Physical Search Problems Applying Economic Search Models

Yonatan Aumann and **Noam Hazon** and **Sarit Kraus** and **David Sarne**
Department of Computer Science
Bar Ilan University
Ramat Gan, 52900, Israel
{aumann,hazonn,sarit,sarned}@cs.biu.ac.il

Abstract

This paper considers the problem of an agent searching for a resource or a tangible good in a physical environment, where at each stage of its search it observes one source where this good can be found. The cost of acquiring the resource or good at a given source is uncertain (a-priori), and the agent can observe its true value only when physically arriving at the source. Sample applications involving this type of search include agents in exploration and patrol missions (e.g., an agent seeking to find the best location to deploy sensing equipment along its path). The uniqueness of these settings is that the expense of observing the source on each step of the process derives from the last source the agent explored. We analyze three variants of the problem, differing in their objective: minimizing the total expected cost, maximizing the success probability given an initial budget, and minimizing the budget necessary to obtain a given success probability. For each variant, we first introduce and analyze the problem with a single agent, either providing a polynomial solution to the problem or proving it is NP-Complete. We also introduce an innovative fully polynomial time approximation scheme algorithm for the minimum budget variant. Finally, the results for the single agent case are generalized to multi-agent settings.

Introduction

Frequently, in order to successfully complete its task, an agent may need to *explore* (i.e., search) its environment and choose among different available options. For example, an agent seeking to purchase a product over the internet needs to query several electronic merchants in order to learn their posted prices; a robot searching for a resource or a tangible item needs to travel to possible locations where the resource is available and learn the configuration in which it is available as well as the difficulty of obtaining it there. In these environments, the benefit associated with an opportunity is revealed only upon observing it. The only knowledge available to the agent prior to observing the opportunity is the probability associated with each possible benefit value of each prospect.

While the exploration in virtual environments can sometimes be considered costless, in physical environments traveling and observing typically also entails a cost. Furthermore, as the agent travels to a new location the cost associated with exploring other unexplored locations changes. For example, consider a Rover robot with the goal of mining a certain mineral. Potential mining locations may be identified based on a satellite image, each associated with some uncertainty regarding the difficulty of mining there. In order to assess the amount of battery power required for mining at a specific location, the robot needs to physically visit there. The robot's battery is thus used not only for mining the mineral but also for traveling from one potential location to another. Consequently, an agent's strategy in an environment associated with search costs should maximize the *overall* benefit resulting from the search process, defined as the value of the option used eventually minus the costs accumulated along the process, rather than merely finding the best valued option.

In this paper we study the problem of finding optimal strategies for agents acting in such physical environments. Models that incorporate search costs as part of an economic search process have attracted the attention of many researchers in various areas, prompting several reviews over the years (Lippman and McCall 1976; McMillan and Rothschild 1994). These search models have developed to a point where their total contribution is referred to as *search theory*. Nevertheless, these economic-based search models, as well as their extensions over the years into multi-agent environments (Choi and Liu 2000; Sarne and Kraus 2005), assume that the cost associated with observing a given opportunity is stationary (i.e., does not change along the search process). While this permissive assumption facilitates the analysis of search models, it is frequently impractical in the physical world. The use of changing search costs suggests an optimal search strategy structure different from the one used in traditional economic search models: other than merely deciding when to terminate its search, the agent also needs to integrate into its decision making process exploration sequence considerations.

Changing search costs have been previously considered in the MAS domain in the context of Graph Search Problems (Koutsoupias, Papadimitriou, and Yannakakis 1996). Here, the agent is seeking a single item, and a distribution is defined over all probability of finding it at each of the graph's nodes (Ausiello, Leonardi, and Marchetti-

Spaccamela 2000). Nevertheless, upon arriving at a node the success factor is binary: either the item is there or not. Extensions of these applications to scenarios where the item is mobile are of the same character (Gal 1980; Koopman 1980).

This paper thus bridges the gap between classical economic search theory (which is mostly suitable for virtual or non-dimensional worlds) and the changing search cost constraint imposed by operating in physical MAS environments. Specifically, we consider physical settings where the opportunities are aligned along a path (Hazon and Kaminka 2005) (either closed or a non-closed one) and the cost of observing the true value of any unexplored source depends on its distance (along the path) from the agent's current position. For exposition purposes we use in the remaining of the paper the classical procurement application where the goal of the search is purchasing a product and the value of each observed opportunity represents a price.

We consider three variants of the problem, differing in their objective. The first (*Min-Expected-Cost*) is the problem of an agent that aims to minimize the expected total cost of completing its task. The second (*Max-Probability*) considers an agent that is given an initial budget for the task (which it cannot exceed) and needs to act in a way that maximizes the probability it will complete its task (e.g., reach at least one opportunity with a budget large enough to successfully buy the product). In the last variant (*Min-Budget*) the agent is requested to guarantee a pre-defined probability of completing the task, and needs to minimize the overall budget that will be required to achieve the said success probability. While the first variant fits mostly product procurement applications, the two latter variants fit well into applications of robots engaged in remote exploration, operating with a limited amount of battery power (i.e., a budget).

The contributions of the paper are threefold: First, the paper is the first to introduce single and multi-agent costly search with changing costs, a model which we believe is highly applicable in real-world settings. To the best of our knowledge this important search model has not been investigated to date, neither in the rich economic search theory literature nor in MAS and robotics research. Second, it thoroughly analyzes three different variants of the problem, both for the single agent and multi-agent case and identifies unique characteristics of their optimal strategy. For some of the variants it proves the existence of a polynomial solution. For others it proves the hardness of the problem. Finally, the paper presents an innovative fully polynomial time approximation scheme algorithm for the budget minimization problem.

Summary of Results. We first consider the single agent case. We prove that in general metric spaces all three problem variants are NP-hard. Thus, as mentioned, we focus on the path setting. For this case we provide a polynomial algorithm for the *Min-Expected-Cost* problem. We show the other two problems (*Min-Budget* and *Max-probability*) to be NP-complete even for the path setting. Thus, we consider further restrictions and also provide an approximation scheme. We show that both problems are polynomial if the number of possible prices is constant. For the *Min-Budget* problem, we also provide an FPTAS (fully-polynomial-time-approximation-scheme), such that for any $\epsilon > 0$, providing a $(1 + \epsilon)$ approximation in time $O(\text{poly}(n\epsilon^{-1}))$, where n is the size of the input.

For the multi-agent case, we show that if the number of agents is fixed, then all of the single-agent algorithms extend to k-agents, with the time bounds growing exponentially in k. Therefore the computation of the agents' strategies can be performed whenever the number of agents is relatively moderate, a scenario characterizing most physical environments where several agents cooperate in exploration and search. If the number of agents is part of the input then *Min-Budget* and *Max-Probability* are NP-complete even on the path and even with a single price. Table 1 presents a summary of the results. Empty entries represent open problems.

Problem Formulation

We are provided with m points - $S = \{u_1, \ldots, u_m\}$, which represent the store locations, together with a distance function *dis* : $S \times S \rightarrow R^+$ - determining the travel costs between any two stores. We are also provided with the agent's initial location, u_s, which is assumed WLOG (without loss of generality) to be at one of the stores (the product's price at this store may be ∞). In addition, we are provided with a price probability function $p^i(c)$ - stating the probability that the price at store i is c. Let D be the set of distinct prices with non-zero probability, and $d = |D|$. We assume that the actual price at a store is only revealed once the agent reaches the store. The multi-agent case will be defined in the last section. Given these inputs, the goal is roughly to obtain the product at the minimal total cost, including both travel costs and purchase price. Since we are dealing with probabilities, this rough goal can be interpreted in three different concrete formulations:

1. *Min-Expected-Cost*: minimize the expected cost of purchasing the product.
2. *Min-Budget:* given a success probability p_{succ} minimize the initial budget necessary to guarantee purchase with probability at least p_{succ}.
3. *Max-Probability*: given a total budget B, maximize the probability to purchase the product.

In all the above problems, the optimization problem entails determining the strategy (order) in which to visit the different stores, and if and when to terminate the search. For the *Min-Expected-Cost* problem we assume that an agent can purchase the product even after leaving the store (say by phone).

Unfortunately, for general distance functions (e.g. the stores are located in a general metric space), all three of the above problems are NP-hard. To prove this we first convert the problems into their decision versions. In the *Min-Expected-Cost-Decide* problem this translate to: we are given a set of points S, a distance function *dis* : $S \times S \rightarrow R^+$, an agent's initial location u_s, a price-probability function $p^{\cdot}(\cdot)$, and a maximum expected cost M, decide whether

		Min-Expected-Cost	*Max-Probability*	*Min-Budget*
General metric spaces		NP-Hard	NP-Complete	NP-Complete
Path - general case	Single agent	$O(d^2m^2)$	NP-Complete	NP-Complete
	k agents	$O(d^2(\frac{m}{k})^{2k})$		
	k is a parameter			
Path - single price	Single agent	not defined	$O(m)$	$O(m)$
	k agents		$O((\frac{m}{k})^{2k})$	$O((\frac{m}{k})^{2k})$
	k is a parameter		NP-Complete	NP-Complete
Path - d prices, k agents		$O(d^2(\frac{m}{k})^{2k})$	$O(2^{-kd}(\frac{e\cdot m}{kd})^{2kd})$	$O(2^{-kd}(\frac{e\cdot m}{kd})^{2kd})$
Path -single agent $(1+\epsilon)$ approximation				$O(n\epsilon^{-6})$
Path - k agents $(1+k\epsilon)$ approximation				$O(n\epsilon^{-k6})$

Table 1: Summary of results: n is the input size, m - the number of points (store locations), d - the number of different possible prices, k - the number of agents.

there is a policy with an expected cost at most M. In the *Min-Budget-Decide* problem, the input is the same, only that instead of a target expected cost, we are given a minimum success probability p_{succ} and maximum budget B, and we have to decide whether a success probability of at least p_{succ} can be obtained with budget at most B. The exact same formulation also constitutes the decision version of the *Max-Probability* problem. We prove that for general metric spaces all these problems are NP complete. Thus, we focus on the case that the stores are all located on a single path. We denote these problems *Min-Budget (path), Max-Probability (path),* and *Min-Expected-Cost (path)*, respectively. In this case we can assume that, WLOG all points are on the line, and do away with the distance function *dis*. Rather, the distance between u_i and u_j is simply $|u_i - u_j|$. Furthermore, WLOG we may assume that the stores are ordered from left-to-right, i.e. $u_1 < u_2 < \cdots < u_m$. In the following, when we refer to *Min-Budget*, *Max-Probability* and *Min-Expected-Cost* we refer to their *path* variants, unless otherwise specified.

Multi-Agent. In the multi agent case, we assume k agents, operating in the same underlaying physical setting as in the single agent case, i.e. a set of *stores* S, a distance function *dis* between the points, and a price probability function for each store. In this case, however, different agents may have different initial location, which are provided as a vector $(u_s^{(1)}, \ldots, u_s^{(k)})$. We assume full (wireless) communication between agents. In theory, agents may move in parallel, but since minimizing time is not an objective, we may assume WLOG that at any given time only one agents moves. When an agent reaches a store and finds the price at this location, it communicates this price to all other agents. Then, a central decision is made whether to purchase the product (and where) and if not what agent should move next and to where. We assume that all resources and costs are shared among all the agents. Therefore, in *Multi-agent Min-Expected-Cost* problem the agents try to minimize the expected total cost, which includes the travel costs of all agents plus the final purchase price (which is one of the prices that the agents have sampled). In *Multi-agent Min-Budget* and *multi-agent Max-Probability* problems, the initial budget is for the use of all the agents, and the success probability is for any of the agents to purchase, at any location.

Minimize-Expected-Cost

Hardness in General Metric Spaces

Theorem 1 *For general metric spaces* Min-Expected-Cost-Decide *is NP-Hard.*

Proof. The proof is by reduction from Hamiltonian path, defined as follows. Given a graph $G = (V, E)$ with $V = \{v_1, \ldots, v_n\}$, decide whether there is a simple path $(v_{i_1}, v_{i_2}, ..., v_{i_n})$ in G covering all nodes of V. The reduction is as follows. Given a graph $G = (V, E)$ with $V = \{v_1, \ldots, v_n\}$, set S (the set of stores) to be $S = \{u_s\} \cup \{u_1, \ldots, u_n\}$, where u_s is the designated start location, and $\{u_1, \ldots, u_n\}$ correspond to $\{v_1, \ldots, v_n\}$. The distances are defined as follows. For all $i, j = 1, \ldots, n$, *dis*$(u_s, u_i) = 2n$, and *dis*(u_i, u_j) is the length of the shortest path between v_i and v_j in G. For all i, $p^i(0) = 0.5$, and $p^i(\infty) = 0.5$, and for u_s, $p^s(n!) = 1$. Finally, set $M = 2n + \sum_{j=1}^{n} 2^{-j}(j-1) + 2^{-n}(n! + n - 1)$.

Suppose that there is an Hamiltonian path $H = (v_{i_1}, v_{i_2}, ..., v_{i_n})$ in G. Then, the following policy achieves an expected cost of exactly M. Starting in u_s move to u_{i_1} and continue traversing according to the Hamiltonian path. If at any point u_i along the way the price is 0, purchase and stop. Otherwise continue to the node in the path. If at all points along the path the price was ∞, purchase from store u_s, where the price is $n!$. The expected cost of this policy is as follows. The price of the initial step (from u_s to u_{i_1}) is a fixed $2n$. For each j, the probability to obtain price 0 at u_{i_j} but not before is 2^{-j}. The cost of reaching u_{i_j} from u_{i_1} is $j-1$. The probability that no u_j has price 0 is 2^{-n}, in which case the purchase price is $n!$, plus $n-1$ wasted steps. The total expected cost is thus exactly M.

Conversely, suppose that there is no Hamiltonian path in G. Clearly, since the price at u_s is so large, any optimal strategy must check all nodes/stores $\{u_1, \ldots, u_n\}$ before pur-

chasing at u_s. Since there is no Hamiltonian path in G, any such exploration would be strictly more expensive than one with a Hamiltonian path. Thus, the expected cost would be strictly more than M. □

Solution for the Path

When all stores are located on a path, the *Min-Expected-Cost* problem can be modeled as finite-horizon Markov decision process (MDP), as follows. Note that on the path, at any point in time the points/stores visited by the agent constitute a contiguous interval, which we call the *visited interval*. Clearly, the algorithm need only make decisions at store locations. Furthermore, decisions can be limited to times when the agent is at one of the two stores edges of the *visited interval*. At each such location, the agent has only three possible actions: "go right" - extending the visited-interval one store to the right, "go left" - extending the visited-interval one store to the left, or "stop" - stopping the search and buying the product at the best price so far. Also note that *after* the agent has already visited the interval $[u_\ell, u_r]$, how exactly it covered this interval does not matter for any future decision; the costs have already been incurred. Accordingly, the states of the MDP are quadruplets $[\ell, r, e, c]$, such that $\ell \leq s \leq r$, $e \in \{\ell, r\}$, and $c \in D$, representing the situation that the agents visited stores u_ℓ through u_r, it is currently at location u_e, and the best price encountered so far is c. The terminal states are *Buy(c)* and all states of the form $[1, m, e, c]$, and the terminal cost is c. For all other states there are two or three possible actions - "go right" (provided that $r < m$), "go left" (provided that $1 < \ell$), or "stop". The cost of "go right" on the state $[\ell, r, e, c]$ is $(u_{r+1} - u_e)$, while the cost of "go-left" is $(u_e - u_{\ell-1})$. The cost of "stop" is always 0. Given the state $[\ell, r, e, c]$ and move "go-right", there is probability $p^{r+1}(c')$ to transition to state $[\ell, r+1, r+1, c']$, for $c' < c$. With the remaining probability, the transition is to state $[\ell, r+1, r+1, c]$. Transition to all other states has zero probability. Transitions for the "go left" action are analogous. Given the state $[\ell, r, e, c]$ and the action "stop", there is probability 1 to transition to state *Buy(c)*. This fully defines the MDP. The optimal strategy for finite-horizon MDPs can be determined using dynamic programming (see (Puterman 1994, Ch.4)). In our case, the complexity can be brought down to $O(d^2m^2)$ steps (using $O(dm^2)$ space).

Min-Budget and Max-Probability

NP Completeness

Unlike the Min-Expected-Cost problem, the other two problems are NP-complete even on a path.

Theorem 2 Min-Budget-Decide *problem is NP-Complete even on a path.*

Proof. Given an optimal policy it is easy to compute its total cost and success probability in $O(n)$ steps, therefore *Min-Budget-Decide* is in NP. The proof of NP-Hardness is by reduction from the knapsack problem, defined as follows. Given a *knapsack* of capacity $C > 0$ and N items, where each item has value $v_i \in \mathbb{Z}^+$ and size $s_i \in \mathbb{Z}^+$, determine whether there is a selection of items ($\delta_i = 1$ if selected, 0 if not) that fits into the knapsack, i.e. $\sum_{i=1}^{N} \delta_i s_i \leq C$, and the total value, $\sum_{i=1}^{N} \delta_i v_i$, is at least V.

Given an instance of the knapsack problem we build an instance for *Min-Budget-Decide* problem as follows. We assume WLOG that all the points are on the line. Our line consists of $2N+2$ stores. N stores corresponds to the knapsack items, denoted by $u_{k_1}, ..., u_{k_N}$. The other $N+2$ stores are denoted $u_{g_0}, u_{g_1}, ..., u_{g_{N+1}}$, where u_{g_0} is the agent's initial location. Let $T = 2 \cdot \sum_{i=1}^{N} s_i$ and $maxV = N \cdot \max_i v_i$. For each odd i, u_{g_i} is to the right of u_{g_0} and $u_{g_{i+2}}$ is to the right of u_{g_i}. For each even i ($i \neq 0$), u_{g_i} is to the left of u_{g_0} and $u_{g_{i+2}}$ is to the left of u_{g_i}. We set $|u_0 - u_1| = |u_0 - u_2| = T$ and for each $i > 0$ also, $|u_{g_i} - u_{g_{i+2}}| = T$. If N is odd (even) u_{k_N} is on the right (left) side of u_{g_i} and it is the rightmost (leftmost) point. As for the other u_{k_i} points, u_{k_i} is located between u_{g_i} and $u_{g_{i+2}}$, if i is odd, and between $u_{g_{i+2}}$ and u_{g_i} otherwise. For both cases, $|u_{g_i} - u_{k_i}| = s_i$. See figure 1 for an illustration.

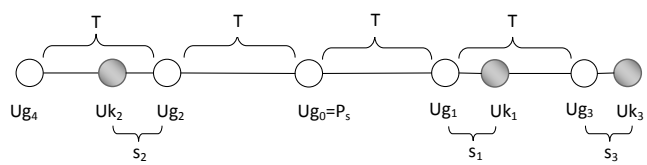

Figure 1: Reduction of knapsack to Min-Budget-Decide problem used in proof of Theorem 2, for N=3.

We set $B = T \cdot \sum_{j=1}^{N+1} j + 2C + 1$ and for each i set $X^i = T \cdot \sum_{j=1}^{i} j + 2 \cdot \sum_{j=1}^{i-1} s_j$. At store $u_{g_{N+1}}$ either the product is available at the price of 1 with probability $1 - 2^{-maxV}$, or not available at any price. On any other store u_{g_i}, either the product is available at the price of $B - X^i$ with the same probability, or not available at all. At any store u_{k_i}, either the product is available at the price of $B - X^i - s_i$, with probability $1 - 2^{-maxV}$, or not available at any price. Finally, we set $p_{succ} = 1 - 2^{-maxV \cdot (N+1)} \cdot 2^{-V}$.

Suppose there is a selection of items that fit the knapsack with a total value of at least V, and consider the following policy: go right from u_{g_0} to u_{g_1}. Then for each $i = 1, 2, .., N$, if $\delta_i = 0$ (item i was not selected) change direction and go to the other side to $u_{g_{i+1}}$. Otherwise, continue in the current direction to u_{k_i} and only then change direction to $u_{g_{i+1}}$. This policy's total travel cost is $\sum_{i=1}^{N}(i \cdot T + \delta_i \cdot 2s_i) + (N+1) \cdot T = T \cdot \sum_{i=1}^{N+1} i + 2C = B - 1$, thus the agent has enough budget to reach all u_{g_i}, and u_{k_i} with $\delta_i = 1$. When the agent reaches u_{g_i}, $i < N+1$ it has already spent on traveling cost exactly $T \cdot \sum_{j=1}^{i} j + 2 \cdot \sum_{j=1}^{i-1}(\delta_j \cdot s_j) \leq X^i$ so the agent has a probability of $1 - 2^{-maxV}$ to purchase the product at this store. When it reaches $u_{g_{N+1}}$ its on the end of its tour and since the agent's total traveling cost is $B - 1$, here it also has a probability of $1 - 2^{-maxV}$ to purchase the product. When it reaches u_{k_i} it has already spent exactly $T \cdot \sum_{j=1}^{i} j + 2 \cdot \sum_{j=1}^{i-1}(\delta_j \cdot s_j) + s_i \leq X^i + s_i$ so the agent has a probability of $1 - 2^{-v_i}$ to purchase the product in this store. In total, the success probability is $1 - (2^{-maxV \cdot (N+1)}$.

$\prod_{i=1}^{N} 2^{-v_i \cdot \delta_i}) \geq 1 - (2^{-maxV \cdot (N+1)} \cdot 2^{-V}) = p_{succ}$ as required.

Suppose there is a policy, plc with a total travel cost that is less than or equal B, and its success probability is at least p_{succ}. Hence, plc's failure probability is at most $1 - p_{succ} = 2^{-maxV \cdot (N+1)} \cdot 2^{-V}$. Since $maxV = N \cdot \max_i v_i$, plc must reach all the $N+1$ stores u_{g_i} with enough budget. Hence, plc must go right from u_{g_0} to u_{g_1} and then to each other u_{g_i} before $u_{g_{i+1}}$. Therefore plc goes in a zigzag movement from one side of u_s to the other side and so on repeatedly. plc also has to select some u_{k_i} to reach with enough budget. Thus, plc has to reach these u_{k_i} right after the corresponding store u_{g_i}. We use $\gamma_i = 1$ to indicate the event in which plc selects to reach u_{k_i} right after u_{g_i}, and $\gamma_i = 0$ to denote the complementary event. plc's total traveling cost is less than or equal $B - 1$ to be able to purchase the product also at the last store, $u_{g_{N+1}}$, so $T \cdot \sum_{j=1}^{N+1} j + 2 \cdot \sum_{j=1}^{N} \gamma_j \cdot s_j \leq T \cdot \sum_{j=1}^{N+1} j + 2C$. Thus, $\sum_{j=1}^{N} \gamma_j \cdot s_j \leq C$. Also, $p_{succ} = 1 - 2^{-maxV \cdot (N+1)} \cdot 2^{-V} \leq 1 - 2^{-maxV \cdot (N+1)} \cdot \prod_{i=1}^{N} 2^{-v_i \cdot \gamma_i} \Rightarrow 2^{-V} \leq \prod_{i=1}^{N} 2^{-v_i \cdot \gamma_i} \Rightarrow V \geq \sum_{i=1}^{N} v_i \cdot \gamma_i$. Setting $\delta_i = \gamma_i$ gives a selection of items that fit the knapsack. □

Thus, we either need to consider restricted instances or consider approximations. We do both.

Restricted Case: Bounded Number of Prices

We consider the restricted case when the number of possible prices, d, is bounded. For brevity, we focus on the *Min-Budget* problem. The same algorithm and similar analysis work also for the *Max-Probability* problem. Consider first the case where there is only one possible price c_0. At any store i, either the product is available at this price, with probability $p_i = p^i(c_0)$, or not available at any price. In this setting we show that the problem can be solved in $O(m)$ steps. This is based on the following lemma, stating that in this case, at most one direction change is necessary.

Lemma 1 *Consider a price c_0 and suppose that in the optimal strategy starting at point u_s the area covered while the remaining budget is at least c_0 is the interval $[u_\ell, u_r]$. Then, WLOG we may assume that the optimal strategy is either $(u_s \rightarrowtail u_r \rightarrowtail u_\ell)$ or $(u_s \rightarrowtail u_\ell \rightarrowtail u_r)$.*

Proof. Any other route would take more cost to cover the same interval. □

Using this observation, we immediately obtain an $O(m^3)$ algorithm for the single price case: consider both possible options for each interval $[u_\ell, u_r]$, and for each compute the total cost and the resulting probability. Choose the option which requires the lowest budget but still has a success probability of at least p_{succ}. With a little more care, the complexity can be reduced to $O(m)$. First note that since there is only a single price c_0, we can add c_0 to the budget at the end, and assume that the product will be provided at stores for free, provided that it is available. Now, consider the strategy of first moving right and then switching to the left. In this case, we need only consider the *minimal* intervals that provide the desired success probability, and for each compute the necessary budget. This can be performed incrementally, in a total of $O(m)$ work for all such minimal intervals, since at most one point can be added and one deleted at any given time. Similarly for the strategy of first moving left and then switching to the right. The details are provided in Algorithm 1.

Algorithm 1 OptimalPolicyForSinglePrice(Success probability p_{succ}, single price c_0)

```
u_r ← leftmost point on right of u_s s.t. 1−∏_{i=s}^{r} 1−p_i ≥ p_succ
ℓ ← s
B_min^RL ← ∞
while ℓ ≥ 0 and r ≥ s do
    B ← 2|u_r − u_s| + |u_s − u_ℓ|
    if B < B_min^RL then
        B_min^RL ← B
    r ← r − 1
    while 1 − ∏_{i=ℓ}^{r} 1 − p_i < p_succ do
        ℓ ← ℓ − 1
u_ℓ ← rightmost point to left of u_s s.t. 1−∏_{i=ℓ}^{s} 1−p_i ≥ p_succ
r ← s
B_min^LR ← ∞
while r ≤ m and ℓ ≤ s do
    B ← 2|u_s − u_ℓ| + |u_r − u_s|
    if B < B_min^LR then
        B_min^LR ← B
    ℓ ← ℓ + 1
    while 1 − ∏_{i=ℓ}^{r} 1 − p_i < p_succ do
        r ← r + 1
return min{B_min^RL, B_min^LR} + c_0
```

Next, consider the case that there may be several different available prices, but their number, d, is fixed. We provide a polynomial algorithm for this case (though exponential in d). First note that in the Min-Budget problem, we seek to minimize the initial budget B necessary so as to guarantee a success probability of at least p_{succ} given this initial budget. Once the budget has been allocated, however, there is no requirement to minimize the actual expenditure. Thus, at any store, if the product is available for a price no greater than the remaining budget, it is purchased immediately and the search is over. If the product has a price beyond the current available budget, the product will not be purchased at this store under any circumstances. Denote $D = \{c_1, c_2, \ldots, c_d\}$, with $c_1 > c_2 > \cdots > c_d$. For each c_i there is an interval $I_i = [u_\ell, u_r]$ of points covered while the remaining budget was at least c_i. Furthermore, for all i, $I_i \subseteq I_{i+1}$. Thus, consider the *incremental* area covered with remaining budget c_i, $\Delta_i = I_i - I_{i-1}$ (with $\Delta_1 = I_1$). Each Δ_i is a union of an interval at left of u_s and an interval at the right of u_s (both possibly empty). The next lemma, which is the multi-price analogue of Lemma 1, states that there are only two possible optimal strategies to cover each Δ_i:

Lemma 2 *Consider the optimal strategy and the incremental areas Δ_i ($i = 1, \ldots, d$) defined by this strategy. For $c_i \in D$, let u_{ℓ_i} be the leftmost point in Δ_i and u_{r_i} the rightmost point. Suppose that in the optimal strategy the covering of Δ_i starts at point u_{s_i}. Then, WLOG we may assume that the optimal strategy is either $(u_{s_i} \rightarrowtail u_{r_i} \rightarrowtail u_{\ell_i})$ or $(u_{s_i} \rightarrowtail u_{\ell_i} \rightarrowtail u_{r_i})$. Furthermore, the starting point for covering Δ_{i+1} is the ending point of covering Δ_i.*

Proof. The areas Δ_i fully determine the success probability of the strategy. Any strategy other than the ones specified in the lemma would require more travel budget, without enlarging any Δ_i. $\square$

Thus, the optimal strategy is fully determined by the leftmost and rightmost points of each Δ_i, together with the choice for the ending points of covering each area. We can thus consider all possible cases and choose the one with the lowest budget which provides the necessary success probability. There are $\frac{m^{2d}}{(2d)!} \leq (\frac{em}{2d})^{2d}$ ways for choosing the external points of the Δ_i's, and there are a total of 2^d options to consider for the covering of each. For each option, computing the budget and probability takes $O(m)$ steps. Thus, the total time is $O(m2^d(\frac{em}{2d})^{2d})$. Similar algorithms can also be applied for the *Max-Probability* problem. In all, we obtain:

Theorem 3 *Min-Budget (path) and Max-Probability (path) can be solved in $O(m)$ steps for a single price and $O(m2^d(\frac{em}{2d})^{2d})$ for d prices.*

Min-Budget Approximation

Next, we provide a FPTAS (fully-polynomial-time-approximation-scheme) for the Min-Budget problem. The idea is to force the agent move in quantum steps of some fixed size δ. In this case the tour taken by the agent can be divided into *segments*, each of size δ. Furthermore, the agent's decision points are restricted to the ends of these segments, except for the case where along the way the agent has sufficient budget to purchase the product at a store, in which case it does so and stops. We call such a movement of the agent a *δ-resolution tour*. Note that the larger δ the less decision points there are, and the complexity of the problem decreases. Given $0 < \epsilon < 1$, we show that with a proper choice of δ we can guarantee a $(1+\epsilon)$ approximation to the optimum, while maintaining a complexity of $O(n\text{poly}(1/\epsilon))$, where n is the size of the input.

Our algorithm is based on computing for (essentially) each initial possible budget B, the maximal achievable success probability, and then pick the minimum budget with probability at least p_{succ}. Note that once the interval $[\ell, r]$ has been covered without purchasing the product, the only information that matters for any future decision is (i) the remaining budget, and (ii) the current location. The exact (fruitless) way in which this interval was covered is, at this point, immaterial. This, "memoryless" nature calls, again, for a dynamic programming algorithm for determining . We now provide a dynamic programming algorithm to compute the optimal δ-resolution tour. WLOG assume that $u_s = 0$ (the initial location is at the origin). For integral i, let $w_i = i\delta$. The points w_i, which we call the *resolution points*, are the only decision points for the algorithm. Set L and R to be such that w_L is the rightmost w_i to the left of all the stores and w_R the leftmost w_i to the right of all stores. We define two tables, $\mathsf{fail}[\cdot,\cdot,\cdot,\cdot]$ and $\mathsf{act}[\cdot,\cdot,\cdot,\cdot]$, such that for all ℓ, r, $L \leq \ell \leq 0 \leq r \leq R$, $e \in \{\ell, r\}$ (one of the end points), and budget B, $\mathsf{fail}[\ell, r, e, B]$ is the minimal failure probability[1] achievable for purchasing at the stores *outside* $[w_\ell, w_r]$, assuming a remaining budget of B, and starting at location w_e. Similarly, $\mathsf{act}[\ell, r, e, B]$ is the best act to perform in this situation ("left", "right", or "stop"). Given an initial budget B, the best achievable success probability is $(1-\mathsf{fail}[0,0,0,B])$ and the first move is $\mathsf{act}[0,0,0,B]$. It remains to show how to compute the tables. The computation of the tables is performed from the outside in, by induction on the number of remaining points. For $\ell = L$ and $r = R$, there are no more stores to search and $\mathsf{fail}[L, R, e, B] = 1$ for any e and B. Assume that the values are known for i remaining points, we show how to compute for $i+1$ remaining points. Consider $\mathsf{cost}[\ell, r, e, B]$ with $i+1$ remaining points. Then, the least failure probability obtainable by a decision to move right (to w_{r+1}) is:

$$F_R = \left(1 - \sum_{c \leq B-\delta} p^{r+1}(c)\right) \mathsf{fail}[\ell, r+1, r+1, B-\delta]$$

Similarly, the least failure probability obtainable by a decision to move left (to $w_{\ell-1}$) is:

$$F_L = \left(1 - \sum_{c \leq B-\delta} p^{\ell-1}(c)\right) \mathsf{fail}[\ell-1, r, \ell-1, B-\delta]$$

Thus, we can choose the act providing the least failure probability, determining both $\mathsf{act}[\ell, r, e, B]$ and $\mathsf{fail}[\ell, r, e, B]$. In practice, we compute the table only for B's in integral multiples of δ. This can add at most δ to the optimum. Also, we may place a bound $B^\delta_{\max}$ on the maximal B we consider in the table. In this case, we start filling the table with $w_L = -B^\delta_{\max}/\delta$ and $w_R = B^\delta_{\max}/\delta$, the furthest point reachable with budget $B^\delta_{\max}$.

Next, we show how to choose δ and prove the approximation ratio. Set $\lambda = \epsilon/9$. Let $\alpha = \min\{|u_s - u_{s+1}|, |u_s - u_{s-1}|\}$ - the minimum budget necessary to move away from the starting point, and $\beta = m^2|u_m - u_1| + \max\{c : \exists i, p^i(c) > 0\}$ - an upper bound on the total usable budget. We start by setting $\delta = \lambda^2\alpha$ and double it until $\delta > \lambda^2\beta$, performing the computation for all such values of δ. For such value of δ, we fill the tables (from scratch) for all values of B's in integral multiples of δ up to $B^\delta_{\max} = 2\lambda^{-2}\delta$. We now prove that for at least one of the choices of δ we obtain a $(1+\epsilon)$ approximation.

Consider a success probability p_{succ} and suppose that optimally this success probability can be obtained with budget B_{opt} using the tour T_{opt}. By *tour* we mean a list of actions ("right", "left" or "stop") at each decision point (which, in this case, are all store locations). We convert T_{opt} to a δ-resolution tour, T^δ_{opt}, as follows. For any $i \geq 0$, when T_{opt} moves for the first time to the right of w_i then T^δ_{opt} moves all the way to w_{i+1}. Similarly, for $i \leq 0$, when T_{opt} moves for the first time to the left of w_i then T^δ_{opt} moves all the way to w_{i-1}.

Note that T^δ_{opt} requires additional travel costs only when it "overshoots", i.e. when it goes all the way to the resolution

[1] Technically, it is easy to work with the failure probability instead of the success probability.

point while T_{opt}would not. This can either happen (i) in the last step, or (ii) when T_{opt} makes a direction change. Type (i) can happen only once and costs at most δ. Type (ii) can happen at most once for each resolution point, and costs at most 2δ. Suppose that T^{δ}_{opt} makes t *turns* (i.e. t directions changes). Then, the total additional travel cost of the tour T^{δ}_{opt} over T_{opt} is at most $(2t+1)\delta$. Furthermore, if we use T_{opt} with budget B_{opt} and T^{δ}_{opt} with budget $B_{opt}+(2t+1)\delta$ then at any store, the available budget under T^{δ}_{opt} is at least that available with T_{opt}. Thus, T^{δ}_{opt} is a δ-resolution tour that with budget at most $B_{opt}+(2t+1)\delta$ succeeds with probability $\geq p_{succ}$. Hence, our dynamic algorithm, which finds the optimal such δ-resolution tour will find a tour with budget $B^{\delta}_{opt} \leq B_{opt}+(2t+2)\delta$ obtaining at least the same success probability (one additional δ for the integral multiples of δ in the tables).

Since T^{δ}_{opt} has t-turns, T_{opt} must also have t-turns, with targets at t *distinct* resolution segments. For any i, the i-th such turn (of T_{opt}) necessarily means that T_{opt}moves to a point at least $(i-1)$ segments away, i.e. a distance of at least $(i-1)\delta$. Thus, for B_{opt}, which is at least the travel cost of T_{opt}, we have:[2]

$$B_{opt} \geq \sum_{i=1}^{t}(i-1)\delta = \frac{(t-1)(t)}{2}\delta \geq \frac{t^2}{4}\delta \qquad (1)$$

On the other hand, since we consider all options for δ in multiples of 2, there must be a $\hat{\delta}$ such that:

$$\lambda^{-2}\hat{\delta} \geq B_{opt} \geq \frac{\lambda^{-2}}{2}\hat{\delta} \qquad (2)$$

Combining (1) and (2) we get that $t \leq 2\lambda^{-1}$. Thus, the approximation ratio is:

$$\frac{B^{\hat{\delta}}_{opt}}{B_{opt}} \leq \frac{B_{opt}+2(t+1)\hat{\delta}}{B_{opt}} \leq 1+\frac{2(t+1)\hat{\delta}}{\lambda^{-2}\hat{\delta}/2} \qquad (3)$$

$$\leq 1+(8\lambda+4\lambda^2) \leq 1+\epsilon \qquad (4)$$

Also, combining (2) and (4) we get that

$$B^{\hat{\delta}}_{opt} \leq B_{opt}(1+\epsilon) \leq 2\lambda^{-2}\hat{\delta} = B^{\hat{\delta}}_{\max}$$

Hence, the tables with resolution $\hat{\delta}$ consider this budget, and $B^{\hat{\delta}}_{opt}$ will be found.

It remains to analyze the complexity of the algorithm. For any given δ there are $B^{\delta}_{\max}/\delta = 2\lambda^{-2}$ budgets we consider and at most this number of resolution points at each side of u_s, for each, there are two entries in the table. Thus, the size of the table is $\leq 8\lambda^{-6} = O(\epsilon^{-6})$. The computation of each entry takes $O(1)$ steps. We consider δ in powers of 2 up to $\beta \leq 2^n$, where n is the size of the input. Thus, the total computation time is $O(n\epsilon^{-6})$. We obtain:

Theorem 4 *For any $\epsilon > 0$, the Min-Budget problem can be approximated with a $(1+\epsilon)$ factor in $O(n\epsilon^{-6})$ steps.*

[2]Assuming that $t > 1$. If $t = 0, 1$ the additional cost is easily small by (2).

Multi-Agent

The algorithms for the single-agent case can be extended to the multi-agent case as follows.

Theorem 5 *With k agents:*

- *Multi-agent Min-Expected-Cost (path) can be solved in $O(d^2(\frac{m}{k})^{2k})$.*
- *Multi-agent Min-Budget (path) and multi-agent Max-Probability (path) with d possible prices can be solved in $O(m2^{-kd}(\frac{em}{kd})^{2kd})$.*
- *For any $\epsilon > 0$, multi-agent Min-Budget (path) can be approximated to within a factor of $(1+k\epsilon)$ in $O(n\epsilon^{-6k})$ steps (for arbitrary number of prices).*

The algorithms are analogous to the ones for the single agent case, with the additional complexity of coordinating between the agents. The details are omitted.

While the complexity in the multi-agent case grows exponentially in the number of agents, in most physical environments where several agents cooperate in exploration and search, the number of agents is relatively moderate. In these cases the computation of the agents' strategies is efficiently facilitated by the principles of the algorithmic approach presented in this paper.

If the number of agents is not fixed (i.e. part of the input) then, the complexity of all three variants grows exponentially. Most striking perhaps is that Multi-agent *Min-Budget* and *Max-Probability* problems are NP-complete even on the path with a *single price*. To prove this we again formulate the problems into a decision version- *Multi-Min-Budget-Decide* - Given a set of points S, a distance function *dis* : $S \times S \rightarrow R^+$, initial locations for all agents $(u_s^{(1)}, \ldots, u_s^{(k)})$, a price-probability function $p^{\cdot}(\cdot)$, a minimum success probability p_{succ} and maximum budget B, decide if success probability of at least p_{succ} can be achieved with a maximum budget B.

Theorem 6 Multi-Min-Budget-Decide *is NP-complete even on the path with a* single price.

Proof. An optimal policy defines for each time step which agent should move and in which direction. Since there are at most $2m$ time steps, it is easy to compute the success probability and the total cost in $O(m)$ steps, therefore the problem is in NP. The NP-Hard reduction is from the knapsack problem.

We assume WLOG that all the points are on the line. We use N agents and our line consists of $2N$ stores. N stores corresponds to the knapsack items, denote them by $u_{k_1}, ..., u_{k_N}$. The other N points are the starting point of the agents, $\{u_s^{(i)}\}_{i=1,..,N}$. We set the left most point to $u_s^{(1)}$ and the right most point to u_{k_N}. For all $1 \leq i \leq N-1$ set u_{k_i} right after $u_s^{(i)}$ and $u_s^{(i+1)}$ right after u_{k_i}. Set $|u_s^{(i)} - u_{k_i}| = s_i$ and $|u_{k_i} - u_s^{(i+1)}| = B+1$. See figure 2 for an illustration.

The price at all the nodes is $c_0 = 1$ and $p^{k_i}(1) = 1-2^{-v_i}$. Finally, set $B = C+1$ and $p_{succ} = 1-2^{-V}$.

For every agent i, the only possible movement is to node p_{k_i}, denote by $\gamma_i = 1$ if agent i moves to p_{k_i}, and 0 if not. Therefore, there is a selection of items that fit, i.e,

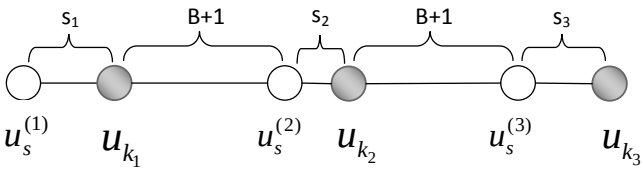

Figure 2: Reduction of knapsack to Multi-Min-Budget-Decide problem used in proof of Theorem 6, for N=3.

$\sum_{i=1}^{N} \delta_i s_i \leq C$, and the total value, $\sum_{i=1}^{N} \delta_i v_i$, is at least V iff there is a selection of agents that move such that $\sum_{i=1}^{N} \gamma_i s_i \leq B$, and the total probability $1 - \prod_{i=1}^{N} \gamma_i 2^{-v_i}$, is at least $p_{succ} = 1 - 2^{-V}$. □

This is in contrast to the single agent case where the single price case can be solved in $O(n)$ steps.

Discussion, Conclusions and Future Work

The integration of a changing search cost into economic search models is important as it improves the realism and applicability of the modeled problem. At the same time, it also dramatically increases the complexity of determining the agents' optimal strategies, precluding simple solutions such as easily computable reservation values (see for example "Pandora's problem" (Weitzman 1979)).

This paper, which is the first to consider economic search problems with non-stationary costs, considers physical environments where the search is being conducted in a metric space, with a focus on the the case of a path (i.e., a line). It presents a polynomial solution for the *Min-Expected-Cost* problem and an innovative approximation algorithm for the *Min-Budget* problem, which is proven to be NP-Complete.

The richness of the analysis given in the paper, covering three different variants of the problem both for a single-agent and multi-agent scenarios, lays the foundations for further analysis as part of future research. In particular, we see a great importance in extending the multi-agent models to scenarios where each agent is operating with a private budget (e.g. multiple Rovers, each equipped with a battery of its own), and finding efficient approximations for the Min-Expected-Cost problem where the number of agents is part of the input.

Acknowledgments. We are grateful to the reviewers for many important comments and suggestions, and especially for calling to our attention the relationship between the *Min-Expected-Cost* problem and MDPs. The research reported in this paper was supported in part by NSF IIS0705587 and ISF. Sarit Kraus is also affiliated with UMIACS.

References

Ausiello, G.; Leonardi, S.; and Marchetti-Spaccamela, A. 2000. *Algorithms and Complexity*. Springer Berlin / Heidelberg. chapter On Salesmen, Repairmen, Spiders, and Other Traveling Agents, 1–16.

Choi, S., and Liu, J. 2000. Optimal time-constrained trading strategies for autonomous agents. In *Proc. of MAMA'2000*.

Gal, S. 1980. *Search Games*. New York: Academic Press.

Hazon, N., and Kaminka, G. A. 2005. Redundancy, efficiency, and robustness in multi-robot coverage. In *IEEE Int. Conference on Robotics and Automation, (ICRA-05)*.

Koopman, B. O. 1980. *Search and Screening: General Principles with Historical Applications*. New York: Pergamon Press.

Koutsoupias, E.; Papadimitriou, C. H.; and Yannakakis, M. 1996. Searching a fixed graph. In *ICALP '96: Proceedings of the 23rd International Colloquium on Automata, Languages and Programming*, 280–289. London, UK: Springer-Verlag.

Lippman, S., and McCall, J. 1976. The economics of job search: A survey. *Economic Inquiry* 14:155–189.

McMillan, J., and Rothschild, M. 1994. Search. In Aumann, R., and Amsterdam, S., eds., *Handbook of Game Theory with Economic Applications*. Elsevier. chapter 27, 905–927.

Puterman, M. L. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience.

Sarne, D., and Kraus, S. 2005. Cooperative exploration in the electronic marketplace. In *Proceedings of AAAI'05*, 158–163.

Weitzman, M. L. 1979. Optimal search for the best alternative. *Econometrica* 47(3):641–54.