Free-Riding and Free-Labor in Combinatorial Agency

Moshe Babaioff[1], Michal Feldman[2], and Noam Nisan[3]

[1] Microsoft Research - Silicon Valley
moshe@microsoft.com
[2] School of Business Administration, Hebrew University of Jerusalem
michal.feldman@huji.ac.il
[3] School of Computer Science, Hebrew University of Jerusalem
noam@cs.huji.ac.il

Abstract. This paper studies a setting where a principal needs to motivate teams of agents whose efforts lead to an outcome that stochastically depends on the combination of agents' actions, which are not directly observable by the principal. In [1] we suggest and study a basic "combinatorial agency" model for this setting. In this paper we expose a somewhat surprising phenomenon found in this setting: cases where the principal can gain by asking agents to *reduce* their effort level, even when this increased effort comes *for free*. This phenomenon cannot occur in a setting where the principal can observe the agents' actions, but we show that it can occur in the hidden-actions setting. We prove that for the family of technologies that exhibit "increasing returns to scale" this phenomenon cannot happen, and that in some sense this is a maximal family of technologies for which the phenomenon cannot occur. Finally, we relate our results to a basic question in production design in firms.

1 Introduction

Background: Combinatorial Agency

The well studied principal-agent problem deals with how a "principal" can motivate a rational "agent" to exert costly effort towards the welfare of the principal. The difficulty in this model is that the agent's action (i.e. whether he exerts effort or not) is unobservable by the principal and only the final outcome, which is probabilistic and also influenced by other factors, is observable. "Unobservable" here is meant in a wide sense that includes "not precisely measurable", "costly to determine", or "non-contractible" (meaning that it can not be upheld in "a court of law"). This problem is well studied in many contexts in classical economic theory and we refer the reader to introductory texts on economic theory for background (e.g. [11] Chapter 14). The solution is based on the observation that a properly designed contract, in which the payments are contingent upon the final outcome, can influence a rational agent to exert the required effort.

In [1] we initiated a general study of handling *combinations* of agents rather than a single agent. While much work was previously done on motivating teams of agents [8, 13, 9, 3], our emphasis in [1] was on dealing with the complex combinatorial structure of dependencies between agents' actions.

In the general model presented in [1], each of n agents has a set of possible *actions*, the combination of actions by the players results in some *outcome*, where this happens probabilistically. The main part of the specification of a problem in this model is a function that specifies this distribution for each n-tuple of agents' actions ("the technology"). Additionally, the problem specifies the principal's utility for each possible outcome, and for each agent, the agent's cost for each possible action. The principal motivates the agents by offering to each of them a *contract* that specifies a payment for each possible outcome of the whole project, with the goal of maximizing his expected net utility. Key here is that the actions of the players are non-observable and thus the contract cannot make the payments directly contingent on the actions of the players, but rather only on the outcome of the whole project.

Given a set of contracts, the agents will each optimize his own utility; i.e., will choose the action that maximizes his expected payment minus the cost of the action. Since the outcome depends on the actions of all players together, the agents are put in a game here and are assumed to reach a Nash Equilibrium (NE). The principal's problem is that of designing the *optimal contract*: i.e. the vector of contracts to the different agents that induce an equilibrium that will optimize his expected utility from the outcome minus his expected total payment.

We refer the reader to our earlier paper [1] for further motivation and details. Several other papers study different issues in the combinatorial agency model. Mixed strategies were studied in [2], while [6] studied random audits. In this paper we deal with a rather surprising (to us) phenomena that we have discovered in this model: the possible advantage of "throwing away" some free agents' effort (effort increase with no increase in cost).

Our Results

We focus on the case of two possible outcomes ("binary outcome"): either the project succeeds (generating value v to the principal) or fails (value 0). We generalize the model of [1] and allow for more than two actions for each agent. In this multiple-actions setting it is natural to assume that each agent has a linear order over his actions that corresponds to the actions' cost, and that more effort (according to the linear order) does not decrease the project's probability of success. An agent *wastes free labor* if he plays an action for which there exists another action with the same cost and is better according the linear order (as the project's success probability can increase with no increase in the agent's cost). A contract wastes free labor if at least one of the agents plays an action that wastes free labor. Is it possible that the principal's optimal contract will waste free labor? In the observable-actions case the principal can never gain by such a waste. Somewhat surprisingly, in the hidden-actions case we are able to present an example for which the principal can gain by wasting free-labor (Section 3). The fundamental reason for that is that free labor increases *free riding*, and reduces the motivation of other agents to exert effort.

To measure the principal's loss from using free labor we define the *Price of Free-Labor (POFL)*. POFL is defined to be the worse ratio (over all values v) between the principal's utility in the optimal contract and the best contract that must use all free labor (Section 4). Our goal is to characterize technologies for which free labor is never wasted. We show that for technologies that exhibit "increasing returns to scale (IRS)", free labor is never wasted (Section 5). Informally, the IRS property ensures that an increase in effort of all agents but one increases the marginal contribution of that agent due to an increase in his effort. An example for such a technology is the AND technology in which agents are perfect complements, each agent has a sub-task and the project succeeds only if all agents succeed in their sub-tasks. Thus, the IRS condition is sufficient to ensure that free labor is not wasted. Is it also necessary? It is easy to construct arbitrary technologies that do not exhibit IRS yet do not waste free labor.[4] Therefore we focus on a natural and large family of technologies: "structured technologies", and aim to prove a complementary result for a natural form of free labor in that family.

In a "structured technology" each agent has a sub-task to perform and the project's success is a deterministic Boolean function of the set of successful sub-tasks. The success of a sub-task executed by an agent is determined independently and stochastically as a function of his effort. If he exert no effort he bears no cost and the success probability is low, while if he exerts effort the cost is positive and the success probability is higher. Free labor is introduced by the principal's ability to remove agents altogether. Suppose that a given technology function specifies the underlying technological feasibility, but now the range of possible technologies that the principal can apriori choose among is given by the *sub-technologies* of the given one. I.e. the principal can apriori choose a subset of the agents and completely removing the others – in which case all the subtasks of the removed agents will surely fail.[5] Removed agents as well as agents that do not exert effort bear no cost. If an agent supply some positive success probability for his sub-task without any effort then removing the agent corresponds to a waste of free labor.

We ask the following question: for which technology functions a waste of free labor will never occur (independent of the exact parameters of the agents' success probabilities in their sub-tasks) ? We show that *any* structured technology will waste free labor for some choice of parameters, with a single exception: for the AND function, with any choice of parameters, free labor should always be used.

Finally, we draw a connection between this phenomenon and the much discussed question of process-based (PB) vs. function-based (FB) division of labor [10, 12, 14]: Suppose that a firm produces a product (task) that is composed of two parts (sub-tasks): A and B. Two workers (agents) $A1$ and $A2$ can each perform a sub-task of type A and two other workers $B1$ and $B2$ can each perform a sub-task of type B. One can consider two natural ways of organizing the production in the firm:

- **Function based:** Two "divisions", each consisting of one agent of *each* type. The project succeeds if at least one division is successful. The success here can be represented by ($A1$ AND $B1$) OR ($A2$ AND $B2$).
- **Process based:** Two "divisions", each consisting of two agents of the *same* type. In this case there is an "A division" (with $A1, A2$) and a "B division" (with $B1, B2$). The success here can be represented by ($A1$ OR $A2$) AND ($B1$ or $B2$).

Notice that the process-based organization is superior in terms of probability of success: the function-based alternative simply discards the possibility of success due to ($A1$ AND $B2$) OR ($A2$ AND $B1$). Yet, our results show that in an agent-based setting with hidden actions, the function-based approach may still be superior due to lower level of possible free-riding. We discuss the connection

[4] Actually, if each action has a different cost this holds trivially.

[5] One could assign different costs to the different sub-technologies, but we just look at the simplest question without any associated costs.

to the issue of free labor at Section 7. This result seems to be in line with the main intuitive reasons for choosing function-based organization (see [14]).

Due to lack of space we defer all proofs to the full version of the paper (which can be found on the authors' web sites).

2 Model and Preliminaries

Our main interest is in the simple "binary action, binary outcome" scenario where each agent has two possible actions (”exert effort” or ”shirk”) and there are two possible outcomes (”failure”, ”success”). In order to study phenomena in this setting, we will need to work within a more general model in which agents have general actions, but the outcome is still binary. This falls within the general framework of [1], and generalizes the "binary action" sub-model.

A principal employs a set of agents N of size n. Each agent $i \in N$ has a possible set of actions A_i, and a cost (effort) $c_i(a_i) \geq 0$ for each possible action $a_i \in A_i$ ($c_i : A_i \rightarrow \Re_+$). The actions of all players determine, in a probabilistic way, a "contractible" outcome, $o \in \{0, 1\}$, where the outcomes 0 and 1 denote project failure and success, respectively (binary-outcome). The outcome is determined according to a success function $t : A_1 \times \ldots \times A_n \rightarrow [0, 1]$, where $t(a_1, \ldots, a_n)$ denotes the probability of project success where players play with the action profile $a = (a_1, \ldots, a_n) \in A_1 \times \ldots \times A_n = A$. We use the notation $(t, c(\cdot))$ to denote a technology (a success function and a cost function for each agent).

The principal's value of a successful project is given by a scalar $v > 0$, where he gains no value from a project failure. The idea is that the actions of the players are unobservable, but the final outcome o is observed by him and others, and he may design enforceable contracts based on this outcome. We assume that the principal can pay the agents but not fine them (known as the *limited liability* constraint). The contract to agent i is thus given by a scalar value $p_i \geq 0$ that denotes the payment that i gets in case of project success. If the project fails, the agent gets no money.

Given this setting, the agents have been put in a game, where the utility of agent i under the profile of actions $a = (a_1, \ldots, a_n)$ is given by $u_i(a) = p_i \cdot t(a) - c_i(a_i)$. As usual, we denote by $a_{-i} \in A_{-i}$ the $(n-1)$-dimensional vector of the actions of all agents excluding agent i. i.e., $a_{-i} = (a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n)$. The agents will be assumed to reach Nash equilibrium, if such an equilibrium exists. The principal's likes to design the contracts p_i as to maximize his own expected utility $u(a, v) = t(a) \cdot (v - \sum_{i \in N} p_i)$, where the actions $a_1, \ldots, a_n$ are at Nash-equilibrium. In the case of multiple Nash equilibria, in our model we let the principal choose the desired one, and "suggest" it to the agents, thus focusing on the "best" Nash equilibrium.[6]

As we wish to concentrate on motivating agents, rather than on the coordination between agents, we assume that more effort by an agent always leads to a better probability of success. Formally, we assume that the actions of each agent are ordered according to the amount of effort, i.e. for any i there is a linear order $\succ_i$ on A_i that is consistent with the costs, $a_i \succ_i a'_i \Rightarrow c_i(a_i) \geq c_i(a'_i)$, and the

[6] A variant, which is similar in spirit to "strong implementation" in mechanism design, and discussed here, would be to take the worst Nash equilibrium, or even, stronger yet, to require that only a single equilibrium exists.

success function t is monotone non-decreasing, $\forall i \in N, \forall a_{-i} \in A_{-i}$ we have that $a_i \succ_i a'_i \Rightarrow t(a_i, a_{-i}) \geq t(a'_i, a_{-i})$. We also assume that $t(a) > 0$ for any $a \in A$. We denote $a_i \succeq_i a'_i$ if $a_i \succ_i a'_i$ or $a_i = a'_i$.

We start with the characterization of Nash equilibrium in this setting.

Observation 1 *The profile of actions $a \in A$ is a Nash equilibrium* [7] *under the payments $(p_1, p_2, \ldots, p_n)$ (agent i is paid $p_i \geq 0$ if the project succeeds and 0 if not) if and only if for any agent $i \in N$ the payment p_i satisfies* [8]

$$max_{a'_i \prec_i a_i} \frac{c_i(a_i) - c_i(a'_i)}{t_i(a_i, a_{-i}) - t_i(a'_i, a_{-i})} \leq p_i \leq min_{a'_i \succ_i a_i} \frac{c_i(a'_i) - c_i(a_i)}{t_i(a'_i, a_{-i}) - t_i(a_i, a_{-i})}$$

Moreover, to get the lowest cost payments that induce $a \in A$ as a Nash equilibrium, the lower bound weak inequality must hold as equality.

Given the technology and the value v of the principal from a successful project, the principal's goal is to maximize his utility, i.e. to determine a profile of actions $a \in A$, which gives the highest utility $u(a, v)$ in equilibrium, as calculated above. We call a profile of actions $a \in A$ that maximizes the principal's utility for the value v, an *optimal contract* for v. A simple but crucial observation, generalizing a similar one in [1], shows that the optimal contract exhibits some monotonicity properties in the value.

Lemma 1. (**Monotonicity lemma**)*: For any technology $(t, c(\cdot))$ the expected utility of the principal at the optimal contracts, the success probability of the optimal contracts, and the expected payment of the optimal contract, are all monotonically non-decreasing with the principal's value v.*

A similar lemma also holds in the observable-actions case, and is also showed there.

The principal can determine the action profile played by the agents in equilibrium by changing the agents' contracts (payment in case of success). As the value of v increases, the principal may change the profile of actions obtained at the equilibrium. It turn out that it is helpful to look at values v in which there is a change in the contracted action profile, and we call such points (values) "transition points".

Definition 1. *$v \in \Re_+$ is a* transition point *for technology $(t, c(\cdot))$ if for any $\epsilon_1 > 0$ and $\epsilon_2 > 0$, the set of optimal contracts for the value $v - \epsilon_1$ is different from the set of optimal contracts for the value $v + \epsilon_2$.*

Some of the results in this paper will be related to success functions for which the marginal contribution of any agent is non-decreasing in the effort of the other agents, we say that in such a case the function exhibits increasing returns to scale. Formally, for two action profiles $a, b \in A$ we denote $b \succeq a$ if for all j, $b_j \succeq_j a_j$.

[7] Note that, unlike in the Boolean action case studied in [1], it is possible that some profile of actions cannot be a Nash equilibrium with any payments, as no payments satisfy all these conditions.

[8] If t is not strictly monotone, it might be that for some a'_i it holds that $t_i(a_i, a_{-i}) = t_i(a'_i, a_{-i})$. In this case for $a \in A$ to be a NE, it must be the case that $c_i(a'_i) \geq c_i(a_i)$. In this case we interpret the above conditions as follows. The upper bound inequality holds for any $p_i \geq 0$ (as $c_i(a'_i) \geq c_i(a_i)$ for any $a'_i \succ_i a_i$). The lower bound inequality holds if for $a'_i \prec_i a_i$, $c_i(a'_i) = c_i(a_i)$.

Definition 2. *A technology success function t exhibits (weakly)* **increasing returns to scale (IRS)** *if for every i, and every* $b \succeq a$

$$t_i(b_i, b_{-i}) - t_i(a_i, b_{-i}) \geq t_i(b_i, a_{-i}) - t_i(a_i, a_{-i})$$

If a technology success function exhibits IRS we also say that the technology exhibits IRS.

2.1 Structured Technology Functions

Much of our focus will be on technology functions whose structure can be described easily as being derived from independent agent sub-tasks – called *structured technology functions*. This subclass will first give us some natural examples of technology functions, and will also provide a succinct and natural way to represent technology success functions.

In a structured technology function, each individual succeeds or fails in his own "sub-task" independently. The project's success or failure deterministically depends, maybe in a complex way, on the set of successful sub-tasks. Thus we will assume a monotone Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$ which denotes whether the project succeeds as a function of the success of the n agents' tasks.

A model with a structured technology success function is a special case of the *binary-outcome, binary-action model* [1]. In this model, the action space of each agent has two possible actions: 0 (shirk) and 1 (exert effort). The cost of shirking is 0, while the cost of exerting effort is $c_i > 0$.

A structure technology function t is defined by $t(a_1, \ldots, a_n)$ being the probability that $f(x_1, \ldots, x_n) = 1$ where the bits $x_1, \ldots, x_n$ are chosen according to the following distribution: if $a_i = 0$ then $x_i = 1$ with probability $\gamma_i \in [0, 1)$ (and $x_i = 0$ with probability $1 - \gamma_i$); otherwise, i.e. if $a_i = 1$, then $x_i = 1$ with probability $\delta_i > \gamma_i$ (and $x_i = 0$ with probability $1 - \delta_i$). We denote $x = (x_1, \ldots, x_n)$.

The question of the representation of the technology function is now reduced to that of representing the underlying monotone Boolean function f. In the most general case, the function f can be given by a general monotone Boolean circuit. An especially natural sub-class of functions in the structured technologies setting would be functions that can be represented as a *read-once network* – a graph with a given source and sink, where every edge is labeled by a different player. The project succeeds if the edges that belong to player's whose task succeeded form a path between the source and the sink[9].

A few simple examples should be in order here:

1. The "AND" technology: $f(x_1, \ldots, x_n)$ is the logical conjunction of x_i ($f(x) = \bigwedge_{i \in N} x_i$). Thus the project succeeds only if all agents succeed in their tasks. This is shown graphically as a read-once network in Figure 1(a).
2. The "OR" technology: $f(x_1, \ldots, x_n)$ is the logical disjunction of x_i ($f(x) = \bigvee_{i \in N} x_i$). Thus the project succeeds if at least one of the agents succeed in their tasks. This is shown graphically as a read-once network in Figure 1(b).
3. The "Or-of-Ands" (OOA) technology: $f(x_1, \ldots, x_n)$ is the logical disjunction of conjunctions. Thus the project succeeds if in at least one clause all agents succeed in their tasks. This is shown graphically as a read-once network in

[9] One may view this representation as directly corresponding to the project of delivering a message from the source to the sink in a real network of computers, with the edges being controlled by selfish agents.

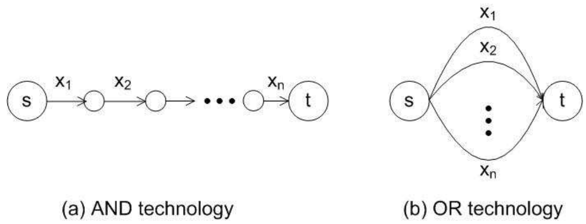

Fig. 1. Graphical representations of (a) AND and (b) OR technologies.

Figure 2(a) The simplest case is the one in which there are n_c clauses, each of length n_l; $n = n_c \cdot n_l$ (thus $f(x) = \bigvee_{j=1}^{n_c}(\bigwedge_{k=1}^{n_l} x_k^j)$).

4. The "And-of-Ors" (AOO) technology: $f(x_1, \ldots, x_n)$ is the logical conjunction of disjunctions. Thus the project succeeds if at least one agent from each disjunctive-form-clause succeeds in his tasks. This is shown graphically as a read-once network in Figure 2(b) The simplest case is the one in which there are n_c clauses of equal length n_l (thus $f(x) = \bigwedge_{j=1}^{n_l}(\bigvee_{k=1}^{n_c} x_k^j)$).

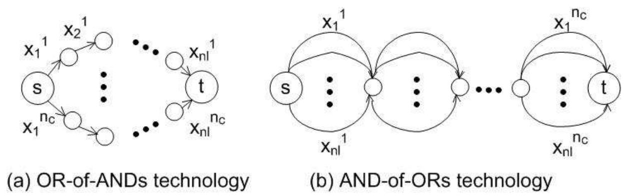

Fig. 2. Graphical representations of (a) OOA and (b) AOO technologies.

A success function t is called *anonymous* if it is symmetric with respect to the players. I.e. $t(a_1, \ldots, a_n)$ depends only on $\sum_i a_i$. A technology (t, c) is *anonymous* if t is anonymous and the cost c is identical to all agents (there exists a c such that for any agent i, $c_i = c$). Of the examples presented above, if we assume that the cost c is identical to all agents and that there exists a γ such that for any agent i, $\gamma_i = 1 - \delta_i = \gamma$, then the AND and OR technologies are anonymous (while for $n_l, n_c \geq 2$, the AOO and OOA technologies are not anonymous).

2.2 Sub-Technologies

The model of structured technologies (from [1]) presented above assumes that the technology function is exogenously given. In this paper we wish to ask how would the principal choose the technology function, had he had control over it. Obviously, this question is not interesting in its unrestricted form since the principal will always choose a technology in which all agents succeed with probability 1 with no cost. Yet, this question turns out to be interesting under reasonable restrictions, and it also connected to the issue of free labor. In this paper we suggest to study the "removal" model in which the principal is allowed to remove an agent, thus ensuring he will *certainly* fail in his sub-task (instead of succeeding with low probability γ_i). It seems natural to assume that if an agent is removed his cost of action is still 0.

In the "removal" model, we formally introduce the possibility of removing an agent as follows. We change the set of actions of any agent i to be $A_i = \{\emptyset, 0, 1\}$, with $1 \succ_i 0 \succ_i \emptyset$. The additional $\emptyset$ action is the action for which the agent does not participate ("removed"), and has 0 cost. If agent i is removed ($a_i = \emptyset$ and the cost to i is 0) his task will always fail, that is, $x_i = 1$ with probability 0. By removing the set of agents S the principal essentially fixes $x_i = 0$ for any $i \in S$, and this creates a restricted Boolean function $f^{|S=0}$ on the bits of the rest of the agents (S^c). We call such a restricted function a *sub-technology*.

In terms of the graphical representation as a read-once network as in Figures 1, this simply means that we allow the principal to erase, ex-ante, some of the edges. Equivalently, originally, choosing the right subset of agents to contract with was determining which agent i succeeds with probability δ_i, where the others succeeded with probability γ_i. Now, the principal can decide, within the group of non-contracted agents, a group that succeed with probability 0 rather than γ_i.

Note that by not removing an agent (and not contracting with him) the principal essentially get some "free labor" as with the same cost of 0 he get an increase in success probability. Observe that this model introduces free labor *only* for the lowest cost (0 cost) actions. This is so as for any strictly monotonic technology, it is impossible to induce Nash equilibrium in which an agent chooses a non-zero cost action that wastes his free labor (By Observation 1 if $a \in A$ is a Nash Equilibrium under p, and $a'_i \succ_i a_i$ with $t_i(a'_i, a_{-i}) > t_i(a_i, a_{-i})$ then it must be the case that $c_i(a'_i) > c_i(a_i)$.)

3 Free-Labor might be Costly: an Example

For the "removal" model the following example demonstrates that the principal might be better off not using all free labor. It shows that for some OR technology with two agents, for some values the principal is better off removing one agent (discarding his free labor) and contracting with the other.

Example 1. Consider an anonymous OR technology with two agents ($n = 2$), $c = 1$ and $\gamma = 1 - \delta = 0.2$. The optimal contract is obtained when the principal contracts with no agent for $0 \leq v \leq 3.65...$, with one agent for $3.65... \leq v \leq 118.75$, and with both agents for $v \geq 118.75$. However, if we allow the principal to ex-ante remove agents from the network, then, for example, when $v = 4$, the principal obtains a utility of more than 1.867 if the other agent does not participate, compared to a utility of 1.61, if the other agent does participate. It turns out that for $3.04... \leq v \leq 118.75$, the optimal contract is achieved when the principal contracts with a single agent and removes the second one.

Obviously, this example strikes us as counter-intuitive because there is unutilized "free-labor" – the principal prefers that the second agent will not participate despite the fact that he increases the probability of success with no additional cost. Yet, free labor increases *free riding* which results with a lower utility for the principal overall.

We note that the phenomena of costly free labor has also been identified in work on selfish routing [5, 7] and in hiring teams with no hidden-actions [4].

In what follows, we will formally define the concept of free-labor and study technologies in which free labor is always used and technologies in which it does not.

4 The Price of Free-Labor (POFL)

We next like to define a measure of the loss to the principal due to not be able to discard free labor. We begin by formally defining the meaning of wasting free labor.

Recall that our focus here is on motivating agents, rather than on the coordination between agents, thus, we are only interested in (weakly) monotone success functions. That is:

$$\forall i \in N, \forall a_{-i} \in A_{-i} \quad a_i \succ_i a'_i \Rightarrow t(a_i, a_{-i}) \geq t(a'_i, a_{-i})$$

Definition 3. *For a given agent i, action $a_i \in A_i$* wastes free-labor *if there exists an action $a'_i \in A_i$, such that $a'_i \succ_i a_i$ while $c(a'_i) = c(a_i)$.*

Note that if a_i wastes free labor then it is possible to (weakly) improve the project success by moving to a'_i with no increase in cost. The *contract $a \in A$ wastes free labor* if for some agent i, action a_i wastes free-labor. The two action profiles $a' \in A$ and $a \in A$ *correspond to the same costs* if for any agent i, $c(a'_i) = c(a_i)$.

Definition 4. *Given a technology $(t, c(\cdot))$ with agents' action spaces $A_1, \ldots, A_n$, the sub-technology that* utilizes all free-labor *is the technology $(t, c(\cdot))$ with agents' action spaces $A'_1, \ldots, A'_n$, obtained by restricting the action space for each agent i to the set of actions that does not waste free labor, that is*
$A'_i = \{a_i \in A_i | a_i$ does not waste free-labor$\}$.

The sub-technology that utilizes all free-labor restricts each agent to actions that do no waste free-labor. In the particular case of structured technologies with the "removal" model, this means that no agent is ever removed.

We are now ready to define the measure on the damage to the principal if he is restricted to the sub-technology that utilizes all free-labor.

Definition 5. *The* price of free-labor *$POFL(t, c(\cdot))$ of a technology $(t, c(\cdot))$ is defined as the ratio between the principal's utility under the optimal contract, and the principal's utility under the optimal contract in the case that he is restricted to the sub-technology that utilizes all free-labor.*

Formally, for a given value v, let $a^*(v) \in A_1 \times \ldots \times A_n = A$ be an optimal contract for v in A, and let $e^*(v) \in A'_1 \times \ldots \times A'_n = A'$ be an optimal contract for v in the sub-technology that utilizes all free-labor (with action spaces A' as defined in Definition 4). The price of free-labor is defined to be

$$POFL(t, c(\cdot)) = Sup_{v>0} \frac{u(a^*(v), v)}{u(e^*(v), v)}$$

By definition we need to find the supremum over a continuum of values. Yet, we are able to show that the POFL is obtained at one of finitely many important points, the transition points between optimal contracts.

Lemma 2. *For any technology $(t, c(\cdot))$ with finite action spaces ($|A_i| < \infty$ for all $i \in N$) the price of free-labor is obtained at a transition point (of either the original technology or the sub-technology with no waste of free-labor).*

Note that the lemma implies that the POFL is obtained, and that it is obtained at a finite positive value.

5 Technologies with Trivial POFL

In this section we consider general technologies and identify a set of technologies for which the POFL is 1, and no free-labor is ever wasted. We need one additional technical condition. A cost function $c_i : A_i \rightarrow \Re_+$ has *finite image* if there exists a number $K < \infty$ such that $|Image(c_i)| < K$. This means that there are only finitely different possible costs for all the actions[10]. A technology $(t, c(\cdot))$ has *finite cost image* if for any $i \in N$, the cost function $c_i(\cdot)$ has a finite image.

Theorem 1. *For any technology $(t, c(\cdot))$ that exhibits IRS and has finite image, the price of free-labor is 1. That is, for any value v, there exists an optimal contract (out of A) that does not waste any free labor.*

The theorem presents a family of technologies for which the price of free-labor is trivial. A natural question is at what extend this family is maximal. In the next section we show that for structured technologies it is maximal in a sense. Specifically, we show that for any function that is not AND (which ensures IRS), there are parameters such that the price is not trivial.

6 Sub-Technologies: Only *AND* Ensures Trivial POFL

In the previous section we have seen that technologies that exhibit IRS have trivial POFL. It is easy to show that AND technology exhibits IRS (even in the "removal" model).

Observation 2 *The AND technology exhibits IRS.*

From Theorem 1 we derive the following corollary.

Corollary 1. *The price of free-labor for AND technology in the "removal" model is trivial (1).*

For the "removal" model we can actually present a weaker condition than IRS that ensures that there exists an optimal contract that is non-excluding (all agents participate, none removed). The new condition requires that for any agent i, the increase in success probability when he changes his action from shirking to exerting effort, (weakly) increases when all removed agents are added (becoming participating agents). This condition (which is formally defined and discussed at the full version of the paper) is sufficient to ensure the existence of an optimal contract that is non-excluding. Which structured technologies satisfy this condition? A technology is determined by the Boolean success function and the parameters of the agents. We are interested in finding with functions ensures that the technology has trivial POFL for any choice of agents' parameters.

[10] The actions space A_i may still be infinite.

We show that the AND function is the *only* monotone function which *ensures* that $POFL$ is trivial, out of all technologies that are based on a Boolean function. That is, given any monotone Boolean function that is not an AND function, there exist values for γ_i and δ_i such that the POFL is greater than 1. This is a result of the fact that any non-AND function has an OR function "embedded" in it, and for OR, by Example 1, there exists a constant $\zeta > 1$ such that $POFL > \zeta$.

Lemma 3. *Let $f : \{0,1\}^n \rightarrow \{0,1\}$ for $n \geq 2$ be a monotone Boolean function that is not constant and not a conjunction of some subset of the input bits. Then there exists an assignment to all but two of the bits such that the restricted function is a disjunction of the two bits.*

Finally we present the main result of this section, showing that the AND function is the only function that ensures trivial POFL.

Corollary 2. *Let f be any monotone Boolean function that is not constant and not a conjunction of some subset of the input bits (an AND function). Then there exists a set of parameters $\{\gamma_i, \delta_i\}_{i \in N}$ such that the POFL of the structured technology with the above parameters (and identical cost $c = 1$) is greater than ζ, for some constant $\zeta > 1$.*

7 Process-based vs. Function-based Technologies

We now present another natural example that may be viewed as having implications on the controversy of *process-based* (PB) versus *function-based* (FB) team formation approaches [10, 14]. In the PB approach, each member of the team is in charge of a different stage in the production process of a single product, and the product is successfully produced only if all stages have succeeded in at least one team. In contrast, an FB team accommodates agents who all work on the same stage of the production process, and the product is successfully produced if there was at least one successful agent in each stage.

The PB and FB approaches can be represented by the OOA and AOO networks, respectively. Clearly, in the FB approach the product will be produced with higher probability (since in the PB approach, a failure of a single stage determines a failure of his team's product). However, in the hidden-actions case the principal sometimes favor PB teams due to the high level of free-riding in FB teams, as demonstrated in the following example.

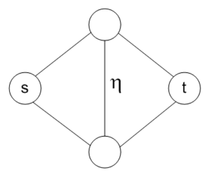

Fig. 3. A network that exhibits Braess-like paradox. As μ changes from 0 to 1, the network moves from Process-Based to Function-Based (and from OOA to AOO).

Example 2. Consider the network demonstrated in Figure 3, where the middle edge connects the middle points of the upper and lower paths, and has a success probability of η. Both the OAA and the AOO networks with $n_l = 2$ and $n_c = 2$ are special cases of this network with $\eta = 0$ and $\eta = 1$, respectively.

Clearly, the probability that a message sent from node s reaches node t is better when $\eta = 1$; namely, in the AOO network. This implies that for sufficiently large value of v, AOO is better for the principal. Nevertheless, due to the high level of free-riding in the AOO network compared to OOA, there exist values for which the optimal contract under the OOA network achieves a better utility than the AOO network. For example, in the case that for all i, $\gamma_i = 1 - \delta_i = 0.2$, $c_i = 1$, and $v = 110$, the optimal contract in the AOO network is to contract with one agent from each OR-component, which yields utility of 74.17..., while in the OOA network, the optimal utility level is 75.59.., which is achieved when contracting with all four agents.

One can think of the edge that succeeds with probability η as an edge that is controlled by an agent with cost of 0 to supply both $\eta = 0$ and $\eta = 1$. Our example above can be viewed as showing that the principal is better off wasting the free labor of that agent as for the presented paraments he prefers that agent to take the action with $\eta = 0$ (although the agent can supply $\eta = 1$ with no additional cost) as it decreases free riding by the other agents.

References

1. M. Babaioff, M. Feldman, and N. Nisan. Combinatorial Agency. In *ACM Conference on Electronic Commerce (EC'06)*, pages 18–28, 2006.
2. M. Babaioff, M. Feldman, and N. Nisan. Mixed strategies in combinatorial agency. In *Workshop on Internet and Network Economics (WINE)*, pages 353–364, 2006.
3. Y. K. Che and S. W. Yoo. Optimal Incentives in Teams. *American Economic Review*, 91:525–541, 2001.
4. N. Chen and A. R. Karlin. Cheap labor can be expensive. In *ACM-SIAM Symposium on Discrete Algorithms (SODA'07)*, pages 707–715, 2007.
5. R. Cole, Y. Dodis, and T. Roughgarten. Pricing Networks with Selfish Routing. In *Workshop on Economics of Peer-to-Peer Systems*, 2003.
6. R. Eidenbenz and S. Schmid. Combinatorial agency with audits. In *IEEE International Conference on Game Theory for Networks (GameNets)*, 2009.
7. E. Elkind. The Costs of Cheap Labor Are Hard to Measure: Edge Deletion and VCG Payments in Graphs. In *ACM Conference on Electronic Commerce*, 2005.
8. B. Holmstrom. Moral Hazard in Teams. *Bell Journal of Economics*, 13:324–340, 1982.
9. H. Itoh. Incentives to Help Multi-Agent Situations. *Econometrica*, 59:611–636, 1991.
10. T. W. Malone and S. A. Smith. Modeling the Performance of Organizational Structures. *Operations Research*, 36:421–436, 1988.
11. A. Mass-Colell, M. Whinston, and J. Green. *Microeconomic Theory.* Oxford University Press, 1995.
12. J. J. Rotemberg. Process- Versus Function-Based Hierarchies. *Journal of Economics and Management Strategy*, 8:453–487, 1999.
13. R. Strausz. Moral Hazard in Sequential Teams. *Departmental Working Paper. Free University of Berlin*, 1996.
14. E. Winter. Collocation and Incentives: Function vs. Process-Based Teams. In *Working Papaer*, 2005.