INCREASED EFFICIENCY THROUGH PRICING IN ONLINE LABOR MARKETS

Rica Gonen*
Department of Management and Economics
The Open University of Israel
The Dorothy de Rothschild Campus, University Road, Raanana 4353701
gonenr@openu.ac.il

Daphne Raban
Faculty of Management,
University of Haifa, Israel, 610 Jacobs Building, Haifa, 3498838
draban@univ.haifa.ac.il

Colin Brady
Auctura Ltd,
Shraga Refaeli, Petach Tikva 4906415 Israel
cbrady2002@kellogg.northwestern.edu

Maya Mazor
Faculty of Management,
University of Haifa, Israel, 610 Jacobs Building, Haifa, 3498838
mayamazor@gmail.com

ABSTRACT

We present a matching-pricing algorithm for online information labor markets motivated by exploratory study of Amazon Mechanical Turk (AMT). The algorithm addresses many of the challenges surfaced by the AMT study. We find that AMT pricing is largely done using rules of thumb, which results in inaccurate pricing. Such pricing leads to either excessive costs for employers or low income for workers and likely reduces the rate of tasks completed. We offer an improved mechanism based on a descending pricing function sensitive to the demand for work and the supply of workers. This approach grants employers the flexibility to design tasks to optimize cost and quality while empowering workers to select tasks that meet their occupation and income expectations. Additionally our pricing algorithm holds several unique advantages which include: motivating workers to quickly perform tasks, revealing workers' true market demand at corresponding prices, allowing workers to guarantee the assignment of complementary tasks assuring reduced cost performance, and bounding employers' total expense.

Keywords: Crowdsourcing algorithm; Information labor markets; Online labor markets; Dynamic pricing; Amazon mechanical Turk

1. Introduction

The last decade's innovation in electronic commerce exchanges for market goods has also extended to markets for information workers. Some examples include ODesk, Elance, Vworker, Freelancer, Guru and Amazon Mechanical Turk, which are also known as crowdsourcing services. Crowdsourcing.org reported that in 2011, revenue for crowdsourcing providers was $376 million, up from $215 million in 2010 and $141 million in 2009. Overall, they found that there were approximately 6.3 million workers in the crowdsourcing market, suggesting this is a robust and growing market. In this paper we use the term "information labor market" interchangeably with the term "crowdsourcing" to refer to online exchanges for work and workers. Our research shows that by using a matching and pricing algorithm with increased efficiency, the crowdsourcing market would become more attractive to workers and employers alike.

Common to these sites is the focus on matching employers (buyers) with information workers (sellers) with the issue of labor pricing left to the parties to resolve using a reverse auction or by displaying fixed prices. Usually the employer states a payment offer and the prospective employee decides whether to agree. In the case of Scriptlance

* Corresponding author

(recently acquired by Freelancer), a form of reverse auction is employed but it does not offer an optimization between quality, price, and timing as we offer in the current paper. Our exploratory study of the Amazon Mechanical Turk (AMT) labor market, presented in subsection 1.3, reveals that AMT is mainly employed by companies using arbitrary pricing rules. Online information labor markets and in particular Amazon Mechanical Turk, hold promise for quickly turning around work and therefore would benefit substantially if payment negotiation could be automated to become as reliable and efficient as possible.

Motivated by the behavior and type of employers we discovered in the Amazon Mechanical Turk online information labor market, we offer a matching and pricing algorithm for online information labor markets. Our algorithm holds several advantages for reducing negotiating time and increasing economic efficiency. The algorithm is based on a descending pricing function sensitive to the demand for work and the supply of workers, giving employers the flexibility to design tasks in a way that optimizes cost and quality while empowering workers to select the tasks that fill their occupation and income expectations and guarantee the matching of complementary tasks' as bundles of work. We theoretically prove the economic and algorithmic properties of our matching-pricing algorithm and then evaluate the algorithm in a variety of experiments on synthetic data.

In the following we introduce the notion of crowdsourcing with a focus on the unique issues of pricing information work, leading to the rationale of dynamic pricing as the solution for improving market clearance. Findings from an exploratory study of the Amazon Mechanical Turk information labor market are presented as evidence for the current market deficiencies that our algorithm resolves.

1.1 Background

Crowdsourcing is a popular term that describes a special case of labor markets in which companies employ self-selected individuals who are not their employees [Howe 2006]. Typical crowdsourcing assignments include problem-solving, product/process innovation, and repetitive or routine tasks which require human intelligence. One crowdsourcing model is competition where a task is performed by many but a prize is offered to one person who performed best (e.g. Innocentive). In another model, the work needed is divided into many tasks so that each task can be accomplished quickly and easily. Piecing the tasks back together generates the entire work needed. When the whole work is done online and the tasks are information-related, such as translating, proofreading, or writing code, we refer to the corresponding market as an information labor market and this is the focus of the algorithm we present.

One of the challenges of information labor markets is the pricing of the tasks. For example, a study of the Amazon Mechanical Turk market (the market is described in detail in subsection 1.3) found that raising the reward for each task increased the quantity of individual tasks accomplished but not the quality and accuracy of the work performed. The implication is that paying high rewards results in faster, but not better performance [Mason and Watts 2010]. Another study of the Amazon Mechanical Turk market found that if two similar tasks are given to the same worker, increasing the reward for the second task leads to higher effort and quality while decreasing the reward for the second task results in lower effort and quality [Yin, Chen, Sun 2013]. A possible implication might be that allowing workers to perform a bundle of tasks with similar nature (as we suggest in our matching-pricing algorithm) and increased rewards can lead to better performance. While the online environment is uniquely suitable for automating the pricing negotiation method, this option is rarely used in crowdsourcing. The present research offers a solution that is more efficient for workers and employers alike.

According to Faridani et al. [2011] increasing the reward for a task decreases the demand for the task. High reward tasks are a signal for complex and involved tasks that decrease the utility of high rewards for the worker. Improper pricing or scheduling often results in task starvation and loss of capital for these markets. For example, it is believed that workers have an expected hourly wage in mind and they tend not to accept underpriced tasks that need more time per unit reward than what they have in mind. Overpriced tasks are also undesirable since requesters may invest excess capital qualifying a deluge of data from eager respondents. Part of the pricing issue stems from the nature of information. Pricing information-related tasks is complex because information is an experience good, the value of which is revealed only after use. Since pricing is usually done before the information work commences, value and pricing is based on intuition or subjective perception [Raban and Rafaeli 2006; Raban 2007]. The next section unpacks information pricing and its relation to crowdsourcing in information labor markets.

1.2 Information Pricing

Pricing information goods and services is crucial and significant in an information economy; yet, the unusual properties of information create challenges for pricing information products and services [Raban 2007]. In a broad review of information pricing strategies Chang and Yuan show that pricing more often favors sellers than buyers [Chang and Yuan 2007].

An innovation in static information pricing procedures is *reverse pricing*, where the buyer initiates the price. Such demand-oriented pricing with a (covert) set minimum price is practiced by eBay, with its "Best Offer"

function. Here the buyer may submit up to three price offers below the publicly set price. If one of his offers is above the minimum determined by the seller, which is invisible to buyers, he may win the item [Linde and Stock 2011]. A newer variant of determining prices from the buyers' side is the (open) Pay-What-You-Want approach. Here the seller foregoes a minimum price and accepts the buyer's price offer without reservation [Kim, Natter, Spann 2009]. Such offers have become more frequent lately, e.g. from music groups such as Radiohead and Nine Inch Nails, who make their music available for download for a certain period of time, merely providing the option of a voluntary payment [Linde and Stock 2011].

Since the value of information changes with time and among people, dynamic pricing procedures may be more suitable for information-related exchanges, such as information labor markets. Individual price negotiation has been used for a long time in traditional markets, however, it is not very practical in business-to-consumer markets and especially when the work to be performed is information-based and divided into multiple tasks and sub-tasks as in crowdsourcing.

This paper offers an algorithm for dynamic online pricing for the benefit of buyers and sellers of information work alike. The crowdsourcing algorithm is based on a descending pricing function that reflects the market supply and demand for specific tasks. Initially, all tasks in a certain work assignment are equally priced. When the market opens and workers start expressing their preferences, dynamic pricing takes place such that prices for tasks that are in high demand will decrease more rapidly than prices for low-demand tasks. This scheme motivates workers to act quickly in performing tasks in order to enjoy the higher prices in the early stages of trade. Employers will enjoy fast response to the work they require and in that process they will learn about the employees' labor market behavior for future reference. When an employer has many open tasks the demand for workers is high, justifying a high initial price. As tasks are filled, the employer is able to reduce task price bids because the demand for workers had diminished. The crowdsourcing algorithm allows the workers to express demand for bundles of tasks guaranteeing the workers receive a bundle of tasks of their choice to perform as part of the same job. Workers can thus bid more efficiently plan their workload and their expected income. The algorithm is flexible for the employer and the employees and holds several important advantages including:

- A full display of open tasks and their prices enables workers to select the tasks that are most suitable for them. Workers can thus plan their workload and their expected income. They can also choose tasks based on their similarity in order to increase their own efficiency and improve their income.
- Task selection by workers informs the employers about the true market demand for information work and the corresponding pricing.
- The employer has flexibility to decide on the division of work into tasks, on the quality level, and on the multiplicity of task fulfillment for each task separately.
- A quality threshold ensures that workers perform above a minimum quality level.
- Assessing the performance of the tasks in the work assignment is assigned as a task in itself to aid the employer in selecting the best information work output from multiple identical tasks.
- The pricing scheme bounds the expense for the employer (proven in the theoretical analysis), which is beneficial for budget management and reduces total payment (shown by the experiments).
- While discriminatory pricing may attract criticism, in our algorithm pricing is based on employee self-selection. We expect that when prices go below a person's minimum expected wage, the person will not select a task.
- The algorithm is uniquely tailored to information labor markets by enabling a multiplicity of identical tasks and selection of the most suitable resolved task for inclusion in the final product of work while optimizing market prices.

Before proceeding to the formal algorithm, we present findings from our exploratory study of the Amazon Mechanical Turk information labor market as evidence for the current market deficiencies that the algorithm presented in section 2 resolves. More importantly, the exploration of AMT informs that it is a labor market where the buyers are mainly companies rather than individuals. Applying the crowdsourcing algorithm to companies offering multiple tasks can make a significant difference in terms of financial control of projects.

1.3 Amazon Mechanical Turk

Probably the best-known implementation of crowdsourcing is the Amazon Mechanical Turk (AMT)[1]. The Amazon Mechanical Turk is a popular crowdsourcing information labor marketplace, established by Amazon in 2005. The original Mechanical Turk was a fake chess-playing machine invented in the 18th century. The secret of the Turk machine was a human chess master, hidden inside the machine.

[1] https://www.mturk.com/mturk/welcome

The idea of the AMT system is that there are many information-related tasks that people can do more effectively than computers. Moreover, it can be extremely difficult (if not impossible) for computers to perform certain tasks such as rewriting translated sentences, transcribing, or tagging. The AMT service gives businesses or individuals, called *requesters*, access to a diverse group of workers and gives workers a selection of thousands of micro-tasks that require human intelligence, called HITs (Human Intelligence Tasks).

The AMT marketplace boasts more than 400,000 workers in over 100 countries [Ross and others 2010]. This enables requesters to have work done 24 hours a day, 7 days a week, and to get work done by a more diverse group of workers than they could by staffing workers locally.

Workers are sometimes referred to as "Turkers". Requesters determine the amount that they will pay for each completed HIT, called a *reward*. The rewards can be low as $0.01 per HIT. Each HIT is completed by multiple Turkers, which allows the Requesters to choose the higher quality HITs produced. The Requester may require that all workers meet a particular set of qualifications or a minimum percentage of previously accepted tasks. In addition, the requester may reject submitted work.

AMT provides two methods for paying workers: fixed rates called rewards on each HIT, and bonuses to individual workers for especially good work. Sometimes the bonus amount is explicitly mentioned in advance and sometimes it is not. This lack of clarity on the use of rewards and bonuses hints that labor pricing in AMT is inefficient: employers are unsure about market prices and the quality of work that will be provided so they may separate payment into two parts: reward and bonus. This separation is likely to create uncertainty for workers and possibly diminish their motivation to work for unclear wages. Two stages of payment make budgeting less predictable for employers. The drawbacks of two-stage payment systems highlight the advantage of seeking a one-stage payment model as our algorithm describes.

We used a web crawler written in Python to gather all available information from AMT and store that information in PostgreSQL during March 17, 2011 - May 18, 2011. We collected 47,967 HIT groups from 3,079 Requesters with a total $47,549 reward (without bonuses). All available HITs were crawled and stored with the following fields: groupid, requester, title, description, keywords, rewards, number of HITs available within the HIT group, qualifications required and time of expiration. Amazon Mechanical Turk groups similar HITs together based on their common properties, such as Title, Description, Reward, Keywords and Time.

At the AMT site, the bonus does not appear as a separate field, but as a part of a HIT title or description. Moreover, the bonus presentation is not uniform: verbal, numerical, mixed. In order to examine the relation between the reward and bonus variables we isolated the bonus. This was done by filtering the data, categorization and pattern identification. In some cases bonus identification and separation were done manually.

1.3.1 AMT Findings

Data analysis shows that a surprisingly high rate, 42%, of total collected HIT groups, includes a bonus offer, mostly practiced by companies. The use of bonuses hints that the employers are seeking a way to reduce financial risk and increase quality by splitting the payment into two parts: reward and bonus. We discovered that there are three major commercial requesters who regularly use the bonus mechanism as part of the total payment: CastingWords, SpeechInk and QuestionSwami. Table 1 shows the frequencies of HIT groups by Requesters and bonus offer.

Table 1: Number of HIT groups in the study period, Amazon Mechanical Turk

	HIT Groups	CastingWords	SpeechInk	QuestionSwami	Other Requestors
With Bonus	20,327	17,114	2,423	478	312
No Bonus	27,640	4,614	6,970	91	15,965
Total	47,967	21,728	9,393	569	16,277

The total dollar value of HITs is presented in Table 2. By analyzing the collected data we can learn about risk preferences of the major requestors. For example, CastingWords uses the ratio 1:2 between reward and bonus, SpeechInk uses 2:1, as presented in Table 2. We assume that this is an attempt to reduce financial risk and at the same time to raise the quality of work using contingent incentives. Yet, the practices revealed here indicate employers seem to rely on rules of thumb. There is no "golden rule" for balancing payment level and work quality as well as for finding the true market price emanating from supply and demand of tasks and workers respectively.

Next, we present the online labor market algorithm that offers high efficiency pricing agreements between employers and workers to promote market clearing eventually leading to a higher rate of market use and satisfaction.

Table 2: Total value of HIT groups (U.S. $) in the study period, Amazon Mechanical Turk

	HITs with Bonus			HITs without Bonus
	Reward ($)	Bonus ($)	Total ($)	Reward ($)
QuestionSwami	2,001	18,309	20,311	12
SpeechInk	2,349	4,699	7,049	6,776
CastingWords	36,185	18,092	54,278	14,738
Other Requestors	7,296	1,536	8,833	121,511
Total	47,834	42,638	90,472	143,037

2. The Matching and Pricing Algorithm

2.1 General Description

In this section we present a general matching and pricing algorithm for the online information labor market problem. In our model there are m nonidentical tasks that are offered simultaneously by the same employer to a potential set of workers N. Workers arrive one-by-one in an online fashion and may or may not be matched to tasks. In order to insure the quality of work that the employer will receive, in our model each task $j \in \{1,\ldots,m\}$, is composed of k_j identical copies of the task such that each task can be performed by multiple workers (i.e., k_j workers). Workers have costs for bundles of tasks, i.e., workers have a minimum amount of money they are expecting to be paid for performing a bundle of tasks (bundle of tasks – meaning a collection of different tasks performed by the worker together. For formal definition see next subsection). A *bundle* of tasks is a binary vector $(d^1 \ldots d^m)$, where $d^j \in \{0,1\}$ indicates whether task j is in the bundle or not. Meaning $d^j = 1$ if it is and 0 otherwise. Specifically, each worker i has a privately known *cost* function c_i that assigns a non-negative value for each *bundle* of tasks, $c_i : \{0,1\} \times \ldots \times \{0,1\} \to R^+$.

As in reality, not all workers perform their tasks in a satisfactory manner from the employer point of view. Each worker in our model has a quality score associated with him. We denote a worker's quality score by $q_i \in R^+$. q_i is assumed to be a known public value and as such is also known to the employer. Throughout this section we assume that there are n workers i, $i \in N$, that their quality score is higher than some quality score threshold H, i.e., that their quality score is high enough to perform the tasks. The goal of the matching and pricing algorithm is to minimize the total qualified workers' expected payment of the employer: find a match of workers with high quality score and tasks that partitions the available tasks into bundles $S_1 \ldots S_n$ and that minimizes $\sum_i c_i(S_i)$.

The algorithm we present is a two-stage mechanism. The first stage matches the qualified workers to sets of tasks while minimizing the sum of workers' expected payment. The second stage matches another set of qualified workers to the tasks of ranking the first stage task performance. The algorithm allows general setting such as workers with different costs for different bundles of tasks. It also runs in polynomial time even though finding the optimal match in such general setting is NP-hard. The approximation ratio of this algorithm is nearly as good as can be achieved in polynomial time. All the above properties are achieved while maintaining the employer's satisfaction of the work quality.

2.2 Definitions

Instead of discussing a matching mechanism where each task j is made up of k_j copies, it will be simpler to normalize by k_j, and assume that each task is composed of 1 copy, but the copy is divisible. (Our formalism will allow expressing the underlying indivisibility of each original copy.) Thus our problem is as follows: There are m tasks, each composed of 1 divisible copy. A bundle of tasks is a vector $d = (d^1 \ldots d^m)$, where $d^j \in \{0,1/k_j\}$, meaning a bundle of tasks is a collection of different tasks' fractions. A cost is a function $c : d \to R^+$ such that c is monotone in each coordinate and $c(\vec{0}) = 0$ (normalization), meaning that the cost of bundle (0,0,…0)=0 for every worker and the cost of bundle that performs task 2 and 5 cannot be higher than the

cost of bundle that performs task 2, 5 and 8 for a particular worker i.

Our algorithm uses an abstract black-box representation of costs. The black box can answer specified queries about the cost function, and the mechanism can only ask these queries. The idea of the black-box representation is to present the worker with task payment prices and allow the worker to find the bundle of tasks that maximizes the worker's return for his work, i.e., to maximize his utility.

Definition 1 *(performance black-box) A performance black-box for cost* c *accepts as input a vector of tasks payments* $(p^1 \ldots p^m)$ *and outputs the offered performance for the tasks at these price payments, i.e. it outputs the vector* $d = (d^1 \ldots d^m)$ *that maximizes a worker utility* $< d, p > -c(d) = \sum_j d^j p^j - c(d^1 \ldots d^m)$.

In a concrete algorithmic implementation, the costs will be given in some "bidding language", and our algorithms will work efficiently (in polynomial time) as long as the bidding language allows efficient computation of answers to performance black-box queries.

A match between tasks and workers is an allocation of bundles of different tasks to workers such that every task is not allocated to more workers than the number of copies the employer designated for that task.

Definition 2 *(match) A match is a collection of* n *non-negative vectors* $d_1 \ldots d_n$, *where* d_i^j *specifies whether task* j *was assigned to worker* i, *i.e.,* $1/k_j$ *of the task was assigned to him. A match is feasible if for all* j, $\sum_i d_i^j \leq 1$.

A cost of a match is the sum of all the workers' costs for the bundle allocated to them by the match.

Definition 3 *(cost of a match) The cost of a match A is* $C(A) = \sum_i c_i(d_i)$.

An H-quality match is a match where all workers in the allocation have a quality score that is higher than H.

Definition 4 *(H-quality match) A match is a quality match if each worker* i *in the match has a quality score* q_i *that is higher than H.*

An H-optimal match is the lowest cost match where all workers in the allocation have a quality score that is higher than H.

Definition 5 *(H-optimal match) A match is optimal if it achieves the minimal cost of any feasible match that is an H-quality match.*

Definition 6 *(Incentive compatibility) An information labor mechanism is incentive compatible if for every worker* i, *every cost* c_i, *all declarations of the other workers* c_{-i}, *and any cost* c_i' *such that* d_i *and* p_i *are the mechanism's bundle and payment output for* i *with input* (c_i, c_{-i}) *and* d_i' *and* p_i' *are the mechanism's bundle and payment output for* i *with input* (c_i', c_{-i}), *then it must hold that:*

$$p_i' - c_i'(d_i') - (p_i - c_i'(d_i)) \geq p_i' - c_i(d_i') - (p_i - c_i(d_i))$$

2.3 The Algorithm

The algorithm uses two concepts to fulfill the mechanism's incentive compatibility: non-anonymous prices and the performance black box. When designing an incentive compatible non-anonymous pricing mechanism one must consider how to differ payments given to different workers for the same task. A natural approach is to establish that payment change with time or workers availability. Moreover, to preserve incentive compatibility it is clear that the price paid for a worker's tasks cannot depend on the worker's declared cost itself. In our algorithm the payments given to a worker for tasks are constructed according to the actions of the workers that arrived before him. The

payments only decrease as we are interested in designing an incentive compatible mechanism and the possibility of payments increasing will lead workers to hedge and delay their participation in the market, hoping for a higher payment in the future. More specifically, payments decrease according to worker willingness to perform, meaning that each time a task is matched to a worker, the payment for it decreases for the next worker.

Given the algorithm will create non-anonymous payments by decreasing the price according to willingness to perform; one must also consider how payments will decrease and how to set the initial payment. These design parameters impact the quality of the solution to the cost minimization problem, as is shown at the end of this section. The basic problem with setting the initial payment is conceptualized as follows. If the initial payment offered is too high the algorithm might initially match many workers with high costs and will not be able to match workers with low costs who arrive later on. If the initial payment offered is too low, the algorithm might not match many workers throughout the system and would end up matching only a few of the workers, thereby not completing the tasks. The same ideas apply to the payment decrease rule. If the algorithm decreases the payment too slowly one might expect many workers with high costs to arrive early on, and the algorithm will not be able to match the low cost workers when they arrive later on. If the algorithm decreases the payment offered rapidly with every match, the algorithm might not match many arriving workers and would end up matching only a few workers, thereby not completing the tasks.

The logic of the algorithm's first step is simple: at any point in time the performance of every task j is offered a payment of p^j. The prices p^j are initialized identically and a quality score threshold is set. The workers arrive one after the other. When a worker i arrives if his quality score is above the employer's quality threshold, he is informed of his set of offered payment prices and the algorithm queries his performance black-box (aka PBB_i) about his preferred bundle of tasks. The worker's paid price for the bundle of tasks matched to him is the sum of the offered payment prices of the bundle of tasks. The algorithm decreases offered payments p^j whenever copies of that task are matched. The decrease in payment price is exponential with a rate g per copy matched.

Notice that the match and the payment to each worker is determined as he arrives while the initial payment price, the price decrease rate and the quality score threshold parameters must be chosen in advance, meaning that the parameter choice needs to be made before any workers arrive.

While determining quality score threshold depends on the type of tasks at hand and the identity of the employer, determining price parameters requires some knowledge on the future costs. Therefore we make the following assumption: There exists a priori known bounds $c_{\max}$ and $c_{\min}$ such that: $c_{\max} \geq \min_i c_i((1/k_1,...,1/k_m))$ and $c_{\min} \leq \min_{i,d_i^j} (c_i((d_i^1,...,d_i^{j-1},1/k_j,d_i^{j+1},...,d_i^m)) - c_i((d_i^1,...,d_i^{j-1},0,d_i^{j+1},...,d_i^m)))$ Where $q_i \geq H$.

That is $c_{\max}$ is larger or equal to the minimum of workers' cost for performing the bundle that includes all the tasks.

$c_{\min}$ is less or equal to the minimum marginal cost in performing any task by any of the workers.

The logic of the algorithm's second step is simple: run the first step a second time such that every task j becomes the task of ranking the quality of executing task j in the first run. Naturally, from incentive compatible considerations, the set of workers participating in the second run, i.e. the algorithm's second step, cannot be the same workers that were chosen to perform the tasks in the first run.

FIRST STEP

Set quality score to be H, $r = \min_j k_j, g = \left(\frac{1}{r} \frac{c_{\max}}{2m \cdot c_{\min}} \right)^{\frac{r}{r-1}}$

For each task j set $p_1^j = c_{\max} / 2m$

For each arriving worker $i \in N$ such that $q_i \geq H$

- Present PBB_i with prices p_i^j for all j, PBB_i outputs $d_i = \left(d_i^1 ... d_i^m \right)$
- Match worker i according to d_i and pay $p_i^{sum} = \sum_j^m d_i^j p_i^j$
- Update for all j $p_{i+1}^j = p_i^j / g^{d_i^j}$

End

SECOND STEP

For each task j

- Task j=Rank j's copies execution performance.

Run first step algorithm with set of workers $\bar{N}$ such that $N \cap \bar{N} = \varnothing$

2.4 Analysis of the Algorithm

The correctness of the algorithm involves three elements: incentive compatibility, validity (i.e. that no task is over-matched, meaning no task's copies are matched to more workers than the number of available copies), and approximation ratio. Lemma 1 proves incentive compatibility and shows that no worker can gain utility by acting according to a different cost function than his true cost function. Lemma 2 proves validity and shows that no task is over-matched, meaning no task's copies are matched to more workers than the number of available copies of the task. Lemma 3 proves the approximation ratio of the algorithm to the target function of minimizing the sum of the costs of matched workers. Lemma 3's proof is achieved by bounding the total expense of the employer for having all of his tasks performed. It shows that by the definition of the performance black box we know that each worker i is paid no less than his cost for the bundle of tasks he is assigned and therefore the total employer expense is an upper bound for the total matched workers' cost. We start with incentive compatibility:

Lemma 1 *The Online Information Labor Market Algorithm is incentive compatible.*

Proof. First note that the algorithm uses two sets of workers, one for running the first step and the other for running the second step. Therefore the workers in the second step are performing the task of ranking the execution of other tasks they were not related to or involved in. As the second step of the algorithm runs the first step of the algorithm it suffices to prove that the first step is indeed incentive compatible.

In order to prove that the first step is incentive compatible we need to show that for every worker i, every cost c_i, all declarations of the other workers c_{-i}, and any cost c_i' such that d_i and p_i are the mechanism's bundle

and payment output for i with input (c_i, c_{-i}) and d_i' and p_i' are the mechanism's bundle and payment output for i with input (c_i', c_{-i}), then it must hold that $p_i' - c_i'(d_i') - (p_i - c_i'(d_i)) \geq p_i' - c_i(d_i') - (p_i - c_i(d_i))$.

- From the first step of the algorithm we know that the items payment prices presented to PBB_i are independent of his costs as the initial price and g are independent of the workers' costs and i's offered payments are determined by the workers preceding him. Therefore every item payment price presented to PBB_i when worker i's cost is c_i' is identical to the item payment price presented to PBB_i when worker i's cost is c_i. On the one hand from the performance black-box definition (definition 1) we know that since worker i is matched bundle of tasks d_i' when his cost is c_i' then $p_i' - c_i'(d_i')$ is the maximum utility worker i can achieve with cost c_i' and the payment prices that are offered to him and therefore $p_i' - c_i'(d_i') \geq p_i - c_i'(d_i)$. On the other hand from the performance black-box definition we know that since worker i is matched bundle of tasks d_i when his cost is c_i then $p_i - c_i(d_i)$ is the maximum utility worker i can achieve with cost c_i and the payment prices that are offered to him and therefore $p_i - c_i(d_i) \geq p_i' - c_i(d_i')$. Thus we conclude $p_i' - c_i'(d_i') - (p_i - c_i'(d_i)) \geq p_i' - c_i(d_i') - (p_i - c_i(d_i))$.

As the algorithm's second step runs the algorithm's first step, it suffices to focus our proofs of validity and approximation ratio on the algorithm's first step. From here on in the analysis we refer to the algorithm's first step as *the algorithm*.

The next step proves the validity of the algorithm, i.e. that it never matches more than the available copies of each task. A simple way to verify that the mechanism does not match more than the number of copies available of each task is to ensure that the offered price for each of the tasks reaches some minimum cost. This cost is set such that no worker can achieve a positive marginal utility by being matched to a copy of this task at this minimum low offered price.

Let p_*^j denote the price of task j after k_j copies of the task were matched to workers.

Lemma 2 *For every task j, if k_j copies of task j were matched to workers then $p_*^j < c_{\min}$. Meaning no task j will be matched to more than k_j workers.*

Proof. Assume to the contrary that $k_j + 1$ copies were matched of some task j. Let i be the worker that was matched with the $k_j + 1$ copy of task j. Then it follows that:

$$p_i^j = \frac{\frac{c_{\max}}{2m}}{g^{\frac{k_j}{k_j}}} < \frac{\frac{c_{\max}}{2m}}{g^{\frac{k_j - 1}{k_j}}} < \frac{\frac{c_{\max}}{2m}}{g^{\frac{r-1}{r}}} = \frac{\frac{c_{\max}}{2m}}{\frac{1}{r}\frac{c_{\max}}{2mc_{\min}}} = \frac{c_{\min}}{\frac{1}{r}}$$

Meaning that $\frac{1}{r} p_i^j < c_{\min}$. Thus as worker i demand for task j is at most $d_i^j \leq \frac{1}{r}$, the payment to worker i for task j is less than his marginal cost for it. This contradicts the definition of his performance black-box PBB_i, as PBB_i will choose not to include task j in the output bundle and by that increase worker i's utility.

The final step is to prove a bound on the approximation quality. Recall that $C(A)$ denote the total sum of

workers' costs for the match of algorithm A. I.e. $C(A) = \sum_i c_i(d_i)$, where $(d_1,...,d_n)$ is the match produced by A. Let $C(ALG)$ denote the total sum of workers' costs for the match produced by our algorithm. Let $C(OPT)$ denote the total sum of costs for the match produced by the H-optimal mechanism meaning an offline mechanism that sees all workers and all costs and finds the H-quality match of workers to tasks that gives the lowest sum of costs. The goal is to prove that $C(OPT) \geq w \cdot C(ALG)$ where w is the approximation ratio.

The w approximation ratio is shown by proving the following lemma. Note that the match produced by our algorithm is also an H-quality match.

Lemma 3 $C(ALG) \cdot 2r\left(1-\frac{1}{g^{\frac{1}{r}}}\right) \leq C(OPT).$

Proof. By PBB_i definition, we know that each worker i is paid no less than his cost for the bundle of tasks he is assigned to and therefore when worker i is matched we have

$$c_i(d_i) \leq \sum_j d_i^j \frac{c_{\max}}{2mg^x}.$$

where x is the accumulative ratio of task j's copies that were matched before worker i arrived. Summing the last inequality for all workers we conclude that the total payment, i.e., the total employer expense is an upper bound for the total cost,

$$C(ALG) = \sum_i c_i(d_i) \leq \sum_i \sum_j d_i^j \frac{c_{\max}}{2mg^x} = \sum_j \sum_i d_i^j \frac{c_{\max}}{2mg^x}$$

Thus if $E^j = \sum_i d_i^j \frac{c_{\max}}{2mg^x}$ is the total expense for performing task j then $C(ALG) \leq \sum_j E^j$. We denote by $\Delta_i E^j$ the expense for performing task j by worker i, meaning, $\Delta_i E^j = d_i^j \frac{c_{\max}}{2mg^x}$, then $E^j = \sum_i \Delta_i E^j$. As we will look at the expense made by the employer for a single worker on some task j, denote $\Delta E^j = \Delta_i E^j = y\frac{c_{\max}}{2mg^x}$, where $y = d_i^j$.

The expense is achieved by exponentially decreasing payment prices with a rate g per copy of task that is matched. In order to give a refined bound on the expense we will first compute the payment price decline of a continuous match of a single worker for some task j and then compute the maximum ratio between the algorithm expense with the discrete copy match and the expense of such continuous match.

We denote by $\Delta\tilde{E}^j$ worker i's continuous payment price decline along match y for some task j. $\Delta\tilde{E}^j$ can be computed as the integral of $\frac{1}{g^t}$ in the segment x and $x+y$ multiplied by the integration constant $c_{\max}/2m$,

$$\frac{\Delta\tilde{E}^j 2m}{c_{\max}} = \int_x^{x+y} \frac{1}{g^t} dt = -\frac{1}{g^t} \cdot \frac{1}{\ln g} \Big|_x^{x+y} = \frac{1}{\ln g}\left(-\frac{1}{g^{x+y}} + \frac{1}{g^x}\right) = \frac{1}{\ln g \cdot g^x}\left(1-\frac{1}{g^y}\right)$$

$\frac{\Delta\tilde{E}^j}{\Delta E^j} = \frac{\frac{1}{\ln g}\cdot\frac{1}{g^x}\left(1-\frac{1}{g^y}\right)}{\frac{y}{g^x}}$ is the ratio between $\Delta\tilde{E}^j$ and ΔE^j and is maximized when y is maximized.

Since the largest demand for performing of any task is $1/r$ the ratio between $\Delta\tilde{E}^j$ and ΔE^j can be bounded:

$$\max_{y\le\frac{1}{r}}\left\{\frac{\Delta\tilde{E}^j}{\Delta E^j}\right\} = \max_{y\le\frac{1}{r}}\left\{\frac{1}{\ln g}\cdot\frac{\left(1-\frac{1}{g^y}\right)}{y}\right\} = \frac{1}{\ln g}\cdot\frac{1-\frac{1}{g^{\frac{1}{r}}}}{\frac{1}{r}} = \frac{r}{\ln g}\cdot\left(1-\frac{1}{g^{\frac{1}{r}}}\right). \quad (1)$$

Denote $\Delta\tilde{E}^j = \Delta_i\tilde{E}^j$ and recall that $\Delta E^j = \Delta_i E^j$ then $\sum_i \Delta_i\tilde{E}^j = \tilde{E}^j$ is the total expense spent on task j in a continuous match of the whole copy of j. From equation (1) we concluded that

$$E^j = \sum_i \Delta_i E^j \le \frac{\frac{\ln g}{r}}{1-\frac{1}{g^{\frac{1}{r}}}}\sum_i \Delta_i\tilde{E}^j = \frac{\frac{\ln g}{r}}{1-\frac{1}{g^{\frac{1}{r}}}}\tilde{E}^j.$$

It follows that

$$E^j \le \frac{\frac{\ln g}{r}}{1-\frac{1}{g^{\frac{1}{r}}}}\int_0^1 \frac{c_{\max}}{2mg^x}dx = \frac{\frac{\ln g}{r}}{1-\frac{1}{g^{\frac{1}{r}}}}\cdot\left(-\frac{c_{\max}}{2mg^x}\cdot\frac{1}{\ln g}\right)\Big|_0^1$$

$$= \frac{\ln g}{r\ln g\left(1-\frac{1}{g^{\frac{1}{r}}}\right)}\left[\frac{c_{\max}}{2mg^0}-\frac{c_{\max}}{2mg^1}\right] = \frac{\left(\frac{c_{\max}}{2m}-p_*^j\right)}{r\left(1-\frac{1}{g^{\frac{1}{r}}}\right)}.$$

since $\tilde{E}^j = \int_0^1 \frac{c_{\max}}{2mg^x}dx$.

By summing the above on all m tasks, we achieve the following bound:

$$C(ALG) \le \sum_j E^j \le \frac{\frac{c_{\max}}{2}-\sum_j p_*^j}{r\left(1-\frac{1}{g^{\frac{1}{r}}}\right)} \quad (2)$$

from (2) we conclude

$$C(ALG)r\left(1-\frac{1}{g^{\frac{1}{r}}}\right)+\sum_j p_*^j \le \frac{c_{\max}}{2}.$$

As one should expect the H-optimal mechanism to match at least the worker with the minimum of workes' cost for performing the bundle that includes all the tasks (and with quality score greater than H), we can conclude that $c_{\max} \le C(OPT)$ holds.

Therefore

$$C(ALG)2r\left(1-\frac{1}{g^{\frac{1}{r}}}\right) \le C(OPT).$$

and Lemma 3 is concluded.

Theorem 1 *There exists an Online Information Labor Market Algorithm that is computationally efficient, incentive compatible, and achieves an* $O\left(r\left[1-\left(\frac{rmc_{\min}}{c_{\max}}\right)^{\frac{1}{r-1}}\right]\right)$ *approximation to the optimal match.*

Proof. Replacing $g=\left(\frac{1}{r}\frac{c_{\max}}{2m\cdot c_{\min}}\right)^{\frac{r}{r-1}}$ in Lemma 3's approximation ratio, one can see that:

$$2r\left(1-\frac{1}{g^{\frac{1}{r}}}\right)=2r\left(1-\frac{1}{\left(\left(\frac{1}{r}\frac{c_{\max}}{2m\cdot c_{\min}}\right)^{\frac{r}{r-1}}\right)^{\frac{1}{r}}}\right)$$

$$=2r\left[1-\left(\frac{r2mc_{\min}}{c_{\max}}\right)^{\frac{1}{r-1}}\right]=O\left(r\left[1-\left(\frac{rmc_{\min}}{c_{\max}}\right)^{\frac{1}{r-1}}\right]\right) \quad (3)$$

Combining (3) with Lemma 1, we conclude Theorem 1.

It is worth noting that our approximation bound becomes logarithmic for $r=O(\log n)$ and it is close to the best possible in polynomial time as the online information labor market problem is not only NP-hard but also hard to approximate.

3. Algorithmic Crowdsourcing Literature Review

The problem of designing mechanism for procurement auctions has been studied by the algorithmic community over the past decade. The earlier work by [Archer and Tardos 2002] focused on minimizing payments for complex objective functions and is not directly applicable to our setting. Recently other algorithms for online labor markets were suggested. One can categorize the suggested algorithms as addressing four main aspects of the relevant task assignment and pricing. The first category consists of task assignment algorithms for learning workers' performance (quality scores) and ignores economic aspects of pricing. The second category includes pricing strategies based on

current behavior (scraped data) and regression models rather than on economic auction theory. This line also ignores workers' performance aspects (quality score). The third category suggests using prediction markets pricing schemes, which are exponential time to compute. The fourth category, which is the closest in spirit to our work, employs algorithms that incentivize workers' truthful cooperation with the market by constructing pricing approximation mechanisms. This fourth category ignores aspects of workers' performance (quality score) and offers the somewhat unrealistic model of algorithms for the assignment of a single task, specific details follow.

First Category - task assignment for learning workers' performance:

Ho and Wortman Vaughan [2012] and Ho, Jabbari and Wortman Vaughan [2013] focused their work on a learning crowdsourcing algorithm. Both papers' algorithms learn the quality score of workers based on machine learning algorithms - Ho and Wortman Vaughan [2012] based on a bandit exploration/exploitation algorithm and Ho, Jabbari and Wortman Vaughan [2013] based on repeated labeling algorithm. Similar to our setting, in their settings m nonidentical tasks are offered simultaneously by the same employer to a potential set of n online arriving workers. Each task is composed of multiple identical copies of the task such that each task can be performed by multiple workers. Unlike our setting, their settings assume that each worker has a different quality score for each task and that workers have no cost for their work (i.e. workers accept any work assign to them at any price). In addition, Ho, Jabbari and Wortman Vaughan [2013] also assume that the tasks are only of the specific type of binary classification problem while our tasks can take any general form.

Our algorithm differs from both of the above algorithms in number of ways. First, they do not assume that the employer has an a priori knowledge of each worker's quality score but rather focus their attention on the employer assigning the tasks such that the quality score is revealed through the assignment algorithm. Second, they do not have prices for the tasks or costs for the workers to perform the tasks and therefore assume that workers must accept whatever assignment they receive, i.e., every arriving worker is assigned a task. Third, Ho and Wortman Vaughan [2012] assign each worker a single task, meaning that bundling of tasks is not possible. In Ho, Jabbari and Wortman Vaughan [2013] multiple tasks are assigned to a single worker in stages, i.e., a worker is assigned other tasks after observing his performance on earlier tasks. Fourth, while our algorithm's performance goal is to minimize the sum of the assigned workers' cost while offering task prices that reveal workers' true demand, Ho and Wortman Vaughan's [2012] goal is to maximize the sum of the workers' quality scores on the tasks they were assigned. Ho, Jabbari and Wortman Vaughan's [2013] goal is to maximize the sum of the workers' quality scores on the assigned tasks while minimizing the number of copies of each task performed and learning the correct answer to their binary tasks.

Second Category - pricing strategies based on current behavior:

Faridani, Hartmann, and Ipeirotis [2011] approached the algorithmic problem of crowdsourcing pricing by going from practice to theory. They collected data on Amazon Mechanical Turk and used survival analysis to find the minimum reward for a task that will lead to the task completion on the desired time. Unlike our bundling model and as their model based on data collected on Amazon Mechanical Turk, Faridani, Hartmann, and Ipeirotis [2011] assumes that each worker can accept only one task at a time. Unlike our heterogeneous quality score and since performance of tasks is hard to observe while collecting data on task allocations, Faridani, Hartmann, and Ipeirotis [2011] assumes that all workers have identical quality score and that their quality score is high enough to perform any task offered to them. While both algorithms, Faridani, Hartmann, and Ipeirotis [2011] and ours, computes dynamic prices for the different tasks, the two pricing approaches result in different properties of the labor market. On the one hand, our algorithm outputs prices that are only decreasing, allowing for the revelation of the true workers' tasks demand and preventing workers manipulations such as hedging. On the other hand, Faridani, Hartmann, and Ipeirotis' [2011] algorithm outputs prices that can alternately increase or decrease over time. The price fluctuations as suggested in the above work can lead to workers manipulating the market. For instance, workers may wait for the price to increase before they obligate to complete a task and therefore the algorithm cannot guarantee revelation of the workers' true demand for tasks and thereby prevent worker manipulations as ours can. A detailed analysis of the effect of time on bidding reveals that sniping is indeed a known practice in online auctions [Kuruzovich 2012].

Third Category – pricing tasks with existing prediction markets' pricing scheme

Ipeirotis and Horton [2011] suggested using the pricing scheme implemented for some of the recent experimental prediction markets. Unfortunately the suggested pricing scheme introduced combinatorial prices which are NP-hard to compute (take exponential time to compute in practice) or limit the bundle of tasks that can be matched to the workers. Our pricing algorithm allows a polynomial time computation of the task-bundle's prices

while also allowing complete flexibility in choosing the worker's preferred bundle of tasks[2].

Fourth Category - pricing approximation mechanisms for a single task assignment:

Singer and Mittal [2011] designed dynamic pricing algorithm for crowdsourcing mechanisms based on Singer [2010] reverse auction. In their model the employer has a single task and wishes to buy the performance of this task from n online arriving workers. Unlike Singer and Mittal's [2011] model, our model allows for an unlimited number of tasks rather than a single task. Contrary to our model Singer and Mittal [2011] assumes that workers have no quality score, meaning that all workers can perform the given task. In Singer and Mittal [2011] and our model, each worker has a cost for performing a copy of the task and he may not report his cost truthfully. Unlike our model, the employer has an a priori known budget for paying the workers and a priori known values for the performance of different numbers of duplicate tasks. In our model, the employer predetermines the number of copies of a given task to be performed, allowing simpler communicational input to exist. (An employer's value function can take exponential time to communicate to the mechanism). Singer and Mittal's [2011] mechanism's goal is to maximize the number of task copies performed while not paying the workers more than the known budget. The approximation to Singer [2010] and Singer and Mittal's [2011] mechanism's goal was improved for special cases in Bei et al [2012] and Dobzinski et al. [2011]. Unlike the aforementioned goal our mechanism's goal is to minimize the sum of the allocated workers' costs while performing the predetermined (multiple) tasks and the tasks' copies. Our minimization of the sum of costs leads to a bounded minimum expense for the employer as opposed to a pre-known maximum expense (the budget) assumed by Singer and Mittal [2011].

4. Synthetic Experiments

In this section we evaluate the performance of our matching and pricing algorithm through simulations on synthetically generated data. We decided to simulate the lower end of the tasks available on Amazon Mechanical Turk as the lower price range represents a disproportionate majority of the tasks available on Amazon Mechanical Turk. Tasks in this group also tend to be more uniform and in many cases more suitable for automation than tasks in the middle and upper price ranges. In particular these tasks are typically involve one of copying simple information from a web page, tagging a short document, deciphering a captcha-like inscription, identifying symbols in a picture, and answering a survey.

As a comparison with our algorithm, we run four other information labor market mechanisms on the same data. The four other information labor market mechanisms we run on the synthetically generated data are: fixed price, optimal offline algorithm, optimal offline incentive compatible algorithm, and our pricing algorithm where no bundling allowed. Our matching pricing algorithm performed quite well in comparison to various information labor market mechanisms. We show that:

1. The matching pricing algorithm benefits both the workers and the employer. On the one hand, early responding workers are offered a higher price for tasks than in the fixed-price algorithm and on the other hand, the employer pays a much lower total amount in the matching pricing algorithm than in the fixed-price algorithm, saving significant amounts of money for performing the very same tasks.
2. Conclusion 1 still holds even if we do not allow workers to bundle with our matching pricing algorithm.
3. Bundling is beneficial in multiple ways - workers' utility is higher when bundling is allowed than when it is not allowed. The algorithm with bundling is more efficient than otherwise and the larger the number of tasks the cheaper it is for the employer to allow bundling.
4. Our online matching pricing algorithm is almost as efficient as possible for an incentive-compatible information labor market.

In our simulation we created n=500 workers, each with different capabilities for performing three job types such as copying simple information from a web page, tagging a short document, identifying symbols in a picture, etc. The job types have m=40 to m=940 nonidentical tasks associated with them, creating 120 to 27,120 total of nonidentical tasks. Each nonidentical task can be performed up to 10 times by 10 different workers in order to assure top performance. We create each worker with a cost function such that nonidentical tasks of the same job type result in complementarities and allow a power function learning curve. I.e., if a worker tags 80 different photos tagging the 81st photo should cost much less than tagging the first one. In a given run, a worker's cost for performing a single task ranges between \$0.02 and \$0.06 on the lower end. The simulation sets the upper end to 10 times the lower end so the upper end cost averages \$0.40. To create the graphs we simulated each data point 5000 times and plotted the average values.

To test the performance of our matching pricing algorithm we first compared it against a fixed price algorithm.

[2]See definition 1.

The fixed price algorithm simulates the Amazon Mechanical Turk's pricing behavior. In the fixed price algorithm a single price is offered to the workers, and if their cost is below that price they accept and otherwise reject. We picked $0.0825 as the fixed price for the simulation, which was the average price, offered on Amazon Mechanical Turk for 1,000 low price, high volume, highly automatable, low skill tasks we sampled. As can be seen in Figure 1, we initialized the matching pricing algorithm's price ($0.13) higher than the fixed price mechanism's price. Nevertheless, as can be seen in Figure 2 the employer's total payment in the fixed price mechanism is always higher than total payment in our matching pricing algorithm. Moreover, for 10,000 tasks and up the total payment gap grows to be at least twice as much.

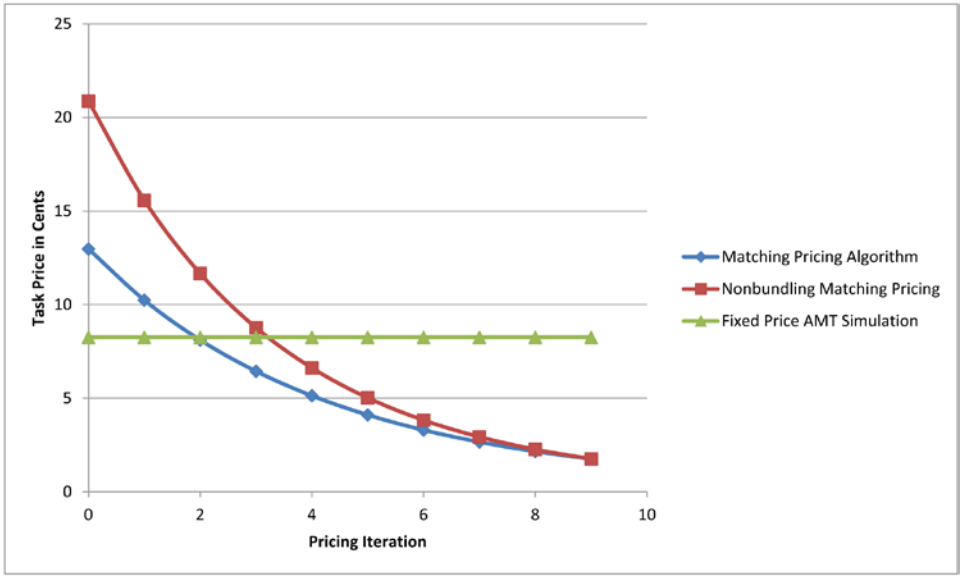

Figure 1. A comparison of prices over time in our matching pricing algorithm with and without bundling capabilities and prices in an AMT fixed price simulation.

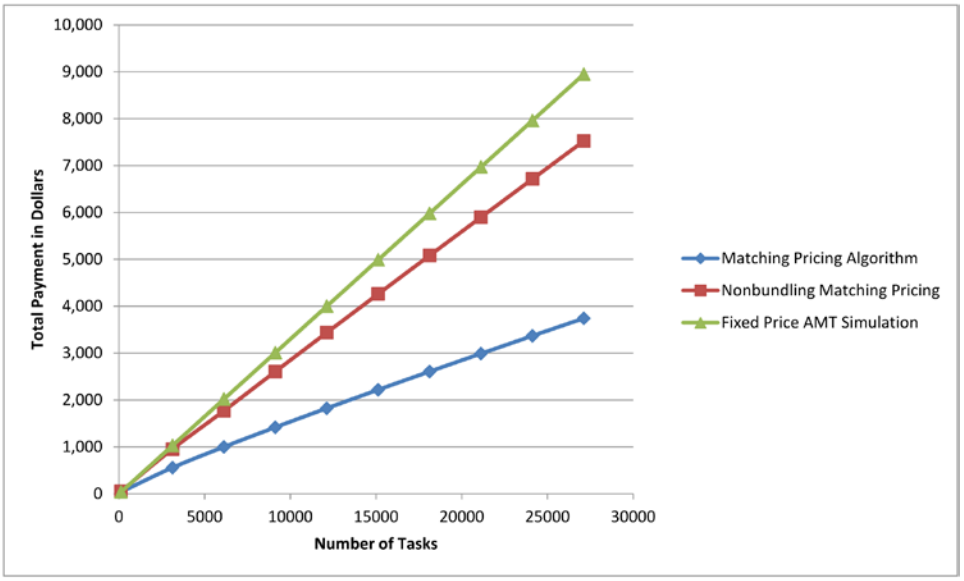

Figure 2. A comparison of total payments made by the employer in our matching pricing algorithm with total payments made by the employer in a matching pricing algorithm with no bundling capabilities and in an AMT fixed price simulation.

The conclusion that we draw from Figure 1 and Figure 2 is that the matching pricing algorithm is benefiting both the workers and the employer. On the one hand the workers are offered a higher price for the tasks up front and on the other hand the employer ends up saving significant amounts of money for performing the very same tasks.

We continue by testing our matching pricing algorithm performance when bundling is not allowed. To simulate an algorithm that does not allow bundling we ran our matching pricing algorithm but allowed every worker that is matched only a single task matching and not a bundle of tasks. As can be seen in Figure 1 we initialized the non-bundling matching pricing algorithm's price ($0.21) much higher than the fixed price mechanism's price. Nevertheless as can be seen in Figure 2 the employer's total payment in the fixed price mechanism is still higher than total payment in the non-bundling matching pricing algorithm starting from 5000 tasks and up.

The conclusion that we draw from Figure 1 and Figure 2 is that even if our pricing scheme is used without the bundling capabilities still our algorithm is benefiting both the workers and the employer. On the one hand, the workers are offered a higher price for the tasks up front and on the other hand the employer ends up saving significant amounts of money for performing the very same tasks.

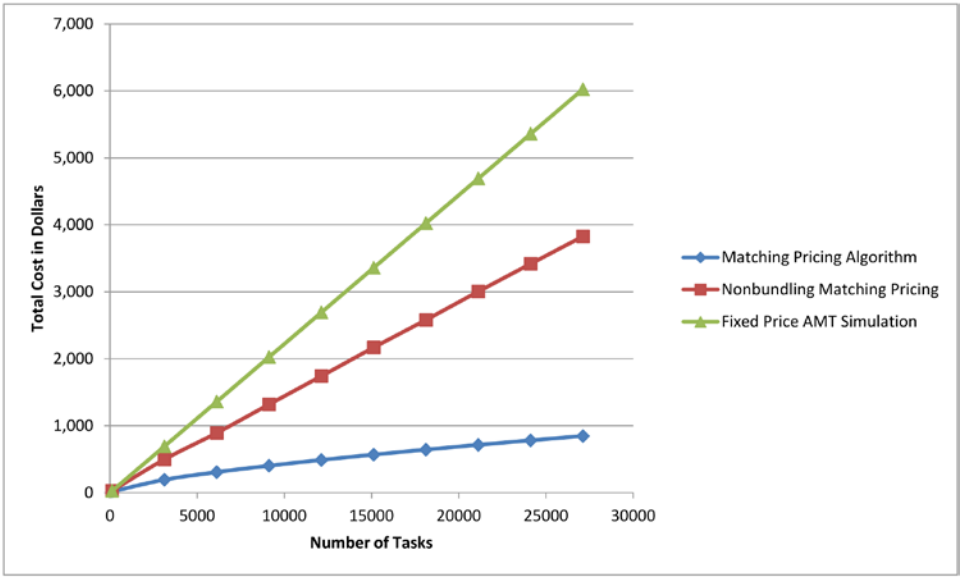

Figure 3. A comparison of total workers' costs incurred in our matching pricing algorithm with total workers' costs in a matching pricing algorithm with no bundling capabilities and total workers' costs in an AMT fixed price simulation.

We further conclude from Figure 3 that the matching pricing algorithm is much more efficient than the non-bundling matching pricing algorithm as it matches workers with lower costs than the non-bundling matching pricing algorithm does. Moreover from Figure 2 together with Figure 3 we conclude that the total utility of the workers matched by our matching pricing algorithm is higher than the total utility of the workers matched by the non-bundling matching pricing algorithm as the gap in costs between the algorithms is larger than the gap in payments. Finally we conclude from Figure 2 that the larger the number of tasks is the cheaper it is for the employer to allow bundling.

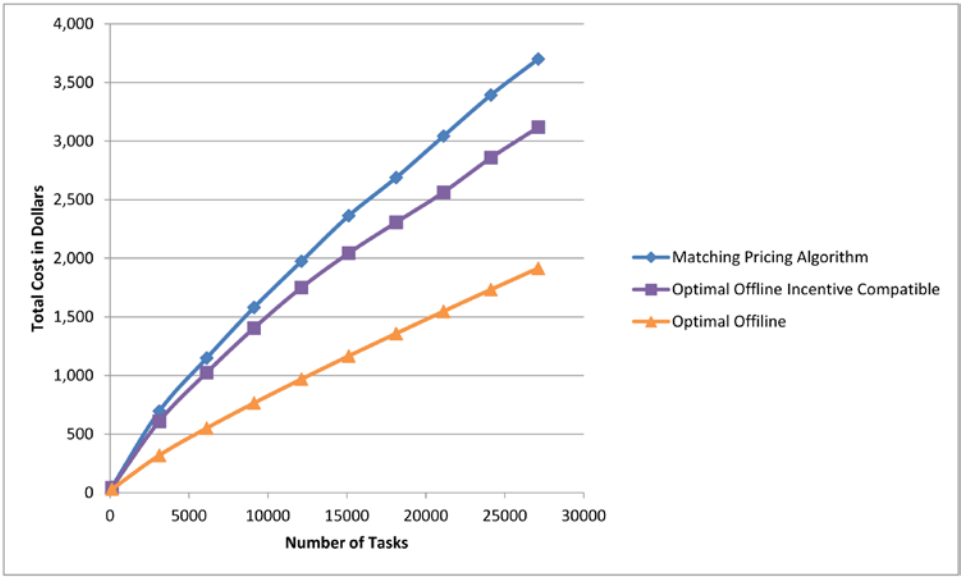

Figure 4. A comparison of total workers' costs incurred in our matching pricing algorithm with total workers' costs in an optimal offline algorithm and total workers' costs in an optimal offline incentive compatible algorithm.

We conclude by testing our matching pricing algorithm's performance against several theoretical benchmarks. The first benchmark is the optimal offline algorithm, which has full knowledge of workers' costs. This benchmark was used to analytically compare our matching pricing algorithm's performance in section 2. The second benchmark is the optimal offline incentive compatible mechanism. This mechanism does not have knowledge of workers' costs and is an offline mechanism, i.e., all workers submit their bids to the mechanism and wait for the mechanism to collect all the bids before deciding on an allocation. In both benchmarks all costs are known a priori and will therefore always outperform our mechanism. Nevertheless from Figure 4 we conclude that our online incentive compatible matching pricing algorithm, where the workers' costs are not a priori known, is almost as efficient as the offline incentive compatible mechanisms and therefore is almost as efficient as possible for an incentive-compatible information labor market. From Figure 4 one can also see that at the point where the gap between the optimal offline mechanism and the matching pricing algorithm is the largest, the 27120 task point, although our matching pricing algorithm is only guaranteed to be within a competitive factor of 2.65 of the optimal offline solution, the simulations show that this ratio is as small as 1.96.

5. Conclusions and Future Work

Current online labor markets offer no solutions for the need to price work quickly, take advantage of complementary tasks, and perform according to real-time supply and demand. The algorithm presented in this work fills this gap by offering an automated pricing scheme that allows for task bundling and reacts to changes in supply and demand. The algorithm's analysis shows that under the suggested pricing scheme, market clearing is highly efficient and the employer's expense is bounded. The experiments validating the algorithm show that pricing schemes that do not react to supply and demand such as a fixed price are not as efficient and result in significantly higher expense for the employer and lower utility for the faster performing workers. Moreover, even a pricing scheme that reacts to changes in supply and demand but does not allow task bundling results in higher expenses for the employer and lower utility for the workers.

Our observation regarding the current state of online labor markets is motivated by our study of the Amazon Mechanical Turk online information labor market. Our study highlights several issues related to AMT's operation. First, AMT's employers are predominantly companies rather than individuals, which supports the notion that companies are adopting crowdsourcing as a form of employment. Second, companies usually split the payment between upfront reward and post factum bonus. This method demonstrates the companies' need for flexibility in pricing and risk reduction. Last, the data does not support a clear pricing pattern. Motivated by these observations we set out to develop the crowdsourcing algorithm with fully transparent task and pricing information in order to encourage workers to start working, provided they meet the quality threshold. The algorithm adjusts task pricing based on ongoing supply of tasks and demand for work. This approach is uniquely suited for the network environment as it can deal with a plurality of concurrent sellers and buyers with multiple tasks to fulfill. The market

host, the intermediary, can decide whether to charge membership, value-added, or usage fees.

Having developed an auction application which is founded on theory, both the theory and the application itself may be extended to explore additional influences on pricing. First, it would be interesting to implement the algorithm in an existing online labor market such as AMT and observe whether known behavioral biases come into play and to what extent [Kuruzovich 2012]. Following the work of Hou [2007], a future study could investigate cultural influences or incorporate cultural artifacts such as photos or emoticons. Finally, the algorithm can be extended by incorporating more elaborate quality-control mechanisms to monitor the actual quality of work submitted or by incorporating additional pricing functions.

Acknowledgement

This work is partly supported by the EC funded project SocIoS (Project number: 257774. Call Identifier FP7 ICT 2009.1.2: Internet of Services, Software and Virtualisation). For more information about the project and its partners see http://www.sociosproject.eu.

REFERENCES

Archer A. and Tardos É. 2002. Frugal path mechanisms. Proceedings of the thirteenth annual ACM-SIAM symposium on discrete algorithmsSociety for Industrial and Applied Mathematics. 991 p.

Bei Xiaohui, Chen Ning, Gravin Nick and Lu Pinyan. 2012. Budget feasible mechanism design: From prior-free to bayesian. Proceedings of the 44th symposium on theory of computingNew York. ACM. 449 p.

Chang WL and Yuan ST. 2007. An overview of information goods pricing. International Journal of Electronic Business 5(3):294-314.

Dobzinski Shahar, Papadimitriou Christos H. and Singer Yaron. 2011. Mechanisms for complement-free procurement. Proceedings of the 12th ACM conference on electronic commerceSan Jose. ACM. 273 p.

Faridani S., Hartmann B. and Ipeirotis P. G. 2011. What's the right price? pricing tasks for finishing on time. Proceedings of HCOMP11: The 3rd workshop on human computationSan Francisco. AAAI. 26 p.

Ho Chien-Ju and Vaughan Jennifer Wortman. 2012. Online task assignment in crowdsourcing markets. Proceedings of the twenty-sixth AAAI conference on artificial intelligenceToronto. . 45 p.

Ho Chien-ju, Jabbari Shahin and Vaughan Jennifer W. 2013. Adaptive task assignment for crowdsourced classification. Proceedings of the 30th international conference on machine learning (ICML-13)Atlanta. . 534 p.

Hou J. 2007. Price determinants in online auctions: A comparative study of eBay china and US. Journal of Electronic Commerce Research 8(3):172-83.

Howe J. 2006. The rise of crowdsourcing. Wired Magazine 14(6):1-4.

Ipeirotis P. G. and Horton J. J. 2011. The need for standardization in crowdsourcing. Proceedings of the workshop on crowdsourcing and human computation at CHI, 2011Vancouver. .

Kim JY, Natter M, Spann M. 2009. Pay what you want: A new participative pricing mechanism. J Market 73(1):44-58.

Kuruzovich J. 2012. Time and online auctions. Journal of Electronic Commerce Research 13(1):23-32.

Linde F and Stock WG. 2011. Information markets: A strategic guideline for the I-commerce. Berlin ; New York, NY: De Gruyter Saur.

Mason W and Watts DJ. 2010. Financial incentives and the performance of crowds. ACM SIGKDD Explorations Newsletter 11(2):100-8.

Raban DR. 2007. User-centered evaluation of information: A research challenge. Internet Research 17(3):306-22.

Raban DR and Rafaeli S. 2006. The effect of source nature and status on the subjective value of information. Journal of the American Society for Information Science and Technology 57(3):321-9.

Ross J., Irani L., Silberman M., Zaldivar A. and Tomlinson B. 2010. Who are the crowdworkers?: Shifting demographics in mechanical turk. Proceedings of the 28th of the international conference on human factors in computing systemsAtlanta. ACM. 2863 p.

Singer Y. and Mittal M. 2011. Pricing tasks in online labor markets. Proceedings of HCOMP11: The 3rd workshop on human computationSan Francisco. AAAI. 55 p.

Singer Yaron. 2010. Budget feasible mechanisms. 51st annual IEEE symposium on Foundations of computer science (FOCS)Las Vegas. IEEE. 765 p.

APPENDIX A

A.1 Presenting the Algorithm as a Reverse Auction

In subsection 2.3 we presented our algorithm in a posted prices format, meaning that the employer posts offered payments for the tasks and the workers choose whether to perform the tasks for the offered prices or not. In this subsection, we claim that the same algorithm we presented in subsection 2.3 can also be presented in a reverse auction format. Meaning that workers bid on bundles of tasks, the employer chooses the assignment of workers to his tasks, and pays the workers based on the bids he collected. As our model captures the online nature of labor markets and workers arrive one-by-one in a continuous stream, the employer must also make assignment decisions in an online fashion. Our reverse auction assigns workers to tasks before the employer can collect all of the workers' bids. Therefore, the reverse auction in this case cannot be designed like classic reverse auctions where all the bids are collected prior to any assignment and payment. In order to design a reverse auction in an online setting the employer must base his new assignment decision on the previous assignment decisions he made. Thus at every decision point the employer computes a threshold payment that helps him decide whether to accept a bid from a worker or not.

The algorithm's first step in the reverse auction format is as follows:

At any point in time, the employer sets a privately known payment threshold p^j for the performance of every task j. The thresholds p^j are initialized identically and a quality score threshold is set. The workers arrive one after the other and bid on bundles of tasks. If an arriving worker i has a quality score above the employer's quality threshold and at least one of his bundle bids is lower than the bundle's sum of thresholds, than the employer assigns a bundle of tasks to that worker. The worker's payment is the bundle's sum of thresholds, and the worker is assigned the bundle that maximizes his utility. The algorithm decreases payment thresholds p^j whenever copies of that task are matched. The decrease in payment threshold is exponential with a rate g per copy matched.

The algorithm's second step in the reverse auction format is identical to the second step of the algorithm presented in subsection 2.3.

The reason we presented the above reverse auction as a posted prices algorithm in subsection 2.3 is that the employer in the reverse auction model has computational difficulty in computing the utility maximizing bundle assignment for every worker. The posted prices algorithm avoids this problem by querying the performance black box.