.g. [32]). Therefore, each type t_i is characterized by a reservation price, rp_i, such that t_i accepts an offer o iff $o \geq rp_i$ (for the milestone version, there is a reservation price for each milestone). We show that under this assumption TCP is tractable, and develop the associated agent.

4.1.1 Optimal Policy RPBA Algorithm.

We first describe the optimal policy algorithm of the reservation price agent for the case of a single assignment with no milestones. We then describe the construction for the setting with milestones which is more complex.

Optimal Policy RPBA Algorithm for Basic TCP For any price x, denote by $p(x)$ the fraction of types with reservation price x, and $P(x) = \sum_{x' \leq x} p(x')$. Recall that the number of types and thus the number of different reservation prices is ℓ, let $rp_1 < rp_2 < \cdots < rp_\ell$ be the different reservation prices, and set $rp_0 = 0$. Clearly there is no sense in making any offer other than one of the reservation prices. Further note that under the reservation price assumption, for any deterministic policy (for the requester) there is necessarily some reservation price rp_s, such that any worker with a reservation price greater than rp_s will reject all offers (which the requester proposes according to the policy), and any worker with a reservation price lower than or equal to rp_s will accept some offer. Since we do not know s, we test all possible values for s. For each, we compute the optimal policy assuming all and only agents with a reservation price of at most rp_s will accept an offer (as explained below). We then calculate the expected cost for each of these policies and choose the policy with the lowest expected cost.

Suppose we want to accept all and only workers with a reservation price of at most $\leq rp_s$. Clearly, since workers with a reservation price of rp_s need to accept, we must make an offer of at least rp_s. However, since workers with a higher reservation price must reject, we cannot make a higher offer. Thus, the offer rp_s must be made, and it will be the last offer.

For workers with lower reservation prices, it would be best to offer each specific worker its exact reservation price. However, there is a cost for each offer. Thus, there is a tradeoff between not offering much more than necessary and minimizing the number of offers. We determine the optimal offer sequence using dynamic programming, as follows.

We construct a table T of size $s \times s$, as follows. Consider j, i with $1 \leq j \leq i \leq s$. Suppose a worker has rejected the offer rp_{j-1}, entry $T(i, j)$ stores the expected cost of completing the interaction with this worker, under the optimal strategy, with the following constraints:

- The next offer must be rp_i.
- All and only workers with reservation price $\leq rp_s$ accept an offer.

We now show how to fill-in the entries of T. Recall that C_c denotes the cost for calling in a worker and that C_o denotes the cost for proposing an offer. For a given $i \geq j$, let $p_a(i, j)$

$T(s,s) \leftarrow C_o + p_a(s,s) \cdot rp_s$
for $i = s-1$ downto 1 do
  $c_r(i) \leftarrow \begin{cases} 0 & \text{if } i = s \\ \min_{i':i+1 \le i' \le s}\{T(i', i+1)\} & \text{else} \end{cases}$
  for $j = i$ downto 1 do
    $T(i,j) \leftarrow C_o + p_a(i,j) \cdot rp_i + (1 - p_a(i,j))c_r(i)$

Fig. 1 Computing the Table $T(i,j)$

be the fraction of workers with a reservation price of at least rp_j who would *accept* offer rp_i:

$$p_a(i,j) = \frac{\sum_{x=rp_j}^{rp_i} p(x)}{\sum_{x \ge rp_j} p(x)}$$

Since the offer rp_s must be made to workers with a reservation price of rp_s, we have:

$$T(s,s) = C_o + p_a(s,s) \cdot rp_s \tag{1}$$

The expected cost of completing the interaction with current worker after *rejection* of the offer rp_i is:

$$c_r(i) = \min_{i' \ge i+1} \{T(i', i+1)\} \tag{2}$$

for $i < s$, and 0 for $i = s$ (since we assume that rp_s is our top offer).

For other i, j, we have:

$$T(i,j) = C_o + p_a(i,j) \cdot rp_i + (1 - p_a(i,j))c_r(i) \tag{3}$$

Using (1), (2) and (3), table T is filled out in descending order of j and i (see Figure 1).

The overall expected cost per worker, assuming that all and only workers with reservation prices of, at most, rp_s accept, is:

$$\min_i \{T(i,1)\} + C_c \tag{4}$$

The expected number of workers with a reservation price of, at most, rp_s is $P(rp_s)$. Thus, if only workers with a reservation price of, at most, rp_s accept, the expected cost until the assignment is performed is:

$$cost(s) = \frac{\min_i\{T(i,1)\} + C_c}{P(rp_s)}$$

We iterate through all possible values of s to find the optimal one. By saving the indexes found while computing the minimum in Equations 2 and 4, the algorithm outputs a policy for the RPBA (Reservation Price Based Agent), detailing exactly what offers to make at any given stage, including the decision of if and when to abandon the existing worker and seek another one. More explicitly, once the optimal s is found (denoted s^*), the highest offer made by RPBA is rp_{s^*} and if a worker rejects the offer rp_{s^*}, RPBA will call in for the next worker. The first offer made by RPBA is determined by Equation 4 (when computing the table assuming rp_{s^*} as the highest offer), if for example $\arg\min_i\{T(i,1)\} = \alpha$, the first offer for RPBA is rp_α. All proceeding offers are determined by Equation (2). For example, $\arg\min_{i' \ge \alpha+1}\{T(i', \alpha+1)\} = \beta$ (when computing the table assuming rp_{s^*} as the highest offer) indicates that after rejection of rp_α, RPBA will offer rp_β. The algorithm runs in $O(l^3)$ time and $O(l^2)$ space (where ℓ is the number of different types / reservation prices).

Optimal Policy RPBA Algorithm for Upfront M-TCP We denote by $\text{RP}_t(m)$ the reservation price of worker with a type t for milestone m. We further assume that the workers and thus their types are well ordered with regards to their reservation prices, i.e. if for milestone m, $\text{RP}_{t_1}(m) > \text{RP}_{t_2}(m)$ then for any other milestones m', also $\text{RP}_{t_1}(m') \geq \text{RP}_{t_2}(m')$. We call this *consistency*.

The algorithm for the setting with milestones with upfront reward determination is identical to the basic TCP, since the bargaining is only performed in the first stage. The only difference is that the reservation prices are replaced by the cluster types and the associated cost is replaced by the total cost for the worker to complete the full assignment (all milestones). Due to the consistency assumption, if a worker accepts a schedule, he should complete the full assignment.

Optimal Policy RPBA Algorithm for Stepwise M-TCP We compute the optimal policy under the RPBA assumptions using dynamic programming. We start by sorting the types based on their reservation price, such that type t_i requires no more than type t_{i+1} for each of the milestone. This is possible due to the consistency assumption. Recall that the number of types is ℓ.

Note that once an optimal policy is constructed, all workers face the same deterministic policy. Therefore, for this optimal policy there is some type t_s, that any worker with a type greater than s will not complete the assignment, and any worker with a type smaller or equal to s will complete the assignment. Since we do not know s we try all possible values (from 1 to ℓ). For each, we compute the optimal policy assuming all and only agents of type at most t_s can complete the assignment (details follow). We then calculate the expected cost for each of these policies and choose the policy with the lowest expected cost per assignment.

Clearly, under the RPBA assumptions, any offer for a milestone must correspond to RP of some type for that milestone. Denote by RP_m^j the RP of type t_j for milestone m. Also note that for any given milestone offers must always be monotonically increasing.

Given a new worker, the agent does not know its type. As the interaction continues, the agent obtains tighter bounds on the worker's possible type. Specifically, given a milestone m and a worker of type j denote by h the highest index such that $RP_m^h = RP_m^j$. If the worker accepts an offer of RP_m^j on milestone m, then its type is at most h, and if it rejects RP_m^j then its type is at least $h+1$. Denote by $\underline{j}$ and $\overline{j}$ the currently known bounds on the minimum and maximum possible index types of the current worker, respectively.

Suppose we want to accept all and only workers of type at most t_s (as described above). We construct $O(\ell^2)$ tables, one for each of the possible combinations of $(\underline{j}, \overline{j})$. In each table, the columns represent the milestones (k columns) and the rows represent the types. In table $T[\underline{j}, \overline{j}]$, entry (i, j) stores the expected cost for completing the interaction with this worker using the optimal policy, assuming:

- The type of the worker is between $\underline{j}$ and $\overline{j}$, inclusive, and the agent knows that this is the case.
- Any worker with a type greater than t_s should be rejected (once this becomes known).
- The worker has completed all milestone up to $i-1$, inclusive.
- The agent must make the offer RP_i^j at milestone i.

All entries that correspond to a type smaller than $\underline{j}$ are ignored, since for any milestone m any offers lower than $RP_m^{\underline{j}}$ will definitely be rejected. The table ends at $\overline{j}$ since the offer corresponding to $\overline{j}$ will definitely be accepted and there is no sense in making higher offers.

Given $\underline{j}, \overline{j}, i, j$ we compute the acceptance probability of RP_i^j by a worker with type $\underline{j} \leq t \leq \overline{j}$ based on the cluster prior, using Bayesian rule. Denote this probability by $p_a(\underline{j}, \overline{j}, i, j)$.

The expected cost for a worker with type t, $\underline{j} \leq t \leq \overline{j}$ to complete the assignment after accepting offer RP_i^j on milestone i, can be computed by:

$$c_a(\underline{j}, \overline{j}, i, j) = \begin{cases} 0 \text{ (Last milestone)} & \text{if } i = \ell \\ \min_{\underline{j} \leq j' \leq h} (T[\underline{j}, h](i+1, j') & \text{else} \end{cases}$$

Similarly, the expected cost for a worker with type t, $\underline{j} \leq t \leq \overline{j}$ to complete the assignment after rejecting offer RP_i^j on milestone i, can be computed by :

$$c_r(\underline{j}, \overline{j}, i, j) = \min_{h+1 \leq j' \leq \overline{j}} (T[h+1, \overline{j}](i, j')$$

Finally we set $T[\underline{j}, \overline{j}](i, j)$ using the following formula:

$$T[\underline{j}, \overline{j}](i, j) = \begin{cases} 0 \text{ (Terminate process)} & \text{if } \underline{j} > s \\ C_o + p_a \cdot (RP_i^j + c_a) + (1 - p_a) c_r & \text{else} \end{cases}$$

where, $p_a = p_a(\underline{j}, \overline{j}, i, j)$, $c_a = c_a(\underline{j}, \overline{j}, i, j)$, and $c_r = c_r(\underline{j}, \overline{j}, i, j)$. The tables are filled in descending order of the milestone number (i), $\overline{j}$, $\underline{j}$ and worker type (j).

The final expected cost given t_s is the minimum on the first row of $T[1, l]$ plus C_c.

Indexes to the entries that lead to local minimum expected cost (found while computing c_r and c_a) are saved during the process, so once the minimum expected cost is found the policy tree can be composed.

The algorithm runs in $O(k^2 \cdot l^4)$ time and $O(k^2 \cdot l^3)$ space.

The output of the algorithm is a policy of the RPBA (Reservation Price Based Agent), detailing exactly what offers to make at any given stage, including the decision if and when to abandon the existing worker and seek another one.

4.2 No Bargaining Agent

The reservation price based agent is based on the assumption that people have a fixed, a priori, reward they require for the task, which is not influenced by the offer history. Such behavior, while possibly rational, is not always exhibited in practice. Frequently, the interaction with the requester largely determines the desired price (so long as it is within a reasonable range). In particular, just knowing that the price is negotiable seems to push up the price. Thus, we now turn to the other extreme and assume that if a worker rejects an offer, it will reject any following offer, unless is it much much higher - to the extend that it is always preferable to bring in a new worker rather than better the offer to an existing worker. Thus, the agent can effectively make only one offer to each worker. We term this kind of agent the *No Bargaining Agent (NBA)*. This agent makes only one offer for each assignment or milestone (to any given worker) and never makes a second offer if rejected. Again, we begin by describing the algorithm for the basic TCP and then describe the M-TCP settings.

Finding an optimal policy for Basic TCP. Let $u(x)$ be the fraction of workers that accept a first-time offer x (out of the entire set of workers). (This fraction can be easily calculated simply by adding the distribution of all types which accept the offer x.) Given the acceptance distribution, we wish to find the optimal policy. Due to the no-bargaining policy, once an offer is rejected, NBA does not attempt to make another offer, but calls for the next worker. Therefore, for the no-milestone case, NBA's policy consists of a single offer. With a policy of offering x, the expected cost per worker is:

$$C_c + C_o + x \cdot u(x)$$

The expected number of workers sampled until a worker accepts the offer is $1/u(x)$. Thus, the expected cost per completed assignment is:

$$\mathit{cost}(x) = \frac{C_c + C_o + x \cdot u(x)}{u(x)} \tag{5}$$

Using this cost as our fitness function, we perform a search to find the optimal offer x. Note, that even if we perform an exhaustive search, since, in the formalized model there are only m possible offers, the complexity NBA is $O(m \cdot \ell)$.

Finding an optimal policy for M-TCP As in the case of the RPBA we assume that the workers' valuations are consistent. I.e. if worker of type t_i accepts an offer o on milestone m and rejects an offer o' on milestone m', any other worker with any other type t'_i who rejects offer o on milestone m rejects offer o' on milestone m'. Since the NBA agent offers a single offer for each of the milestones and does not perform any bargaining, its policy is identical for both M-TCP with stepwise reward determination and for M-TCP with upfront reward determination. Given the acceptance distribution, we wish to find the optimal policy. Due to the no-bargaining policy, once an offer is rejected, NBA does not attempt to make another offer, but calls for the next worker. Therefore NBA's policy consists of a single offer for each milestone. Consider such a sequence $\mathbf{o}$ of offers, $\mathbf{o} = (o_1, o_2, \ldots, o_k)$. Consider workers w and w' and milestones m_i and $m_{i'}$. By consistency, if worker w accepts $\mathbf{o}$ on m_i, and w' accepts $\mathbf{o}$ on $m_{i'}$, then either w also accepts $\mathbf{o}$ on $m_{i'}$ or w' accepts $\mathbf{o}$ on m_i. Thus, either $S^i(o_i) \subseteq S^{i'}(o_{i'})$ or $S^i(o_i) \supseteq S^{i'}(o_{i'})$. Thus, the fraction of workers that will accept the offers for both milestones is: $\min\{u^i(o_i), u^{i'}(o_{i'})\}$. Thus, the fraction of workers that will accept the offers for milestones m_1 through m_{i_0}, denoted $e_{i_0}(\mathbf{o})$, is:

$$e_{i_0}(\mathbf{o}) = min_{i \le i_0}(u^i(o_i)) \tag{6}$$

Therefore given $\mathbf{o}$ the fraction of workers that will complete the full assignment is simply $e_k(\mathbf{o})$. The estimated expected cost of a single worker (whether he completes the assignment or not) is given by:

$$C_c + C_o + \Sigma_{i \le k}(o_i + \mathbf{1}\{i < k\} \cdot C_o) \cdot e_i(\mathbf{o}) \tag{7}$$

The intuition behind (7) is that C_c must be paid just for calling the worker, C_o must be paid for the first offer. The offer for milestone m_i (o_i) and the cost to make the consecutive offer C_o is only paid if the worker accepts the offers for milestones m_1 through m_i ($e_i(\mathbf{o})$). Accepting the offer for the last milestone m_k requires the agent to pay for that offer (o_k) but not for making the consecutive offer (since there is none).

Dividing by $e_k(\mathbf{o})$, we obtain that the expected cost for completing the assignment is:

$$Cost = \frac{C_c + C_o + \Sigma_{i \leq k}(o_i + \mathbf{1}\{i < k\} \cdot C_o) \cdot e_i(\mathbf{o})}{e_k(\mathbf{o})} \quad (8)$$

Using this cost as our fitness function, we perform a search to find the optimal $\mathbf{o}$ (i.e. the one with the lowest cost).

5 Type Elicitation

The formulation of TCP assumes that the type decision policies and the type distribution are given. In reality, however, this is rarely the case. We therefore propose two methods for the type elicitation, each method is designed to suit each of the proposed agents.

5.1 Type Elicitation for RPBA

In order to construct a policy for the agent, we must know the reservation prices of workers, or the distribution thereof. Since we are not given this distribution, we approximate it by sampling a subset of the workers and eliciting the reservation prices by means of a truthful auctioning mechanism.

We consider the following two methods, in which the dominant policy for each player is to bid for his truthful reservation price. The first is the Vickrey auction [33]. A Vickrey auction is a sealed bid auction where each worker submits a bid and the worker with the lowest bid performs the task and is paid the amount requested by the second lowest bid. Allocating tasks using a Vickrey auction was used, for example, in [34]. The second method is the Becker-DeGroot-Marschak (BDM) mechanism [35]. In the BDM mechanism, the worker bids for an amount, then a computer selects a number by random, if the random number is lower than the bid, no work is done; however, if the random number is greater than the bid, the worker performs the task and is paid the sum given by the random number.

Noussair et al. [36] compare the Becker-DeGroot-Marschak (BDM) mechanism with the Vickrey Auction and show that although both methods do not disclose people's true reservation prices, the Vickrey auction is more effective and better than BDM for eliciting human's true reservation price.

We therefore, sample a subset of the workers at random and perform a Vickrey auction with them. In order for workers to have a reasonable chance of winning, we divide the set of workers into subsets of three workers, where each worker only competes with the bids offered by the other workers in its subset. Since we use a crowdsourcing environment, each bid submitted is compared to the bids submitted by the previous 2 workers. This method sustains the dominant policy of bidding truthfully.

In both settings with milestones, we do not consider all reservation prices for each of the milestones instead we cluster the data according to the bids received for each of the milestones. Unfortunately, some workers did not fill out the biddings seriously, therefore, we use noise reduction techniques to eliminate noisy bids. We use the nearest neighbor based noise detection described in [37]. We then use the expectation-maximization (EM) to cluster the data. EM is most appropriate since the data is assumed to be sampled from Gaussians with possibly different standard deviations. Determining the number of clusters is done using the Bayesian Information Criterion (BIC) implementation in MCLUST (see [38]), as described in detail in [39]. Each cluster is associated with a type.

5.2 Type Elicitation for NBA

The best offer for the NBA to make depends on what offers would and would not be accepted, and the distribution thereof. We thus wish to estimate this distribution. We assume that the portion of workers who would accept a first-time offer of x follows a sigmoidal distribution (in x) (The use of a sigmoid function to describe the probability that people accept offers is common in the literature, see [40,41]). We approximate the distribution by choosing several points, then sample a subset of workers and obtain their acceptance fraction for these points (different workers for different points). These workers are only asked if they would be willing to complete an assignment for a given amount, they aren't required to actually perform the assignment. Once we receive their responses we can interpolate the sigmoid from these values. In the settings with milestones, we sample points and interpolate the sigmoid for each of the milestones. In order to reduce the number of workers needed in this learning phase, each worker is asked for each of the milestones if he would be willing to complete it for a given sum.

6 Experimental Evaluation

All of our experiments were performed using Amazon's Mechanical Turk service (AMT) [24][3].

The basic TCP was composed of a single milestone (by definition). Both M-TCP with stepwise reward determination and M-TCP with upfront reward determination were composed of five milestones ($k = 5$). We set C_c (cost for calling a new worker) to 20 cents, and C_o (cost per offer) to 4 cents. These are the same settings as used in [27], and have been chosen, since they yield policies for NBA and RPBA which differ significantly from one another. In all of the experiments the subjects were only asked if they were willing to complete an assignment for a given amount, and were not told anything regarding the requester's costs or incentives. This is similar to how these assignments are offered in reality.

We conducted two sets of experiments with varied parameters. The first set of experiments was presented in detail in [27], and only briefly reviewed here. The second set of experiments is newly published in this paper, and described in detail below. Table 2 summarizes the differences between the two sets of experiments.

Table 2 Comparison between previously published experiments and new set of experiments

	Previous experiments	Current experiments
Culture	USA	India
Assignment	Distinctive shape	Text encoding
NBA Elicitation	Actual	Questionnaires
Results	NBA outperformed others	NBA and RPBA outperformed Experts

6.1 Manually Designed Agents

In order to evaluate the quality of NBA and RPBA, we also constructed an additional set of agents. We interviewed four people with significant experience as requesters in Mechanical

[3] For a comparison between AMT and other recruitment methods see [42].

Turk (whom we refer to as experts). After explaining the problem, each expert was requested to compose a policy. We considered the average cost among the four experts as the cost for the manually designed *Experts' agent*.

6.2 Previously Published Experiments

In the first set of experiments all assignments and milestones were composed of a set of simple puzzles where the subjects were required to find a distinctive shape among other shapes. The different milestones varied in the number of puzzles needed to be solved by the worker and the number of shapes. Subjects were USA residents. NBA price elicitation was conducted using subjects who were asked if they were willing to complete an assignment for a certain amount and if accepted they had to actually complete that assignment. Our results showed that NBA significantly outperformed both RPBA and the Experts.

6.3 New Experiments

The Assignments All assignments and milestones were composed of paragraphs which the subjects were required to encode by writing the number of letters in each word. The paragraphs differed in their length. The number of words in each paragraph was clearly stated. Please refer to Figure 2 for a screen-shot example. The workers were given the following instructions: "Please encode the given paragraph using the following method: each word should be replaced with the number of letters it has, any white spaces (spaces, tabs or line breaks) should be replaced with a single space. Any other symbols should be ignored. For example the following sentence: 'Ducks are sometimes confused with several types of unrelated water birds with similar forms, such as loons or divers, grebes, gallinules, and coots.' should be encoded as: 5 3 9 8 4 7 5 2 9 5 5 4 7 5 4 2 5 2 6 6 10 3 5 If you make any mistakes you will be asked to fix them, but you will not receive any indication on where they were." In all of the experiments, all of the subjects were first required to encode a short paragraph (without being paid any additional amount) to make sure that they understood the nature of the assignment.

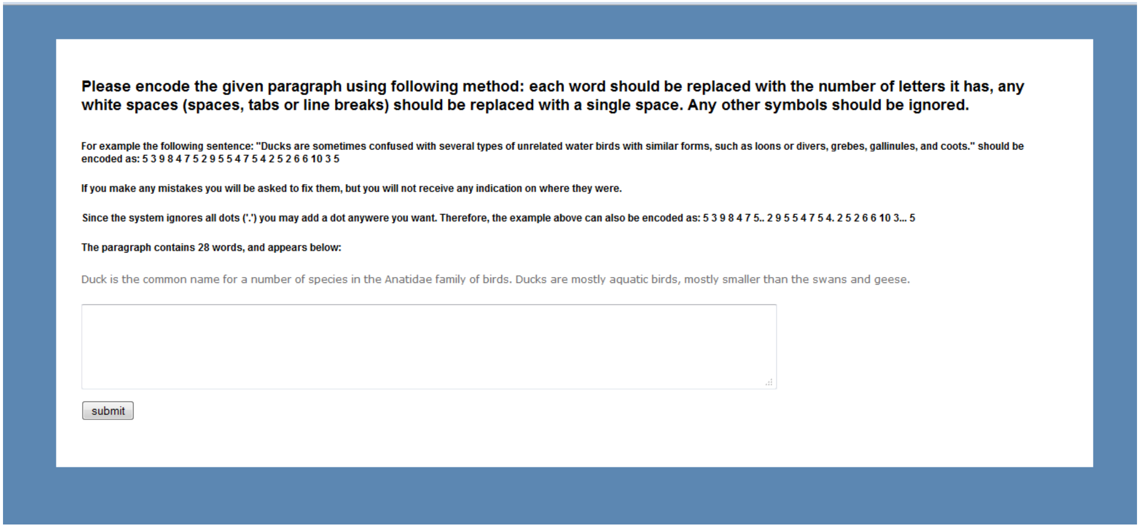
In the basic TCP the requester needed to complete 25 instances of this core task. In the M-TCP, each assignment was consisted 5 milestones, each of which was an encoding task as described above. The requester needed to complete 25 such (5 milestone) assignments. For each such TCP (basic or M-TCP), we applied: the NPBA agent, the NBA agent, and the four Experts's agents.

Workers Participation consisted of a total of 1117 subjects, all from India, of which 34.1% were females and 65.9% were males. Subjects' ages ranged from 18 to 76, with a mean of 29.25 and median of 27. All subjects were paid 11 cents for participating in the study. Any extra credit gained in the assignment was given to the subjects as a bonus. A large majority of the subjects, 62.3%, had a bachelors degree, 22.7% had a masters degree, 0.3% had a PhD, and the remainder, 14.7%, had no academic degree. The subjects were required to fill out a questionnaire asking for their age, gender and education before they started with the assignment.

The 1117 subjects were employed in the following experiments:

- RPBA type elicitation: 182 subjects participated in the Vickrey Auction for the RPBA agent type elicitation.

Fig. 2 Example for a paragraph encoding request



- NBA type elicitation: 90 subjects participated in the NBA type elicitation process.
- Assignments: the remaining 845 subjects participated as human workers for the different agent. Of these, 151 faced the RPBA agent, 154 faced the NBA agent, and 540 faced the human agents (the number of subjects that participated in each of the groups depended on the number of subjects that were required for each of the agents to meet its goal).

6.4 Results

We ran 25 instances of the experiment for each of the agents, i.e. each agent had to accomplish the goal 25 times. The manually designed agents were required to accomplish 25 goals each, leading to a total of 100 goals for all four experts. Participants consisted of 845 subjects.

Tables 3, 4 and 5 provide the results obtained for the basic TCP, the M-TCP with stepwise reward determination and the M-TCP with upfront reward determination, respectively. Results show the average number of workers called, the average number of offers given and the average cost per assignment performed. The last column shows the acceptance rate - the fraction of accepted offers. Combining all experiments together, Figure 3 shows the average performance of the three agents over all three settings we tested. Clearly, NBA performs the best, though only very slightly better than RPBA. For the statistical test, we compared all three agent using the ANOVA test, and both RPBA and NBA performed significantly better than the experts ($p < 0.001$). Table 6 shows the final average cost over all three settings where each expert is shown separately. As can be seen, neither of the experts performed better than the automated agents, however Expert #3, performed far worse than all the rest. However, even after removing Expert #3 from the results, the automated agents still significantly outperform the other three experts ($p < 0.01$). It might be interesting to note, that Expert #3, was the only expert who proposed only a single offer to each worker (for each

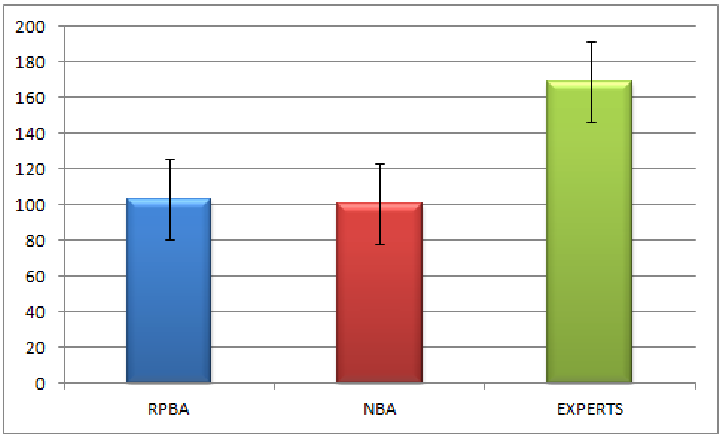


Fig. 3 Average cost per goal over all three settings

milestone), which is similar to NBA's approach. However, his mal-performance is explained by his offers being too high, as shown in Table 7.

Table 3 Average per Goal (completion of a single milestone) in Basic TCP

agent	workers	no. offers	cost	offer accept. rate	assignment completion rate
RPBA	1.32	1.4	52	91.4%	71.4%
NBA	1.16	1.16	48.9	90.0%	87.0%
EXPERTS	1.24	1.31	47.8	91.9%	76.4%

Table 4 Average per Goal (completion of all 5 milestones) in M-TCP, Stepwise Reward Determination

agent	workers	no. offers	cost	offer accept. rate	milestone completion rate
RPBA	2.52	10.16	142.7	88.6%	84.8%
NBA	2.64	9.71	133.8	87.3%	83.1%
EXPERTS	2.13	10.23	221.1	81.3%	76.5%

Table 5 Average per Goal (completion of all 5 milestones) in M-TCP, Upfront Reward Determination

agent	workers	no. offers	cost	offer accept. rate	milestone completion rate
RPBA	2.2	8.68	114	91.2%	84.8%
NBA	2.36	8.96	119.5	92.4%	85.3%
EXPERTS	2.03	8.69	238.1	92.8%	87.4%

We conclude the results section by comparing the prediction provided by the two automated agents to the actual data. Table 8 compares the fraction of subjects who accepted and completed the first offer with the prediction provided by both automated agents, for each of the settings. The final column presents the Root-Mean-Square Deviation (RMSD). We show

Table 6 Average cost per goal over all three settings

agent	average cost
RPBA	102.9
NBA	100.7
EXPERT #1	118.2
EXPERT #2	121.1
EXPERT #3	294.1
EXPERT #4	142.5

Table 7 Average offer over all three settings

agent	average cost
RPBA	10.78
NBA	10.82
EXPERT #1	7.98
EXPERT #2	13.19
EXPERT #3	45.17
EXPERT #4	16.26

the results only for the first offer, since all participants were offered the first offer, and any other offer was either offered only upon rejection to the first offer or upon completion of it. As can be seen by the table, while the prediction of the NBA agent seems reasonable, the prediction of the RPBA agent seems unacceptably off. This result suggests that price elicitation by sampling a subset of workers and interpolating a sigmoid, is much more accurate than using a Vickrey Auction for the same purpose.

Table 8 A comparison between prediction and actual completion rate for first offer

settings	agent	prediction	actual	RMSD
Basic TCP	RPBA	47.3%	73.5%	0.513
Basic TCP	NBA	56.8%	83.1%	0.457
Stepwise M-TCP	RPBA	52.0%	100.0%	0.47
Stepwise M-TCP	NBA	81.5%	93.3%	0.276
Upfront M-TCP	RPBA	31.8%	88.9%	0.652
Upfront M-TCP	NBA	81.5%	93.1%	0.279

7 Discussion

Although the RPBA performed as good as the NBA agent, we recommend using the NBA agent for the following reasons:

- Since the NBA agent only gives a single offer, it is likely to be easier to implement and maintain.
- When using the NBA agent all workers are paid identically, this seems more fair towards the workers.
- data collection is simpler with the NBA agent since it just requires questionnaires (rather than performing an auction using the RPBA agent)

– The prediction of the NBA is much better than the RPBA's prediction. This may indicate that under different settings, the NBA agent might significantly outperform the RPBA agent.
– In the previous published experiments provided above (Section 6.2), using different settings and a different assignment, a variant of the NBA agent significantly outperformed the RPBA agent.
– Due to the NBA's price elicitation method, it can use any data gained while interacting with humans to refine its believes on human reaction and regenerate a more accurate strategy every interaction.

Examining the detailed dynamics of the different versions exhibits how the willingness of the other agents to bargain harmed their performance. In the basic TCP experiments, there were 5 cases in which a worker rejected some offer and the agent (not NBA) replied with an improved offer. Of these workers, only a single worker ended up completing the assignment. For the remaining 4, all subsequent offers were rejected, and the cost thereof wasted. In the multiple milestone experiments the effect was even more pronounced. In all, in the M-TCP experiments there were 19 workers that rejected an offer and were offered a better one (not by NBA). Of these, only a single worker accepted one of these future offers. However, even that worker did not end up completing the entire assignment. Thus, bargaining seems to be fruitless for the most part, mostly adding to the expenses and seldom bringing about the desired outcome.

Also worth noting is that in the Milestone Task Completion Problem, while the Experts' agents encountered the greatest overall average cost, it also exhibited a smaller average number of workers per assignment completion (see Tables 4-5). This might suggest that in the Milestone Task Completion Problem, after investing some money in having the workers complete their initial milestones, people tend to make offers that are too high for the consecutive milestones. The reason for this is a subject for further research, but perhaps it can be linked to the sunk cost effect (see [43]), where in order to not appear wasteful in the present, people tend to invest more money in a situation where greater money has already been invested, regardless of future expected expenses and utility. Also, the starting offer of the Experts' Agents was higher than that of the automated one, perhaps also linked to the human preference not to "fail".

Finally, it is interesting to note the differences in the results of the two variants of milestones-TCP - the upfront reward determination schedule and the stepwise one. Both automated agents (RPBA and NBA) performed better in the upfront schedule than in the stepwise one, reducing the cost in $\sim 15\%$ on average in the upfront schedule than in the stepwise one. This is the behavior we expected since the commitment on the side of the requester would tend to lower the cost required by the workers. Interestingly, the Experts' Agents performed slightly worse in the upfront schedule. We do not know the reason for this, and further experimentation is necessary in order to validate that such behavior is consistent and to examine the possible reasons.

In all of the experiments that we conducted, the offers were usually less than $\$0.25$. Although these values may seem low, they are very common when dealing with microwork and crowdsourcing. It has been shown that although people's behavior is still affected by the size of the stakes, general trends in human behavior can also be observed as the stakes are raised [44, 17]. Therefore, we believe that the methods suggested can also perform well for higher stakes, however, high stakes are not as common in a crowdsourcing environment.

In this work, we did not take into consideration the quality of the work performed by the workers. Rather, the choice of workers was based solely on cost, and it was implicitly assumed that the wage does not affect the quality of the work. This may not be totally true in all cases, but we believe it is a good first-order approximation. Indeed, many microworks tasks are sufficiently simple that it may be assumed that if a worker is already engaged in the task, the payment level will not affect the quality. Furthermore, most crowd-sourcing applications (including AMT) rate workers based on their performance, and requesters can determine the quality of the workers they employ (e.g. 95% approval rate). In such cases, workers do not want to stain their reputation, and it may be assumed that they continue their high-quality work, regardless of wage. That said, they may also be situations where the quality of the work, and not only cost, must be taken into account. Wage negotiation for such settings is not considered in this work, and is a subject for future research.

We end the discussion by considering the changes that would occur in NBA and RPBA policies as the values of C_c (cost for calling in the next worker) and C_o (cost for making another offer) are modified. If C_c were higher, both NBA and RPBA would propose higher offers, since they would try to have the current worker complete the task and avoid paying the additional fee of calling in another worker. We also assume that the Experts would raise their offers if C_c were higher. If C_o were lower, RPBA would propose more offers, and in the extreme case where $C_o = 0$, RPBA would propose all offers up to a certain threshold. If C_o were higher, RPBA would propose less offers, as each offer is more expensive, however, the offers themselves would be higher, since rejection is more expensive. Since NBA would propose a single offer in any case, changes to C_o would have minor effects on the resulted policy (changes might occur only because C_o is added to C_c).

In this work we enhanced results obtain by previous work [27] using a different assignment and a different culture. We based our learning on questionnaires, rather than actual offers which significantly reduces the overall cost. We prove that the Task Completion Problem is NP-Hard. We presented the full algorithms for both NBA and RPBA agents and elaborated both methods.

8 Conclusions & Future Work

Determining the appropriate sum paid to workers performing microwork when using a crowdsourcing environment is a difficult problem. When a requester needs to recruit a large number of people to perform some assignment, even human experts find it difficult to design the most effective strategy to minimize costs. Ensuring that people complete complex assignments with several milestones is even more challenging, but necessary in many settings.

In this paper we presented a reward strategy that succeeded in facing these challenges by building a general model of the human response to offers and their attitude toward negotiation. Based on this model, we designed an automated agent that interacts with the humans and is successful in minimizing the costs. The accuracy of the human model has benefited from applying principles adopted from behavioral science. We strongly believe that this methodology is useful, and essential, in designing self-interested agents that interact successfully with people in other domains as well.

In the future, we suggest considering the following extension as an avenue for possible improvement of NBA:

Expertise and Boredom: When the milestones are relativity similar to each other, a worker may experience either expertise or boredom when performing more than one milestone. If this happens, this may change the necessary reward structure (e.g. a bored worker may need to be paid more). How exactly to incorporate this into NBA is a subject for future research.

9 Acknowledgments

This paper has evolved from a paper presented at the AAAI-2012 conference [27]. We thank Avi Rosenfeld and Shira Abuhatzera for their helpful comments. This work is supported in part by the following grants: ERC grant #267523, MURI grant #W911NF-08-1-0144, ARO grants W911NF0910206 and W911NF1110344 and ISF grant #1401/09.

References

1. J. Howe, The rise of crowdsourcing, Wired magazine 14 (6) (2006) 1–4.
2. G. Adda, B. Sagot, K. Fort, J. Mariani, et al., Crowdsourcing for language resource development: Critical analysis of amazon mechanical turk overpowering use, in: LTC 2011: Proceedings of the 5th Language and Technology Conference, 2011.
3. A. Rubinstein, Perfect equilibrium in a bargaining model, Econometrica 50 (1) (1982) 97–109. doi:10.2307/1912531.
4. H. Raiffa, The art and science of negotiation / Howard Raiffa, Belknap Press of Harvard University Press, Cambridge, Mass. :, 1982.
5. I. Stahl, Bargaining theory, Stockholm School of Economics.
6. F. Di Giunta, N. Gatti, Bargaining over multiple issues in finite horizon alternating-offers protocol, Annals of Mathematics and Artificial Intelligence 47 (2006) 251–271.
7. R. Lin, S. Kraus, J. Wilkenfeld, J. Barry, Negotiating with bounded rational agents in environments with incomplete information using an automated agent, Artif. Intell. 172 (6-7) (2008) 823–851.
8. S. D. Ramchurn, C. Sierra, L. Godo, N. R. Jennings, Negotiating using rewards, Artif. Intell. 171 (2007) 805–837.
9. R. Davis, R. G. Smith, Negotiation as a metaphor for distributed problem solving, Artificial intelligence 20 (1) (1983) 63–109.
10. G. Zlotkin, J. S. Rosenschein, Mechanism design for automated negotiation, and its application to task oriented domains, Artif. Intell. 86 (2) (1996) 195–244.
11. D. Sarne, S. Kraus, Solving the auction-based task allocation problem in an open environment, in: Proc. Of AAAI, 2005, pp. 164–169.
12. Y. Chevaleyre, P. E. Dunne, U. Endriss, J. Lang, M. Lemaître, N. Maudet, J. A. Padget, S. Phelps, J. A. Rodríguez-Aguilar, P. Sousa, Issues in multiagent resource allocation, Informatica (Slovenia) 30 (1) (2006) 3–31.
13. S. Airiau, U. Endriss, Multiagent resource allocation with sharable items: Simple protocols and nash equilibria, in: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2010), 2010.
14. T. S. Ferguson, Who Solved the Secretary Problem?, Statistical Science 4 (3) (1989) 282–289.
15. T. Sandholm, A. Gilpin, Sequences of take-it-or-leave-it offers: Near-optimal auctions without full valuation revelation, Autonomous Agents and Multiagent Systems (AAMAS).
16. W. Gaith, R. Schmittberger, B. Schwarze, An experimental analysis of ultimatum bargaining, J. Econ. Behav. Organ. 3 (1982) 367–388.
17. L. A. Cameron, Raising the stakes in the ultimatum game: Experimental evidence from Indonesia, Economic Inquiry 37 (1) (1999) 47–59.
18. R. Katz, S. Kraus, Efficient agents for cliff-edge environments with a large set of decision options, in: AAMAS, 2006, pp. 697–704.
19. U. Gneezy, E. Haruvy, A. Roth, Bargaining under a deadline: evidence from the reverse ultimatum game, Games and Economic Behavior 45 (2003) 347–368.
20. A. W. Woolley, C. F. Chabris, A. Pentland, N. Hashmi, T. W. Malone, Evidence for a collective intelligence factor in the performance of human groups, Science 330 (6004) (2010) 686–688.
21. K. Galuscak, M. Keeney, D. Nicolitsas, F. R. Smets, P. Strzelecki, M. Vodopivec, The determination of wages of newly hired employees - survey evidence on internal versus external factors, Working Paper Series 1153, European Central Bank (2010).

22. G. d. Walque, O. Pierrard, H. Sneessens, R. Wouters, Sequential bargaining in a new keynesian model with frictional unemployment and staggered wage negotiation, Working Paper Research 157, National Bank of Belgium (2009).
23. J. J. Horton, R. J. Zeckhauser, Algorithmic wage negotiations: Applications to paid crowdsourcing, CrowdConf 2010.
24. Amazon, Mechanical Turk services, http://www.mturk.com/ (2010).
25. S. Faridani, B. Hartmann, P. Ipeirotis, What's the right price? pricing tasks for finishing on time, in: AAAI, 2011.
26. J. Riley, R. Zeckhauser, When to haggle, when to hold firm, The Quarterly Journal of Economics 98 (2) (1983) 267–289.
27. A. Azaria, Y. Aumann, S. Kraus, Automated strategies for determining rewards for humanwork, in: AAAI, 2012.
28. H. Overstreet, Influencing human behavior, The People's institute publishing, 1925.
29. D. Carnegie, How To Win Friends And Influence People, Simon and Schuster, 1964.
30. D. Ariely, G. Loewenstein, D. Prelec, Coherent arbitrariness: Stable demand curves without stable preferences, Journal of Business Research 59 (10-11) (2006) 1053–1062.
31. R. M. Karp, Reducibility among combinatorial problems, Springer, 1972.
32. M. L. Weitzman, Optimal search for the best alternative, Econometrica: Journal of the Econometric Society (1979) 641–654.
33. W. Vickrey, Counterspeculation, Auctions, and Competitive Sealed Tenders, Journal of Finance 16 (1) (1961) 8–37.
34. D. Sarne, S. Kraus, Solving the auction-based task allocation problem in an open environment, in: AAAI, 2005, pp. 164–169.
35. G. M. Becker, M. H. DeGroot, J. Marschak, Measuring utility by a single-response sequential method., Behav Sci 9 (3) (1964) 226–232.
36. C. Noussair, S. Robin, B. Ruffieux, Mrevealing consumers' willingness-to-pay: A comparison of the bdm mechanism and the vickrey auction, Journal of economic psychology 25 (2004) 725–741.
37. S. Byers, A. Raftery, Nearest-neighbor clutter removal for estimating features in spatial point processes, Journal of the American Statistical Association 93 (1998) 577–584.
38. C. Fraley, A. E. Raftery, Model-based clustering, discriminant analysis and density estimation, Journal of the American Statistical Association 97 (2002) 611–631.
39. C. Fraley, A. E. Raftery, How many clusters? which clustering method? answers via model-based cluster analysis, The Computer Journal 41 (8).
40. N. Peled, Y. Gal, S. Kraus, A Study of Computational and Human Strategies in Revelation Games, AAMAS-11.
41. Y. Gal, A. Pfeffer, Modeling reciprocity in human bilateral negotiation, in: Proc. of AAAI, 2007.
42. G. Paolacci, J. Chandler, P. G. Ipeirotis, Running experiments on Amazon Mechanical Turk, Judgment and Decision Making 5 (5).
43. H. R. Arkes, C. Blumer, The psychology of sunk cost, Organizational Behavior and Human Decision Processes 35 (1) (1985) 124 – 140.
44. S. J. Kachelmeier, M. Shehata, Examining risk preferences under high monetary incentives: Experimental evidence from the people's republic of china, American Economic Review 82 (5) (1992) 1120–41.