The Oil Searching Problem

Andrew McGregor[1], Krzysztof Onak[2⋆], and Rina Panigrahy[3]

[1] University of Massachusetts, Amherst. Email: mcgregor@cs.umass.edu
[2] Massachusetts Institute of Technology. Email: konak@mit.edu
[3] Microsoft Research Silicon Valley. Email: rina@microsoft.com

Abstract. Given n potential oil locations, where each has oil at a certain depth, we seek good trade-offs between the number of oil sources found and the total amount of drilling performed. The cost of exploring a location is proportional to the depth to which it is drilled. The algorithm has no clue about the depths of the oil sources at the different locations or even if there are any. Abstraction of the oil searching problem applies naturally to several life contexts. Consider a researcher who wants to decide which research problems to invest time into. A natural dilemma whether to invest all the time into a few problems, or share time across many problems. When you have spent a lot of time on one problem with no success, should you continue or move to another problem?

One could study this problem using a competitive analysis that compares the cost of an algorithm to that of an adversary that knows the depths of the oil sources, but the competitive ratio of the best algorithm for this problem is $\Omega(n)$. Instead we measure the performance of a strategy by comparing it to a weaker adversary that knows the set of depths of the oil sources but does not know which location has what depth. Surprisingly, we find that to find k oil sources there is a strategy whose cost is close to that of any adversary that has this limited knowledge of only the set of depths. In particular, we show that if any adversary can find k oil sources with drilling cost B while knowing the set of depths, our strategy finds $k-\tilde{O}(k^{5/6})$ sources with drilling cost $B(1+o(1))$. This proves that our strategy is very close to the best possible strategy in the total absence of information.

1 Introduction

Consider n potential oil locations where each has oil at a certain depth. The cost of exploring a location is proportional to the depth to which it is drilled. If a location has been drilled to depth d_1 at some point in the past, then drilling it do depth $d_2 > d_1$ costs $d_2 - d_1$. The algorithm has no clue about the depths of the oil sources at the different locations or even if there are any. The natural question then is what is a good strategy to go about drilling for oil. Our abstraction of the oil searching problem applies naturally to several life contexts. Consider a researcher who wants to decide which research problems to invest time into. A natural dilemma is that of whether one should invest all the time into few problems, or share time across many problems. When

⋆ The research was initiated during a summer internship at Microsoft Research Silicon Valley. The author is supported in part by a Symantec Research Fellowship, NSF grant 0732334, and NSF grant 0728645.

you have spent a lot of time at one problem with no success, should you continue or move to another problem? Is it better to continue since you already invested so much time or should you cut your losses and move to another problem? These dilemmas cut across several common decision-making processes including management strategies, investment decisions, and career changes.

While it is typical to study such an 'online' problem under competitive analysis that compares the cost incurred by an algorithm to that of an adversary that knows the depths of the oil sources, we note that in this problem the competitive ratio of the best algorithm is $\Omega(n)$; e.g., consider one location with oil at depth 1 and the other $n-1$ locations having no oil. While giving some insight, competitive analysis doesn't fully capture what constitutes a good strategy in practice. When should we give up on a research problem (drilling location) and move on to other? The approach we take here to measure the performance of a strategy is to compare its performance to a weaker adversary that knows the set of depths of the oil sources but does not know which location has what depth. Surprisingly, we find that to find k oil sources there is a strategy whose cost is close to that of any adversary that has this limited knowledge of only the set of depths. In particular, we show that if any adversary can find k oil sources in drilling cost B while knowing the set of depths, our strategy finds $k - \tilde{O}(k^{5/6})$ sources[4] of oil in budget $B(1+o(1))$. This essentially proves our strategy is 'very close' in performance to the best possible strategy in the total absence of information. When k is constant, our strategy incurs at most $O(\log n)$ times the cost of any strategy that knows the set of depths. We show that this is the best possible ratio.

Search Games: Our problem is closely related to the class of problems known as *search games*. In general, there are two players: the Hider and the Searcher. The Hider is hiding in a space (which can be a weighted graph, or some continuous metric space), and the goal of the Searcher is to locate the Hider by traversing the space and wants to minimize the traversed distance. An extensive description of search games can be found in the book of Gal [1] or the book of Alpern and Gal [2]. A well-known example of a search game is looking for lost key on a road when we do not know in which direction the key is and at what distance. It is easy to show that a near-optimal strategy is to alternate search direction doubling the search distance iteratively. Search games find applications in robotics when a robot searches for an object in unknown terrain.

A special case of the oil searching problem, where we are only looking for one oil source has been studied as the *cow-path problem*, where a cow searches for a grass field starting at a junction of several roads. A sequence of papers resulted in an optimal solution to this problem [3–5] in terms of the competitive-ratio of distance travelled to the distance from the field. As we stated before, our analysis style is very different as we do not compare ourselves to an all knowing adversary. Lastly we note that, in the same proceedings, Kirkpatrick [6] also studies variations of the cow-path and oil searching problems. His focus is on finding a single oil source.

Multi-armed bandits: Our problem is also related to work on multi-armed bandits (see, e.g., [7–9]) and there has been work in the multi-armed bandit setting that attempts to model some of the issues that arise when drilling for oil (see, e.g., [10–12]). In this problem we consider several arms of a bandit each of which is has a known "state." At

[4] the $\tilde{O}$ hides factors of the form $\log n$

each time step, one of the arms can be played resulting in a random payoff according to a distribution determined by the current state of the arm. Given a current state and the payoff, the arm may deterministically move into a new state. Various objective functions are considered including maximizing the discounted reward over an infinite horizon or minimizing the regret over a finite number of steps. Our objective functions do not map to either of these objective function although, in spirit, the goal is similar.

The Problems: We are given n locations and each contains oil at some depth from $\{1, 2, 3, \ldots\}$. The set of depths of the oil is $\{d_1, \ldots, d_n\}$. The algorithm has no information about the set of depths. The adversary on the other hand knows the set of depths but does not know which location has which depth – the n locations are assigned a random permutation of these depths which is unknown to the adversary. Let $n_{\leq d}$ be the number of oil wells whose depth is at most d. The *cost of obliviousness* is the ratio between the performance (appropriately defined) of an optimal algorithm that knows the set of depths to an optimal algorithm that has no information of the depths whatsoever.

Our Contributions: We present an algorithm that expects to find $k - \tilde{O}(k^{5/6})$ sources of oil in budget $B(1 + o(1))$, where the adversary, who knows the set of depths, would expect to find at most k sources in expected budget B. We develop the algorithm in stages, first studying the problem for one oil source (in Section 2) getting a cost of obliviousness of $O(\log n)$; we also show that this is tight. Next in Section 3 we study a variant of the problem where the set of depths are chosen from a distribution; again the adversary knows the distribution and the algorithm doesn't. Section 3.1 investigates the strategy the adversary who knows the distribution should use; Section 3.2 shows how the algorithm can emulate this strategy by first trying to learn the distribution during the initial few drills. In Section 4, we generalize our algorithm to the case when the set of depths is simply a set and is not necessarily chosen from a distribution (note the subtle difference between the two cases as picking from a distribution corresponds to selecting the depth of each location independently with replacement whereas when there is a set of n depths, these depths are matched to the locations in some permutation.) Again the algorithm essentially treats the set as a distribution and makes use of the algorithm for the distribution case. In the Appendix we extend the problem to trees.

2 Warm-up: Finding a single oil source

In this section we consider the expected amount of drilling required to find one oil source or to find one oil source with constant probability. We show that there exists an oblivious algorithm that in expectation uses at most a factor $O(\log n)$ more drilling than an algorithm that knows the set of the oil depths. This is best possible.

We start by showing that if we wish to maximize the probability of finding a single oil source given some budget, we may restrict to algorithms that choose $k \leq n$ locations at random and drill to some set of depths at these locations.

Lemma 1. *Consider a fixed set of depths. Let $\mathcal{A}$ be an algorithm that for any assignment of depths to locations, finds oil with probability p using budget B. There is a set* *of k depths $b_1, \ldots, b_k$ satisfying $\sum b_i \leq B$ such that the probability that one finds oil by drilling to depths b_i in k different random locations is at least p.*

Proof. Assume that the locations are randomly permuted before $\mathcal{A}$ starts solving the problem. This does not impact the probability with which $\mathcal{A}$ solves the problem. Consider coin tosses of $\mathcal{A}$. For some setting of coin tosses the probability of finding oil is at least p. We simulate $\mathcal{A}$ for this setting of coin tosses. We tell the algorithm that it hasn't found oil, as long as it hasn't drilled enough to find oil for all permutations of locations. We stop when $\mathcal{A}$ stops, or we know it must have found oil. This exhibits $\mathcal{A}$'s exploration pattern. We set k to the number of locations $\mathcal{A}$ drilled in. We also set b_i, $1 \leq i \leq k$, to the depths $\mathcal{A}$ drilled to in consecutive locations. Clearly $\sum b_i \leq B$, because $\mathcal{A}$ does at most B drilling. By using the same exploration pattern as $\mathcal{A}$ with locations permuted at random, and therefore, with drilling applied to random locations, one can also find oil with probability at least p. □

Lemma 2. *Drilling to depths $b_1 \geq \ldots \geq b_k$ at $k \leq n$ random, distinct locations finds oil with probability $1-\prod_{i=1}^{k}(1 - n_{\leq b_i}/(n-i+1))$[5], which is at most $\sum_{i=1}^{k} n_{\leq b_i}/(n-i+1)$.*

Proof. The proof is by induction on k: For $k = 1$, drilling at a random location to depth b_1, yields oil with probability $n_{\leq b_1}/n$. Given that oil is not found after drilling to depths $b_1, \ldots, b_{k-1}$, the probability that oil is found at depth at most b_k when drilling at the next random location is $n_{\leq b_k}/(n-k+1)$ because $n-k+1$ locations remain and$n_{\leq b_k}$ of them still have oil at depth at most b_k since we drilled to depths greater than b_k in previous steps and did not find oil. The probability of finding an oil source is exactly $1 - \prod_{i=1}^{k}(1 - n_{\leq b_i}/(n-i+1))$.

Furthermore, we have:

$$\prod_{i=1}^{k}\left(1 - \frac{n_{\leq b_i}}{n-i+1}\right) \geq 1 - \sum_{i=1}^{k}\frac{n_{\leq b_i}}{n-i+1},$$

that is,

$$1 - \prod_{i=1}^{k}\left(1 - \frac{n_{\leq b_i}}{n-i+1}\right) \leq \sum_{i=1}^{k}\frac{n_{\leq b_i}}{n-i+1}.$$

□

Using the above two lemmas, we get the following fact.

Lemma 3. *Finding an oil source with probability p requires $(p/4) \cdot \min_d (dn/n_{\leq d})$ drilling.*

Proof. Consider any algorithm that uses budget B to find oil with probability p. By Lemma 1, there is a sequence $b_1 \geq b_2 \geq \ldots \geq b_k$ of depths such that $\sum_{i=1}^{k} b_i \leq B$, and drilling at k different random locations to depths b_i, $1 \leq i \leq k$, exposes an oil source with probability at least p.

[5] Dynamic programming can maximize these probabilities for $\sum b_i \leq B$ in $\mathrm{poly}(B, n)$ time.

Algorithm 1: For finding a single source of oil when the depths are not known.

$b := 1$
while *oil not found* **do**
  for $i = 0, \dots, \lceil \log n \rceil$ **do**
    Drill in $\min\{2^i, n\}$ random locations to depth $b/2^i$
  $b := 2b$

Let $k' = \min\{k, \lceil n/2 \rceil\}$. We claim that the sequence $b_1, b_2, \dots, b_{k'}$ finds oil with probability at least $p/2$. If $k = k'$, the claim is trivially true. Otherwise, consider two sequences: $b_1, \dots, b_{k'}$, and $b_{k'+1}, \dots, b_k$. When applied to random locations, they find an oil source with probability p_1 and p_2, respectively. It must be the case that $p_1 + p_2 \geq p$, since otherwise, by the union bound, the sequence $b_1, \dots, b_k$ would find oil with probability less than p. Furthermore, $p_1 \geq p_2$, because even the sequence b_1, $\dots$, $b_{k-k'}$, where $k - k' \leq k'$, is such that $b_i \geq b_{k'+i}$, $1 \leq i \leq k - k'$, and therefore, this sequence exposes oil with probability at least the same as $b_{k'+1}, \dots b_k$. If follows that $p_1 \geq p/2$.

For the sequence $b_1, \dots, b_{k'}$, we have by Lemma 2 that

$$p/2 \leq \sum_{i=1}^{k'} \frac{n_{\leq b_i}}{n - i + 1} \leq \sum_{i=1}^{k'} \frac{n_{\leq b_i}}{n/2},$$

that is,

$$p/4 \leq \sum_{i=1}^{k'} \frac{n_{\leq b_i}}{n}.$$

In the above inequality, a unit of drilling assigned to a given b_i, contributes $n_{\leq b_i}/(b_i n)$ to the right-hand side, and no unit can contribute more than $\max_d(n_{\leq d}/(dn))$. Therefore,

$$p/4 \leq B \cdot \max_d \frac{n_{\leq d}}{dn},$$

and

$$B \geq \frac{p}{4} \cdot \min_d \frac{dn}{n_{\leq d}}.$$

□

Theorem 4. *Algorithm 1 finds a single oil source with constant probability using at most a factor $O(\log n)$ more drilling than that required by an algorithm that knows the set of the oil depths.*

Proof. Suppose all depths are powers of 2 and note that this assumption at most doubles the amount of drilling. The cost of the k-th round in Algorithm 1 is $O(2^k \log n)$, and therefore, the total cost of the first k rounds is also $O(2^k \log n)$. Let $d^* = \operatorname{argmin}_d (dn/n_{\leq d})$, and let OPT be the optimal amount of drilling to find a single oil source with the given constant probability. By Lemma 3, $\mathrm{OPT} = \Omega(d^* n/n_{\leq d^*})$. The algorithm finds a source of oil with constant probability by the time it reaches level $\log(d^* n/n_{\leq d^*})$. But at this point at most $O(\mathrm{OPT} \log n)$ drilling has been performed. □

Theorem 5. *Any oblivious algorithm that finds a single oil source with constant probability may need to use a factor $\Omega(\log n)$ more drilling than that required by an algorithm that knows the set of the oil depths.*

Proof. Consider the following sets $\mathcal{I}_i$ of depths, for $i \in [\log n]$. $\mathcal{I}_i$ consists of 2^i oil sources at depth 2^i, and the other oil sources are at depth ∞. For each i, there is an algorithm that finds oil with probability $1/2$ by drilling to depth 2^i in $n/2^i$ locations, which gives n drilling in total. Assume that each $\mathcal{I}_i$ appears with the same probability. Suppose that the oblivious algorithm finds oil with constant probability by drilling only d units in total. By Lemma 1, we can assume that the algorithm chooses a few depths d_1 to d_k that it drills to in different locations. We know that $\sum d_i = d$. The expected number of oil sources found by this strategy bounds the probability of finding at least one oil source. Let X_i be the event that drilling to depth d_i exhibits oil. We have

$$\Pr[\exists X_i = 1] \leq \sum_i E[X_i] = \sum_i \sum_{1 \leq j \leq \log d_i} \frac{1}{\lfloor \log n \rfloor} \cdot \frac{2^j}{n} \leq \frac{\sum_i 2d_i}{n \cdot \lfloor \log n \rfloor} = \frac{2d}{n \cdot \lfloor \log n \rfloor}.$$

Hence, the oblivious algorithm must drill $\Omega(n \log n)$ in total, which is a factor $\Omega(\log n)$ more than the optimal strategy drills for any of the inputs. □

Remark: Similarly, one can show that for a given set of depths, Algorithm 1 uses in expectation at most $O(\log n)$ factor more than an algorithm that knows the input and minimizes the expected amount of drilling to find one oil source. Any oblivious algorithm must use $\Omega(\log n)$ more drilling in some cases.

3 Multiple sources when depths are chosen from a distribution

If the adversary can find k oil sources in B drilling cost, the objective of the algorithm is to match this cost as well as possible. In this section, we will consider a special case where the depths of the oil sources are independently chosen from the same distribution and there are infinitely many locations to dig; again the adversary knows the distribution and the algorithm does not. In the next section we will generalize such an algorithm to the case when there are finitely many locations and the depths are not chosen from a distribution but simply assigned using a set of n depths that is known to the adversary.

3.1 Adversary's strategy when distribution is known

In this section we will show a simple procedure for minimizing the expected amount of drilling to find k oil sources for the adversary who knows the distribution. It turns out that when there are infinitely many locations, the best algorithm for the adversary is to simply to dig upto a fixed depth repeatedly; that fixed depth depends on the distribution. Consider an algorithm for the adversary that in a given location drills until a fixed depth

d, unless it finds oil earlier. Let X be a random variable equal to the depth at which oil occurs. The expected payoff, i.e., the number of oil sources found by the algorithm, divided by the expected amount of drilling equals

$$g(d) = \frac{E[\text{payoff}]}{E[\text{amount of drilling}]} = \frac{\Pr[X \le d]}{d \cdot \Pr[X > d] + \sum_{1 \le j \le d} j \cdot \Pr[X = j]}.$$

Define $x^\star$ to be the smallest $d \in \mathbb{Z}_+ \cup \{\infty\}$ that maximizes the above value. The next lemma shows that no algorithm exploring one location can achieve a better ratio of the expected payoff to the expected amount of drilling than $g(x^\star)$. The proof starts by showing that any algorithm that explores one oil source is equivalent to an algorithm that selects a depth d from some distribution, and drills to d. Then, the ratio of the expected payoff to the expected amount of drilling cannot be greater than the maximum such ratio for depths in the distributions, which is bounded by $g(x^\star)$.

Lemma 6. $E[\text{payoff}]/E[\text{amount of drilling}] \le g(x^\star)$ *for any algorithm $\mathcal{A}$ that explores only one location.*

Proof (Proof of Lemma 6). We show that $\mathcal{A}$ can be transformed into an algorithm $\mathcal{B}$ that picks d from some distribution over $\mathbb{Z}_+ \cup \{\infty\}$, and then drills to depth d, unless if finds oil before. If that is the case, and the probability that $d = i$ is p_i, then

$$\frac{E[\mathcal{B}\text{'s payoff}]}{E[\mathcal{B}\text{'s amount of drilling}]} = \frac{\sum p_i \cdot E[\mathcal{B}\text{'s amount of drilling}|d=i] \cdot g(i)}{\sum p_i \cdot E[\mathcal{B}\text{'s amount of drilling}|d=i]} \le g(x^\star)$$

Consider the behavior of $\mathcal{A}$. Let q_i be the probability that $\mathcal{A}$ stops drilling, after it discovers that there is no oil at depth i. Define $p_i = q_i \cdot \prod_{1 \le j < i}(1 - q_j)$, for finite i, and we set p_∞ to $\prod_{j=1}^{\infty}(1 - q_j)$. The sum of all p_i is 1. Let t_i be the probability that there is oil at depth i, given there is no oil at depth at most $i - 1$. The algorithm has no impact on this probability, since it is independent of its knowledge.

We now show that $\mathcal{A}$ behaves in the same way as $\mathcal{B}$ for the probabilities that we have defined. The probability that $\mathcal{A}$ stops, not having found oil up to depth i is $(1 - t_i)q_i \cdot \prod_{1 \le j < i}(1 - q_j)(1 - t_j)$. The same probability for $\mathcal{B}$ is

$$p_i \cdot \prod_{1 \le j \le i}(1 - t_j) = q_i \prod_{1 \le j < i}(1 - q_j) \prod_{1 \le j \le i}(1 - t_j) = (1 - t_i)q_i \cdot \prod_{1 \le j < i}(1 - q_j)(1 - t_j),$$

that is, it is identical. The probability that $\mathcal{A}$ finds oil at depth i is $t_i \cdot \prod_{1 \le j < i}(1 - t_j)(1 - q_j)$. For $\mathcal{B}$, the probability is

$$\sum_{j \ge i} p_j t_i \prod_{1 \le k < i}(1 - t_k) = \left(\prod_{1 \le k < i}(1 - q_k)\right) t_i \prod_{1 \le k < i}(1 - t_k) = t_i \cdot \prod_{1 \le k < i}(1 - t_k)(1 - q_k).$$

Hence, $\mathcal{A}$ and $\mathcal{B}$ behave in the very same way. □

The proof of the next lemma shows that even if drilling in multiple locations, the ratio of the expected payoff to the expected amount of drilling is still at most $g(x^\star)$. This gives a lower bound on the expected amount of drilling to find k oil sources.

Algorithm 2: For finding k oil sources when the distribution of depths are known.

while *less than k oil sources found* **do**

pick a new location, and drill until depth $x^\star$ or until oil found

Lemma 7. *Let $\mathcal{A}$ be an algorithm that finds exactly k oil sources. $\mathcal{A}$'s expected amount of drilling is at least $k/g(x^\star)$.*

Proof. Let the random variable Y_i, $i \in \mathbb{Z}_+$, be an indicator of whether $\mathcal{A}$ found oil at location i. Let D_i be the amount of drilling $\mathcal{A}$ did at location i. We have

$$\frac{k}{E[\text{amount of drilling}]} = \frac{E[\sum Y_i]}{E[\sum D_i]} = \frac{\sum E[Y_i]}{\sum E[D_i]}.$$

For each i, let p_i be the probability that $\mathcal{A}$ drills at location i. By Lemma 6, we have $E[Y_i|\mathcal{A} \text{ drills at } i]/E[D_i|\mathcal{A} \text{ drills at } i] \leq g(x^\star)$. Hence,

$$\frac{\sum E[Y_i]}{\sum E[D_i]} = \frac{\sum p_i \cdot E[Y_i|\mathcal{A} \text{ drills at } i]}{\sum p_i \cdot E[D_i|\mathcal{A} \text{ drills at } i]} \leq \frac{\sum p_i \cdot g(x^\star) \cdot E[D_i|\mathcal{A} \text{ drills at } i]}{\sum p_i \cdot E[D_i|\mathcal{A} \text{ drills at } i]} = g(x^\star).$$

Thus, the expected amount of drilling must be at least $k/g(x^\star)$. □

Theorem 8. *Algorithm 2 minimizes the expected amount of drilling to find k oil sources, when the distribution is known.*

Proof. By Lemma 7 we know that the expected amount of drilling to find k oil sources is at least $k/g(x^\star)$. It suffices to show that the expected time to find one oil source is exactly $1/g(x^\star)$. Let $p = \Pr[X \leq x^\star]$, and let D be the expected amount of drilling in a new location. By definition, $p/D = g(x^\star)$. Since the depth at each location is independent, by Wald's theorem, the expected amount of drilling until one oil source is found is exactly $D \cdot \sum_{i=0}^{\infty}(1-p)^i = p/g(x^\star) \cdot 1/p = 1/g(x^\star)$. □

3.2 Extending to an algorithm when distribution is unknown

We will now show how the adversary's strategy can be extended to an algorithm that does not know the distribution. The essential idea is to estimate the distribution in the initial few drillings, and then emulate the adversary's strategy. Our oblivious algorithm finds k oil sources with probability $1-\delta$, and performs in expectation at most a factor $(1+\tilde{O}(\sqrt[5]{k^{-1}\log\delta^{-1}}))$ more drilling than the expected amount of drilling for the adversary's algorithm. Let B_1 be the min. expected amount of drilling to find one oil source.

Lemma 9. *There is an algorithm that approximates B_1 up to a factor of $O(\log B_1)$ with probability $1-\delta$. The expected amount of drilling is $O(B_1 \cdot \log B_1 \cdot \log(1/\delta))$.*

Algorithm 3: For finding a single source of oil when the distribution is unknown.

Let $b := 1$
while *no oil found* **do**
  For each $i \in \{1, \ldots, \log b\}$, drill 2^i locations up to depth $b/2^i$
  $b := \sqrt{2} \cdot b$
Return b

Proof. Note that the minimum expected budget to get one oil source and the minimum budget to get a single oil source with probability $1/2$ are within a constant factor. Thus, it suffices to approximate the latter. To do this we run Algorithm 3 a total of $O(\log \delta^{-1})$ times. There is a constant C_1 such that the probability that we find oil using a budget $b \leq C_1 \cdot B_1 / \log B_1$ is less than $1/4$. Otherwise, we could get a better minimum expected budget to find a single oil source. On the other hand, there is a constant C_2 such that the probability that $b \geq C_2 \cdot B_1$ is less than $1/4$. Hence, by the Chernoff bounds, if we have $O(\log(1/\delta))$ samples and take the median of them, we get a value that is between $C_1 \cdot B_1 / \log B_1$ and $b \geq C_2 \cdot B_1$ with probability $1 - \delta$. Once b gets greater than some constant times B_1, it finds oil in each iteration with probability greater than $1/2$. Since the total budget spent grows by a factor smaller than and bounded away from 2, the expected amount of drilling in each run of the auxiliary algorithm is $O(B_1 \log B_1)$. □

Fact 10 *Let X be a Bernoulli variable and let $\delta, \epsilon \in (0,1)$. $O(\delta^{-1}\epsilon^{-2})$ samples of X suffice to distinguish $\Pr[X = 0] \leq \delta$ from $\Pr[X = 0] \geq \delta(1+\epsilon)$ with probability $2/3$.*

Lemma 11. *Let t satisfy $\Pr[X > t] \leq \epsilon$ and $x^\star > t$. Then, $g(t) \geq g(x^\star) \cdot (1 - \epsilon)$.*

Proof. Let T and D be the expected amount of drilling if we stop at depth t and $x^\star$ respectively. Then $g(t) = (1 - \Pr[X > t])/T \geq (1 - \epsilon)/D \geq (1 - \epsilon) g(x^\star)$. □

Lemma 12. *There is an algorithm that finds t such that $g(t) \geq g(x^\star)(1 - \epsilon)$ with probability $1 - \delta$. The expected amount of drilling used is $\tilde{O}(B_1 \epsilon^{-4} \cdot \log(1/\delta))$.*

Proof. We first run the algorithm of Lemma 9 to approximate the minimum expected budget B_1 required to discover a single oil source. We get a budget upper-bound $b = O(B_1 \log B_1)$. We now argue that our desired t is $O(B_1 \log B_1 \log(1/\epsilon))$. By Lemma 11, we need not care about big depths. Since we can find oil with probability at least $1/2$ up to depth $O(B_1 \log B_1)$ using a budget of at most b, it makes no sense to drill deeper than $d = O(B_1 \log B_1 \log(1/\epsilon))$ to find oil with probability higher than $1 - \epsilon$ in any location, since there is a more effective method that explores only small depths. Next we round up each drilling depth to a power of $(1+\epsilon)$. This decreases $g(x)$ by at most a factor of $1 + \epsilon$ for any x, We now have $s = \log_{1+\epsilon} d$ possible values for t. We have

$$\frac{g((1+\epsilon)^i)}{1+\epsilon} \leq \frac{\Pr[X \leq (1+\epsilon)^i]}{1 + \sum_{0 \leq j \leq i-1} \epsilon(1+\epsilon)^j \cdot \Pr[X \geq (1+\epsilon)^j]} \leq g((1+\epsilon)^i) \; .$$

Now using $O(d\epsilon^{-4} \log(s/\delta))$ drilling, we learn for each of the possible choices $(1+\epsilon)^i$ for t the probability $\Pr[X \geq (1+\epsilon)^i]$ up to an additive term of ϵ^2. If the probability is smaller than ϵ for some $(1+\epsilon)^i$, we can assume by Lemma 11 that t is bounded by $(1+\epsilon)^i$. Otherwise, our approximation gives a good multiplicative approximation for the denominator of $g((1+\epsilon)^i)$.

It remains to estimate the numerator. We are only interested in i such that the probability of finding oil up to depth $(1+\epsilon)^i$ using a total budget of b is $\Omega(1)$. This means that we are interested in $(1+\epsilon)^i$ such that $\Pr[X \leq (1+\epsilon)^i] = \Omega((1+\epsilon)^i/b)$. To approximate each $\Pr[X \geq (1+\epsilon)^i]$ up to a factor of $1+\epsilon$, given it is larger than $\Omega((1+\epsilon)^i/b)$, it suffices to use $O\left(s(\log s)b\epsilon^{-2}\log(s\delta^{-1})\right)$ drilling by Fact 10. Hence with $\tilde{O}(B_1\epsilon^{-4}\log(1/\delta))$ drilling we find t with $g(t) \geq g(x^\star)/(1+\epsilon)^{O(1)}$ with probability $1-\delta$. Rescaling ϵ we get the required approximation. □

Corollary 13. *There is an algorithm that with probability* $1-\delta$ *uses* $B\left(1+\tilde{O}\left(\sqrt[5]{\log(1/\delta)k^{-1}}\right)\right)$ *drilling in expectation to find* k *oil sources, while the minimum expected amount of drilling to find* k *oil sources is* $B = kB_1$.

Proof. We use the algorithm described in the proof of Lemma 12 to find a t such that $g(t)$ approximates $g(x^\star)$. By running an algorithm that drills up to depth t, with probability $1-\delta$, the expected amount of drilling we use to find k oil sources is $kB_1(1+\epsilon)+\tilde{O}\left(B_1\epsilon^{-4}\cdot\log(1/\delta)\right)$. Setting $\epsilon = \sqrt[5]{\log(1/\delta)/k}$ gives the result. □

4 Generalizing to an arbitrary set of depths

We will now genaralize the algorithms to the case when the depths of the locations are a (random) permutation of a set of n depths that is known only to the adversary.

4.1 Adversary's strategy when set of depths is known

In this section we will study algorithms for the case when the set of depths are known but not the depth of each oil source separately. We will provide an efficient (polynomial time) algorithm for the adversary whose perfomance is close to that of the best possible (perhaps exponential time) algorithm. If any algorithm finds k' oil sources in expectation using B amount of drilling in expectation then the proposed algorithm can find k sources in expectation using $B(1+\epsilon)$ expected drilling where $k' \leq k + \tilde{O}(\sqrt{k/\epsilon})$.

Assume that all the depths in the input instance are rounded up to powers of $(1+\epsilon)$; this only increases the budget by a factor of $(1+\epsilon)$. Denote different depths by $H_i = (1+\epsilon)^i$ for $i \leq r = O(\epsilon^{-1}\log B)$. Define $h_i = H_i - H_{i-1}$, where $H_{-1} = 0$.

We will say that (B, k) is an *achievable* solution if there exists an algorithm whose expected drilling is B and the expected number of oil sources found is k. We say that the pair (B, k) is *feasible* in the following program if the program has a solution. We write n_d and $n_{\geq d}$ to denote the number of oil sources at depth exactly d and at least d, respectively. Here $m_i = n_{H_i}/n_{\geq H_i}$. x_i corresponds to the number of locations that we drill from depth H_{i-1} to H_i, which results in expectation in $x_i m_i$ oil source discoveries.

$$k \leq \sum \min(m_i x_i, x_i - x_{i+1}); \quad B \geq \sum h_i x_i; \quad x_r \leq \ldots \leq x_0 \leq n \qquad (1)$$

Algorithm 4: For finding k oil with a known set of depths in budget B

1. Compute a (B, k) solution to Program 2.
2. Start drilling x_0 random locations to depth h_0.
3. Let A_i be the number of locations drilled to depth H_i where no oil was found. Among these, choose $\min(x_{i+1}, A_i)$ at random and drill these to depth H_{i+1}.

The first inequality corresponds to stating that at least k oil sources are found in expectation. Note that $m_i x_i$ is the expected number of oil sources found while drilling from depth H_{i-1} to H_i; however it cannot exceed $x_i - x_{i+1}$. The second inequality ensures that the total budget is at most B. The third simply ensures that the solution is meaningful as the number of wells drilled to increasing depths must be decreasing. We show (proof deferred to the Appendix) that Program 1 is equivalent to:

$$k \leq \sum m_i x_i; \quad B \geq \sum h_i x_i; \quad \forall\, i \in [r-1] : x_{i+1} \leq (1-m_i)x_i; \quad x_0 \leq n \tag{2}$$

Lemma 14. *If (B, k) is feasible in Program 2 then Algorithm 4 finds k oil sources in expectation while spending budget B.*

Proof. Consider Algorithm 4 and let Y_i be the number of locations that we drill from depth H_{i-1} to H_i. Since $Y_i \leq x_i$ the total amount of drilling is at most B. We will argue that the expected number of oil sources it finds is at least k. To prove this, we alter the algorithm so that even if less than x_i locations are available to drill to depth H_i as we have found oil in some of them, we will pretend to also continue drilling $x_i - Y_i$ locations where we found oil earlier. This can only increase the cost but will not change the number of oil sources found. The number of locations drilled to depth H_i is exactly x_i. The number of oil sources G_i found at depth H_i is $x_i - A_i$. Now, $E[G_i] = x_i - E[A_i]$. But $E[A_i] = (1-m_i)E[Y_i] \leq (1-m_i)x_i$. So $E[G_i] \geq m_i x_i$, and hence, $\sum_i E[G_i] \geq \sum_i m_i x_i \geq k$. □

Next we will lower bound the performance of the best possible algorithm. We will show that if (B, k) is achievable, then it must be feasible in the following program.

$$k \leq \sum_i \min((m_i+\epsilon_i)x_i+\epsilon'_i, x_i - x_{i+1}); \;\; B = \sum_i h_i x_i; \;\; x_{i+1} \leq \ldots \leq x_0 \leq n \tag{3}$$

where $\epsilon_i = \tilde{O}(\sqrt{m_i/n_{\geq H_i}} + 1/n_{\geq H_i})$ and $\epsilon'_i = m_i/(n_{\geq H_i})^{\Omega(1)}$ as guaranteed by the following lemma (proof deferred to the Appendix) for $n_{\geq H_i}$ boxes with $m_i n_{\geq H_i}$ of them containing gold.

Lemma 15 (Boxes of Gold). *Consider n boxes of which pn contain a gold ingot. For any randomized algorithm that opens B boxes and finds G ingots, $E[G] \leq (p + \epsilon)E[B] + \epsilon'$ where $\epsilon = \tilde{O}(\sqrt{p/n} + 1/n), \epsilon' = p/n^{\Omega(1)}$.*

Lemma 16. *If (B, k') is an achievable solution for an instance of the oil searching problem with known depths then it is a feasible solution for Program 3*

Proof. Assume that there is an algorithm $\mathcal{A}^*$ that spends budget B in expectation and finds k' oil in expectation. Let $x_i = E$[number of wells drilled to depth at least H_i]. Let $g_i = E$[number of wells found at depth H_i]. Then, by Lemma 15, $g_i \leq (m_i+\epsilon_i)x_i+\epsilon'_i$ as we can think of each location with the depth at least H_i as a box and the ones with oil at depth H_i as boxes containing gold. Also clearly $g_i \leq x_i - x_{i+1}$ as we can continue drilling to depth H_{i+1} only if we have not found gold already. So $k' = \sum g_i \leq \sum_i \min((m_i+\epsilon_i)x_i + \epsilon'_i, x_i - x_{i+1})$ and $B = \sum_i h_i x_i$. □

Lemma 17. *If (B, k') is feasible for Program 3 then (B, k) is feasible for Program 1 where $k' \leq k + \tilde{O}(\sqrt{k/\epsilon})$.*

Proof. Consider the feasible solution in Program 3 that satisfies $k' \leq \sum_i \min((m_i + \epsilon_i)x_i+\epsilon'_i, x_i-x_{i+1})$. Substituting this solution in Program 1 is feasible if $k = \sum_i \min(m_i x_i, x_i - x_{i+1})$. Let Q denote the set $\{i : (m_i+\epsilon_i)x_i + \epsilon'_i \leq x_i - x_{i+1}\}$. Now $k' - k \leq \sum_{i\in Q} x_i\epsilon_i + \epsilon'_i \leq \tilde{O}(\sum_{i\in Q}\sqrt{m_i x_i})) + o(1)$. Since $\sum_{i\in Q} m_i x_i \leq k'$, this is at most $\tilde{O}(\sqrt{k}' r) = \tilde{O}(\sqrt{k/\epsilon})$. □

The above lemmas, with the depth rounding, gives the main theorem of this section:

Theorem 18. *If (B, k') is an achievable solution for an instance of the oil searching problem with known depths then Algorithm 4 finds in expectation k oil sources in expected cost $B(1+\epsilon)$ where $k' \leq k + \tilde{O}(\sqrt{k/\epsilon})$.*

4.2 Extending to an algorithm that does not know the set of depths

We now study algorithms that are oblivious to the set of depths. We compare our solution to that which can be achieved by the adversary with knowledge of the set of depths but not the depth of each source separately. We show that if an adversary that knows the set of depths, expects to find k' oil sources, and expected to perform B drilling in expectation, then our algorithm expects to find k oil sources where $k' \leq k + \tilde{O}(k^{5/6})$ sources and peforms $B(1 + o(1))$ drilling in expectation.

Let us view the set of depths S as a distribution $D(S)$ obtained by picking a random location from the set S. Any algorithm that drills one location at random from the set S is an algorithm that drills one location with depth from the distribution $D(S)$. Now we know from the results in Section 3.1 that the optimal k/B is obtained by drilling up to depth $d = x^*$, which maximizes $g(d)$. Let us say that a solution to Program 2 is tight if the first two inequalities involving B and k are equalities. Any tight solution to Program 2 can be converted to an algorithm for drilling locations from distribution $D(S)$ achieving and vice-versa. For instance, an algorithm that drills all locations up to depth $d = H_s$ can be realized by setting $x_0 = n$ and $x_{i+1} = (1-m_i)x_i$, for all $i+1 \leq s$, and $x_{i+1} = 0$, for all $i + 1 \geq s + 1$, in Program 2. Let (B_s, k_s) denote the values of B and k in this tight solution. Similarly we can view a tight solution to Program 2 as a strategy to drill one location drawn from the distribution $D(S)$ by scaling all x_i by x_0 resulting in $(B/x_0, k/x_0)$ expected budget and payoff. The strategy continues drilling from H_i to H_{i+1} with probability $x_{i+1}/((1-m_i)x_i)$. Clearly $k_s/B_s = g(H_s)$. The algorithm of Lemma 12 can be used to compute a near optimal depth H_t so that $g(H_t) \geq g(x^*)(1-\epsilon)$. This is done by sampling locations from S with replacement.

Lemma 19. *(B, k) is a tight solution for Program 2 if and only if it can be expressed as a linear combination $\sum_i \alpha_i(B_i, k_i)$ such that $\sum_i \alpha_i \leq 1$.*

Proof. Without loss of generality we may assume that $x_0 = n$ as any solution to Program 2 can be scaled to satisfy this. We know that any such solution (B, k) to Program 2 can be viewed as an algorithm to drill one location chosen from distribution $D(S)$ achieving expected budget and payoff: $(B/n, k/n)$. We know from the proof of Lemma 6 that any such algorithm can be viewed as a distribution or linear combination of the strategies that always drill a location up to H_i. The latter result corresponds to $(B_s/n, k_s/n)$ expected budget and payoff. The only if part is trivial. □

Theorem 20. *If (B, k) is a tight solution for Program 2, then there is a $(B, k(1-\epsilon))$ solution for Program 2 such that $x_{i+1} = (1 - m_i)x_i$, for all $i \leq t-1$ and either $x_0 = n$, or $x_t > 0$. This essentially corresponds to a solution that first explores all locations to depth up to H_t, and only then drills to greater depths.*

Proof. Express (B, k) as $\sum_i \alpha_i(B_i, k_i)$. The essential observation is that shifting budget to α_t from other α_i can only increase k. It suffices to show that we can convert the (B, k) solution to a $(B, k(1-\epsilon))$ solution, where either $\alpha_i = 0$, for all $i < t$, and $\sum_i \alpha_i = 1$, or only α_t is non-zero and others 0. Assume first for simplicity that $g(H_t) = g(x^*)$. Hence transferring budget from any other (B_i, k_i) to (B_t, k_t) only increases k. However we need to respect the constraint $\sum_i \alpha_i \leq 1$. Also note that transferring budget from components $i < t$ only decreases $\sum_i \alpha_i$ as B_i is non-decreasing in i. So we can always move to an improved (or same quality) solution where $\alpha_i = 0$ for all $i < t$. Further, as long as $\sum_i \alpha_i < 1$ we can transfer budget from higher components $i > t$ till $\sum_i \alpha_i$ hits 1. We stop only if either $\alpha_i = 0$ for all $i \neq t$ or $\sum_i \alpha_i = 1$ and $\alpha_i = 0$ for all $i < t$. Now if $g(H_t) = g(x^*)(1-\epsilon)$ then each transfer of budget will have a rate of return that is smaller by a factor of ϵ. So total payoff loss is at most ϵk. □

Remark: Observe that the above theorem also implies that there is an optimal solution to 2 that first spends all its budget drilling to x^* and then recursively solves the remaining instance. The solution is thus to drill according to a sequence of depths x^*'s for the different instances until either all locations are drilled or budget is exhausted. Also (assuming $g(H_t) = g(x^*)$) the $g(x^*)$'s found in the different recursions is non-increasing as otherwise there is a better value of x^* at the recursion after which it increased. Further if we use H_t at each recursive step instead of x^* the total loss in payoff is at most ϵk as the budget moved in the different recursions is disjoint (the budget moved in a recursion is never moved again in another recursion). If $\tilde{B}$ is the expected budget and $\tilde{k}$ is the expected payoff used in a recursion then $\tilde{B}/\tilde{k} = g(H_t)$.

See Algorithm 5. The depth of the recursion is at most $r = \log G/\epsilon$ as there are only r distinct depths. This gives:

Lemma 21. *If (B, k) is a feasible solution for Program 2 then Algorithm 5 finds $k(1-\epsilon)$ oil sources in expectation with budget B (ignoring the cost of computing $g(H_t)$).*

We also need to take into account the cost $O(\epsilon^{-4}\log(1/\delta)/g(x^*))$ of computing $g(H_t)$ at the beginning each recursive step. Besides this the only non-determinism in the budget used and payoff is in the last step of the recursion; at previous steps both are fixed as we are dealing with a set of depths.

Algorithm 5: Finding multiple sources with budget B and unknown set of depths

1. Treating the set of depths as a distribution, compute t such that $g(H_t) \geq g(x^*)(1-\epsilon)$. This is done with the algorithm (of Lemma 12) for estimating $g(x^*)$ on the distribution $D(S)$ by sampling locations with replacement.
2. Drill all locations to depth H_t (except those that hit oil earlier) unless budget is exhausted. If budget is left over we have found all oil sources at depth at most H_t. The remaining locations are all drilled to depth H_t.
3. In such a case recursively explore the remaining locations with the remaining budget.

Theorem 22. *If (B, k') is an achievable solution for the oil searching problem with known depths then with probability $1 - r\delta$, algorithm 5 finds in expectation $k(1-\epsilon) - \tilde{O}(\log(1/\delta)/\epsilon^5)$ sources in expectation, where $k' \leq k + \tilde{(}\sqrt{k/\epsilon})$, and spends $B(1+\epsilon)$ budget. For $\epsilon = o(k^{-1/6})$, this amounts to $k - \tilde{O}(k^{5/6})$ sources in budget $B(1+o(1))$.*

Proof. If (B, k') is a feasible solution for the oil searching problem with known depths then there is a (B, k) solution to Program 2 where $k' \leq k + O(\min\{k, \sqrt{kr}\})$. To bound the cost of computing $g(H_t)$, we increase B by a factor of ϵ and in each recursion only allocate at most ϵ fraction of the remaining budget to computing $g(H_t)$. If cost of computing $g(H_t)$ is more than ϵ fraction of remaining budget in a certain recursion, then we can stop the algorithm as this means $O(\log(1/\delta)/(\epsilon^5 g(x^*)) \geq \epsilon B$, which means we can find at most $g(x^\star)B \leq O(\log(1/\delta)/\epsilon^5)$ more sources in expectation. Thus, if we stop, we are only giving up $O(\log(1/\delta)/\epsilon^5)$ sources. □

References

1. Gal, S.: Search Games. Academic Press (1980)
2. Alpern, S., Gal, S.: The Theory of Search Games and Rendezvous. Kluwer Academic Publishers (2003)
3. Baeza-Yates, R.A., Culberson, J.C., Rawlins, G.J.E.: Searching in the plane. Inf. Comput. **106**(2) (1993) 234–252
4. Kao, M.Y., Reif, J.H., Tate, S.R.: Searching in an unknown environment: An optimal randomized algorithm for the cow-path problem. Inf. Comput. **131**(1) (1996) 63–79
5. Kao, M.Y., Ma, Y., Sipser, M., Yin, Y.L.: Optimal constructions of hybrid algorithms. J. Algorithms **29**(1) (1998) 142–164
6. Kirkpatrick, D.: Hyperbolic dovetailing. In: ESA. (2009)
7. Robbins, H.: Some aspects of the sequential design of experiments. Bull. Amer. Math. Soc **58** (1952) 527–535
8. Gittins, J.C., Jones, D.M.: A dynamic allocation index for the sequential design of experiments. In: Progress in Statistics: European Meeting of Statisticians. Volume eds 1. (1974) 241–266
9. Whittle, P.: Arm-acquiring bandits. The Annals of Probability **9** (1981) 284–292
10. Benkherouf, L., Pitts, S.: On a multidimensional oil exploration problem. Journal of Applied Mathematics and Stochastic Analysis **2005**(2) (2005) 97–118
11. Benkherouf, L., Glazebrook, K., Owen, R.: Gittins indices and oil exploration. J. Roy. Statist. Soc. Ser. B **54** (1992) 229–241
12. Grayson, C.: Decisions Under Uncertainty: Drilling Decisions by Oil and Gas Operators. Harvard, Division of Research, Graduate School of Business Administration (1960)

Algorithm 6: For finding an oil source in a tree T with budget B

1. Let d be the degree of the root of T.
2. Assign d tokens to one random branch outgoing from the root. Assign $d/2$ tokens to two other remaining branches outgoing from the root. Assign $d/4$ to four other branches outgoing from the root. Continue in this way until you assign some number of tokens to each branch.
3. Divide B between branches proportionally to the number of tokens assigned to each branch.
4. For each branch follow the corresponding edge until you run out of budget or you reach a internal vertex v. In the latter case, recursively run TreeExhaustive(B', T'), where B' is the remaining budget, and T' is the subtree rooted at v.

A Extension to Finding Oil in Symmetric Trees

In this section, we consider searching for oil in trees. We focus on weighted rooted trees that are "fully symmetric". Formally, a tree is *symmetric* if for each internal node v, all the subtrees rooted at children of v are identical. We assume that all weights are positive integers. The exploration starts at the root, and on each path from the root to a leaf oil appears at exactly one place that is at integral distance from the root. We want to minimize the expected drilling time until one oil source is found. We compare against an algorithm that knows all the oil depths and how they are organized in the tree, but is initially unaware of which subtree is which at any internal node. In this section, we prove the following theorem.

Theorem 23. *For symmetric trees, there is an oblivious algorithm that finds a single oil source using in expectation at most $O(\prod_{i=1}^{k} 2(1 + \log d_i))$ times the optimal expected amount of drilling.*

The reason why we focus on symmetric trees is that in the asymmetric case, an algorithm may almost directly go to the closest oil source by observing asymmetries of the tree. Therefore, it may be impossible to give a good bound on the cost of obliviousness.

Exhaustive Strategies. We shall bound the difference between the budget used by an arbitrary algorithm and the budget required by an *exhaustive strategy*. We define the exhaustive strategy as Algorithm 6.

Lemma 24. *Let T be a symmetric tree with k levels of internal nodes. Let the number of children of every node on level i be d_i. Let $\mathcal{A}$ be an algorithm with budget B that finds oil in this setting with probability at least $1/2$. Let $B' = B \cdot \prod_{i=1}^{k} 2(1 + \log d_i)$. The exhaustive strategy with budget B' also finds oil with probability at least $1/2$.*

Proof. Simulate $\mathcal{A}$ on the tree and tell $\mathcal{A}$ that no oil has been found as long as that is possible. Look at the exploration pattern produced by the algorithm. If the algorithm is not deterministic and it may produce multiple patterns, based on its coin tosses, there must be a pattern that maximizes the probability of finding oil. In such a case, consider that pattern. We will bound the pattern by an exhaustive strategy. The proof proceeds

Algorithm 7: For finding a single oil source in a tree

Let $b := 1$ be the initial budget.
while *no oil found* **do**
    Run Algorithm 6 with T and b
    $b := \sqrt{2} \cdot b$

by induction in a bottom-up manner, and we show that at level i, we increase the budget by at most a factor of $2(1 + \log d_i)$. Consider an internal node with d children, and suppose that all subtrees of the nodes are already explored via exhaustive strategies. Suppose that the whole strategy uses a budget of B. We sort the budgets $B_1 \geq \ldots \geq B_d$ assigned to subtrees, and find the smallest budget $B^\star$ such that if we sort subbudgets $B_1^\star \geq \ldots \geq B_d^\star$ used by the exhaustive strategy with budget $B^\star$, they will dominate the current subbudgets, i.e., $B_i^\star \geq B_i$ for all i. It is clear that by replacing the current strategy with the exhaustive strategy of budget $B^\star$, we do not decrease the probability of finding oil. We also claim that $B^\star$ is at most $2(1 + \log d_i)$ times greater than the budget before the modification. Since $B^\star$ is minimal, there is an i such that $B_i^\star = B_i$. Let j be the block index corresponding to $B_i^\star$ in the exhaustive strategy budget assignment. We claim that the entire budget $B^\star t_j w_j / (\sum_k t_k w_k)$ of the block is at most 2 times B. This is easy to see by noticing that either $j = 1$, in which case the claim is trivial, or $j > 1$, and B is greater than $B_i^\star \cdot 2^{j-2}$, which in turn is at least a half of the budget of the j-th block. Since we have at most $1 + \log d$ blocks, the modification grows the initial budget B at most $2(1 + \log d)$ times. □

Proof (Proof of Theorem 23). Let $\mathcal{A}$ be an optimal algorithm. If the expected amount of drilling of $\mathcal{A}$ is B, then $\mathcal{A}'$ that simulates $\mathcal{A}$ and stops whenever it has used $2B$ drilling finds oil with probability at least $1/2$, by Markov's bound. Hence, by Lemma 24, there is a budget $B' = B \cdot O(\prod_{i=1}^{k} 2(1 + \log d_i))$ such that any exhaustive strategy with budget $\geq B'$ finds oil with probability at least $1/2$.

Consider Algorithm 7. What is the expected amount of drilling that this algorithm needs to find oil? We can assume that the algorithm does not find oil as long as $b < B'$. The amount of drilling the algorithm uses by then is $O(B')$. As soon as $b \geq B'$, the algorithm finds oil with probability at least $1/2$ in each iteration. Thus, the expected amount of drilling is at most $O(B') + B'\sqrt{2} + 2^{-1}B'\left(\sqrt{2}\right)^2 + 2^{-2}B'\left(\sqrt{2}\right)^3 + \cdots = O(B')$. □

B Omitted Proofs

B.1 Equivalence of Program 1 and Program 2

Lemma 25. *(B, k) is feasible in Program 1 iff it is feasible in Program 2.*

Proof. Clearly a feasible solution to Program 2 is also a solution to Program 1 because $x_i - x_{i+1} \geq x_i - (1 - m_i)x_i = m_i x_i$. For the other direction we will show how to convert a feasible solution in Program 1 to a solution where each $x_{i+1} \leq (1-m_i)x_i$. We prove this by induction on j, the largest i for which this is not the case. Consider $k^* = \sum_{i \geq j} \min(m_j x_j, x_j - x_{j+1}) = x_j - x_{j+1} + \sum_{i>j} \min(m_i x_i, x_i - x_{i+1})$ and $B^* = \sum_{i \geq j} h_i x_i = h_j(x_j - x_{j+1}) + h_j x_{j+1} + \sum_{i>j} h_i x_i$. Observe that $\sum_{i>j} \min(m_i x_i, x_i - x_{i+1})$ can be written as αx_{j+1}, where $\alpha \leq 1$, and $h_j x_{j+1} + \sum_{i>j} h_i x_i$ can be written as $\beta h_j x_{j+1}$ where $\beta \geq 1$. So $k^* = x_j - x_{j+1} + \alpha x_{j+1}$ and $B^* = h_j(x_j - x_{j+1} + \beta x_{j+1})$. So by scaling down all x_i's for $i > j$ so that $x_{j+1} = x_j - m_j x_j$ we can only increase k^* and decrease B^*. □

B.2 Lemma 15

Proof. Without loss of generality we consider an algorithm that opens the boxes in some (random) predetermined order. With respect to this ordering define the following random variables, let G_l be the number of ingots found among the first $n/2$ boxes if the total number of boxes opened is at most $n/2$ and 0 otherwise. Let G_r be the total number of ingots found if the total number of boxes opened is greater that $n/2$. Let B_l be the number of boxes opened if this is at most $n/2$ and 0 otherwise. Let B_r be the total number of boxes opened if this is greater than $n/2$. So $G = G_l + G_r$ and $B = B_l + B_r$.

We wish to show,

$$E[G] \leq (p + \epsilon)E[B] + 1/n^{\Omega(1)} .$$

Let X_t be the number of boxes of gold among the first t boxes. Then, by the sampling without replacement version of the Hoeffding bounds,

$$\Pr[X_t/t > p + \tilde{O}(\sqrt{p/n} + 1/n)] \leq 1/n^{\Omega(1)}.$$

Hence with probability at least for $1 - 1/n^{\Omega(1)}$, $X_t/t \leq p + \epsilon$ where for all $t > n/2$ where $\epsilon = O(\sqrt{p/n} + 1/n)$. Call this event A_1 Let Y_t be the number of boxes of gold among the last t boxes. By an identical argument to that above, we know that with probability at least for $1 - 1/n^{\Omega(1)}$, $Y_t/t \leq p + \epsilon$ for all $t > n/2$. Call this event A_2 and let $A = A_1 \cap A_2$.

Let p_t be the probability that the algorithm opens t boxes conditioned on the event A. Then,

$$\frac{E[G_r|A]}{E[B_r|A]} \leq \frac{\sum_{t>n/2} p_t t(p+\epsilon)}{n \sum_{t>n/2} p_t} \leq (p + \epsilon) .$$

Now consider $E[G_l|A]/E[B_l|A]$. Conditioned on the event A, we know that each new box that is opened has gold with probability at most $(p + \epsilon)$. Hence, we may argue that that random process that reveals gold in each box independently with probability $(p + \epsilon)$ stochastically dominates the process where pn ingots are randomly distributed between the n boxes. But once we assume independence, it is easy to show that $E[G_l|A]/E[B_l|A] \leq (p + \epsilon)$.

Consequently,

$$\frac{E[G|A]}{E[B|A]} = \frac{E[G_l + G_r|A]}{E[B_l + B_r|A]} \leq \max\left(\frac{E[G_l|A]}{E[B_l|A]}, \frac{E[G_r|A]}{E[B_r|A]}\right) \leq (p + \epsilon) .$$

Finally,

$$
\begin{aligned}
E[G] &= \Pr[\neg A]E[G|\neg A] + \Pr[A]E[G|A] \\
&\le \Pr[\neg A]pn + \Pr[A]E[B|A](p+\epsilon) \\
&\le \Pr[\neg A]pn + E[B](p+\epsilon) \\
&\le p/n^{\Omega(1)} + E[B](p+\epsilon)
\end{aligned}
$$

□