

# Managing Innovation In A Crowd

Daron Acemoglu\*

Mohamed Mostagir<sup>†</sup>

Asuman Ozdaglar<sup>‡</sup>

Preliminary Version

## Abstract

Crowdsourcing is an emerging technology where innovation and production are sourced out to the public through an open call. At the center of crowdsourcing is a resource allocation problem: there is an abundance of workers but a scarcity of high skills, and an easy task assigned to a skilled worker is a waste of resources. This problem is complicated by the fact that the exact difficulties of innovation tasks may not be known in advance, so tasks that require skilled labor cannot be identified and allocated ahead of time. We show that the solution to this problem takes the form of a skill hierarchy, where tasks are first attempted by low-skilled labor and high-skilled workers only engage with a task if workers with lesser skills are unable to finish it. Organizing these hierarchies in crowdsourcing is difficult because firms have little or no information about the skills of the workers they can hire and the firm-worker relationship is fleeting and temporary, providing an incentive for workers to misrepresent their skills. This complicates the firm's problem: it now wants to find an optimal assignment of workers to tasks even though it knows neither the difficulties of the tasks nor the skills of the workers.

We give a dynamic pricing mechanism for tasks that utilizes the concept of *self-selection*—the idea that workers know what they are capable of and optimize their decisions accordingly. Each time a task is attempted and not finished, its price (reward upon completion) goes up. By correctly setting the prices, the mechanism provides an incentive for workers to sort themselves into an optimal hierarchy, i.e. workers participate in the same level of the hierarchy that would be produced if the firm had knowledge of the workers' skills, ultimately leading to the desired optimal matching between workers and tasks.

---

\*Department of Economics, MIT. email: [daron@mit.edu](mailto:daron@mit.edu)

<sup>†</sup>Laboratory for Information and Decision Systems, MIT. email: [mosta@mit.edu](mailto:mosta@mit.edu)

<sup>‡</sup>Laboratory for Information and Decision Systems, MIT. email: [asuman@mit.edu](mailto:asuman@mit.edu)

# 1 Introduction

A few years ago, Goldcorp, a gold mining company based in Vancouver, terminated its mining operations and was on the brink of bankruptcy as a result of increases in production costs and its inability to reliably estimate the value and location of gold on its property. As a final maneuver, the company published its geological data online and asked people to submit their estimates and estimation methods in exchange for prize money that totalled a little over half a million dollars. The crowd identified 110 targets that led to the extraction of a staggering \$8 million ounces of gold, transforming the \$100 million company into a \$9 billion powerhouse. Today, Goldcorp is the second largest gold company in the world.<sup>1</sup> More recently, the website “fold.it” provided the public with a chance to design and fold proteins, a computational biology task that routinely stumps supercomputers. Within three weeks, the crowd of participants were able to determine the structure of an AIDS-related enzyme, a task that the scientific community has unsuccessfully tackled for over a decade, and that represents a significant step forward in finding a cure for the disease.

These two examples represent a growing number of success stories where innovation is sourced out to the public – an emerging technology known as crowdsourcing, and one that promises to transform our idea of a firm from a highly stringent structure into a more dynamic and malleable construct. Historically, improvements in information and communication technology have always led to major revisions in the nature of the firm,<sup>2</sup> and crowdsourcing is poised to be the natural next step in this chain of evolution. The idea that innovation can happen outside of the confinements of a firm is not new (Von Hippel 1976, Chesbrough, Vanhaverbeke, and West 2008, Von Hippel 2009), but it is only now that improvements in technology have made it possible to facilitate this phenomenon on an unprecedented scale.

Despite its successes, crowdsourcing remains a technology in infancy, using mostly ad hoc methods and mechanisms that may or may not work, and a wealth of challenges in resource allocation, pricing, and incentive design need to be addressed in order to bring it to its full potential. On one

---

<sup>1</sup> “Innovation in the age of mass collaboration”, Bloomberg Businessweek 2007-02-01.

<sup>2</sup> Mail railways, telegrams, and later telephones played a key role in the emergence of the American corporation at the early 20<sup>th</sup> century. See Chandler (1977).

hand, a firm can now have immediate access to an unlimited supply of labor and a wide pool of talent and skills, but extracting the good from the bad and managing this pool of workers is fraught with difficulties. As with most new technologies, a rush towards development may fail to leverage existing knowledge and possibly reinvent inferior alternatives.<sup>3</sup> In this paper, we take a step in the direction of a unified systematic analysis of these systems based on established concepts from labor economics and the theory of the firm, as well as methodologies from dynamic optimization, mechanism design, and optimal control.

The prevalent idea of a crowdsourcing task is what has become known as a Human Intelligence Task, or a HIT. These are simple tasks that can be performed by almost any one (for example, determining whether a photograph contains a certain object or not). Because of their easy nature, these tasks are priced extremely low, paying a fraction of a cent on average. Tasks that require more skill and effort (like for example, transcribing audio) pay more. A common feature of these tasks is that their difficulty can be determined to a reasonable degree of accuracy beforehand, and hence they can be priced accordingly given the set of skills available in the market. In addition, there is an abundance of workers whose skills enable them to successfully complete these tasks, and therefore there is little or no concern about having a short supply of labor. Understanding how to price these kinds of tasks has been the focus of most work in this area (Horton and Chilton 2010, Faridani, Hartmann, and Ipeirotis 2011).

The critical departure in our work from most of the existing literature is that we focus on tasks that require innovation. The primary difference between innovation tasks and HITs is that the difficulty of an innovation task *cannot be determined in advance*, but like HITs, its successful completion can be verified. This holds true for tasks that require a modicum of innovation (for example, a manager may verify that a piece of code performs a desired function without knowing how to write such code himself) to examples like the ones at the beginning of this paper. This presents an immediate problem because while there is an abundance of workers, there is usually a scarcity of high skills, and an easy task assigned to a skilled worker is a waste of resources, since

---

<sup>3</sup>For example, the Generalized Second Price auction used in computational advertising lacks the basic and economically-desirable property of efficiency; something that the VCG mechanism –which has existed for decades– provides. Lack of efficiency introduces randomness into revenues.

that worker can handle a more demanding and difficult job. With no prior knowledge of difficulties, tasks that require skilled workers cannot be identified and allocated ahead of time.

The second and bigger problem is that unlike in a standard firm, crowdsourced workers are completely free to decide whether or not to participate in a task. Normally, one would use prices (the price of a task is the reward upon its successful completion) as an instrument to align the incentives of workers with that of the firm or an employer, so that for example the most skilled workers choose to work on the most difficult tasks because they pay the highest. But pricing tasks correctly requires advanced knowledge of their difficulties, which is not the case in our setting. Tasks cannot be priced based on workers' input because of information asymmetry, since a worker who engages in a task knows more about its difficulty than the employer does. This leads to a host of incentive problems that prevent truthful elicitation of the task's difficulty from workers. Even assuming the existence of probabilistic information about difficulties, a straightforward assignment of potentially difficult tasks to skilled workers cannot avoid scenarios where firms end up paying too much for a task that is not too difficult.

This last point relates our work to the literature on experts (Wolinsky 1993), where the primary difficulty is judging whether or not a service that an expert provided is justified. This difficulty arises from informational asymmetries between the expert and the party that receives and pays for the service. Recent work shows how an expert can reveal information in a way that manipulates the receiver even when all parties involved are rational (Kamenica and Gentzkow 2011). Our problem can be viewed as a problem of matching tasks to experts, where an expert is defined as a worker whose expertise allows him to successfully complete the task. A standard tool that overcomes the problem of information asymmetry in this setting is reputation. Experts develop reputations and are incentivized to provide accurate appraisals of difficulty in order to maintain that reputation. However, recent experimental work suggests that experts have no incentive to build reputations as long as they can avoid liabilities (Dulleck, Kerschbamer, and Sutter 2011). Unfortunately, at this point crowdsourcing platforms lack both a decent reputation tracking system and any meaningful way to enforce liabilities. This is further complicated by the myriad of ways in which reputation can be manipulated in a digital environment. As it stands, reputation of crowdsourced workers

cannot be considered a reliable instrument to utilize in the process of assigning tasks to workers. One advantage of the approach we present in this paper is its lack of reliance on reputation.

Our problem can be modeled as a stochastic matching problem. In this problem, nodes in a graph are connected by edges and these edges have 'success probabilities' attached to them. A matching of two nodes is successful with a probability equal to the probability of the edge that joins these nodes, and success (or failure) is realized upon the actual match. In our setting, one node is a worker and the other is a task, and the probability of successful match indicates how likely it is that the difficulty of the task is within the worker's skill. There has been recent interest in this problem because of its applications to online dating and ad allocation (Feldman, Mehta, Mirrokni, and Muthukrishnan 2009, Bansal, Gupta, Li, Mestre, Nagarajan, and Rudra 2010). The stochastic nature of the problem leads to an exponentially large dynamic programming formulation that is NP-Hard to solve. No constant-factor approximations are known for models similar to the one we consider in this paper (Chen, Immorlica, Karlin, Mahdian, and Rudra 2009). However, because the supply of workers in a crowdsourcing environment is usually very large, we get around the stochasticity problem by considering a continuum of workers, and hence can appeal to the law of large numbers to uncover structures that we expect will be useful in designing better approximations for a variety of problems in the stochastic matching literature.

## 1.1 Contribution

Our first contribution is conceptual. We argue that through designing mechanisms that provide the correct incentives to innovation workers to behave as if they are embedded in a single organization and working towards a common goal, we can leverage the wealth of knowledge accumulated in strategic management and organizational behavior to deal with the uncertainties of the innovation process, whose undesirable effects can be amplified in a crowdsourced environment. Our technical contributions advance this idea through a rigorous analysis of the problem. First, we characterize the first-best matching of tasks to workers by showing that under the assumption of unknown difficulties and assuming no voluntary participation constraints (i.e. workers do not get to choose whether to participate or not), the optimal solution to the problem takes the form of a *skill hierar-*

*chy*: tasks are first attempted by the least-skilled workers and high-skilled workers only engage with a task if workers with lesser skills are unable to finish it. This ensures that the different skills in the workers pool are fully utilized. We show how to organize the optimal hierarchy by introducing the concept of hierarchical matching. In a hierarchical matching, workers are arranged in groups according to their skill and tasks are offered to groups in a sequential fashion. We then derive the structure of the optimal hierarchy through analyzing a dynamic programming formulation. The idea of knowledge or skill hierarchies was developed in the economics literature in a different setting (Garicano 2000), where workers belong to the same firm and can cooperate on solving problems in order to increase output, and workers in different levels of the hierarchy specialize in solving only certain kinds of problems that complement other existing skills.

We then turn our attention to crowdsourced workers, where voluntary participation becomes an issue. In fact, the problem is more complicated than simple voluntary participation because we would like workers to not just participate, but to work on tasks whose difficulties are commensurate with their skills, which in light of the previous discussion is not easy to achieve given the lack of information about task difficulties. To this end, we develop a pricing mechanism that utilizes the concept of *self-selection* –the idea that workers know what they are capable of and optimize their decisions accordingly (Heckman and Taber 2008). In order to implement the optimal hierarchy, the process of matching workers to tasks proceeds in an iterative fashion, with tasks allocated to workers who choose to participate in the current round. Each time a task is attempted without successfully finishing, its price increases. Thus tasks that remain in the system longer have higher prices and higher difficulties. This provides an incentive for higher-skilled workers to delay their participation and wait for the difficult tasks to filter through, and makes lower-skilled workers participate early since if they wait they run the risk of the remaining tasks being too difficult for them to finish. By correctly setting the prices, this easily-implementable compensation scheme provides an incentive for all workers to participate in the same level of the hierarchy that we derive in the first-best setting, i.e. the setting where the employer knows the skills of the workers and can assign workers to groups at will. Furthermore, it provides a price-discovery mechanism that, in light of informational constraints, ensures that no task is priced higher than it should.

Complementary to our work is the literature on innovation tournaments. An innovation tournament encourages competition amongst participants in order to obtain a prize. One famous example of an innovation tournament is The Netflix Challenge, where participants were promised an award of a million dollars if they can beat Netflix’s in-house algorithm for movie recommendations by a certain margin. The structure of optimal tournaments, in terms of number of competitors and amount, timing, and frequency of awarding prizes is still not well-understood. A common method in the literature models an innovation tournament as simple all-pay auctions and utilizes existing results in that area to draw conclusions about tournaments (Moldovanu and Sela 2006, Chawla, Hartline, and Sivan 2012). Our paper takes a different approach by combining elements of both cooperation and competition, since workers compete for tasks but at the same time, under the mechanism described above, act as if they are part of a cooperative hierarchical organization that is working in tandem on a project. It is of interest to note that some of the theoretical predictions of our model agree with empirical analysis. For example, based on a very large data set, Boudreau, Lacetera, and Lakhani (2011) advise that more unrestricted entry into a contest is useful the more difficult (in a probabilistic sense) an innovation task is. We find that, even though in our setting workers attempt tasks sequentially, the same recommendation holds true.

The paper is organized as follows. Section 2 introduces the model and defines the assignment and implementation problems. Section 3 derives the necessary results to analyze the dynamic programming formulation of the assignment problem. Section 4 provides a pricing scheme for the implementation problem, and Section 5 concludes the paper. Proofs are presented in the appendix.

## 2 Model

### 2.1 Setup

There is a set  $\mathcal{W}$  of workers indexed by  $i \in [0, A]$  and a set  $\mathcal{T}$  of tasks indexed by  $j \in [0, 1]$ . Both sets are equipped with a Lebesgue measure  $\lambda$  and we assume that  $\lambda(\mathcal{T})$  is normalized to one and that  $\lambda(\mathcal{W}) \gg \lambda(\mathcal{T})$ , i.e. there are many more workers than tasks. The difficulty  $d$  of a task in a set  $T \subseteq \mathcal{T}$  is a nonnegative scalar distributed according to  $F_{dT}$ . Similarly, the skill  $s$  of a worker in a

set  $W \subseteq \mathcal{W}$  is a nonnegative scalar distributed according to  $F_{s_W}$ . We will assume that the workers in  $\mathcal{W}$  are ordered in ascending order of skill, so that  $s_i \geq s_j$  if  $i > j$ . Define  $G(s) : R \rightarrow [0, \lambda(\mathcal{W})]$  to be the measure of workers whose skill is less than or equal to  $s$  and let  $G^{-1}(w)$  be its inverse, i.e.  $G^{-1}(w)$  is the skill at or below which a measure  $w$  of workers reside.

A matching of tasks  $T$  to workers  $W$  is a measurable map  $m : T \rightarrow W$  satisfying measure consistency: for every open interval  $Z \subseteq T$ ,  $\lambda(Z) = \lambda(m(Z))$ . Furthermore, if a set  $\bar{T} \subseteq T$  is matched to a set  $\bar{W} \subseteq W$  then  $\exists \hat{T} \subset T \setminus \bar{T}$  s.t.  $m(\hat{T}) = \hat{W}$  and  $\hat{W} \cap \bar{W} = \emptyset$ , i.e. no task is assigned to more than one worker and no worker is assigned more than one task. Define the throughput of a pairing of a worker of skill  $s$  with a task of difficulty  $d$  to be equal to 1 if  $s \geq d$  and 0 otherwise. The throughput of a match  $m$ ,  $\theta(m)$  is the integral of the throughput over all task-worker pairs in  $m$ , and is equal to the amount of tasks successfully completed by workers under  $m$ .

Central to our work is the idea that the matching process can be sequential. Roughly, a first match  $m_1$  assigns some tasks to some workers. A second match  $m_2$  assigns some tasks (possibly containing uncompleted tasks from the first match) to another set of workers and so on. To capture this idea formally, we denote by  $W_i$  and  $T_i$  the sets of workers and tasks that constitute the  $i^{\text{th}}$  match. The set of tasks completed in the  $i^{\text{th}}$  match is denoted by  $T_i^c$  and the set of tasks leftover after the  $i^{\text{th}}$  match is denoted by  $T_i'$ .

**Definition 2.1.** *Let  $\mathcal{T}$  be a set of tasks. A sequential matching  $\mu$  of tasks to workers is a series of matchings  $m_i : T_i \rightarrow W_i$  with the following properties:*

1.  $W_i \cap W_j = \emptyset$  for all  $i$  and  $j$ ,
2.  $T_1 \subseteq \mathcal{T}$  and  $T_i \subseteq \mathcal{T} \setminus \bigcup_{j=1}^{i-1} T_j^c$  for  $i = 2, \dots$

The first part of the definition implies that a worker can only be part of at most one matching. The second simply states that the pool of tasks available at any point is equal to the set of tasks we started with minus the tasks that were completed in earlier matchings. Notice that a one-shot matching  $m$  of  $T$  to  $W$  can be written as a degenerate sequential matching  $\mu$  with  $T_1 = T$  and  $W_1 = W$ . As mentioned, we use  $F_{d_{T_i}}$  and  $F_{s_{W_i}}$  to refer to the distributions of difficulties and skills, respectively, in the task and worker groups at stage  $i$ . Throughout the paper we denote a sequential

matching by  $\mu$  and a regular, one-shot matching by  $m$ . The throughput of a sequential match is the total throughput of the individual matches it comprises:  $\theta(\mu) = \sum_i \theta(m_i)$ .

Let us assume that the time a worker spends working on a task until he realizes that he cannot complete it is much smaller compared to the time spent by workers who are actually able to finish their tasks. We can then interpret the first point of Definition 2.1 in light of this assumption. Workers who are working on their tasks are busy with them and cannot participate in other matches, and those who are not equipped to finish the tasks they were assigned seek opportunities elsewhere, especially, as we will show later, that attempting to get another task from the same employer (i.e. participate in another match) will likely result in a more difficult task than the one originally assigned. This implies that it is in the workers' best interest to quickly decide whether to stick with a task or not in order to move on to another opportunity in case the task they currently have is too difficult for them.

We impose an extra condition on the sets of workers in a sequential match to obtain the following definition.

**Definition 2.2.** *Consider a sequential matching and for all  $i$ , let  $\min\{W_i\} = \{\min_j s_j | j \in W_i\}$  and  $\max\{W_i\} = \{\max_j s_j | j \in W_i\}$  be the skills of the least and most skilled workers, respectively, in  $W_i$ . A sequential matching is **hierarchical** if  $\max\{W_i\} \leq \min\{W_l\}$  whenever  $i < l$ .*

A hierarchical matching is a partition of workers into several groups based on skill, such that workers in group  $i$  are less skilled than workers in group  $j > i$ . Tasks are then offered to the least-skilled group, where some tasks may be completed and some not. The remaining tasks are assigned to the next group and the process repeats.

From the discussion above, a sequential matching is likely to take longer to conclude than a one-shot matching. To capture this, we assume that the employer can afford to have at most  $k$  sequential rounds of matchings. The objective then is to find a throughput-maximizing matching  $\mu^*$  that consists of at most  $k$  matchings. Let us denote the number of individual matchings in a

sequential matching  $\mu$  by  $|\mu|$ , then the employer wants to solve

$$\begin{aligned} & \max_{\mu} \theta(\mu) \\ & s.t. |\mu| \leq k \end{aligned}$$

In this paper, we consider two versions of the employer's problem: the assignment problem, and the implementation problem. As discussed earlier, these correspond to finding the first-best assignment and the implementation of that assignment when the decision to participate is endogenous.

## 2.2 The Assignment Problem

In the assignment problem the employer has the liberty to form groups of workers for the purpose of matching them to tasks, and workers do not get to decide which group they are part of. This setting models a firm environment, where a manager freely organizes his workforce to maximize a certain outcome. Different matchings can be realized depending on what information the employer has about the difficulties of the tasks. We can think of the assignment problem as a problem where the employer and workers have the same goal: to finish as many tasks as possible. Workers facilitate this by truthfully sharing information (for example, about their skills) with the employer and by following the assignment that the employer suggests.

## 2.3 The Implementation Problem

Let us now drop the assumption that the employer can assign tasks to whomever he wants. Instead, workers get to decide whether to participate or not. The employer, knowing neither the difficulties of the tasks nor the skills of the workers has to design a mechanism that entices workers to participate and reveal this information in a way that replicates the outcomes of the assignment problem. We will refer to this as *the implementation problem*.

Obviously, to design a mechanism that solves the implementation problem, the employer has to first solve the corresponding assignment problem. The next section discusses the assignment problem when there are no implementation issues.

### 3 The Assignment Problem: Analysis and Optimal Matching

In this section we analyze the structure of the optimal matching. We restrict our decision variable in stage  $i$  to be the choice of  $W_i$ . This group is assigned a measure of tasks *uniformly at random* from the pool of available tasks, i.e. choosing a particular group of tasks to match to workers is not part of the decision process.

The main result of this section is the following theorem.

**Theorem 3.1.** *Consider a set of tasks  $\mathcal{T}$  of unknown difficulties and a set of workers  $\mathcal{W}$  with  $\lambda(\mathcal{W}) \gg \lambda(\mathcal{T})$ . There exists a hierarchical matching  $\mu^H$  of tasks to workers such that  $\theta(\mu^H) \geq \theta(\mu)$  for all  $\mu$ .*

Theorem 3.1 states that an optimal match can be obtained by arranging workers in a hierarchical fashion. The rest of this section provides a series of lemmas that establish the structure of the optimal matching and ultimately lead to the proof of Theorem 3.1. We first illustrate our ideas through a simple example.

**Example 3.2.** Consider a set of tasks  $\mathcal{T}$  with measure normalized to unity and let the distribution of difficulties  $F_{d_{\mathcal{T}}}$  be  $U(0, 1)$ . Let  $\mathcal{W}$  be a set of workers with  $\lambda(\mathcal{W}) = 2$  and the distribution of skills  $F_{s_{\mathcal{W}}}$ , also  $U(0, 1)$ . Assume that the number of desired groups,  $k$ , is equal to 1, then the optimal one-shot matching assigns all tasks to the group of workers whose skills are in  $(0.5, 1)$  (the most skilled group of workers with measure equal to 1). The throughput of this matching is equal to 0.75, since any task with difficulty less than or equal to 0.5 will be finished with probability 1 and a task with difficulty  $x > 0.5$  will be finished with probability  $2(1 - x)$ , leading to a total throughput of

$$\frac{1}{2} + \int_{0.5}^1 2(1 - x)f_{d_{\mathcal{T}}}(x)dx = \frac{1}{2} + \frac{1}{4} = \frac{3}{4}.$$

Assume now that  $k = 2$ , then it can be verified that the optimal grouping is as follows:  $W_1$  contains all workers with skills in  $(0.25, 0.75)$  and  $W_2$  contains all workers whose skills are in  $(0.75, 1)$ . The matching  $m_1 : \mathcal{T} \rightarrow W_1$  has  $\theta(m_1) = 0.5$  (since any task with difficulty less than 0.25 is finished with probability one, any task whose difficulty is over 0.75 is not finished, and any task whose

difficulty is in  $(0.25, 0.75)$  is finished with probability 0.5). The leftover tasks from this matching constitute the group  $\mathcal{T}'$ , which has measure 0.5 and difficulty distribution  $F_{d_{\mathcal{T}'}}$ . This distribution has two types of tasks, those that are in  $(0.5, 0.75)$  and those that are in  $(0.75, 1)$ , with each type having an equal measure of 0.25. Matching  $m_2 : \mathcal{T}' \rightarrow W_2$  assigns the remaining tasks to  $W_2$  and has  $\theta(m_2) = 0.375$ , since all tasks with difficulties less than 0.75 will be finished, and tasks with difficulties in  $(0.75, 1)$  will be finished with probability 0.5. Thus the total throughput of this matching is equal to  $\theta(m_1) + \theta(m_2) = 0.875$ , which is higher than the one shot matching throughput of 0.75.

Example 3.2 shows the idea behind our approach. The matching with two groups was able to improve on the matching with a single group because in the latter, some of the skilled workers were assigned tasks that could have been done by other less-skilled workers, resulting in under-utilization of skill. The two-group solution filters out some of these task early in the process, so that there is a smaller chance that skilled workers in the second group are held back by getting assigned these easier tasks. Ideally, as we proceed through groups, tasks that would be too easy (meaning that they can be done by less-skilled workers) are completed, leaving more difficult tasks to more skilled workers. It is also worth mentioning that the optimal two group solution does not use any worker whose skill is less than 0.25. This is because the assignment process of tasks to workers within a group is random, and so it is possible to decrease a group's output by adding unskilled labor to a group. In the example, the size of the first workers group is equal to the size of the set of tasks, and so adding less skilled workers to the workers group can only increase the probability that some tasks get assigned to these workers instead of being assigned to someone more skilled. Notice also that the optimal two-group solution is contiguous; once we start the first group at a certain skill, every worker whose skill is above that skill participates. As we will show, all of these properties will turn out to be general properties of the optimal grouping.

We now turn our attention to formalizing the above intuition. We will repeatedly refer to a few concepts throughout this section, so we start by giving the proper definitions here. The first definition is of the probability of task completion in a matching.

**Definition 3.3.** (*Probability of successful completion*) Let  $F_{d_T}$  and  $F_{s_W}$  be the distributions of

difficulties and skills in a group of tasks  $T$  and a group of workers  $W$ , respectively, and consider matching  $m : T \rightarrow W$ . The probability that a task is successfully completed in  $m$  is given by  $\pi(m)$  and can be written as

$$\pi(m) = \int_{d_1}^{d_2} (1 - F_{s_W}(\delta)) f_{d_T}(\delta) d\delta,$$

where  $[d_1, d_2]$  is the domain of difficulties in  $T$  and  $f_{d_T}$  is the density of difficulties.

The probability of successful completion is simply the probability that a worker in  $W$  is matched with a task in  $T$  such that the difficulty of the task is within the worker's skill. A one-shot matching  $m$  of tasks to workers results in a partition of tasks into three regions, defined as follows.

**Definition 3.4.** (*Task regions*) Consider matching  $m : T \rightarrow W$  and assume for simplicity that  $\lambda(W) = \lambda(T)$ . Let  $s_l = \min\{s_i | i \in W\}$  and  $s_h = \max\{s_i | i \in W\}$ . We can divide  $T$  into three regions:

- R1* : These are the tasks that can be done by any worker in  $W$ . The probability that a task is in *R1* is equal to  $F_{d_T}(s_l)$ , and this region has measure  $\lambda(T)F_{d_T}(s_l)$ .
- R2* : These are the tasks that cannot be performed by any worker in  $W$ . The probability that a task is in *R2* is equal to  $1 - F_{d_T}(s_h)$ , and this region has measure  $\lambda(T)(1 - F_{d_T}(s_h))$ .
- R3* : These are the tasks that can be done by some workers in  $W$ , and their difficulties lie in  $(s_l, s_h)$ . The probability that a task is in this region is equal to  $F_{d_T}(s_h) - F_{d_T}(s_l)$ , and this region has measure  $\lambda(T)(F_{d_T}(s_h) - F_{d_T}(s_l))$ .

For two random variables  $A$  and  $B$ , we write  $A \succeq B$  to indicate that  $A$  first-order stochastically dominates  $B$ , i.e.  $A \succeq B$  implies that  $F_A(x) \leq F_B(x)$  for all  $x$ , with strict inequality for some  $x$ .

In order to arrive at the characterization in the beginning of this section, we need to understand how the throughput of a matching depends on the composition of workers and tasks that this matching comprises. One would expect that the same group of workers will perform better and have higher throughput when given an easier (in a stochastic dominance sense) set of tasks. The following lemma formalizes this intuition by examining how throughput is affected when we keep the group of workers constant but change the distribution of the difficulty of the tasks.

**Lemma 3.5.** *Consider a group of workers  $W$  and matchings  $m_1 : T_1 \rightarrow W$  and  $m_2 : T_2 \rightarrow W$ . Let  $\lambda(T_1) = \lambda(T_2)$  and assume  $d_{T_1} \succeq d_{T_2}$ , then  $\theta(m_1) \leq \theta(m_2)$ .*

Lemma 3.5 states that the throughput of the same group of workers cannot improve when tasks become more difficult.

We also define a notion of dominance for worker groups. Recall that the set of workers is arranged in ascending order of skill.

**Definition 3.6.** *(Group dominance) A group of workers  $W_1$  dominates group  $W_2$  if  $\min\{s_i : i \in W_1\} > \max\{s_j : j \in W_2\}$ . We write this as  $W_1 \succ W_2$ .*

The next lemma complements Lemma 3.5 by showing that a group's throughput on the same tasks can only increase if some workers in the group are replaced by an equal number of more skilled workers.

**Lemma 3.7.** *Consider a group of workers  $W$  and matchings  $m : T \rightarrow W$  and  $m' : T \rightarrow W'$ , where  $W' = (W \setminus w) \cup w'$  where  $w \subset W$ ,  $w' \cap W = \emptyset$ ,  $w' \succ w$ , and  $\lambda(w) = \lambda(w')$ , then  $\theta(m) \leq \theta(m')$ .*

When tasks go through a matching, the overall distribution of difficulties in the remaining pool of tasks changes. The following lemma relates the pre- and post- matching difficulty distributions and is central to the proof of Theorem 3.1.

**Lemma 3.8.** *(Stochastic Dominance) Let  $m : T \rightarrow W$  be a matching and  $T^c$  the set of tasks completed in  $m$ . Let  $T' = T \setminus T^c$  and assume that  $T' \neq \emptyset$ . Define  $w = \{i | i \in W \text{ and } \exists t \in T \text{ s.t. } j \in t \text{ and } d_j > s_i\}$ , then  $w = \emptyset$  implies  $F_{d_{T'}} = F_{d_T}$ . Otherwise,  $d_{T'} \succeq d_T$ .*

Lemma 3.8 implies that the distribution of difficulty in  $T'$  is skewed more towards difficult tasks than it is in  $T$ , and therefore the chances of selecting a difficult task from  $T'$  is higher than it is in  $T$ .

The following lemma derives a relationship between group dominance, group ordering, and throughput.

**Lemma 3.9.** *(Swapping Lemma) Let  $T$  be a group of tasks and consider two groups of workers  $W_1$  and  $W_2$  with  $\lambda(W_1) = \lambda(W_2) \leq \frac{\lambda(T)}{2}$  and such that  $W_2 \succ W_1$ . Denote by  $T'_i$  the group*

of tasks remaining from matching  $T$  to  $W_i$ . Let  $\mu^A = (m_1 : T \rightarrow W_1, m_2 : T'_1 \rightarrow W_2)$  and  $\mu^B = (m_3 : T \rightarrow W_2, m_4 : T'_2 \rightarrow W_1)$ , then  $\theta(\mu^B) \leq \theta(\mu^A)$ .

Lemma 3.9 says that for two worker groups with one group dominating the other, throughput increases if we arrange them such that the dominated group comes first. The reasoning for this is that since task assignment is random, when the dominating group comes first it finishes some tasks that the dominated group could have finished, and this comes at the expense of finishing less tasks that the dominating group can finish but the dominated group cannot. On the other hand, having the dominated group come first increases the proportion of tasks that the dominating group can finish but the dominated group cannot, and hence both groups are effectively utilized.

Note that while the combined throughput of both groups in Lemma 3.9 increases, the throughput of the dominant group can only decrease, since by Lemma 3.8, it now faces a more difficult distribution. The throughput of the dominant group may decrease further if the skills in the dominated group increase. This is because more tasks that the dominant group can finish are completed before they reach the group. This is formalized in the following lemma. It is worth stressing again that while the throughput of the dominant group may decrease, the combined throughput can only increase by Lemma 3.9.

**Lemma 3.10.** *Let  $T$  be a group of tasks and consider groups  $W_3 \succ W_1 \succ W_2$  with  $\lambda(W_1) = \lambda(W_2)$  and matchings  $\mu^A = (m_1 : T \rightarrow W_1, m_2 : T'_1 \rightarrow W_3)$  and  $\mu^B = (m_3 : T \rightarrow W_2, m_4 : T'_2 \rightarrow W_3)$ , then  $\theta(m_4) \geq \theta(m_2)$ .*

The following Lemma collects some simple but useful results about the matching throughputs for various scenarios.

**Lemma 3.11.** *Consider matchings  $m : T \rightarrow W$ ,  $m' : T \rightarrow W'$ , and  $m'' : T' \rightarrow W$ . Let  $w$  and  $t$  be a group of workers and a group of tasks such that  $W \cap w = \emptyset$ ,  $t \subset T$ ,  $W' = W \cup w$ , and  $T' = T \setminus t$ , then*

- a. *If  $\lambda(T) = \lambda(W)$  and  $w$  is such that  $\max_{i \in w} s_i < \min_{j \in W} s_j$ , then  $\theta(m) \geq \theta(m')$ .*
- b. *If  $\lambda(T) > \lambda(W)$  and  $w$  is such that  $\lambda(w) = \lambda(T) - \lambda(W)$ , then  $\theta(m) \leq \theta(m')$ .*

c. If  $\lambda(T) \leq \lambda(W)$ , then  $\theta(m'') = \theta(m) - \theta(t \rightarrow W)$ .

Lemma 3.11.a states that if the number of workers in a group is equal to the number of tasks given to that group, then there is no reason to add any more workers whose skills are less than the skills of everyone else in the group. The reason is that this will cause a redistribution of tasks in a way that results in some tasks going to less-skilled workers instead of the workers they were originally assigned to, while the remaining tasks have their original assignment, and hence the throughput can only go down. A simple corollary of this claim is that no optimal assignment will assign a group of tasks to a group of workers such that the number of workers is more than the number of tasks.

**Corollary 3.12.** *In an optimal solution, no tasks  $T$  are assigned to a group  $W$  such that  $\lambda(W) > \lambda(T)$ .*

In contrast to Lemma 3.11.a, Lemma 3.11.b says that extending the size of a group of workers to make it equal to the number of assigned tasks in a matching cannot reduce output. This is because adding more workers in this scenario does not affect the distribution of tasks in the original smaller-sized group. Instead, the effect is that tasks that were not originally assigned due to a mismatch in group size will now have extra workers to attempt them.

Finally, Lemma 3.11.c states that when the number of workers is at least equal to the number of tasks, removing some tasks from a matching does not affect the throughput of the group of workers on the remaining tasks.

The previous lemmas will allow us to identify the solution to the dynamic programming formulation of the problem, which we provide in the next section.

### 3.1 Dynamic Programming Formulation

To start our analysis of the optimal matching, we give a dynamic programming formulation for the problem. The state and action space, along with the transition function, can be given as:

- **State** The state at stage  $\tau$  is written as  $(\mathbb{W}_\tau, T_\tau)$ , where  $\mathbb{W}_\tau$  is the set of remaining workers and  $T_\tau$  are the tasks remaining at the beginning of  $\tau$ .

- **Action** At stage  $\tau$ , the decision or action is what group  $W_\tau \subseteq \mathbb{W}_\tau$  to assign tasks to.
- **Transition** The transition function is derived from the outcome of matching  $m_\tau : T_\tau \rightarrow W_\tau$ . The new state is given by  $(\mathbb{W}_{\tau+1}, T_{\tau+1})$ , where  $\mathbb{W}_{\tau+1} = \mathbb{W}_\tau \setminus W_\tau$  and  $T_{\tau+1} = T_\tau \setminus T_\tau^c$ .

Recall that the number of stages (or equivalently, groups),  $k$ , is an input to the problem. The value function at stage  $\kappa$  satisfies

$$V_\kappa = \max_{W_\kappa \subseteq \mathbb{W}_\kappa} \theta(m_\kappa : T_\kappa \rightarrow W_\kappa) + V_{\kappa+1}(\mathbb{W}_{\kappa+1}, T_{\kappa+1})$$

with  $V_{k+1} = 0$ . Let us examine the solution to the formulation above when  $k = 1$ . In this case, the problem is equivalent to searching for the best possible one-shot matching under the informational constraints of the problem).

**Lemma 3.13.** *Consider a set of tasks  $\mathcal{T}$  and a set of workers  $\mathcal{W}$ , with  $\lambda(\mathcal{W}) > \lambda(\mathcal{T})$ . The optimal one-shot matching  $m^*$  assigns  $\mathcal{T}$  to  $W^* \subset \mathcal{W}$  such that  $\lambda(W^*) = \lambda(\mathcal{T})$  and the minimum skill  $s_l$  in  $W^*$  is equal to  $G^{-1}(\lambda(W^*) - \lambda(\mathcal{T}))$ .*

Lemma 3.13 provides the solution to the problem when  $k = 1$ , i.e. when an employer only does a single matching. Not surprisingly, in that scenario the firm chooses the best workers available to participate in the matching. How would the optimal grouping change for the case  $k = 2$ ? Before answering this question, it would be useful to point out that having more groups is not always better. In fact, consider a setting where the number of tasks is equal to the number of workers, then we have the following result.

**Lemma 3.14.** *Let  $\lambda(\mathcal{T}) = \lambda(\mathcal{W})$ , and denote by  $m^* : \mathcal{T} \rightarrow \mathcal{W}$  the one-shot matching (i.e. the matching with  $k = 1$ ) and by  $\mu^i$  a sequential matching that uses  $i > 1$  groups, then  $\theta(m^*) \geq \theta(\mu^i)$ .*

The intuition for Lemma 3.14 is the following: as the proportion of difficult tasks increases, it becomes more likely that a worker will pick a task that he cannot finish. In a one-shot setting, a difficult task may lead to wasting the efforts of at most one worker, but as more groups are introduced, this task has the potential to 'defeat' other workers as well, as it gets carried through

from one group to the next. These workers could have been better utilized getting other tasks, which is what would happen in the one-shot setting. Thus breaking up a group can only lead to an overall decrease in throughput. This immediately implies that we should not consider having more than one group unless there are more workers than tasks. To avoid uninteresting scenarios, we will assume that for any worker, there is always a measure of tasks, no matter how small, that he can finish. This avoids cases where a group of workers has zero throughput because no one in the group can finish any task. There is no loss of generality here of course, since if such groups exist we can either discard them or simply assign them tasks without affecting the outcome. The preceding discussion is summarized in the following assumption.

**Assumption 3.15.**  $\lambda(\mathcal{W}) \gg \lambda(\mathcal{T})$ . In addition, for any  $w \subset \mathcal{W}$ , we have  $\theta(m : \mathcal{T} \rightarrow w) > 0$ .

Continuing with these ideas, the following lemma shows that the formation of two groups requires no more workers than twice the number of tasks and that the size of the first group of workers,  $W_1$ , has to be equal to 1.<sup>4</sup> Because  $\lambda(\mathcal{T})$  is normalized to 1, the maximum size a group can have is also 1 (recall the discussion after Example 3.2, where we explain that a matching with more workers than tasks cannot improve throughput). Therefore for two groups, we will not need to use more than a total of two measures of workers. Let  $s_l$  be the least skilled worker in the top two measures of workers (using our notation,  $s_l = G^{-1}(\lambda(\mathcal{W}) - 2\lambda(\mathcal{T}))$ ). Substituting a worker whose skill is more than  $s_l$  with a worker whose skill is less than  $s_l$  in either group can only decrease throughput, and so there is no reason to consider workers whose skills are less than  $s_l$ .

Assume now that the second group has measure 1. This cannot be the case in an optimal solution unless the first group has zero throughput, which we have ruled out. Thus  $\lambda(W_2) < 1$ . Assume that  $\lambda(W_1)$  is also less than 1, then again this cannot be optimal because there will be enough unused workers whose skills are higher than or equal to  $s_l$  and that we can add to group  $W_1$  to increase its size to 1, increasing throughput in the process or at worst leaving it unchanged. This provides us with the following result.

---

<sup>4</sup>The assumption that there are many more workers than tasks simplifies the analysis but does not impact the generality of our results.

**Lemma 3.16.** *Let  $W_1^*$  and  $W_2^*$  be the optimal solution for  $k = 2$ . Under Assumption 3.15, we have  $\lambda(W_1^*) = 1$  and  $\lambda(W_2^*) < 1$ , and the minimum skill in a two-group solution is at least equal to  $s_l = G^{-1}(\lambda(\mathcal{W}) - 2\lambda(\mathcal{T}))$ .*

We can now show the following. This proposition will be the basis for our inductive proof of Theorem 3.1.

**Proposition 3.17.** *Under Assumption 3.15, there exists an optimal hierarchical matching for  $k = 2$ .*

The proof of Proposition 3.17 utilizes the results proved thus far to compare the throughputs of different group arrangements. The idea can be simplified as follows: from Lemma 3.8, we know that latter groups (groups with higher indices) face more difficult tasks regardless of the composition of earlier groups. Therefore, by Lemma 3.5, workers with lower skills would struggle if they are placed in latter groups but have a better chance at completing tasks if they are in earlier ones. As an extreme example, consider the first group  $W_1$  and let it contain a unit measure of the most skilled workers available, then any other group that follows  $W_1$  has to have throughput equal to zero, since the workers in these groups have to tackle tasks at which more skilled workers have failed. Conversely, by placing less skilled workers in earlier groups, the (easy) tasks that they finish will not get in the way of more skilled workers later on. Instead, these workers will be assigned a bigger share of tasks that are commensurate with their skills, leading to an overall increase in throughput.

The following result implies that the optimal groups are contiguous.

**Lemma 3.18.** *Consider the optimal grouping  $W^* = (W_1^*, W_2^*)$  for  $k = 2$ , and a worker  $i$  with skill  $s_i$ . If  $i \in W^*$ , then  $\{j | s_j > s_i\} \in W^*$ .*

Using the results of this section and inducting on the number of groups, we arrive at a proof of Theorem 3.1.

The precise nature of the optimal hierarchy is unraveled to reveal the following:

- a) Necessary condition for optimality: at the beginning of round  $i$ , the number of workers left should be strictly higher than the number of remaining tasks, unless  $i = k$ , in which case the

opposite should hold, i.e. the number of leftover tasks should be at least equal to the number of workers remaining.

- b) The optimal groups are contiguous. Let  $s_1 < s_2$ . If workers with skills less than  $s_1$  are assigned tasks and workers with skills higher than  $s_2$  are assigned tasks, then workers in  $(s_1, s_2)$  should be assigned tasks.
- c) Depending on the desired number of groups, the optimal solution may include a threshold for skill  $\bar{s}$  below which no worker is chosen to participate (in Example 3.2,  $\bar{s} = 0.25$ ).

## 4 The Implementation Problem

We now assume that worker participation is endogenous so that the employer cannot assign workers to groups at will, but rather has to provide an incentive for them to participate in specific groups. To do this, the employer computes the first-best solution, which as characterized above consists of discarding all workers whose skills lie beneath a skill threshold  $\bar{s}$  and then having  $k$  contiguous groups. In order to implement the first-best allocation from the assignment problem, we utilize a mechanism that charges a participation or entry fee  $q$  for workers and a tiered payment system for tasks. A task finished in group  $i, i = 1, \dots, k$  pays an amount  $p_i$ , with  $p_i > p_j$  for  $i > j$ . Fix a configuration of groups and let the probability that a worker with skill  $s$  finishes a task when he is a member of group  $W_i$  be given by  $\psi(s, W_i)$ . The expected payment that this worker receives is equal to  $p_i\psi(s, W_i)$ . Groups are in equilibrium when no worker wants to change the group he is in. Denote by group  $W_0$  those workers who do not participate, then a formal definition for an equilibrium is the following

**Definition 4.1.** (*Group Equilibrium*) Groups  $W_1, \dots, W_k$  are in equilibrium if for any skill  $s$ , a worker with skill  $s$  in  $W_i, i = 0, \dots, \kappa$  receives expected payment  $p_i\psi(s, W_i) \geq p_j\psi(s, W_j)$  for all  $j \neq i$ .

Thus for a worker to voluntarily choose to belong to group  $W_j$ , we must have  $0 < p_i\psi(s, W_i) < p_j\psi(s, W_j) < p_l\psi(s, W_l)$  for all  $i < j$  and  $l > j$ . The following result follows from Lemma 3.8.

**Lemma 4.2.** (*Monotonicity of success probabilities*) For all groups  $i$  and all skills  $s$ ,  $\psi(s, W_i)$  is monotonically decreasing in  $i$  and monotonically increasing in  $s$ .

Note however that for a worker in group  $i$ , the probability of success being non-increasing in groups does not guarantee poorer performance in later groups. This is because throughput depends both on the success probability and probability that a worker gets assigned a task to begin with. A worker in a smaller-sized group has a higher chance of getting a task and therefore might have a higher expected throughput and payoff even if the probability of success has decreased. This scenario is ruled out in the optimal solution by Corollary 3.12, since no optimal group has a size larger than the size of the tasks that are assigned to it, which means that the probability of getting a task for any worker is the same across all groups, and hence the determining factor of a worker's output remains his success probability. This discussion is summarized in the following corollary of Lemma 4.2.

**Corollary 4.3.** (*Monotonicity of Throughput*) Denote by  $\theta(s, i)$  the throughput of a worker with skill  $s$  in group  $W_i$ . In the optimal hierarchical solution  $\theta(s, i) \geq \theta(s, j)$  for  $j > i$ .

In order to derive the main result of this section, we need the following assumption

**Assumption 4.4.** (*Comparative Advantage*) Define  $\phi(s) = \frac{\psi(s, W_i)}{\psi(s, W_j)}$  for  $j > i$ , then  $\phi(s)$  is decreasing in  $s$ .

Comparative advantage is a standard assumption in the matching literature in labor economics (Costinot 2009). Informally, it means that more skilled workers perform *relatively* better on more difficult tasks than less skilled workers do. In the context of our model, this means that for two workers in group  $i$  with skills  $s_1$  and  $s_2$  and with  $s_1 > s_2$ , the drop in probability of success for the worker with skill  $s_2$  upon moving to a higher group will be larger compared to the drop for the worker with skill  $s_1$  if he makes the same move.

We now give the main result of this section.

**Theorem 4.5.** Let  $W_1^*, \dots, W_k^*$  be an optimal  $k$ -level hierarchical assignment satisfying Assumption 4.4. There exists a participation fee  $q$  and payments  $p_1, \dots, p_k$  that induce an equilibrium in which

a worker chooses to be in group  $W_i^*$  if and only if he is selected for that group in the optimal allocation.

Let  $s_i = \{\min s | s \in W_i\}$ . The proof of Theorem 4.5 shows that the values for the entry fee  $q$  and payments  $p_1, \dots, p_k$  can be obtained by solving the following simple system of equations.

$$\begin{aligned} p_1 \psi(s_1, W_1) &= q \\ p_i \psi(s_{i+1}, W_i) &= p_{i+1} \psi(s_{i+1}, W_{i+1}) \quad i = 1, 2, \dots, k-1 \end{aligned}$$

The first equation above indicates that the least skilled worker who participates only breaks even in expectation. Because  $\psi(s, W_1)$  is monotonically increasing in  $s$ , any worker whose skill is less than  $s_1$  will have negative expected payoff and therefore will not participate. Similarly, by recalling that groups are contiguous, the second equation gives a condition that makes workers on the boundary between successive groups indifferent. Again, from the monotonicity of  $\psi(s, W_i)$  (increasing in  $s$  and decreasing in  $i$ ), we find that an increase in payment  $p_{i+1}$  over payment  $p_i$  by a factor  $\frac{\psi(s_{i+1}, W_i)}{\psi(s_{i+1}, W_{i+1})}$  will provide less payoffs to a worker with skill less than  $s$  who decides to move to group  $i+1$ , and a worker whose skill is higher than  $s$  and decides to move to group  $i$ . This is because, upon a move to a group with higher index, the drop in probability of successful completion for a less skilled worker is not sufficiently compensated by the increase in payment, and the opposite is true for a more skilled worker who contemplates moving in the opposite direction.

**Example 4.6.** We revisit Example 3.2 to show how the optimal two group assignment can be implemented. We want to find  $q$ ,  $p_1$ , and  $p_2$  that will provide the correct incentives to workers to participate in the groups they were allocated to in that example. Let us denote by the dummy group  $W_0$  the group of workers who choose not to participate. Recall that the optimal groups were (in terms of skill)  $W_0 = (0, 0.25)$ ,  $W_1 = (0.25, 0.75)$ , and  $W_2 = (0.75, 1)$ . By setting  $q = 0.25$ ,  $p_1 = 1$ , and  $p_2 = 1.5$ , we obtain the desired hierarchy. To see this, let us examine a deviation of a worker of skill  $s$  in  $W_0$ . This worker makes 0 in that group, and by deviating to join group  $W_1$  he makes  $p_1 \psi(s, W_1) - q$ . For this example,  $\psi(s, W_1) = s$ , and so the expected payoff from deviation is equal to

$1s - 0.25$ , which is less than zero for  $s < 0.25$ . Similarly, a worker of skill  $s$  in  $W_1$  makes an expected profit of  $1s - 0.25$ , which is higher than 0 but also higher than what he would make had he decided to join  $W_2$  instead. In that case he will be making  $p_2\psi(s, W_2) - q = 1.5(s - 0.25) - 0.25$ , which is less than  $1s - 0.25$  for  $s \in (0.25, 0.75)$ . Finally, a worker of skill  $s$  in  $W_2$  has  $\psi(s, W_2) = \frac{1}{2} + \frac{s-0.75}{0.5}$  for an expected payoff of  $1.5(\frac{1}{2} + \frac{s-0.75}{0.5}) - 0.25$ . This is higher than what a worker whose skill is in  $(0.75, 1)$  stands to gain by moving to  $W_1$ .

A simple implementation of the mechanism announces the entry fee  $q$  and the prices  $p_i$  for each group. A worker chooses either not to participate at all or to participate in exactly one of the groups. A worker participating in group  $i$  is not allowed to participate in group  $j > i$ , although as we explained in the introduction, this worker will have little reason to pay another entry fee and try his luck again with the same employer given that his chances of success have become worse as the tasks are now more difficult.

## 5 Conclusion

In this work, we propose combining new concepts from crowdsourcing innovation with standard ideas about firm and organizational structure. The goal is to be able to take advantage of the wealth of knowledge accumulated in strategic management in order to maximize impact of crowdsourced innovation endeavors. Innovation involves varying degrees of uncertainty, and this uncertainty has tremendous effects on participation, project costs, and eventual outcomes. By designing mechanisms that incentivize innovation workers to behave as if they are embedded in a single organization and working towards a common goal, we can leverage much of our understanding about organizational structure to deal with these uncertainties and advance the practice of crowd innovation to the next level. We have shown how workers can be incentivized to partition themselves in a way that improves the utilization of skills. Taking further inspiration from the standard model of the firm, a natural next step is to think about providing incentives to crowdsourced workers, who potentially have little knowledge of one another, to endogenously and dynamically form teams that lead to even higher efficiency and more utilization of skills.

Fortunately, the abundance of crowdsourcing platforms means that ideas like the one in this paper can be immediately deployed and tested. We are currently working on an experimental design to validate the hierarchical model with real workers and tasks, so that we can investigate how it can be further improved. We expect that while workers in our model choose what tasks to work on based only on monetary compensation, there will be factors, behavioral and otherwise, that can influence these choices. We hope that our model will evolve to accommodate these factors and to ultimately provide a real impact in the new innovation marketplace.

## References

- BANSAL, N., A. GUPTA, J. LI, J. MESTRE, V. NAGARAJAN, AND A. RUDRA (2010): “When LP is the cure for your matching woes: improved bounds for stochastic matchings,” *Algorithms-ESA 2010*, pp. 218–229.
- BOUDREAU, K., N. LACETERA, AND K. LAKHANI (2011): “Incentives and problem uncertainty in innovation contests: An empirical analysis,” *Management Science*, 57(5), 843–863.
- CHANDLER, A. (1977): *The visible hand: The managerial revolution in American business*. Belknap Press.
- CHAWLA, S., J. HARTLINE, AND B. SIVAN (2012): “Optimal crowdsourcing contests,” in *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 856–868. SIAM.
- CHEN, N., N. IMMORLICA, A. KARLIN, M. MAHDIAN, AND A. RUDRA (2009): “Approximating matches made in heaven,” *Automata, Languages and Programming*, pp. 266–278.
- CHESBROUGH, H., W. VANHAVERBEKE, AND J. WEST (2008): *Open Innovation: Researching a New Paradigm: Researching a New Paradigm*. OUP Oxford.
- COASE, R. (2007): “The nature of the firm,” *Economica*, 4(16), 386–405.

- COSTINOT, A. (2009): “An Elementary Theory of Comparative Advantage,” *Econometrica*, pp. 1165–1192.
- DULLECK, U., R. KERSCHBAMER, AND M. SUTTER (2011): “The Economics of Credence Goods: An Experiment on the Role of Liability, Verifiability, Reputation, and Competition,” *American Economic Review*, 101, 526–555.
- FARIDANI, S., B. HARTMANN, AND P. IPEIROTIS (2011): “Whats the right price? pricing tasks for finishing on time,” in *Proc. of AAAI Workshop on Human Computation*.
- FELDMAN, J., A. MEHTA, V. MIRROKNI, AND S. MUTHUKRISHNAN (2009): “Online stochastic matching: Beating  $1-1/e$ ,” in *Foundations of Computer Science, 2009. FOCS’09. 50th Annual IEEE Symposium on*, pp. 117–126. IEEE.
- FULLERTON, R., AND R. MCAFEE (1999): “Auctionin Entry into Tournaments,” *Journal of Political Economy*, 107(3), 573–605.
- GARICANO, L. (2000): “Hierarchies and the Organization of Knowledge in Production,” *Journal of Political Economy*, 108(5), 874–904.
- GERSHKOV, A., AND M. PERRY (2009): “Tournaments with midterm reviews,” *Games and Economic Behavior*, 66(1), 162–190.
- GHOSH, A., AND P. MCAFEE (2012): “Crowdsourcing with endogenous entry,” in *Proceedings of the 21st international conference on World Wide Web*, pp. 999–1008. ACM.
- HECKMAN, J., AND C. TABER (2008): “The Roy Model,” *The New Palgrave Dictionary of Economics*.
- HORTON, J., AND L. CHILTON (2010): “The labor economics of paid crowdsourcing,” in *Proceedings of the 11th ACM conference on Electronic commerce*, pp. 209–218. ACM.
- KAMENICA, E., AND M. GENTZKOW (2011): “Bayesian Persuasion,” *American Economic Review*, 101(6).

- MOLDOVANU, B., AND A. SELA (2001): “The optimal allocation of prizes in contests,” *American Economic Review*, pp. 542–558.
- (2006): “Contest architecture,” *Journal of Economic Theory*, 126(1), 70–96.
- SCHÖTTNER, A. (2008): “Fixed-prize tournaments versus first-price auctions in innovation contests,” *Economic Theory*, 35(1), 57–71.
- SIEGEL, R. (2009): “All-Pay Contests,” *Econometrica*, 77(1), 71–92.
- VON HIPPEL, E. (1976): “The dominant role of users in the scientific instrument innovation process,” *Research policy*, 5(3), 212–239.
- (2009): “Democratizing innovation: the evolving phenomenon of user innovation,” *International Journal of Innovation Science*, 1(1), 29–40.
- WOLINSKY, A. (1993): “Competition in a market for informed experts’ services,” *The RAND Journal of Economics*, pp. 380–398.

## A Proofs

Throughout this section, we will refer in different ways to the throughput of a sequential match  $\theta(\mu)$  depending on context. Sometimes explicitly as  $\theta(m_1 : T \rightarrow W_1, m_2 : T'_1 \rightarrow W_2, \dots)$ , or simply the shorthand  $\theta(W_1, W_2, \dots)$  to indicate that the set  $T$  is used for the first matching,  $T'_1 = T \setminus T_i^c$  for the second and so on.

**Proof of Lemma 3.5.** Let  $f_W(s)$  be the density function for skills in  $W$ . We rewrite the throughput of  $m_1$  and  $m_2$  as

$$\theta(m_1) = \pi(m_1)\lambda(T_1) = \int_{s_1}^{s_2} F_{d_{T_1}}(s)f_W(s)ds \quad (1)$$

$$\theta(m_2) = \pi(m_2)\lambda(T_2) = \int_{s_1}^{s_2} F_{d_{T_2}}(s)f_W(s)ds \quad (2)$$

Since  $d_{T_1} \succeq d_{T_2}$  implies  $F_{d_{T_1}}(s) \leq F_{d_{T_2}}(s)$  for all  $s$  and  $F_{d_{T_1}}(s) < F_{d_{T_2}}(s)$  for some  $s$ , we have  $(1) \leq (2)$  and hence  $\theta(m_1) \leq \theta(m_2)$ .  $\square$

**Proof of Lemma 3.7.** Denote by  $S$  the set of skills in  $W \cup w'$ . Because  $w' \succ w$ , there is a set  $D$  such that  $F_{s_W}(d_T) = F_{s_{W'}}(d_T)$  for  $d_T \in D$  and  $F_{s_W}(d_T) > F_{s_{W'}}(d_T)$  for  $d_T \in \bar{D}$ , where  $D \cap \bar{D} = \emptyset$  and  $D \cup \bar{D} = S$ . This implies that  $s_{W'} \succeq s_W$ . By writing down the expressions for  $\pi(m)$  and  $\pi(m')$ , we get  $\pi(m) = \int_{d_1}^{d_2} (1 - F_{s_W}(\delta)) f_{d_T}(\delta) d\delta$  and  $\pi(m') = \int_{d_1}^{d_2} (1 - F_{s_{W'}}(\delta)) f_{d_T}(\delta) d\delta$ , where  $(d_1, d_2)$  is the domain of difficulties in  $T$ . It is straightforward to see that, for any  $\delta \in (d_1, d_2)$

$$\begin{aligned} F_{s_W}(\delta) &\geq F_{s_{W'}}(\delta) \\ \int_{d_1}^{d_2} (1 - F_{s_W}(\delta)) f_{d_T}(\delta) d\delta &\leq \int_{d_1}^{d_2} (1 - F_{s_{W'}}(\delta)) f_{d_T}(\delta) d\delta \\ \pi(m) &\leq \pi(m') \end{aligned}$$

Since  $\theta(m) = \pi(m)\lambda(T)$  and  $\theta(m') = \pi(m')\lambda(T)$ , the result follows.  $\square$

**Proof of Lemma 3.8.** The case  $w = \emptyset$  coupled with the assumption that  $T' \neq \emptyset$  implies that  $\lambda(W) < \lambda(T)$ . Because  $w = \emptyset$ ,  $\lambda(T^c) = \lambda(W)$  and  $\lambda(T') = \lambda(T) - \lambda(T^c) = \lambda(T) - \lambda(W)$ . Since the matching  $m : T \rightarrow W$  is random, this is equivalent to selecting a set of measure  $\lambda(T')$  uniformly at random from  $T$ , and hence  $F_{d_{T'}} = F_{d_T}$ .

Now let  $w \neq \emptyset$  and assume for simplicity that  $\lambda(W) = \lambda(T)$ . We can divide  $T$  into three regions as in Definition 3.4. Let  $\gamma_i$  be the probability that a worker in  $W$  finishes a task in region  $i$ , so that  $\gamma_1 = 1$  and  $\gamma_2 = 0$ . From Definition 3.3,  $\gamma_3$  is given by

$$\gamma_3 = \int_{s_l}^{s_h} (1 - F_{s_W}(\delta)) f_{d_T}(\delta) d\delta$$

The set  $T' = T \setminus T^c$  has two regions:

- Region A: This is the same as R2 in Definition 3.4, with measure  $\lambda(T)(1 - F_{d_T}(s_h))$ .
- Region B: This contains the uncompleted tasks from R3, and has measure  $(1 - \gamma_3)\lambda(T)(F_{d_T}(s_h) - F_{d_T}(s_l))$ .

It is straightforward to see that the relative measure of tasks in R2 has increased. Tasks with difficulties over  $s_h$  had a proportion of  $1 - F_{d_T}(s_h)$  in  $T$ , but in  $T'$ , the same tasks now have a proportion

$$\begin{aligned} \frac{\lambda(T)(1 - F_{d_T}(s_h))}{\lambda(T)(1 - F_{d_T}(s_h)) + (1 - \gamma_3)\lambda(T)(F_{d_T}(s_h) - F_{d_T}(s_l))} &= \frac{1 - F_{d_T}(s_h)}{1 - F_{d_T}(s_h) + (1 - \gamma_3)(F_{d_T}(s_h) - F_{d_T}(s_l))} \\ &= \frac{1 - F_{d_T}(s_h)}{1 - F_{d_T}(s_l) - \gamma_3(F_{d_T}(s_h) - F_{d_T}(s_l))} \\ &> 1 - F_{d_T}(s_h) \end{aligned}$$

To understand the distribution of the tasks in Region B, note that the probability  $1 - F_{s_W}(\delta)$  of finishing a task in R3 is decreasing in  $\delta$ . Consider contiguous intervals  $d_1 = (\delta_1, \delta_2)$ , and  $d_2 = (\delta_3, \delta_4)$ , with  $\delta_3 > \delta_2$  and  $\lambda(d_1) = \lambda(d_2)$ . The probability of finishing a task in  $(\delta_1, \delta_2)$  is given by

$$\begin{aligned} \gamma &= \int_{\delta_1}^{\delta_2} (1 - F_{s_W}(\delta)) f_{d_T}(\delta) d\delta \\ &< \int_{\delta_3}^{\delta_4} (1 - F_{s_W}(\delta)) f_{d_T}(\delta) d\delta \\ &= \gamma' \end{aligned}$$

where  $\gamma'$  is the probability of finishing a task in  $(\delta_3, \delta_4)$ . Thus the measure of tasks in  $d_1$  decreases by  $\gamma\lambda(d_1)$  while the measure of tasks in  $d_2$  decreases by  $\gamma'\lambda(d_2) < \gamma\lambda(d_1)$ , which implies that the relative measure of tasks in  $(\delta_3, \delta_4)$  has increased in Region B from its original proportion in R3. Let  $j$  and  $j'$  be two randomly chosen tasks from  $T$  and  $T'$ , respectively, then for any value of difficulty  $d$ ,  $Pr(d_j \leq d) \geq Pr(d_{j'} \leq d)$  and  $Pr(d_j \leq d) > Pr(d_{j'} \leq d)$  for  $d > s_h$ , implying that  $d_{T'} \succeq d_T$ . □

**Proof of Lemma 3.9.** Let  $\lambda(T)$  be normalized to 1 and let  $\lambda(W_1) = \lambda(W_2) = l \leq \frac{1}{2}$ . Assume wlog that the least difficult task in  $T$  and the lowest skill worker in  $W_1$  have value zero. Let  $d_1 = \max\{s_i : i \in W_1\}$  and  $d_2 = \max\{s_j : j \in W_2\}$ , and divide the difficulties in  $T$  into intervals

$(0, d_1)$ ,  $(d_1, d_2)$ , and  $(d_2, d_{max})$ , where  $d_{max}$  is the maximum difficulty in  $T$ . Let

$$F_{d_T}(d) = \begin{cases} x_1 & d = d_1 \\ x_2 & d = d_2 \end{cases}$$

Denote by  $T_{(\underline{d}, \bar{d})}$  the set of tasks in  $T$  whose difficulties lie in  $(\underline{d}, \bar{d})$  and define matchings  $m : T_{(0, d_1)} \rightarrow W_1$  and  $m' : T_{(d_1, d_2)} \rightarrow W_2$ , where  $T_{(0, d_1)}$  and  $T_{(d_1, d_2)}$  denote the tasks in  $T$  whose difficulties are in  $(0, d_1)$  and  $(d_1, d_2)$ , respectively. Recall that the probabilities of finishing a task in  $m$  and  $m'$  are given by  $\pi(m)$  and  $\pi(m')$ , respectively. Because  $W_2 \succ W_1$ ,  $Pr(s_j \geq d) = 1$  for all  $j \in W_2$  and  $d \in (0, d_1)$ . Now define  $\mu^A = (m_1 : T \rightarrow W_1, m_2 : T'_1 \rightarrow W_2)$  and  $\mu^B = (m_3 : T \rightarrow W_2, m_4 : T'_2 \rightarrow W_1)$ , where  $T'_1$  and  $T'_2$  are defined as in the statement of the lemma. We have  $\theta(m_1) = \theta(m) = lx_1\pi(m)$  and  $\lambda(T'_1) = 1 - lx_1\pi(m)$ , with  $d_{T'_1}$  distributed as

$$F_{d_{T'_1}}(d) = \begin{cases} \frac{x_1(1-l\pi(m))}{1-lx_1\pi(m)} & d = d_1 \\ \frac{x_2-lx_1\pi(m)}{1-lx_1\pi(m)} & d = d_2 \end{cases}$$

and hence  $\theta(m_2) = l \left( \frac{x_1(1-l\pi(m)) + \pi(m')(x_2-x_1)}{1-lx_1\pi(m)} \right)$ . This means that

$$\theta(\mu^A) = \theta(m_1) + \theta(m_2) = lx_1\pi(m) + l \left( \frac{x_1(1-l\pi(m)) + \pi(m')(x_2-x_1)}{1-lx_1\pi(m)} \right) \quad (3)$$

Proceeding in the same fashion, we compute  $\theta(m_3) = l(x_1 + (x_2 - x_1)\pi(m'))$ . The probability that a task has difficulty in  $(0, d_1)$  in  $T'_2$  is given by

$$F_{d_{T'_2}}(d_1) = \frac{x_1(1-l)}{1-l(x_1 + \pi(m')(x_2-x_1))}$$

Consequently,  $\theta(m_4) = l\pi(m) \left( \frac{x_1(1-l)}{1-l(x_1 + \pi(m')(x_2-x_1))} \right)$ , and

$$\theta(\mu^B) = \theta(m_3) + \theta(m_4) = l(x_1 + (x_2 - x_1)\pi(m')) + l\pi(m) \left( \frac{x_1(1-l)}{1-l(x_1 + \pi(m')(x_2-x_1))} \right) \quad (4)$$

Since  $d_2 > d_1$  implies  $x_2 > x_1$ , it can be verified that (3) is indeed greater than or equal to (4) for any values of  $l$ ,  $\pi(m)$ , and  $\pi(m')$ , and hence  $\theta(\mu^A) \geq \theta(\mu^B)$ . □

**Proof of Lemma 3.10.** Let  $s_l = \min_s s \in W_3$  and define  $T_f^i = \{t \in T_i' | d_t \leq s_l\}$ . It is enough to note that  $\lambda(T_1') \leq \lambda(T_2')$  and  $T_2^c \cap T_1^c = T_2^c$  and hence  $\lambda(T_2^c) = \lambda(T_1^c) - \tau$  and  $\lambda(T_f^1) = \lambda(T_f^2) - \tau$  for some  $\tau \geq 0$ . This implies that  $\frac{\lambda(T_f^1)}{\lambda(T) - \lambda(T_1^c)} = \frac{\lambda(T_f^2) - \tau}{\lambda(T) - \lambda(T_2^c) - \tau} \leq \frac{\lambda(T_f^2)}{\lambda(T) - \lambda(T_2^c)}$ , i.e. the proportion of the tasks that  $W_3$  can finish has shrunk, and since  $\lambda(T_1') \leq \lambda(T_2')$ ,  $W_3$  now finishes a smaller proportion from a smaller set of tasks in  $m_2$  compared to  $m_4$ , and therefore  $\theta(m_4) \geq \theta(m_2)$ . □

**Proof of Lemma 3.11.** Proof of Part a: Note that  $W \succ w$  implies  $s_W \succeq s_{W'}$ , i.e.  $F_{s_W} \leq F_{s_{W'}}$ , and hence by the same arguments of Lemma 3.7 we have  $\pi(m) \geq \pi(m') \rightarrow \theta(m) \geq \theta(m')$ .

Proof of Part b: Let  $\bar{T} \subset T$  be the set of tasks assigned to  $W$  in  $m$ , then by the assumption that matchings are uniformly random, the exact same set of tasks will be assigned to  $W \subset W'$  in  $m'$ , making  $\theta(m') \geq \theta(m)$ .

Proof of Part c: The same argument as Part b applies, with the bigger set being  $W$  instead of  $T$ . □

**Proof of Lemma 3.13.** Let the domain of difficulties in  $\mathcal{T}$  be given by  $(d_1, d_2)$  and consider the assignment  $m^*$  in the statement of the lemma. Because we normalize  $\lambda(\mathcal{T})$  to 1, we have

$$\theta(m^*) = \lambda(\mathcal{T})\pi(m^*) = \pi(m^*) = \int_{d_1}^{d_2} (1 - F_{s_{W^*}}(\delta)) f_{d_{\mathcal{T}}}(\delta) d\delta \quad (5)$$

Note that any optimal matching  $m$  will not use more than  $\lambda(\mathcal{T})$  workers. Assume  $m' : \mathcal{T} \rightarrow W'$  where  $\lambda(W') > \lambda(\mathcal{T})$  and where  $W'$  is arranged in ascending order of skill over the set  $(s_1, s_2)$ , with  $s_1 = \min_{i \in W'} s_i$  and  $s_2 = \max_{i \in W'} s_i$ . Let  $s' = \max_{i \in W'} s_i$  s.t.  $\lambda((s_i, s_2)) = \lambda(\mathcal{T})$  and define matching  $m'' : \mathcal{T} \rightarrow W''$  such that  $W''$  is the set of workers with skills in  $(s', s_2)$ , then for any

$\delta \in (d_1, d_2)$

$$\begin{aligned} F_{s_{W'}}(\delta) &> F_{s_{W''}}(\delta) \\ \int_{d_1}^{d_2} (1 - F_{s_{W'}}(\delta)) f_{d_{\mathcal{T}}}(\delta) d\delta &< \int_{d_1}^{d_2} (1 - F_{s_{W''}}(\delta)) f_{d_{\mathcal{T}}}(\delta) d\delta \\ \pi(m') &< \pi(m'') \end{aligned}$$

and hence  $\theta(m'') > \theta(m')$ . Similarly, there is always an optimal matching that uses exactly  $\lambda(\mathcal{T})$  workers, since if a matching is optimal and uses  $l < \lambda(\mathcal{T})$  we can add  $l - \lambda(\mathcal{T})$  workers to that matching without affecting the outcome. Finally, by examining (5) and noting that the distribution of skill in any set of workers  $\bar{W}$  with  $\lambda(\bar{W}) = 1$  and  $\bar{W} \neq W^*$  satisfies  $s_{W^*} \succeq s_{\bar{W}}$ , the result follows.  $\square$

**Proof of Lemma 3.14.** Divide  $\mathcal{W}$  into two arbitrary, disjoint groups  $W_1$  and  $W_2$  and let  $\mu^2 = (m_1 : \mathcal{T} \rightarrow W_1, m_2 : \mathcal{T} \setminus \mathcal{T}^c \rightarrow W_2)$  be a sequential matching, where  $\mathcal{T}^c$  are the tasks finished in  $m_1$ . Now consider the following *one-shot* matching  $m^*$  that divides  $\mathcal{W}$  into  $W_1$  and  $W_2$  and divides  $\mathcal{T}$  into  $T_1$  and  $T_2$  with  $\lambda(T_1) = \lambda(W_1)$  and  $\lambda(T_2) = \lambda(W_2)$ . Matching  $m^*$  is not sequential; it simply performs the matchings  $T_1 \rightarrow W_1$  and  $T_2 \rightarrow W_2$  simultaneously. This means that  $\theta(m^*)$  can be written as  $\theta(T_1 \rightarrow W_1) + \theta(T_2 \rightarrow W_2)$ , whereas  $\theta(\mu^2) = \theta(m_1) + \theta(m_2)$ . Because the tasks in  $T_1$  are selected uniformly at random from  $\mathcal{T}$ , we have  $F_{d_{T_1}} = F_{d_{\mathcal{T}}}$ , and hence  $\theta(m_1) = \theta(T_1 \rightarrow W_1)$ . However, by Lemma 3.8, the set of tasks  $\mathcal{T} \setminus \mathcal{T}^c$  is distributed such that  $d_{\mathcal{T} \setminus \mathcal{T}^c} \succeq d_{\mathcal{T}}$ , and since  $F_{d_{T_2}} = F_{d_{\mathcal{T}}}$ ,  $d_{\mathcal{T} \setminus \mathcal{T}^c} \succeq d_{T_2}$ . By Lemma 3.5, we have  $\theta(m_2) \leq \theta(T_2 \rightarrow W_2)$ , leading to  $\theta(m_1) + \theta(m_2) = \theta(\mu^2) \leq \theta(m^*) = \theta(T_1 \rightarrow W_1) + \theta(T_2 \rightarrow W_2)$ . The same argument applies to any value of  $i > 1$ .  $\square$

**Proof of Lemma 3.16.** Assume  $\lambda(W_2) = 1$ . By Assumption 3.15,  $\theta(m_1 : \mathcal{T} \rightarrow W_1) > 0$ , indicating that  $\lambda(\mathcal{T} \setminus \mathcal{T}_1^c) < 1 = \lambda(W_2)$ , where  $\mathcal{T}_1^c$  are the tasks completed in  $m_1$ . By Lemma 3.11.a and Corollary 3.12, there is a group  $W_2' \subset W_2$  with  $\lambda(W_2') = \lambda(\mathcal{T} \setminus \mathcal{T}_1^c)$  such that  $\theta(\mathcal{T} \setminus \mathcal{T}_1^c \rightarrow W_2) \leq \theta(\mathcal{T} \setminus \mathcal{T}_1^c \rightarrow W_2')$ , and hence  $\lambda(W_2) = 1$  cannot be optimal. Now assume that  $\lambda(W_1) < 1$  and consider  $W_1' = W_1 \cup w$ , where  $w$  has size  $1 - \lambda(W_1)$ , then by Lemma 3.11.b,  $\theta(m_1' : \mathcal{T} \rightarrow W_1') \geq$

$\theta(m_1 : \mathcal{T} \rightarrow W_1)$ , and  $W'_1 = 1$  can only increase throughput.

Now consider groups  $W_1$  and  $W_2$  with  $\lambda(W_1) = 1$  and  $\lambda(W_2) < 1$  and let the minimum skill across both groups be equal to  $s_l$  as given in the statement of the lemma. We will show that replacing any group of workers with another less-skilled group will not improve throughput. Consider a group  $\bar{w}$  where  $\max_{i \in \bar{w}} s_i < s_l$  and any group  $w_1 \in W_1$  such that  $\lambda(\bar{w}) = \lambda(w_1)$  and replace group  $w_1$  with  $\bar{w}$  in  $W_1$  to get  $\bar{W}_1$ , then  $\pi(\bar{m}_1 : \mathcal{T} \rightarrow \bar{W}_1) \leq \pi(m_1 : \mathcal{T} \rightarrow W_1)$  and  $\theta(\bar{m}_1 : \mathcal{T} \rightarrow \bar{W}_1) \leq \theta(m_1 : \mathcal{T} \rightarrow W_1)$ . Let  $\bar{\mathcal{T}}_1^c$  be the tasks completed in  $\bar{m}_1$ , then  $\lambda(\bar{\mathcal{T}}_1^c) \leq \lambda(\mathcal{T}_1^c)$  and  $\mathcal{T} \setminus \mathcal{T}_1^c \subseteq \mathcal{T} \setminus \bar{\mathcal{T}}_1^c$ . However,  $\theta(\bar{m}_2 : \mathcal{T} \setminus \bar{\mathcal{T}}_1^c \rightarrow W_2) - \theta(m_2 : \mathcal{T} \setminus \mathcal{T}_1^c \rightarrow W_2) \leq \theta(m_1 : \mathcal{T} \rightarrow W_1) - \theta(\bar{m}_1 : \mathcal{T} \rightarrow \bar{W}_1)$ , since  $\theta(\bar{m}_2) - \theta(m_2) \leq \lambda(\mathcal{T}_1^c) - \lambda(\bar{\mathcal{T}}_1^c)$ . Therefore,  $\theta(\bar{m}_1 : \mathcal{T} \rightarrow \bar{W}_1) + \theta(\bar{m}_2 : \bar{\mathcal{T}}_1^c \rightarrow W_2) \leq \theta(m_1 : \mathcal{T} \rightarrow W_1) + \theta(m_2 : \mathcal{T}_1^c \rightarrow W_2)$ , and throughput does not improve. Similarly, Consider  $w_2 \in W_2$  such that  $\lambda(\bar{w}) = \lambda(w_2)$  and replace group  $w_2$  with  $\bar{w}$  in  $W_2$  to get  $\bar{W}_2$ . The set  $\mathcal{T} \setminus \mathcal{T}_1^c$  does not change but  $\pi(\bar{m}_2 : \mathcal{T} \setminus \mathcal{T}_1^c \rightarrow \bar{W}_2) \leq \pi(m_2 : \mathcal{T} \setminus \mathcal{T}_1^c \rightarrow W_2)$  and therefore  $\theta(\bar{m}_2 : \mathcal{T} \setminus \mathcal{T}_1^c \rightarrow \bar{W}_2) \leq \theta(m_2 : \mathcal{T} \setminus \mathcal{T}_1^c \rightarrow W_2)$ , leading to  $\theta(m_1 : \mathcal{T} \rightarrow W_1) + \theta(\bar{m}_2 : \mathcal{T} \setminus \bar{\mathcal{T}}_1^c \rightarrow \bar{W}_2) \leq \theta(m_1 : \mathcal{T} \rightarrow W_1) + \theta(m_2 : \mathcal{T} \setminus \mathcal{T}_1^c \rightarrow W_2)$ , and throughput again does not improve.

Finally, assume that there is  $w' \in W_i, i = 1, 2$  such that  $w'$  has skill  $s'_i$  and  $s'_i < s_l$ , then by the arguments above we can find a group  $w''$  with skill  $s''_i > s_l > s'_i$  such that  $W'_i = W_i \cup w'' \setminus w'$ ,  $\lambda(W'_i) = \lambda(W_i)$ , and  $W'_i$  has at least the same throughput as  $W_i$ .  $\square$

We prove the following lemma in preparation for proving Lemma 3.17. Roughly, this lemma states that any improvement on a two-group hierarchical arrangement can only be hierarchical, by either moving some of the most skilled workers from the first group to the second, or moving some of the least skilled workers from the second group to the first.

**Lemma A.1.** *Let  $W_1$  and  $W_2$  be two worker groups such that  $W_2 \succ W_1$  and denote by  $\mu$  the matching  $(m_1 : T \rightarrow W_1, m_2 : T'_1 \rightarrow W_2)$ .*

- a. *Consider  $w \subset W_1$  and  $w' \subset W_1$  such that  $w \succ w'$  and  $\lambda(w) = \lambda(w')$  and let  $\mu' = (m'_1 : T \rightarrow W_1 \setminus w, T'_{1'} \rightarrow W_2 \cup w)$  and  $\mu'' = (m''_1 : T \rightarrow W_1 \setminus w', T'_{1''} \rightarrow W_2 \cup w')$ , where  $T'_{1'}$  and  $T'_{1''}$  are the set of tasks remaining after  $m'_1$  and  $m''_1$ , respectively. If  $\theta(\mu'') \geq \theta(\mu)$ , then  $\theta(\mu') \geq \theta(\mu)$ .*
- b. *Consider  $w \subset W_2$  and  $w' \subset W_2$  such that  $w \succ w'$  and  $\lambda(w) = \lambda(w')$  and let  $\mu' = (m'_1 : T \rightarrow$*

$W_1 \cup w, T'_{1'} \rightarrow W_2 \setminus w$ ) and  $\mu'' = (m'' : T \rightarrow W_1 \cup w', T'_{1''} \rightarrow W_2 \setminus w')$ , where  $T'_{1'}$  and  $T'_{1''}$  are the set of tasks remaining after  $m'_{1'}$  and  $m''_{1'}$ , respectively. If  $\theta(\mu') \geq \theta(\mu)$ , then  $\theta(\mu'') \geq \theta(\mu')$ .

**Proof of Lemma A.1.** To prove Part a, assume as in the statement of the lemma that  $\theta(\mu'') \geq \theta(\mu)$ . We can switch  $w$  and  $w'$  in  $\mu'$  to obtain  $\mu''$ . By Lemma 3.9,  $w'$  followed by  $w$  has higher throughput than  $w$  followed by  $w'$  and because  $w \succ w'$ , by Lemma 3.10,  $\theta(T \rightarrow W_1 \setminus w, T'_{1'} \rightarrow W_2) \geq \theta(T \rightarrow W_1 \setminus w', T'_{1'} \rightarrow W_2)$ , and hence  $\theta(\mu') \geq \theta(\mu'')$ . The argument for Part b is similar: if we swap  $w'$  for  $w$  in  $\mu'$  we get  $\mu''$ . Again by Lemmas 3.9 and 3.10 having  $w'$  precede  $w$  and  $W_2 \setminus w$  can only increase throughput compared to  $w$  preceding  $w'$ , and therefore  $\theta(\mu') \leq \theta(\mu'')$ .  $\square$

**Proof of Lemma 3.17.** Assume that  $W_1$  and  $W_2$  constitute an optimal solution for  $k = 2$  and that they are not hierarchical. Construct  $W'_1$  and  $W'_2$  such that  $i \in (W_1 \cup W_2) \rightarrow i \in (W'_1 \cup W'_2)$ ,  $\lambda(W_1) = \lambda(W'_1)$ ,  $\lambda(W_2) = \lambda(W'_2)$ , and  $W'_2 \succ W'_1$ . There exists  $w'_1 \subset W'_1$  and  $w'_2 \subset W'_2$  such that  $W_1 = (W'_1 \setminus w'_1) \cup w'_2$  and  $W_2 = (W'_2 \setminus w'_2) \cup w'_1$ . Thus, by moving  $w'_2$  to  $W'_1$  and moving  $w'_1$  to  $W_2$ , we can reconstruct  $W_1$  and  $W_2$ . However, by Lemma A.1, unless either: a)  $w'_1 = \emptyset$  and  $w'_2$  is such that  $(W'_2 \setminus w'_2) \succ w'_2$  or b)  $w'_2 = \emptyset$  and  $w'_1$  is such that  $w_1 \succ (W'_1 \setminus w'_1)$  – either of which imply that  $W_1$  and  $W_2$  are hierarchical and hence contradict our assumption– there exists  $\bar{W}_1$  and  $\bar{W}_2$  such that  $\bar{W}_1$  and  $\bar{W}_2$  are hierarchical and  $\theta(\bar{W}_1, \bar{W}_2) \geq \theta(W_1, W_2)$ .  $\square$

**Proof of Lemma 3.18.** Assume that there is  $w_1 \succ w_2 \succ w_3$  such that  $w_1 \in W^*$  and  $w_3 \in W^*$ , and  $w_2 \notin W^*$  and  $\lambda(w_1) = \lambda(w_2)$ <sup>5</sup>. The result follows from Lemma 3.7 since the output of  $W^*$  improves by swapping  $w_1$  and  $w_2$ , and hence  $W^*$  cannot be optimal.  $\square$

**Proof of Theorem 3.1.** We prove the theorem by induction on the number of groups. Lemma 3.17 shows that the base case for  $k = 2$  holds. Assume that the result holds for  $k$  and consider  $k + 1$  groups. By the induction hypothesis, the optimal solution for the assignment of the remaining tasks from the matching  $m_1 : \mathcal{T} \rightarrow W_1$  is a hierarchy. Furthermore, Lemma 3.18 applies to  $k$  groups since the output of a group of workers can only increase by substituting a dominated subgroup within a group by a group of equal size that dominates it but is not in the current selection of workers. This implies that the optimal solution for  $k$  groups starts with a worker with skill  $\bar{s}$  and includes

<sup>5</sup>We can always find groups that fulfill this measure requirement under the assumption  $w_1 \in W^*$  and  $w_2 \notin W^*$ .

all workers up to the maximum skill in  $\mathcal{W}$ . Therefore, the optimal solution for  $k + 1$  groups uses at most one more measure of workers than the solution for group  $k$ . This measure consists of workers whose skill lie in  $(G^{-1}(G(s) - 1), s)$ . Assume that the optimal solution uses workers whose skills is less than  $G^{-1}(G(s) - 1)$ , then by Lemma 3.7 we can replace these workers by others whose skills are in  $(G^{-1}(G(s) - 1), s)$  to improve throughput. Consider moving  $w \in W_i, i > 2$  to group  $W_1$ . By Lemmas 3.9 and 3.10, this will decrease throughput. By similar arguments to Lemma A.1, the only move with a potential for throughput maximization is one that removes a group  $w'$  from  $W_2$  such that  $(W_2 \setminus w') \succ w'$ , and adds it to  $W_1$ , and hence the optimal  $k + 1$  groups solution is also hierarchical.  $\square$

**Proof of Lemma 4.2.** Let the distribution of difficulties in group  $i$  be given by  $F_{d_{T_i}}$  and consider group  $j > i$  with difficulty distribution  $F_{d_{T_j}}$ . A worker with skill  $s$  has  $\psi(s, W_i) = F_{d_{T_i}}(s)$  in group  $i$  and  $\psi(s, W_j) = F_{d_{T_j}}(s)$  in group  $j$ . By Lemma 3.8,  $d_{T_j} \succeq d_{T_i}$  and therefore  $F_{d_{T_i}}(s) \geq F_{d_{T_j}}(s)$ , and  $\psi(s, W_i)$  is monotonically decreasing in  $i$ , and since  $F$  is a distribution function, it is monotonically increasing in its argument  $s$ .  $\square$

**Proof of Theorem 4.5.** Denote by  $W_0$  the group that includes workers whose skills are below the minimum skill for participation  $\bar{s}$ . Let  $s_i = \{\min s | s \in W_i^*\}$ , so that  $s_1 = \bar{s}$ . Consider an entry fee  $q$  and payments  $p_1, \dots, p_k$  that satisfy the following system of equations

$$\begin{aligned} p_1 \psi(s_1, W_1^*) &= q \\ p_i \psi(s_{i+1}, W_i^*) &= p_{i+1} \psi(s_{i+1}, W_{i+1}^*) \quad i = 1, 2, \dots, k-1 \end{aligned}$$

The minimum payoff that can be obtained in  $W_1^*$  is equal to  $p_1 \psi(s_1, W_1^*) - q = 0$ . By Lemma 4.2,  $\psi(s', W_1^*) < \psi(s_1, W_1^*)$  for all  $s' < s_1 = \bar{s}$ , and therefore  $p_1 \psi(s', W_1^*) - q < 0$ . Similarly, for  $s > \bar{s}$ ,  $\psi(s, W_1^*) > \psi(\bar{s}, W_1^*)$  and  $p_1 \psi(s, W_1^*) - q > p_1 \psi(s_1, W_1^*) - q \geq 0$ . Consider a worker with skill

$s < s_{i+1}$  in  $W_i^*$ . This worker makes  $p_i\psi(s, W_i^*)$  and upon moving to  $W_{i+1}^*$  makes

$$\begin{aligned}
p_{i+1}\psi(s, W_{i+1}^*) &= p_i \frac{\psi(s_{i+1}, W_i^*)}{\psi(s_{i+1}, W_{i+1}^*)} \psi(s, W_{i+1}^*) \\
&= p_i \psi(s, W_i^*) \frac{\psi(s_{i+1}, W_i^*)}{\psi(s_{i+1}, W_{i+1}^*)} \frac{\psi(s, W_{i+1}^*)}{\psi(s, W_i^*)} \\
&= p_i \psi(s, W_i^*) \frac{\phi(s_{i+1})}{\phi(s)} \\
&\leq p_i \psi(s, W_i^*)
\end{aligned}$$

where the next to last equality is obtained from Assumption 4.4 and the fact that  $s < s_{i+1}$ . The inequality still holds if we replace  $i + 1$  with any  $j > i$ . A symmetrical argument shows that  $p_{i+1}\psi(s, W_{i+1}^*) \geq p_i\psi(s, W_i^*)$  for any  $s > s_{i+1}$ .  $\square$