Executing Complex Cognitive Tasks: Prizes vs. Markets

Debrah Meloso* Peter Bossaerts† Jernej Čopič‡

November 21, 2006

Abstract

Execution of complex cognitive tasks is often analyzed as an exercise of information acquisition and belief updating. We challenge this view in the context of a non-incremental task, namely, the knapsack problem. First, we provide a theoretical argument why Bayesian updating makes little sense in this context. Second, we provide experimental evidence against the Bayesian approach by comparing the quality of problem solving under two treatments: prizes; markets. We find that Bayesian theory cannot make sense of the data: both systems work equally well, while trading is abundant in the market setup and prices are informative but noisy. The experimental data provide suggestions for a new theory of discovery of solutions in non-incremental tasks.

*California Institute of Technology. Corresponding author Debrah Meloso, debrah@caltech.edu. The authors are especially grateful to Bill Zame, and also thank Colin Camerer, Eduardo Faingold, and John Ledyard for helpful comments. Comments from participants of the Social and Information Science Laboratory (SISL) Workshop 2005, and the BDRM conference at UCLA 2006 are also gratefully acknowledged. The study was sponsored under NSF Grant #SES 0317715, and Caltech SISL grant.

†California Institute of Technology and CEPR

‡Cowles Foundation, Yale

1 Introduction

In the standard economic model, whatever is not known is treated as a random variable over which one holds a prior belief. Learning and discovery are modeled as an application of Bayes' law to construct a posterior.[1] Though this may be close to how we think when we try to guess the number of coins in a large closed jar, it is not very descriptive of how Picasso thought of "Guernica". Here we explore this latter form of discovery. We run experiments with this purpose, and also to demonstrate that consideration of the characteristics of the discovery problem at hand, matters in economically relevant situations.

The creation of a piece of art like Guernica is not readily achieved in the laboratory. This is why we resort to a cognitive task that captures the features of discovery that we care about – mainly, that it cannot readily be translated into Bayesian learning. The cognitive task we consider is the knapsack problem. An instance of the knapsack problem is given by a set of indivisible objects defined by a value and a weight parameter. The objective is to find the combination of objects that produces the largest sum of their values, given that the sum of their weights is within a given limit. The knapsack problem suits our objective because it is non-incremental and hard (we will return to these ideas below). We show that, as an implication of non-incrementality, the Bayesian paradigm does not provide a reasonable model of the search for a solution in an instance of the knapsack problem.

Our experiment then serves two purposes. First, it demonstrates that it is relevant to Economics that discovery may be non-incremental and, thus, existing models do not address the implications of this notion. Second, it points out in what ways non-incrementality differs from traditional approaches, thus opening the path for further experimental and theoretical exploration of the topic.

[1] Examples include probabilistic learning in markets (Grossman [1977]); or how individuals learn their own productivity (Jovanovic [1979]). A reduced form of this approach is common in the literature on Intellectual Discovery (see Arrow [62], and Gallini and Scotchmer [2002] for a survey). Commentary and a brief review of the classical study of technological change is given in Arrow [1994].

In our experiment we look at how institutions interact with the task of solving an instance of the knapsack problem. We consider two institutions which we refer to as treatments; a prize and a market treatment. For traditional, Bayesian problems, where discovery is treated as updating of information, the market treatment provides very little incentive for discovery (see, e.g., Grossman and Stiglitz [1980]). On the other hand, we calibrate the two treatments in such a way that the prize to one participant in the prize treatment equals what we pay to the market as a whole in the market treatment, thus providing a large individual incentive conducive to discovery. We find that markets do very well, according to whether somebody finds the solution, and how many persons find it. We are also able to relate the performance of markets with the trading behavior of participants and the pricing of securities.

The feature of the knapsack problem we care most about is non-incrementality. This means that many instances cannot be divided into a small number of parts that can easily be computed, finding the solution of the entire instance through computation of these parts. The knapsack problem is also hard; even though many instances can be solved using heuristics, it is typically not possible to determine before the problem has been solved, what heuristic will work for the instance at hand. Known exact solution algorithms that will always work, are laborious and the time they take rapidly increases with the size of the instance. Both features relate to the fact that no polynomial time algorithm has been found to solve every instance of the knapsack problem.[2] This feature is used in Section 2.3 to show that Bayesian updating is not reasonable in our setting. Loosely speaking, the argument is that in hard-enough instances of the knapsack problem, failure to compute the solution stems partly from the size of the set of possible solutions. This implies that if a subject were not able to compute the optimal solution, then he would also not be able to make any useful updating of his beliefs over the set

[2]There are other problems that satisfy these properties. We choose the knapsack problem because it is well suited for implementing the market treatment. One example of a non-incremental problem is that of finding the "right" linear regression, as pointed out in Aragones et al. [2005]. The findings in this work are also a good example of how intellectual discovery or learning has incremental (running the regression) as well as non-incremental (deciding what regression to run) parts. For an extensive discussion on the incremental and non-incremental features of research and science, see Kuhn [1962]

of possible solutions. Thus, instances of knapsack problems can be constructed such that either there will be no uncertainty (if subjects can compute the solution fast) or subjects will not be able to compute the solution, but the Bayesian paradigm is then also inadequate.

The fact that the knapsack problem is non-incremental affects the possibility of coordination and decentralized computation in the markets we implement. In particular, assigning to different participants the task of finding the value of one security leads to no gain at all in the ease of computation. This is so because our securities correspond to objects in the knapsack problem, each paying a positive dividend only if the corresponding object belongs to the value-maximizing knapsack. This splits up the instance in a very small number of pieces, in which case non-incrementality implies that each piece must be hard to compute (this depends on the instance, as we already remarked). In fact, for the instances we use, for all objects, the problem of determining whether an object belongs to the optimal knapsack or not, is equivalent to finding the optimal knapsack. Thus, even though the securities in the market provide a language that may serve to coordinate decentralized computation of the solution, they do not correspond to tasks for decentralization. It is important to note that distributed computation is not the only possible source of gains from interaction among individuals confronted with a cognitive task. This is suggested by literature in Psychology. For example Maciejovsky and Budescu [2005] reports an extensive experiment on learning in groups, where they find that groups are able to correct individual biases (mistakes) even when groups are composed only of individuals who display these biases when alone.[3]

Within the Bayesian, information aggregation framework, the securities in our market treatment are expected to lead to poor communication and aggregation. This is so because they do not allow for the expression of conditional beliefs like, for example, "conditional on object A being in the optimal knapsack, object C has probability 0.3

[3]The gains or losses from interaction have also been explored in Economics. Blinder and Morgan [2005] report that groups make better decisions than individuals conditional on equal amounts of information, while Cox and Haynes [2006] finds that groups fall victim to the winner's curse in auctions more often than individuals.

of also being there". Ledyard [2005] shows that this absence significantly decreases the informational quality of the market posterior. In a similar setup, Plott and Sunder [1988] shows that a complete set of contingent securities yields better information aggregation than a single security with multiple payoff levels.[4] We thus reiterate that our market treatment is not an attempt to implement the most efficient mechanism for *information aggregation*, but rather a tool to understand non-incremental discovery within the frame of economic institutions.

This paper points out that the description of intellectual discovery is relevant to our understanding of incentives to foster it. It also points out that cognition and computation may be the driving forces of trading and pricing behavior in complex information aggregation setups (e.g., Ledyard [2005]). As noted before, we also wish to better understand non-incremental discovery. To this avail, we construct two ad-hoc measures of difficulty of an instance, design our instances to have a range of difficulties according to these measures, and look for correlation between experimental results and the measures we propose. We find that the difficulty of the approximation algorithm – within a narrow class – that exactly solves an instance, is a good predictor of the number of individuals that solve the instance. In Section 2.2 we explain the two measures we propose and discuss why our results are surprising.

The remainder of the paper is organized as follows: In Section 2 we extensively describe the knapsack problem. In Subsection 2.3 we present an illuminating result about the knapsack problem being outside the scope of Bayesian updating. Section 3 presents and discusses the experimental design, with its different treatments. Section 4 presents the results. Results and future research are discussed in Section 5, which concludes.

[4]We say "similar" because the single security in Plott and Sunder [1988] is more akin to, for example, having only one security paying the value of the optimal knapsack in our setup.

2 The Knapsack Problem

The Knapsack Problem represents the model of intellectual discovery we wish to deal with. In this section we will carefully define the knapsack problem and some of its properties.

An instance of the knapsack problem is defined by a set N, containing n objects (we will refer to n as the *size* of the instance of the knapsack problem) with values $v = (v_1, \ldots, v_n)$, and weights $w = (w_1, \ldots, w_n)$; and a weight limit c. In the knapsack problem, the objective is to find the subset of N that yields the maximal sum of object values, given that the sum of weights does not exceed c. We will call this subset the *optimal knapsack*. The above objective can be stated mathematically as:

$$\begin{aligned} &\max \sum_{j=1}^{n} v_j \theta_j \\ &\text{s.t.} \ \sum_{j=1}^{n} w_j \theta_j \leq c \\ &\theta_j \in \{0, 1\}. \end{aligned}$$

An intuitive but incorrect approach to solving the knapsack problem is the *greedy algorithm*. According to this algorithm, the object with highest value to weight ratio, $\frac{v_j}{w_j}$ (the most *efficient* object) is added first, followed by the object with the second highest value to weight ratio, and so on, until the weight limit is hit. This procedure can yield an arbitrarily bad approximation to the true solution (measured in terms of the value of the knapsack).

There are no known simple algorithms that will solve every instance of the knapsack

problem. On the other hand, the related linear programming problem,

$$\begin{aligned} &\max \sum_{j=1}^{n} v_j \theta_j \\ &\text{s.t. } \sum_{j=1}^{n} w_j \theta_j \leq c \\ &\theta_j \in [0,1], \end{aligned}$$

has a straightforward solution, established by Dantzig [1957] that can always be found in linear time in the size of the instance. The solution of the above linear program is found by first applying the greedy algorithm. The first object that is left out according to the greedy algorithm, is then split so to fill the remainder weight.[5]

The following example serves to fixate notation and illustrate the working of the greedy algorithm:

Example 1 *Here is an instance of the knapsack problem:*

$$v = \begin{bmatrix} 5 \\ 7 \\ 11 \\ 20 \end{bmatrix}, w = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 6 \end{bmatrix}, c = 7. \tag{1}$$

There are four objects ($n = 4$), their values are in v, and their weights in w, the weight limit is $c = 7$.

Use of the greedy algorithm yields the following solution for the instance we consider here:

$$\hat{\theta} = (1, 1, 1, 0)\,.$$

Knapsack $\hat{\theta}$ contains the first three objects, has a total weight of 6*, and a total value of* 23*. However, $\hat{\theta}$ is not the* optimal *knapsack, which illustrates that the greedy algorithm*

[5]There are many references for more on the knapsack problem, its variants, bounds, and exact and approximate algorithms. Two excellent guides are Kellerer et al. [2004] and Martello and Toth [1990].

is not a correct solution method. Instead, the optimal *knapsack is given by*

$$\theta^* = (1, 0, 0, 1)\,,$$

containing objects 1 *and* 4*, and with a total weight of* 7*, and a total value of* 25*.*

2.1 NP-Completeness of the Knapsack Problem

A problem is said to be solvable in polynomial time if there exists an algorithm that solves every instance in a running time that is bounded above by a polynomial function of the instance size, n. A problem Q is $\mathcal{NP}$-complete ($\mathcal{NPC}$) if it is $\mathcal{NP}$ and $\mathcal{NP}$-hard. The former means that a proposed solution to Q can be *verified* in polynomial time, and the latter means that every problem in the class $\mathcal{NP}$ can be transformed into Q in polynomial time. No algorithm running in polynomial time in the instance size is known for any problem in the $\mathcal{NPC}$ class. If such an algorithm were found for one problem in $\mathcal{NPC}$, then this would immediately establish that all problems in $\mathcal{NPC}$ belong to $\mathcal{P}$, the set of problems that can be solved in polynomial time.

To us it is most relevant that no polynomial-time algorithm is known for the knapsack problem. This means that no fast (clever) algorithm can be used to solve every instance. In particular, if a participant in our experiment solves one instance, this does not mean that he has *learned* to solve the problem and will henceforth face no more difficulties.[6] However, it does not mean that every instance of the knapsack problem is difficult. First, if the instance is sufficiently small, the crudest exact solution algorithm will still solve it in a fairly small number of computations (e.g., the largest instance used in our experiment has 12 objects, which implies that the value of only 4096 knapsacks must be computed and compared to find the solution). Second, and more important, certain instances can

[6]One implication of this is that the periods in our experiment are independent instances of discovery, in the sense that having solved one instance does not guarantee a participant will solve every other instance. Still, participants' skills may improve. For example, a participant that starts by using the greedy algorithm, will quickly learn it is not a very good approach, or the clumsy exact algorithm that explores every possible knapsack may be replaced with a more efficient branch-and-bound algorithm.

be exactly solved with fast approximate algorithms, e.g. the greedy algorithm. This is the subject of the next subsection.

2.2 Measures of Difficulty of Instances

There is no straightforward measure of the difficulty of an instance of the knapsack problem. For example, even though in the *worst case* an instance of size $n+1$ may take twice as much time as one of size n, this pattern needn't be satisfied by two arbitrary instances of size $n+1$ and n, respectively. We propose two measures of difficulty, and revisit them when we present our choice of instances for the experiments we ran.

A first intuitive measure is a parameter of *input size*. It is the product of the base-two logarithm of the knapsack capacity times the size of the instance. The base-two logarithm of the knapsack capacity is a proxy for the binary representation of the instance parameters, which in turn represents the amount of storage and information necessary in each step of computation. This is then multiplied by the size (the number of objects) of the instance. In this measure, two instances with equal capacity and size are equally difficult. We propose a second measure which captures less obvious characteristics of an instance.

This second measure relates to the question of heuristic solvability of an instance.[7] A simple approximation algorithm for the knapsack problem may solve an instance exactly. For a class of approximation algorithms described below, we will consider an instance that can be exactly solved with an approximation algorithm in this class to be easier than another instance that cannot be exactly solved with it.

The simplest approximation algorithm for the knapsack problem is the *greedy* procedure, which consists of filling the knapsack in efficiency order - i.e. starting with the objects that have a higher $\frac{v_i}{w_i}$ ratio - until the weight limit is reached. This heuristic is part of a family of approximation algorithms known as the Sahni approximation scheme (Sahni [1975]).[8] The Sahni scheme is parameterized by a number k, referring to the

[7] We use the terms *heuristic* and *approximation algorithm* interchangeably.

[8] Approximation schemes have more desirable properties than approximation algorithms. In the case

specific algorithm, which we will call a *Sahni algorithm of size k*.

A Sahni algorithm of size k looks at all subsets of N (the set of all objects that are considered to enter the knapsack) of cardinality k or less. For each subset, it computes the residual weight in the knapsack after subtracting the weight of the subset, and fills this residual with the remaining objects using the greedy procedure (if no set of k objects fits in the knapsack, then the exact optimal solution is found by the algorithm). The value of all knapsacks constructed in this way is compared, and the one with highest value becomes the approximate solution given by the algorithm. Clearly, the greedy algorithm is a Sahni algorithm of size 0. The Sahni algorithm of size 1 uses the greedy algorithm for every subset of size $n-1$ (there are n such subsets) to fill the capacity that remains after isolating one object. Though the complexity of the greedy algorithm it runs is smaller, it has to run it n times. The complexity added is thus of order n. This is the case for every increase from k to $k+1$. Although a Sahni algorithm of size k is not as straightforward as the greedy algorithm, it is still a very simple heuristic.

Definition 1 *We say that an instance has Sahni-difficulty level k if it can be exactly solved with a Sahni algorithm of size k, but not with a Sahni algorithm of size $k-1$. The higher the k associated to an instance, the harder the instance.*

While specific instances can be solved using simple heuristics, it is impossible to determine a-priori what simple heuristic to use. Given an instance, if one knew a-priori what Sahni algorithm to use in order to find the solution, then the Sahni-k would be a precise measure of difficulty. However, only after the instance has been solved can one know what Sahni algorithm to use. Thus, there is no reason to believe that Sahni-k will be a good predictor of whether a person can solve an instance or not.

In Table 2 we return to the two measures of difficulty and apply them to the instances used in our experimental sessions.

---

of the Sahni scheme, different algorithms from the family can be chosen depending on the desired performance level.

2.3 The Knapsack Problem: A Hard Problem

We wish to draw a parallel between non-incremental intellectual discovery and finding the solution to a hard problem. The knapsack problem allows for instances that are hard according to our criterion because of the following properties:

1. There is a parameter (in our case the size, n), such that an increase of this parameter by 1 unit can in some instances increase the computational complexity of finding the exact solution by an order of magnitude. This allows us to construct problems that are very hard.

2. It is not possible to establish a-priori what approximation algorithm may solve the instance exactly or with a fixed precision. It is also not possible to split the problem in a *small number* of *easy* parts such that solving each of these parts separately will give the right answer.

The above two properties make the problem *non-incremental*. Additionally, the knapsack problem satisfies the following property, which allows us to prove the main result of this section:

3. When n is increased by 1 unit, the complexity of computation of the value function of the instance at each point in the solution space does not increase very much.

When it comes to identifying a hard problem, we just care that we can find *an instance* of the knapsack problem which is hard enough for our purposes and which satisfies property 3. This contrasts the worst-case analysis that was mentioned in Section 2.1, and which is often used in computer science.

The reason we care about properties 1, 2, and 3 is that these properties allow us to make a clear-cut distinction between our setup and rational-expectations models. In rough strokes, a problem that satisfies properties 1, 2, and 3, will either be easy enough to be computable by agents, or it will be too hard to be computable. In the latter case, it will *also be too hard to update any prior belief over the solution set.* Hence, in our

model, there is either no uncertainty, or agents cannot possibly update any prior over the solution set, so that any model involving updating of beliefs cannot be the right one to model this situation. This provides the basis for testing the hypothesis of whether we posed "hard enough instances" of the knapsack problem to our subjects. We now make our argument precise.

Take the following simple Bayesian model to describe solving the knapsack problem as a situation with uncertainty in which agents get new signals and are required to update their beliefs over how likely each point in the solution set is to be the optimum. Call this model the *Uncertainty model.*

Let $\Theta(n)$ be the set of *possible* solutions of an instance of the knapsack problem with n objects. Thus, $\Theta(n)$ is in the case of the knapsack problem the set 2^n. Each agent $i \in I$ (I is some non-empty index set) has a prior belief B_i over $\Theta(n)$ describing for each $\theta \in \Theta(n)$ how likely it is that θ is the solution to the specific instance of the problem. Assume that $B_i(\theta) > 0, \forall \theta \in \Theta(N)$. This is a sensible assumption since the complexity of the problem is on one hand determined by the number of *possible* solutions, so that points with 0 probability do not add any complexity.[9] Now assume that i observes a realization of a random variable $\sigma \in \{0, 1\}$, and there is an updating mapping $B_i(\theta \mid \sigma) : \{0, 1\} \times B_i(\theta) \to [0, 1]$, specifying how the probability of each state changes depending on the realization of σ. We assume that this updating does not reduce the complexity of the problem by an order of magnitude. If this were not the case, and if we want to think of σ as some accessible and simple device available to the agent, then he could just use n such devices and figure out the precise solution in polynomial time. That such a scheme is not readily available to the agent is therefore an assumption consistent with the problem being $\mathcal{NP}$.

[9]This assumption is additionally justified as follows. For example, if all objects have a small w_j and the weight limit is large, one expects an individual to immediately rule out all knapsacks with only one object. One should think of this as a signal, as this can only be established once an agent is presented with the instance; the updated distribution gives probability zero to all knapsacks with only one object. This signal satisfies the assumption we make about signals. The assumption on prior belief can also be replaced with the alternative assumption that the support of $B_i(\theta)$ be of cardinality larger than 2^{n-1}, or in fact with any cardinality that is exponential in n.

Finally, we assume that the complexity of computation of the updated probability $B_i(\theta \mid \sigma)$ is of the same order of magnitude as the computation of the value function $V(.)$ of the instance of the problem at each θ. This assumption is sensible precisely because of 1-3. Namely, suppose that we increase n by 1 unit to n'. Then the set $\Theta(n')$ increases in size by a factor of 2 relative to $\Theta(n)$, or an *order of magnitude*. However, computing $V(\theta)$ either remains unchanged in complexity for $\theta \in \Theta(n) \cap \Theta(n')$, or it at most increases by an additive constant for $\theta \in \Theta(n') \setminus \Theta(n)$.[10] Formally, for each $\theta \in \Theta(n)$ denote by $\eta_{i,B}(\theta, n, \sigma)$ the number of operations needed to compute $B_i(\theta \mid \sigma)$, and denote by $\eta_V(\theta, n)$ the number of operations needed to compute $V(\omega, n)$. To avoid any confusion with computer science, we state our assumption as follows:[11]

$$\frac{\eta_{i,B}(\theta, n, \sigma)}{\eta_V(\theta, n)} \geq \underline{m} > 0, \forall \theta, \forall n \leq \bar{n} < \infty.$$

Here, $\underline{m}$ is a positive constant, *close to* 1, and $\bar{n}$ is some potentially very large constant. It is clear that we can restrict ourselves to finite problems, as when $\bar{n}$ is for instance 2 billions, the solution set of the knapsack problem with so many objects is much larger than the number of atoms in the universe, and there is clearly no sense in theorizing about problems that complex.

To make things more concrete and directly related to our problem, think of σ as a comparison of the values of two different knapsacks. Such a σ is a very natural one, and it clearly satisfies the above assumptions.

Proposition 2 *In the* Uncertainty Model*, assume that* $\underline{m} \geq 1$*. Then, if agent* i *is unable to compute the solution to the instance of the knapsack problem,* $\theta^* \in \Theta(n)$*, then it is also for* i *impossible to compute* $B_i(. \mid \sigma)$*.*

[10] Computing the value of the knapsack is just adding up the values of all the objects, and adding all the weights to verify that the weight limit is not surpassed. Clearly, if we have one new object there is at most one new weight and one new value to add, so that computational complexity can at most increase by 2 operations. Moreover, even at most points $\theta \in \Theta(n') \setminus \Theta(n)$ computing the value doesn't increase in complexity at all.

[11] We could state our assumption in computer-science language as $\eta_V(\theta, n) = O\left(\eta_{i,B}(\theta, n, \sigma)\right)$, but such a statement is one of asymptotic properties where n becomes large. Our statement describes the situation for finite n.

Proof. The proof is obvious: if i were able to compute $B_i(\theta \mid \sigma)$, for all $\theta \in \Theta(n)$, then he could have just as well computed $V(\theta)$ at each $\theta \in \Theta(N)$ in the first place, and then keeping always track of just the maximal θ he would find the precise solution θ^* so that there could be no uncertainty.

We comment that $\underline{m} \geq 1$ is just a convenient device to make our statement particularly simple, and the mapping $B_i(. \mid \sigma)$ is abstract and might be very complex anyway. Even if $\underline{m}$ were equal to $\frac{1}{n}$,[12] a similar statement would apply: the only difference is that then, if the agent were unable to compute the solution for a specific n, he surely would be unable to compute the posterior when n is increased by 1 unit.

3 Experimental Design

The instruction sets can be found by visiting the following web pages (which were part of the web site that was used in the experiments):

http://clef.caltech.edu/exp/eTradeLab13a/instructions.html and
http://clef.caltech.edu/exp/knapsack

The first web-page corresponds to the original experiment, of which we ran four sessions. We will refer to this setup as setup ω. The second web-page corresponds to the version with equal time for both treatments and a variation in the prize setup that is explained below. We will call this setup, setup ν.

3.1 Timing of an Experimental Session

In an *experimental session*, a group of participants are asked to solve several instances of the knapsack problem, cast in terms of *shipping problems*. All instances have a unique

[12] This is the worst things could possibly go for us, as the complexity of computing $B_i(. \mid \sigma; n)$ is at least a constant and the complexity of computing $V(\theta, n)$ is at most of the order n. The constant in the numerator is 1 because all values of the parameters in an instance have the same base-two log.

solution. Each experimental session is divided in ten *periods*, P = 1,...,10. Periods 1 and 2 are *practice* periods; the remaining ones (3 to 10) are *earnings* periods. The explanation that follows applies to both setups – ω and ν – except when specified.

In setup ω, prizes, security prices, cash, and period earnings are expressed in a fictitious currency called *Franc* (F), with an exchange rate of $F1$ = \$0.01. In setup ν, the dollar value of all parameters is preserved, but expressed in dollars. In both setups, earnings from practice periods do not count towards total experiment payoffs. Earnings from earnings periods are accumulated, converted to dollars at a pre-announced exchange rate, and a \$5 sign-up reward is added. In each earnings period, participants are given an instance of the knapsack problem and a specific payment rule. Under the *Market* treatment, participants are paid based on their final holdings of securities whose dividends depend on the optimal solution, as well as their final cash position. Under the *Prize* treatment, they are paid directly for solving the instance of the knapsack problem. All odd-numbered earnings periods are *Market* periods; all even-numbered earnings periods are *Prize* periods.

At the beginning of a period, participants receive a sheet of paper with a full description of an instance of the knapsack problem and an area to mark their proposed solution to the problem. This sheet will be referred to as *Answer Sheet* (see *Shipping Problem Presentation* in the Instructions Sets). Answer Sheets are collected at the end of a Prize period; in a *Market* period, the sheets are collected 30 seconds before the end. In other words, the Answer Sheets are handed in by participants, before security payoffs are revealed, and hence, before the optimal solution, which is implicit in the security payoffs, becomes known.

An entire experimental session lasts approximately 2 1/2 hours, including one half hour for instructions. In setup ω, each prize period lasts up to 7 minutes, and each market period lasts 15 minutes. In setup ν, each period lasts 10 minutes, for both treatments.

3.2 Knapsack Instances

Table 1 lists the instances of the knapsack that were used in the experiment. There are eight different instances; four have n (number of items) equal to 10, and four have $n = 12$.

In each experimental session, two different instances of size $n = 10$ are assigned under the Market treatment, and two different ones are assigned under the Prize treatment. Analogously for instances of size 12. To ensure that all instances are solved under both setups, we run two types of sessions. In experimental sessions of type a, instances I, V, VII, and VIII (see Table 1) are solved under the Market treatment; while instances II, III, IV, and VI are solved under the Prize treatment. In experimental sessions of type b, this organization is reversed: an instance that is solved under the Market treatment in type a is solved under the Prize treatment in type b, and *vice versa*.

3.3 Market Treatment

In the Market treatment, participants are paid through securities. There are as many securities as there are items in the knapsack instance. Each security corresponds to an item. At the end of the period, a security pays \$1 if the corresponding object is in the optimal knapsack; otherwise the security pays nothing.

Participants start with an initial endowment of 5 units of each security, and \$4 cash (this is expressed as $F400$ in setup ω). To provide liquidity, participants are endowed with more securities than the desired average payoff; this is why a *"loan repayment"* is subtracted from total period earnings. In type a experiments the loan repayment is \$23.75; in type b experiments the loan repayment is \$32.50. Because of the loan, participants may lose money. Losses are subtracted from cumulative earnings. If cumulative earnings are negative, the participant is paid only the sign-up reward of \$5, plus earnings from prize periods.

In setup ω, trading is done through a continuous electronic open book system called eTradeLab. Details of this trading interface can be found in the instructions (see afore-

Problem & capacity		Objects											
		A	B	C	D	E	F	G	H	I	J	K	L
I	p	500	350	505	505	640	435	465	50	220	170		
	w	750	406	564	595	803	489	641	177	330	252		
$c = 1900$	θ^*	0	0	1	1	0	1	0	0	0	1		
II	p	300	350	400	450	47	20	8	70	5	5		
	w	205	252	352	447	114	50	28	251	19	20		
$c = 1044$	θ^*	1	0	1	1	0	0	0	0	1	1		
III	p	15	14	3	3	10	9	28	28	31	25	24	1
	w	129	144	77	77	66	60	184	184	229	184	219	72
$c = 850$	θ^*	0	0	0	0	1	0	1	1	1	1	0	0
IV	p	37	72	106	32	45	71	23	44	85	62		
	w	50	820	700	46	220	530	107	180	435	360		
$c = 1500$	θ^*	1	0	0	1	1	0	1	1	1	1		
V	p	2	3	4	5	6	9	8	7	6	5	8	9
	w	3	4	6	3	5	13	6	9	2	4	7	7
$c = 14$	θ^*	0	0	0	1	1	0	0	0	1	1	0	0
VI	p	107	35	120	206	88	34	28	110	88	101	74	53
	w	599	196	670	1204	502	202	145	600	453	601	404	299
$c = 3800$	θ^*	1	1	0	0	1	0	1	1	1	1	1	1
VII	p	201	84	113	303	227	251	129	147	86	127	144	167
	w	192	80	106	288	212	240	121	140	82	120	137	160
$c = 1300$	θ^*	1	0	1	1	1	0	0	1	1	1	0	1
VIII	p	31	141	46	30	74	105	119	160	59	71		
	w	21	97	32	21	52	75	86	116	43	54		
$c = 265$	θ^*	0	1	0	0	1	0	0	1	0	0		

Table 1: Instances of the knapsack problem used in experimental sessions. Objects had common names, not letters. The letters used in the table stand for: A=Anderson, B=Brown, C=Cole, D=Darwin, E=Evans, F=Foster, G=Green, H=Hamilton, I=Ives, J=Jensen, K=Keaton, L=Lee.

mentioned web-page). The system tracks all offers (bids, asks) and transactions, time stamped to the second.

In setup ν, an identical market mechanism is used for trading. However, the interface is different, and orders are submitted by "clicking" on price labels, instead of manually entering a price and quantity. The software for this setup is jMarkets, and is further described in the experiment instructions for setup ν.

While Answer Sheets are collected 30 seconds before the end of the period (i.e., before security payoffs are revealed), participants are not paid depending on their marks on these sheets. Participant earnings for Market periods come exclusively from trading and from the final payoffs of the securities.

3.4 Prize Treatment

The prize treatment in setup ω is substantially different from that in setup ν. We describe them separately, under the names *prize treatment* ω and *prize treatment* ν, respectively.

Prize treatment ω

At any moment during a Prize period, a participant can submit his/her Answer Sheet with the proposed solution by raising his/her hand. The experimenter then checks the marks on the Answer Sheet and announces a winner if the solution is correct. The period ends when all participants have submitted their Answer Sheets, or the time limit has been reached (recall that the time limit is 7 minutes), whichever occurs first. Participants do not get a second chance: once an Answer Sheet is turned in, participants cannot change it. The Prize is set at $F6,600$, i.e., \$66. Ties are resolved by dividing the Prize equally among the winners.

Prize treatment ν

All participants are in constant live communication with the experimenter, through a chat program called Skype. A username and password are assigned to each participant for the duration of the experiment. These are secret, to ensure that participants communicate only with the experimenter. Participants have one opportunity to submit

		Input Size Proxy ($n \log_2 c$)				
		$40-50$	$80-90$	$100-120$	$120-130$	$140-150$
Sahni-difficulty level	0			IV		
	1	V		I		
	2		VIII	III		
	3			II		VI
	6				VII	

Table 2: Difficulty of Knapsack Problem instances used in experiments. The difficulty is measured by the proxy of *input size* (columns) and the Sahni difficulty level described in Section 2.2.

their proposed solution to the instance at hand. The submission is made over the chat program, using a code of ones and zeros, to indicate objects that are in and objects that are not in the optimal knapsack according to their proposed solution. Participants have a time limit of 10 minutes to make submissions. After these 10 minutes elapse, all answer sheets are picked up, after which the correct answer is announced together with the time stamp of the winning submission. The winner is the first participant to submit the correct solution. Unlike in setup ω, in prize treatment ν, the fact that there is a winner is not revealed until the end of the period.

Note that in both setups the Prize is approximately the same as the *aggregate* payment in Market periods (security dividends and cash, minus loans). Notice the difference between a Prize period and a Market period: in a Prize period, only the winner(s) is (are) paid; in a Market period, everyone is paid, through securities and cash.

3.5 Discussion of Experimental Design

3.5.1 Difficulty of Knapsack Problem Instances

The instances we consider have different levels of difficulty according to the two measures given in Subsection 2.2. In Table 2 below, the rows correspond to Sahni difficulty levels, k, while the columns correspond to the described parameter of *input size*. In Table 2, the Northwest corner corresponds to lower complexity. Complexity increases in the Southeast direction.

3.5.2 All-or-nothing Problem

One property of the knapsack problem is that a knapsack that is close to optimal – say, the feasible knapsack with the second-highest value – need not look in any way similar to the optimal knapsack. The latter may be composed of an entirely different set of objects. We want to preserve this property in our experimental treatments. It is to this avail that compensation is never tied to the value of the knapsack that a participant proposes as the optimal knapsack.

It is obvious that compensation in the prize treatment is given only for the exact solution of the problem. We wish to emphasize that this is also the case in the market treatment. Only securities corresponding to objects that belong to the optimal knapsack pay positive dividends. For example, if the knapsack with the second-highest value has an entirely different composition from the optimal knapsack, a participant betting on the objects composing the former will make negative earnings. In Section **??** we further discuss the payoff structure in our experiment.

3.5.3 Securities Do Not Split the Problem

As discussed in Section 2.3, instances of the knapsack problem satisfy property 2, i.e., the solution cannot be found by splitting the problem in a small number of easily-solvable parts. In particular, computing whether an object belongs to the optimal knapsack or not is as hard as solving the instance.[13] Therefore, computation of the solution cannot be decentralized in an obvious fashion by each subject focusing on a specific object.

One example of how the process of finding the solution to an instance can be distributed, is by splitting the set of all knapsacks in groups for comparison. Each person gets a fraction of all knapsacks to make value comparisons. The most valuable feasi-

[13]There are exceptions, corresponding to cases where it is obvious if a certain object belongs or not to the optimal knapsack. For example, if one object weighs more than the sum of all remaining objects and its value is below the maximum value of the remaining objects, then this object clearly does not belong to the optimal knapsack. This cannot be extended to sets of more than one object - if a pair weighs more than the sum of the remainder and has a lower value than the maximal object in the remainder, this does not mean that both objects in the pair are not in the optimal knapsack. We make sure that none of the instances we consider falls in the class where the above described exception applies.

ble knapsack of each group is then compared to the winner of other groups, and the most valuable feasible knapsack can thus be found.[14] It is not clear how this form of distribution can be encoded in the securities in the market treatment.

The securities in the market are the minimal set of securities necessary to express the solution of every instance. They provide a binary code to represent the solution, and may serve as a language to transmit it. However, the securities do not distribute or simplify the problem in any way.

4 Results

We report results from four ω experimental sessions and two ν sessions. Sessions are identified by the date when they were run (yymmdd), the setup (ω or ν), and the experiment type (a or b). Our six sessions are: $\omega 040809a$, $\omega 040929b$, $\omega 041202a$, $\omega 041215b$, $\nu 061112a$, and $\nu 061116b$. All sessions were run in the Social Sciences Experimental Laboratory at Caltech, with Caltech students (undergraduate, graduate, and summer visiting students). Session $\omega 040929b$ had seventeen participants, session $\nu 061112a$ had 14, and session $\nu 061116b$ had sixteen participants; all other sessions had fifteen participants. In sessions of setup ω, the seven-minute time limit for Prize periods was never binding: participants always turned in their answer sheets early. Total earnings before sign-up reward fluctuated between \$0 and \$163. Mean payment amounted to \$31; the median payment was \$20.

Our results are taken from two sources. We have all the information on trades and prices that is collected during market periods, plus we have the answer sheets of all participants for both treatments. These answer sheets are returned to the experimenter for no compensation. Participants are told at the beginning of the earning periods that they must give the answer sheets back to the experimenter, but their answers are completely irrelevant for payment, and participants understand this. Still, we are able

[14] If such an assignment of groups of knapsacks to subjects were exogenously given by a social planner, then this would be the team problem studied by Marschak and Radner [1972].

to collect many answer sheets that are carefully filled out.[15] The number of answer sheets discarded because they are blank or incomprehensible is small, and similar across treatments for setup ω. For setup ν, the number of discarded answer sheets in the prize treatment is very small (approximately 3% of all answer sheets) while the number of discarded sheets (mainly because they are left blank) in the market treatment is large (approximately 30% of all answer sheets).

Result Zero: Instances are Hard. Before we move on to analyze the performance of markets vis-a-vis the prize treatment, we make the point that instances are *hard enough*, as defined in Section 2.3. Proposition 2 says that it is only sensible to use Bayesian updating and thus the standard information aggregation paradigm, if instances are easy enough. If problems are easy enough, prices must immediately collapse to one or zero, depending on whether the object belongs or not to the optimal knapsack.

Direct evidence that problems are hard enough is the fact that only a fraction of participants solve them in both treatments. This is true for all instances (see Table 3). Indirect evidence is the fact that prices never collapse to zero or one.

We can thus pursue our objective of understanding intellectual discovery outside the Bayesian learning framework. We present results from each setup in a separate section. The data for both setups demonstrate that markets do generate the solution to the instances of the knapsack problem, that trade is abundant and prices are informative. Results from setup ω show a very strong correlation between the Sahni difficulty ranking of an instance and the number of participants that solve it. This result disappears in setup ν.

[15] Answer sheets are not always easy to comprehend, in which case we discard them. One example of behavior that leads us to think that answer sheets are carefully filled out is that as we pick up the answers, they often take two or three extra seconds to make sure they hand us the answer they desire. Another example is that answer sheets often have several iterated answers that are scratched, with an arrow or marker pointing to the "surviving" answer.

		Instance							
		I	II	III	IV	V	VI	VII	VIII
Percentage correct answers[a]	Market	20.00	3.13	31.25	62.5	60.00	15.63	0.00	26.67
	Prize	18.75	10.00	20.00	33.33	34.38	3.33	0.00	12.50
R_{rw}	Market	56.9	22.1	75.0	194.9	154.7	51.9	28.7	28.9
	Prize	55.0	33.3	53.3	166.7	50.7	43.7	26.0	27.6

[a] Percentage of total number of participants that marked the correct answer on their answer sheets.

Table 3: Summary of data from participants' answer sheets for setup ω.

4.1 Market Performance

Every instance that is solved under the prize treatment is also solved under the market treatment.

Table 3 shows the fraction of participants that find the correct solution for every instance of the knapsack problem in setup ω. Results are split according to the treatment. There is a big variance across instances in the number of correct solutions and instances that are more frequently solved under the market treatment are also more frequently solved under the prize treatment. Table 3 shows that all instances except instance VII are solved by a significant fraction of participants in both treatments. Table 4 shows the fraction of correct solutions for each instance in setup ν. Percentages are taken with respect to the total number of participants, without adjusting for discarded answer sheets. It is no longer the case that there are more correct answers under the market treatment than under the prize treatment. Still, the solution is always found in the market.

Figures 1 to 4 display the fraction of correct solutions and the number of choices of each object (correctly or incorrectly chosen) taken from participants' answer sheets. Results for setup ν are in Figures 5 to 8. This is simply a graphical representation of the results in 3. It is also interesting to notice in these plots that whenever a large proportion of participants agrees about an object being in the optimal knapsack, this object truly is. Participants are not misled in the markets.

		Instance							
		I	II	III	IV	V	VI	VII	VIII
Percentage correct answers[a]	Market	50.00	18.75	6.25	25.00	35.71	6.25	7.14	21.43
	Prize	18.75	28.57	64.29	57.14	25.00	14.29	0.00	12.5

[a] Percentage of total number of participants that marked the correct answer on their answer sheets.

Table 4: Summary of data from participants' answer sheets for setup ν.

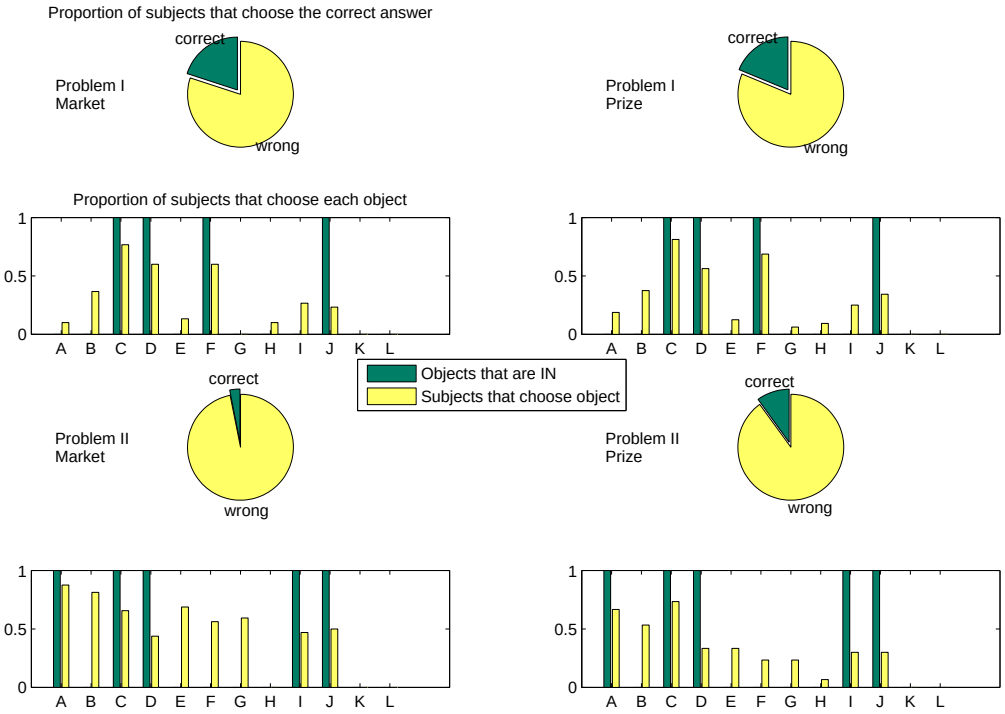

Figure 1: Answer sheets data. Instances I and II.

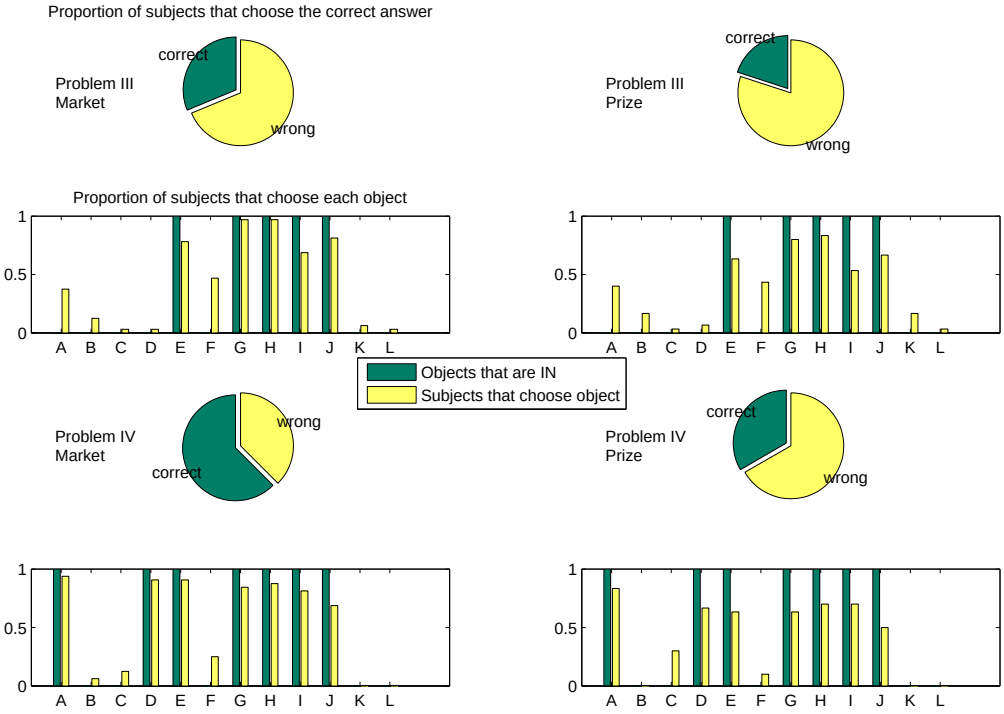

Figure 2: Answer sheets data. Instances III and IV.

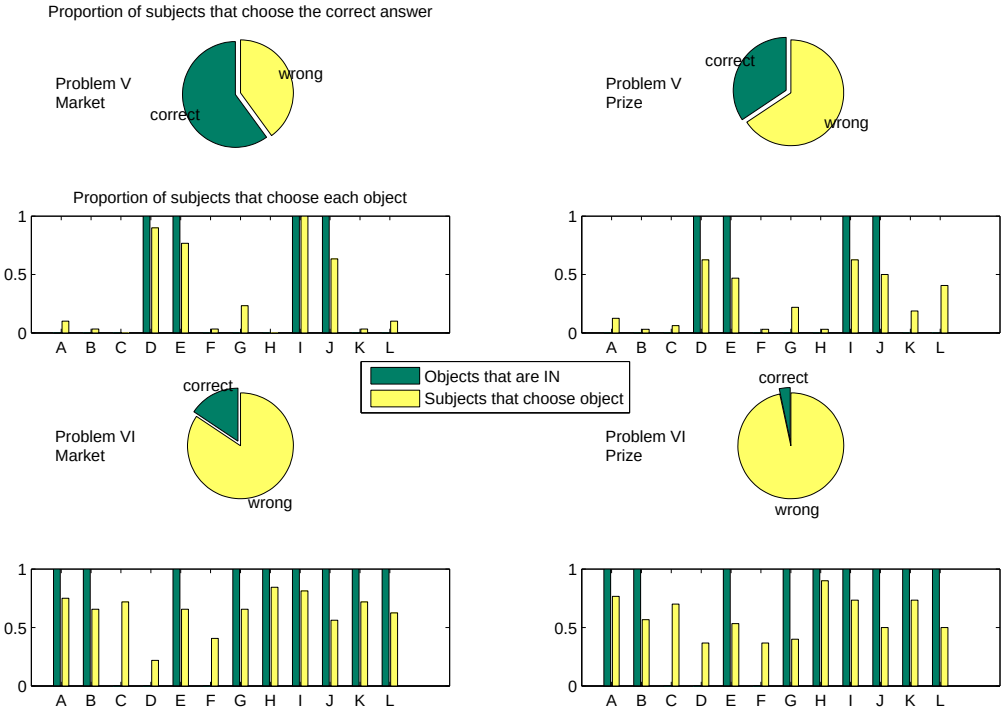

Figure 3: Answer sheets data. Instances V and VI.

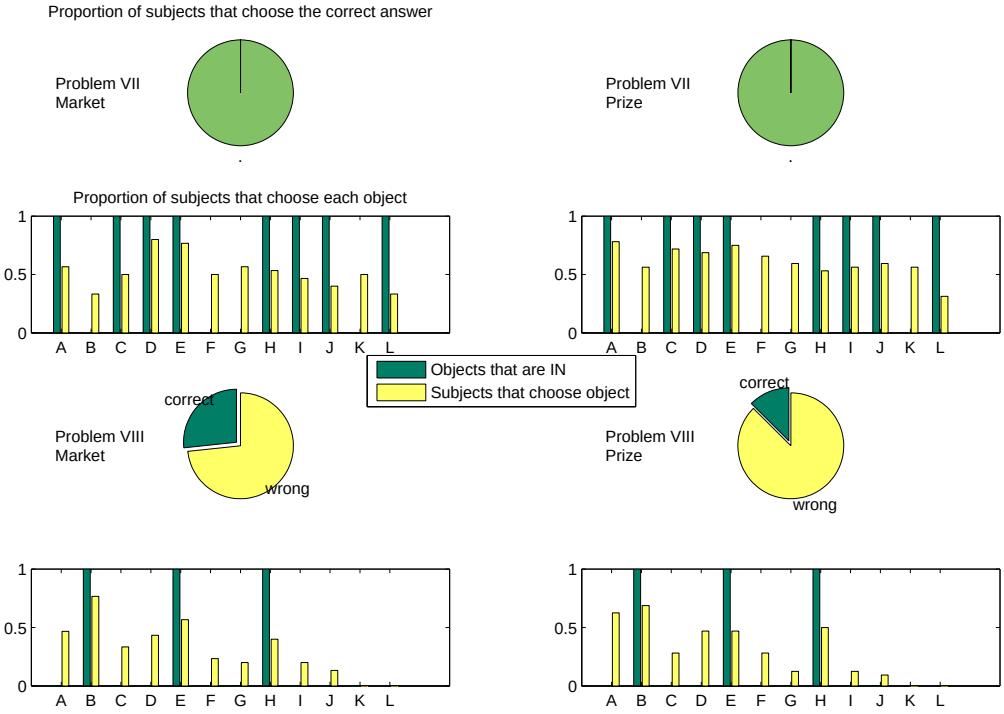

Figure 4: Answer sheets data. Instances VII and VIII.

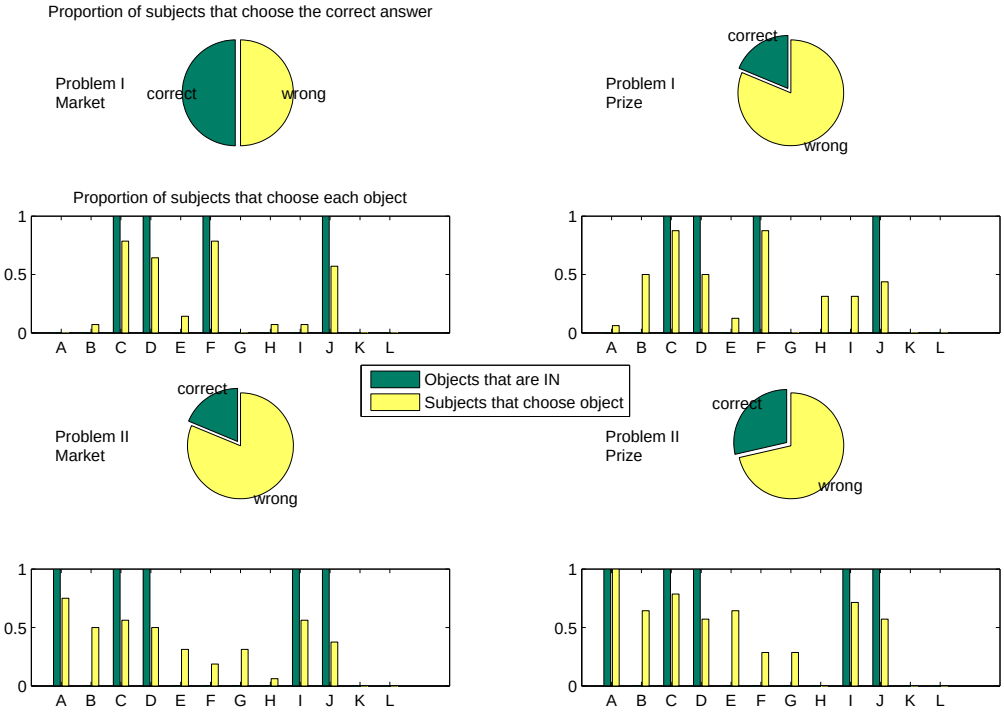

Figure 5: Answer sheets data. Instances I and II. Setup ν.

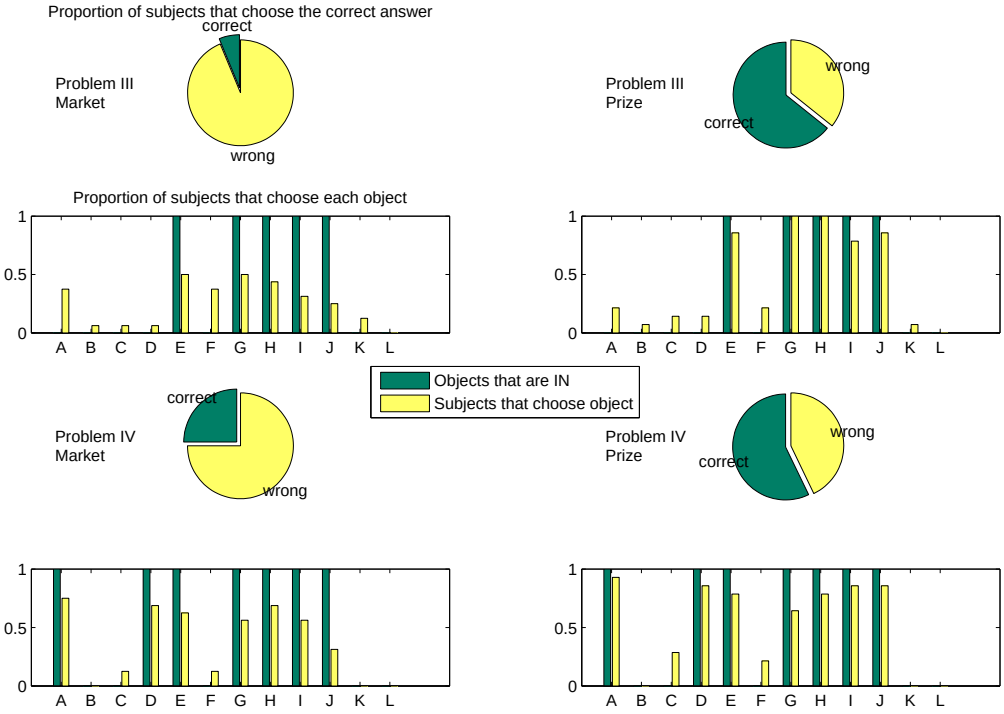

Figure 6: Answer sheets data. Instances III and IV. Setup ν.

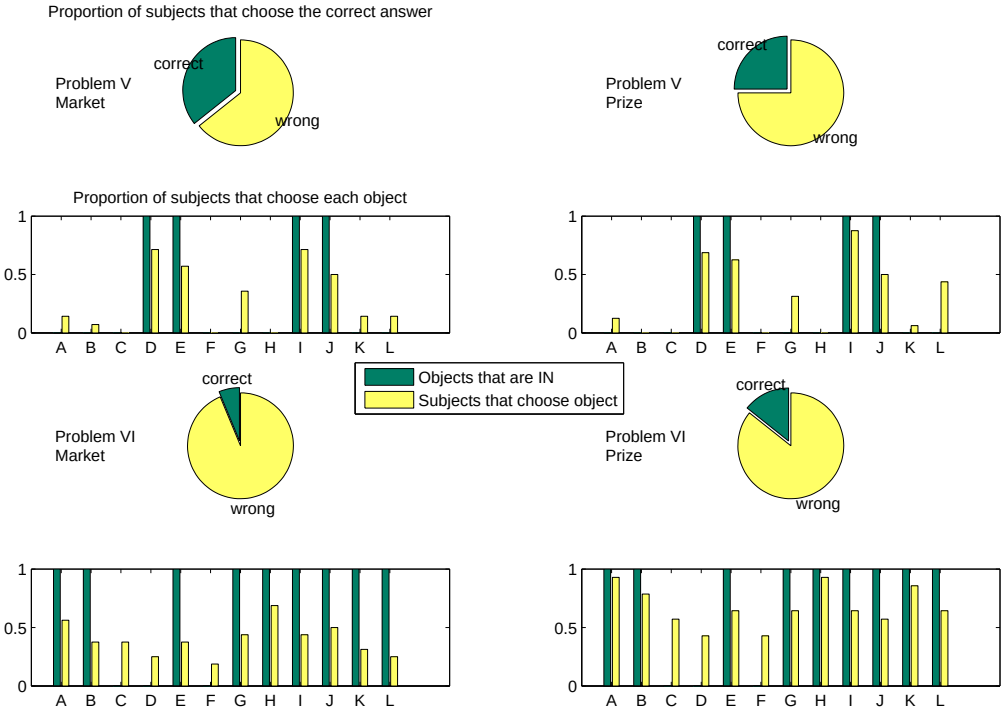

Figure 7: Answer sheets data. Instances V and VI. Setup ν.

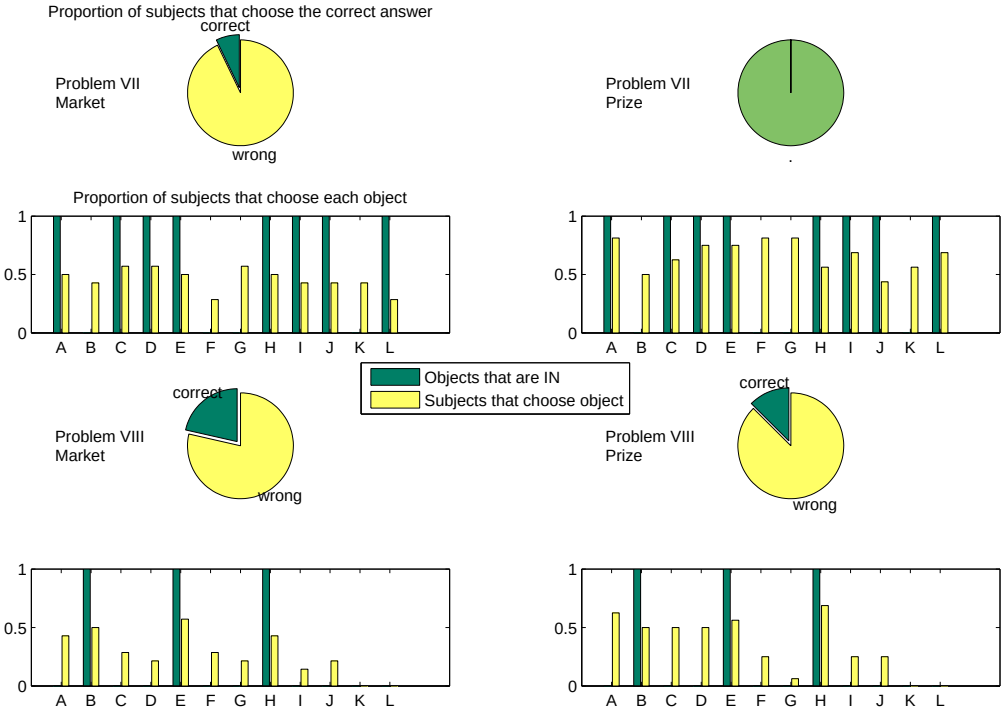

Figure 8: Answer sheets data. Instances VII and VIII. Setup ν.

		Experiment					
		Setup ω				Setup ν	
		040809a	040929b	041202a	041215b	061112a	061116b
Average per person per period	Asks	6.97 (5.1)	5.38 (3.6)	7.68 (4.4)	5.3 (4)	24 (22.1)	22 (22.2)
	Bids	6.28 (3.7)	5.15 (2.9)	6.07 (4.2)	5.03 (3.7)	16.6 (13.9)	20.6 (27)
	Trades	5.43 (3.8)	4.19 (3.5)	5.63 (3.2)	4.07 (2.7)	10.6 (8.4)	7.9 (6.8)
	Volume	11.07 (8.5)	8.79 (6.9)	12.6 (7.8)	12.88 (9.2)	-	-
Average per asset per period	Asks	8.71 (4.1)	7.62 (3.6)	9.6 (4.8)	6.62 (3.9)	31.7 (25)	32 (17.8)
	Bids	7.85 (5.3)	7.29 (3.9)	6.07 (4.2)	6.29 (3.7)	21.1 (9)	30 (13.2)
	Trades	6.79 (3.8)	5.94 (4.0)	7.04 (3.2)	5.08 (3.8)	13.5 (7)	11.5 (7.5)
	Volume	13.83 (9.9)	12.46 (9.0)	15.75 (9.8)	16.1 (11.9)	-	-
Totals	Asks	418	366	461	318	1397	1411
	Bids	377	350	364	302	928	1321
	Trades	326	285	338	244	594	505
	Volume	664	598	756	773	-	-

Table 5: Summaries of trade and bidding activity. In setup ν all trades involve only one unit, so volume equals number of trades.

How does the market do what it does? It could be the case that participants in the market treatment just sat down and tried as hard as in the prize treatment to solve the instance, completely disregarding the market. This is not the case, as attested by the following two results:

Market Result One: Trade and Bidding are Abundant.

Table 5 lists offer and trade statistics. Per-person-and-per-period averages are displayed, as well as per-asset-and-per-period averages. Standard errors are in parentheses. Across all Market periods in all sessions, a large number of offers and trades are recorded.

Table 6 splits statistics across IN and OUT securities. "IN" securities correspond to items that are in the optimal knapsack; "OUT" securities correspond to items that are not in the optimal knapsack. Table 7 displays trading activity per instance. In setup ω,

		Experiment					
		Setup ω				Setup ν	
		040809a	040929b	041202a	041215b	061112a	061116b
Trade volume	IN	15.4	10.7	17.3	12.5	10.3	8.5
	OUT	14.9	17.7	17.12	24.9	16	15.8
Asks	IN	7.9	6.0	9.5	5.3	13.4	22.3
	OUT	10.7	11.7	11.2	10.1	45.7	46.17
Bids	IN	10.9	8.0	9.8	6.3	25	31.6
	OUT	6.8	7.9	7.1	7.7	18	27.8

Table 6: Volume of trade and number of bids and asks, for IN and OUT securities.

each instance is solved under the Market treatment in two experimental sessions, which we refer to as Sessions One and Two. Since the number of items, and hence, securities, differs across instances, Table 7 also reports per-security averages across both Sessions.

Differences in the number of trades and orders are evident across instances, and across securities (IN vs. OUT securities). We can only speculate about the reason of these differences. That is the matter of Section 5. It also catches the eye that the number of bids and asks is much larger in setup ν. This may be driven by the use of the simpler trading interface of jMarkets.

Market Result Two: Prices are Noisy but Informative.
For all but one instance, the distribution of prices of IN securities *first order stochastically dominate* the distribution of prices of OUT securities. Transaction prices reveal information about the optimal solution, but very noisily. The one instance where this dominance relation is not found, is instance VII, for which nobody finds the solution. First order stochastic dominance in this context means that the probability that the price of an IN security is larger than the price of an OUT security is at least 0.5. Figure 9 shows the empirical distribution functions for prices of IN and OUT securities in every instance. in setup ω. For setup ν we show the histograms of prices of IN and OUT securities for every instance. In this setup, the variety in prices was small, thus making the construction of an empirical cdf a pointless exercise. The histograms do display the

	Instance							
	I	II	III	IV	V	VI	VII	VIII
Session One	84	82	81	57	88	65	74	80
Session Two	78	75	57	59	69	53	87	104
n	10	10	12	10	12	12	12	10
average, per asset (ω)	8.1	7.85	5.75	5.8	6.54	4.92	6.71	9.2
Setup ν	135	138	140	142	171	87	115	175

Table 7: Number of trades for each instance of the knapsack problem.

same stochastic dominance feature that is obvious in the empirical cdf figures. Moreover, if we disregard the low price diversity and construct empirical cdfs with these data, we obtain a clear first order stochastic dominance of prices of IN securities, even for instance VII!.

There is clear evidence that participants do *interact* in the market. There is the potential for communication, and prices are an effective language since securities are differentiable based on their prices.

Finally, we report two more characteristics of our experimental markets which might carry some information, but are harder to interpret.

- **Prices are Low.** In setup ω, the median price is \$0.5, while the average price is \$0.48. Average and median prices of IN and OUT securities lie slightly above and below \$0.5, respectively. Both the median and the average become even lower when we add data from setup ν sessions.

- **Excess Supply of OUT Securities.** For every trade, we know whether it was the result of a buyer's submission of a market order (an order that takes an offer that is already waiting in the book and transforms it into a trade), or of a seller's submission of a market order. Among trades that are the result of a buyer's market order, a significantly higher number correspond to OUT securities than to IN securities. Also, most trades of OUT securities are the result of a buyer's market order. The reverse is not true for IN securities.

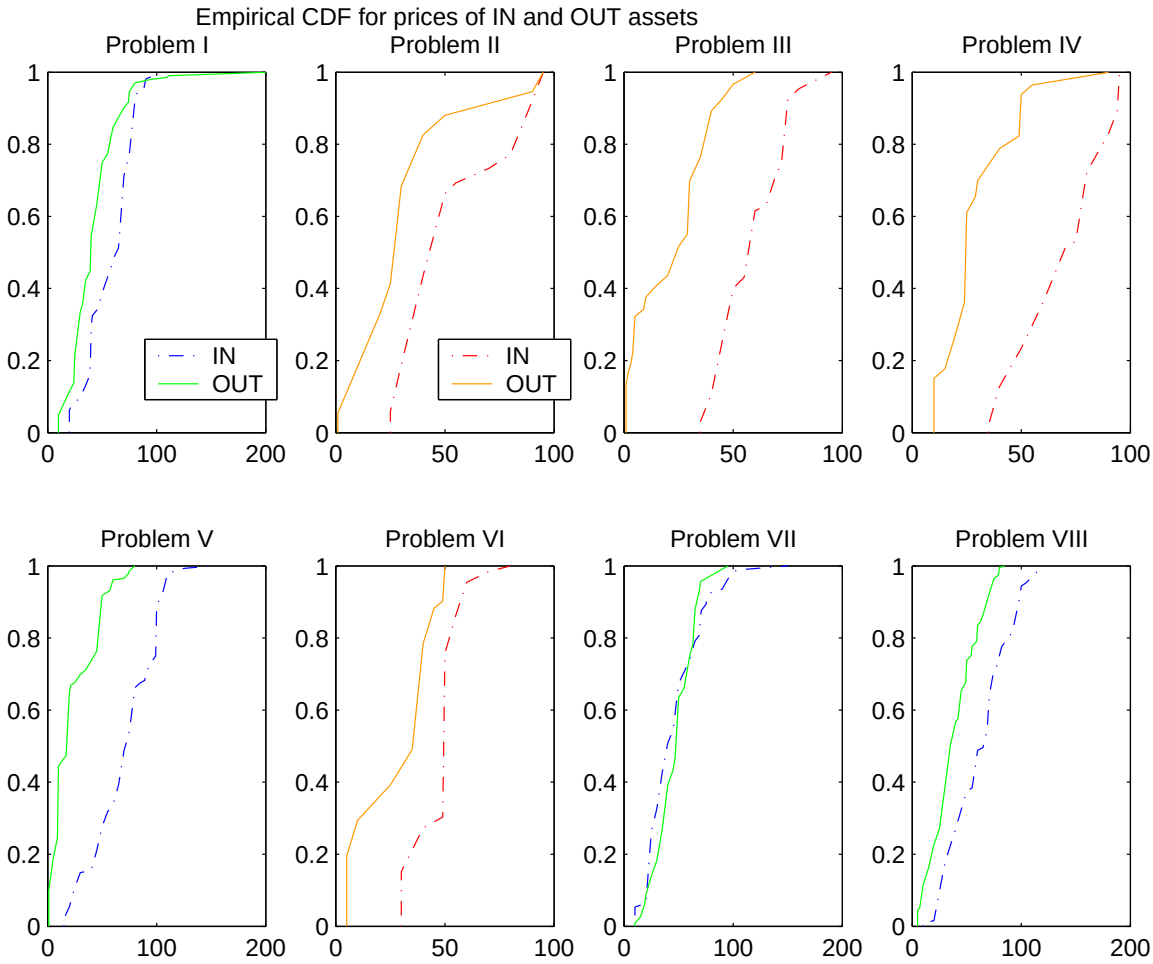

Figure 9: Empirical Cumulative Density Functions for prices of IN and OUT securities. Instances solved under the Market setup in experiment type a have a different color scheme than those solved under the Market setup in experiment type b.

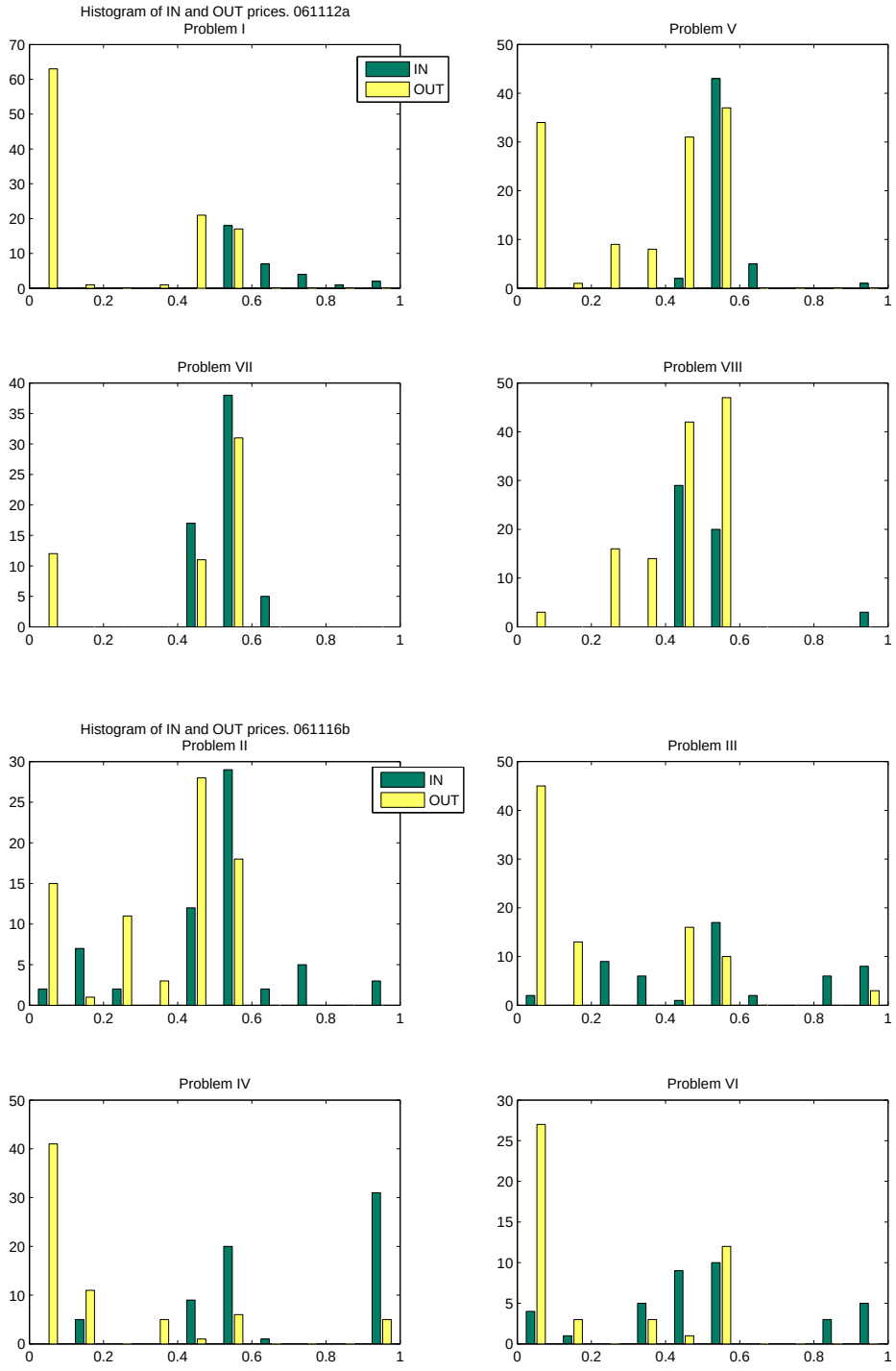

Figure 10: Histogram of prices for the experimental periods of session $\nu 061112a$ and $\nu 061116b$.

4.2 Sahni-k is a good measure of difficulty

If we consider only setup ω, in both treatments, Sahni-k difficulty is a good predictor of the number of correct answers for an instance. We propose and describe this measure in Section 2.2. Here we summarize the ranking of instances that follows from using this measure:

$$IV \prec V \sim I \prec VIII \sim III \prec II \sim VI \prec VII,$$

where $\prec$ indicates that an instance is "easier", and $\sim$ indicates that two instances are not ranked, according to the Sahni-k difficulty measure.

From Table 3 we get the following ordering of instances in the market:

$$\tilde{I}_{IV} > \tilde{I}_{V} > \tilde{I}_{III} > \tilde{I}_{VIII} > \tilde{I}_{I} > \tilde{I}_{VI} > \tilde{I}_{II} > \tilde{I}_{VII},$$

where $\tilde{I}_s$ denotes the fraction of participants that solve instance s correctly. Instances in the prize treatment rank very similarly,

$$\tilde{I}_{V} > \tilde{I}_{IV} > \tilde{I}_{III} > \tilde{I}_{I} > \tilde{I}_{VIII} > \tilde{I}_{II} > \tilde{I}_{VI} > \tilde{I}_{VII}.$$

In both cases the ranking agrees with the Sahni-k ranking for all instances except instance I, which is "harder" in the experiment than predicted by our difficulty measure.

The clear correlation we just mentioned, disappears in setup ν. However, we argue that this is mainly caused by the fact that participants in one of the two sessions of this setup, outperformed participants in the other session, for every instance. This heterogeneity between groups changes the relative ranking of instances that were solved in the market in one session versus those solved in the other session (analogously for instances solved in the prize). To see that this is indeed the case, notice that the ranking within a session and treatment (e.g., the instances solved under the market treatment in session $\nu 061112a$) displays the strong correlation with Sahni difficulty that was noted for setup ω.

5 Conclusion

We create in the laboratory a situation that emulates intellectual discovery with the property that it cannot be reasonably modeled as Bayesian learning. The results point out that this property may be relevant since in a market for discovery we find that: the discovery is made, trading activity is abundant, and prices are informative but noisy.

To exactly what extent these results are driven by the non-incrementality feature of the cognitive task we use to represent discovery, must still be determined. We set the ground for the formulation of hypotheses that can be tested with further experiments.

In particular, the market treatment may deliver the results it does for several different reasons, which may actually interact. We mention three reasons that are worthy of and prone to further experimental study. First, it may be that participants find a code to coordinate on decentralized computation of the solution. Second, it may be that markets provide a "check", where participants try to corroborate whether the algorithm they are using for computation is right or wrong. Third, the markets do provide a way to reduce the risk that comes from attempting to guess the solution on a hunch (heuristic). This may make participants more willing to express their hunches, by means of which prices become informative. This reason may interact with one or both of the previous reasons. It can be studied further by changing the payoff in the market to incorporate the risk of being paid only if one is entirely correct (e.g., have securities pay dividends only if someone finds the solution, or pay per optimal solution held in the final portfolio, not per security).

We have given most thought to the pursuit of the first reason mentioned above. One check of the idea that participants manage to distribute computation is to verify if markets do better in a setup where coordination is easier, while the task is still computationally difficult. Another option is to move away from the cognitive task and instead set up an environment where the selective acquisition of information by different participants may benefit from coordination. In this purely informational framework, will coordination ensue even when it is very difficult to encode (as is the case in our

experiment)?

Another contribution of this paper is to bridge a connection between certain notions of computer science and economic decisions where incentives matter. The description of non-incrementality of a problem in terms of solution algorithms, and the predictive power of the Sahni difficulty measure indicate that there is a bridge. In particular, it is our belief that the theory of intellectual discovery can gain much from the models of individual cognition that are inferred from the study of algorithms.

References

Enriqueta Aragones, Itzhak Gilboa, Andrew Postlewaite, and David Schmeidler. Fact-free learning. *The American Economic Review*, 95(5):1355–1368, 2005.

Kenneth Arrow. Methodological individualism and social knowledge. *The American Economic Review*, 84(2):1–9, 1994.

Kenneth Arrow. The economic implications of learning by doing. *The Review of Economic Studies*, 29(3):155–173, 62.

Alan Blinder and John Morgan. Are two heads better than one? monetary policy by committee. *Journal of Money, Credit, and Banking*, 37(5), 2005.

James Cox and Stephen Haynes. Barking up the right tree: Are small groups rational agents? *Experimental Economics*, 9:209–222, 2006.

G.B. Dantzig. Discrete variable extremum problems. *Operations Research*, 5:266–277, 1957.

Nancy Gallini and Suzanne Scotchmer. Intellectual property: When is it the best incentive system? In Adam Jaffe, Joshua Lerner, and Scott Stern, editors, *Innovation Policy and the Economy*, volume 2, pages 51–78. MIT Press, 2002.

Sanford Grossman. The existence of futures markets, noisy rational expectations, and informational externalities. *The Review of Economic Studies*, 44(3):431–449, 1977.

Sanford Grossman and Joseph Stiglitz. On the impossibility of informationally efficient markets. *The American Economic Review*, 70(3):393–408, 1980.

Boyan Jovanovic. Job matching and the theory of turnover. *The Journal of Political Economy*, 87(5):972–990, 1979.

Hans Kellerer, Ulrich Pferschy, and David Pisinger. *Knapsack Problems*. Springer-Verlag, 2004.

Thomas Kuhn. *The Structure of Scientific Revolutions*. University of Chicago Press, 1962.

John Ledyard. Designing information markets for policy analysis. Presentation Slides, 2005.

Boris Maciejovsky and David Budescu. Is cooperation necessary? learning and knowledge transfer in cooperative groups and competitive auctions. 2005.

Jacob Marschak and Roy Radner. *Economic Theory of Teams*. Yale University Press, 1972.

S. Martello and P. Toth. *Knapsack Problems: Algorithms and Computer Implementations*. Wiley, 1990.

Charles Plott and Shyam Sunder. Rational expectations and the aggregation of diverse information in laboratory security markets. *Econometrica*, 56(5):1085–1118, 1988.

S. Sahni. Approximate algorithms for the 0-1 knapsack problem. *Journal of the ACM*, 22:115–124, 1975.