The Dependent Doors Problem: An Investigation into Sequential Decisions without Feedback

Amos Korman, Yoav Rodeh

The Dependent Doors Problem: An Investigation into Sequential Decisions without Feedback*

Amos Korman[1] and Yoav Rodeh[2]

[1]CNRS and University Paris Diderot, Paris, France, amos.korman@irif.fr
[2]Weizmann Institute of Science, Rehovot, Israel, yoav.rodeh@gmail.com

April 20, 2017

Abstract

We introduce the *dependent doors problem* as an abstraction for situations in which one must perform a sequence of possibly dependent decisions, without receiving feedback information on the effectiveness of previously made actions. Informally, the problem considers a set of d doors that are initially closed, and the aim is to open all of them as fast as possible. To open a door, the algorithm knocks on it and it might open or not according to some probability distribution. This distribution may depend on which other doors are currently open, as well as on which other doors were open during each of the previous knocks on that door. The algorithm aims to minimize the expected time until all doors open. Crucially, it must act at any time without knowing whether or which other doors have already opened. In this work, we focus on scenarios where dependencies between doors are both positively correlated and acyclic.

The fundamental distribution of a door describes the probability it opens in the best of conditions (with respect to other doors being open or closed). We show that if in two configurations of d doors corresponding doors share the same fundamental distribution, then these configurations have the same optimal running time up to a universal constant, no matter what are the dependencies between doors and what are the distributions. We also identify algorithms that are optimal up to a universal constant factor. For the case in which all doors share the same fundamental distribution we additionally provide a simpler algorithm, and a formula to calculate its running time. We furthermore analyse the price of lacking feedback for several configurations governed by standard fundamental distributions. In particular, we show that the price is logarithmic in d for memoryless doors, but can potentially grow to be linear in d for other distributions.

We then turn our attention to investigate precise bounds. Even for the case of two doors, identifying the optimal sequence is an intriguing combinatorial question. Here, we study the case of two cascading memoryless doors. That is, the first door opens on each knock independently with probability p_1. The second door can only open if the first door is open, in which case it will open on each knock independently with probability p_2. We solve this problem almost completely by identifying algorithms that are optimal up to an additive term of 1.

1 Introduction

Often it is the case that one must accomplish multiple tasks whose success probabilities are dependent on each other. In many cases, failure to achieve one task will tend to have a more negative affect on the success probabilities of other tasks. In general, such dependencies may be quite complex, and balancing the work load between different tasks becomes a computational challenge. The situation is further complicated if the ability to detect whether a task has been accomplished is limited. For example, if task B highly depends on task A then until A is accomplished, all efforts invested in B may be completely wasted. How should one divide the effort between these tasks if feedback on the success of A is not available?

In this preliminary work we propose a setting that captures some of the fundamental challenges that are inherent to the process of decision making without feedback. We introduce the *dependent doors*

*This work has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 648032).

problem, informally described as follows. There are $d \geq 2$ doors (representing tasks) which are initially closed, and the aim is to open all of them as fast as possible. To open a door, the algorithm can "knock" on it and it might open or not according to some governing probability distribution, that may depend on other doors being open or closed[1]. We focus on settings in which doors are positively correlated, which informally means that the probability of opening a door is never decreased if another door is open. The governing distributions and their dependencies are known to the algorithm in advance. Crucially, however, during the execution, it gets no direct feedback on whether or not a door has opened unless all d doors have opened, in which case the task is completed.

This research has actually originated from our research on heuristic search on trees [4]. Consider a tree of depth d with a treasure placed at one of its leaves. At each step the algorithm can "check" a vertex, which is child of an already checked vertex. Moreover, for each level of the tree, the algorithm has a way to compare the previously checked vertices on that level. This comparison has the property that if the ancestor of the treasure on that level was already checked, then it will necessarily be considered as the "best" on that level. Note, however, that unless we checked all the vertices on a given level, we can never be sure that the vertex considered as the best among checked vertices in the level is indeed the correct one. With such a guarantee, and assuming that the algorithm gets no other feedback from checked vertices, any reasonable algorithm that is about to check a vertex on a given level, will always choose to check a child of the current best vertex on the level above it. Therefore, the algorithm can be described as a sequence of levels to inspect. Moreover, if we know the different distributions involved, then we are exactly at the situation of the dependent doors problem. See Appendix A for more details on this example.

Another manifestation of d dependent doors can arise in the context of cryptography. Think about a sequence of d cascading encryptions, and separate decryption protocols to attack each of the encryptions. Investing more efforts in decrypting the i'th encryption would increase the chances of breaking it, but only if previous encryptions where already broken. On the other hand, we get no feedback on an encryption being broken unless all of them are.

The case of two doors can serve as an abstraction for exploration vs. exploitation problems, where it is typically the case that deficient performances on the exploration part may result in much waste on the exploitation part [10, 17]. It can also be seen as the question of balance between searching and verifying in algorithms that can be partitioned thus [1, 15]. In both examples, there may be partial or even no feedback in the sense that we don't know that the first procedure succeeded unless the second one also succeeds.

For simplicity, we concentrate on scenarios in which the dependencies are *acyclic*. That is, if we draw the directed dependency graph between doors, then this graph does not contain any directed cycles. The examples of searching and verifying and the heuristic search on trees can both be viewed as acyclic. Moreover, despite the fact that many configurations are not purely acyclic, one can sometimes obtain a useful approximation that is.

To illustrate the problem, consider the following presumably simple case of two dependent memoryless doors. The first door opens on each knock independently with probability $1/2$. The second door can only open if the first door is open, in which case it opens on each knock independently, with probability $1/2$. What is the sequence of knocks that minimizes the expected time to open both doors, remembering that we don't know when door 1 opens? It is easy to see that the alternating sequence $1, 2, 1, 2, 1, 2, \ldots$ results in 6 knocks in expectation. Computer simulations indicate that the best sequence gives a little more than 5.8 and starts with $1, 1, 2, 2, 1, 2, 1, 2, 2, 1, 2, 2, 1, 2, 1, 2$. Applied to this particular scenario, our theoretical lower bound gives 5.747, and our upper bound gives a sequence with expected time 5.832.

1.1 Context and Related Work

This paper falls under the framework of decision making under uncertainty, a large research subject that has received significant amount of attention from researchers in various disciplines, including computer science, operational research, biology, sociology, economy, and even psychology and cognition, see, *e.g.,* [2, 3, 5, 6, 7, 8, 9, 16].

Performing despite limited feedback would fit the framework of reinforced learning [17] and is inherent to the study of exploration vs. exploitation type of problems, including Multi-Armed Bandit problems [10]. In this paper we study the impact of having no feedback whatsoever. Understanding this extreme scenario may serve as an approximation for cases where feedback is highly restricted, or limited in its impact. For

[1] Actually, the distribution associated with some door i may depend on the state of other doors (being open or closed) not only at the current knock, but also at the time of each of the previous knocks on door i.

example, if it turns out that the price of lacking feedback is small, then it may well be worth to avoid investing efforts in complex methods for utilizing the partial feedback.

Of particular interest is the case of two doors. As mentioned, difficulties resulting from the lack of feedback can arise when one aims to find a solution by alternating between two subroutines: Producing promising candidate solutions and verifying these candidates. Numerous strategies are based on this interplay, including heuristics based on brute force or trail and error approaches [1, 15], sample and predict approaches [11, 14, 17], iterative local algorithms [12, 13], and many others. Finding strategies for efficiently balancing these two tasks can be therefore applicable.

1.2 Setting

There are $d \geq 2$ doors and each door can be either open or closed. Doors start closed, and once a door opens it never closes. To open a door, an algorithm can knock on it and it might open or not according to some probability distribution. The goal is to minimize the expected number of knocks until all doors open. Crucially, the algorithm has no feedback on whether or not a door has opened, unless all doors have opened, in which case the task is completed.

The probability that a door opens may depend on the state of other doors (being open or closed) at the time of the current knock as well as on their state during each of the previous knocks on the door. For example, the probability that a certain knock at door i succeeds may depend on the number of previous knocks on door i, but counting only those that were made while some other specific door j was open. The idea behind this definition is that the more time we invest in opening a door the more likely it is to open, and the quality of each knock depends on what is the state of the doors it depends on at the time of the knock.

Below we provide a semi-formal description of the setting. The level of detail is sufficient to understand the content of the main text, which is mainly concerned with independent and cascading configurations. The reader interested in a more formal description of the model is deferred to Appendix B.

A specific setting of doors is called a *configuration* (normally denoted $\mathcal{C}$). This includes a description of all dependencies between doors and the resulting probability distributions. In this paper we assume that the dependency graph of the doors is acyclic, and so we may assume that a configuration describes an ordering of the doors, such that each door depends only on lower index doors. Furthermore, we assume that the correlation between doors is positive, *i.e.*, a door being open can only improve the chances of other doors to open.

Perhaps the simplest configuration is when all doors are *independent* of each other. In this case, door i can be associated with a function $p_i : \mathbb{N} \to [0, 1]$, where $p_i(n)$ is the probability that door i is not open after knocking on it n times. Another family of acyclic configurations are *cascading* configurations. Here, door i cannot open unless all doors of lower index are already open. In this case, the configuration can again be described by a set of functions $\{p_i\}_{i=1}^{d}$, where $p_i(n)$ describes the probability that door i is not open after knocking on it n times, where the count starts only after door $i-1$ is already open.

In general, given a configuration, each door i defines a non-decreasing function $p_i : \mathbb{N} \to [0, 1]$, called the *fundamental distribution* of the door, where $p_i(n)$ is the probability that the door is not open after knocking on it n times in the best of conditions, *i.e.*, assuming all doors of lower index are open. In the case of independent and cascading configurations, the fundamental distribution p_i coincides with the functions mentioned above. Two doors are *similar* if they have the same fundamental distribution. Two configurations are *similar* if for every i, door i of the first configuration is similar to door i of the second.

When designing an algorithm, we will assume that the configuration it is going to run in is known. As there is no feedback, a deterministic algorithm can be thought of as a possibly infinite sequence of door knocks. A randomized algorithm is therefore a distribution over sequences, and as all of them will have expected running time at least as large as that of an optimal sequence (if one exists), the expected running time of a randomized algorithm cannot be any better. Denote by $\mathbb{T}_{\mathcal{C}}(\pi)$, the expected time until all doors open when running sequence π in configuration $\mathcal{C}$. We define $\mathbb{T}_{\mathcal{C}} = \min_{\pi} \mathbb{T}_{\mathcal{C}}(\pi)$. By Claim 23 in Appendix B.3, there exists a sequence achieving this minimum. Therefore, by the aforementioned arguments, we can restrict our discussion to deterministic algorithms only.

If we had feedback we would knock on each door until it opens, and then continue to the next. Denoting by $E_i = \sum_{n=0}^{\infty} p_i(n)$ the expected time to open door i on its own, the expected running time then does not depend on the specific dependencies between doors at all, and is $\sum_i E_i$. Also, this value is clearly optimal. To evaluate the impact of lacking feedback for a configuration $\mathcal{C}$, we therefore define:

$$\texttt{Price}(\mathcal{C}) = \frac{\mathbb{T}_{\mathcal{C}}}{\sum_i E_i}$$

Obviously $\texttt{Price}(\mathcal{C}) \geq 1$, and for example, if all doors start closed and open after just 1 knock, it is in fact equal to 1. Claim 22 in Appendix B.2 shows that $\texttt{Price}(\mathcal{C}) \leq d$.

1.3 Our Results

We have two main results. The first one, presented in Section 2, states that any two similar configurations have the same optimal running time up to a constant factor. We stress that this constant factor is universal in the sense that it does not depend on the specific distributions or on the number of doors d.

Furthermore, given a configuration, we identify an algorithm that is optimal for it up to a constant factor. We then show that for configurations where all doors are similar, there is a much simpler algorithm which is optimal up to a constant factor, and describe a formula that computes its approximate running time. We conclude Section 2 by analysing the price of lacking feedback for several configurations governed by standard fundamental distributions. In particular, we show that the price is logarithmic in d for memoryless doors, but can potentially grow to be linear in d for other distributions.

We then turn our attention to identify exact optimal sequences. Perhaps the simplest case is the case of two cascading memoryless doors. That is, the first door opens on each knock independently with probability p_1. The second door can only open if the first door is open, in which case it opens on each knock independently, with probability p_2. In Section 3 we present our second main result: Algorithms for these configurations that achieve the precise optimal running time up to an additive term of 1.

On the technical side, to establish such an extremely competitive algorithm, we first consider a semi-fractional variant of the problem and find a sequence that achieves the precise optimal bound. We then approximate this semi-fractional sequence to obtain an integer solution losing only an additive term of 1 in the running time. A nice anecdote is that in the case where $p_1 = p_2$ and are very small, the ratio of 2-knocks over 1-knocks in the sequence we get approaches the golden ratio. Also, in this case, the optimal running time approaches $3.58/p_1$ as p_1 goes to zero. It follows that in this case, the price of lacking feedback tends to $3.58/2$ and the price of dependencies, *i.e.*, the multiplicative gap between the cascading and independent settings, tends to $3.58/3$.

2 Near Optimal Algorithms

The following important lemma is proved in Appendix B.1 using a coupling argument:

Lemma 1. *Consider similar configurations $\mathcal{C}, \mathcal{X}$ and $\mathcal{I}$, where $\mathcal{X}$ is cascading and $\mathcal{I}$ is independent. For every sequence π, $\mathbb{T}_{\mathcal{I}}(\pi) \leq \mathbb{T}_{\mathcal{C}}(\pi) \leq \mathbb{T}_{\mathcal{X}}(\pi)$. This also implies that $\mathbb{T}_{\mathcal{I}} \leq \mathbb{T}_{\mathcal{C}} \leq \mathbb{T}_{\mathcal{X}}$.*

The next theorem presents a near optimal sequence of knocks for a given configuration. In fact, by Lemma 1, this sequence is near optimal for any similar configuration, and so we get that the optimal running time for any two similar configurations is the same up to a universal multiplicative factor.

Theorem 2. *There is a polynomial algorithm[2], that given a configuration $\mathcal{C}$ generates a sequence π such that $\mathbb{T}_{\mathcal{C}}(\pi) = \Theta(\mathbb{T}_{\mathcal{I}})$. In fact, $\mathbb{T}_{\mathcal{C}}(\pi) \leq 2 + 4\mathbb{T}_{\mathcal{I}} \leq 2 + 4\mathbb{T}_{\mathcal{C}}$.*

Proof. Denote by $p_1, \ldots, p_d$ the fundamental distributions of the doors of $\mathcal{C}$. For a finite sequence of knocks α, denote by $\texttt{SC}_{\mathcal{C}}(\alpha)$ the probability that after running α in configuration $\mathcal{C}$, some of the doors are still closed. Note that if α is *sorted*, that is, if all knocks on door 1 are done first, followed by the knocks on doors 2, etc., then $\texttt{SC}_{\mathcal{X}}(\alpha) = \texttt{SC}_{\mathcal{I}}(\alpha)$.

We start by showing that for any T, we can construct in polynomial time a finite sequence α_T of length T that maximizes the probability that all doors will open, *i.e.*, minimizes $\texttt{SC}_{\mathcal{I}}(\alpha_T)$. As noted above, if we sort the sequence, this is equal to $\texttt{SC}_{\mathcal{X}}(\alpha_T)$.

The algorithm follows a dynamic programming approach, and calculates a matrix A, where $A[i,t]$ holds the maximal probability that a sequence of length t has of opening all of the doors $1, 2, \ldots, i$. All the entries $A[0, \cdot]$ are just 1, and the key point is that for each i and t, knowing all of the entries in $A[i, \cdot]$, it is easy to calculate $A[i+1, t]$:

$$A[i+1,t] = \max_{k=0}^{t} A[i, t-k] \cdot (1 - p_{i+1}(k))$$

[2] A polynomial algorithm in our setting generates the next knock in the sequence in polynomial time in the index of the knock and in d, assuming that reading any specific value of any of the fundamental distributions of a door takes constant time.

Calculating the whole table takes $O(dT^2)$ time, and $A[d,T]$ will give us the highest probability a sequence of length T can have of opening all doors. Keeping tabs on the choices the max in the formula makes, we can get an optimal sequence α_T, and can take it to be sorted.

Consider the sequence $\pi = \alpha_2 \cdot \alpha_4 \cdots \alpha_{2^n} \cdots$. The complexity of generating this sequence up to place T is $O(dT^2)$, and so this algorithm is polynomial. Our goal will be to compare $\mathbb{T}_{\mathcal{X}}(\pi)$ with $\mathbb{T}_{\mathcal{I}}(\pi^\star)$, where $\pi^\star$ is the optimal sequence for $\mathcal{I}$.

The following observation stems from the fact that for any natural valued random variable X, $\mathtt{E}[X] = \sum_{n=0}^{\infty} \mathtt{Pr}[X > n]$ and $\mathtt{Pr}[X > n]$ is a non-increasing function of n.

Observation 3. Let $\{a_n\}_{n=1}^{\infty}$ be a strictly increasing sequence of natural numbers, and X be some natural valued random variable. Then:

$$\sum_{n=1}^{\infty}(a_{n+1}-a_n)\mathtt{Pr}[X > a_{n+1}] \leq \mathtt{E}[X] \leq a_1 + \sum_{n=1}^{\infty}(a_{n+1}-a_n)\mathtt{Pr}[X > a_n]$$

For a sequence π, denote by $\pi[n]$ the prefix of π of length n. In this terminology, $\mathbb{T}_{\mathcal{C}}(\pi) = \sum_{n=0}^{\infty} \mathtt{SC}_{\mathcal{C}}(\pi[n])$. Setting $a_n = 2 + 4 + \ldots + 2^n$ in the right side of Observation 3, and letting X be the number of rounds until all doors open when using π, we get:

$$\begin{aligned}\mathbb{T}_{\mathcal{X}}(\pi) &\leq 2 + \sum_{n=1}^{\infty} 2^{n+1} \cdot \mathtt{SC}_{\mathcal{X}}(\pi[2+\ldots+2^n]) \leq 2 + \sum_{n=1}^{\infty} 2^{n+1} \cdot \mathtt{SC}_{\mathcal{X}}(\alpha_{2^n}) \\ &= 2 + \sum_{n=1}^{\infty} 2^{n+1} \cdot \mathtt{SC}_{\mathcal{I}}(\alpha_{2^n}) \leq 2 + \sum_{n=1}^{\infty} 2^{n+1} \cdot \mathtt{SC}_{\mathcal{I}}(\pi^\star[2^n]) \leq 2 + 4\mathbb{T}_{\mathcal{I}}(\pi^\star)\end{aligned}$$

The last step is using Observation 3 with $a_n = 2^{n-1}$. Theorem 2 concludes. □

2.1 Configurations where all Doors are Similar

In this section we focus on configurations where all doors have the same fundamental distribution $p(n)$. We provide simple algorithms that are optimal up to a universal constant, and establish the price of lacking feedback with respect to a few natural distributions. Corresponding proofs appear in Appendix C.

2.1.1 Simple Algorithms

Let us consider the following very simple algorithm $A_{\mathtt{simp}}$. It runs in phases, where in each phase it knocks on each door once, in order. As a sequence, we can write $A_{\mathtt{simp}} = (1, 2, \ldots, d)^{\infty}$. Let $X_1, \ldots, X_d$ be i.i.d. random variables taking positive integer values, satisfying $\mathtt{Pr}[X_i > n] = p(n)$. The following is straightforward:

Claim 4. $\mathbb{T}_{\mathcal{I}}(A_{\mathtt{simp}}) = \Theta\left(d \cdot \mathtt{E}[\max\{X_1, \ldots, X_d\}]\right)$

This one is less trivial:

Claim 5. *If all doors are similar then* $\mathbb{T}_{\mathcal{I}}(A_{\mathtt{simp}}) = \Theta(\mathbb{T}_{\mathcal{I}})$

The claim above states that $A_{\mathtt{simp}}$ is optimal up to a multiplicative constant factor in the independent case, where all doors are similar. As a result, we can also show:

Claim 6. *Denote by* α_n *the sequence* $1^{2^n}, \ldots, d^{2^n}$. *If all doors are similar then for any configuration* $\mathcal{C}$, $\mathbb{T}_{\mathcal{C}}(\alpha_0 \cdot \alpha_1 \cdot \alpha_2 \cdots) = \Theta(\mathbb{T}_{\mathcal{C}})$.

In plain words, the above claim states that the following algorithm is optimal up to a universal constant factor for any configuration where all doors are similar: Run in phases where phase n consists of knocking 2^n consecutive times on each door, in order.

2.1.2 On the Price of Lacking Feedback

By Claims 4 and 5, investigating the price of lacking feedback when all doors are similar boils down to understanding the expected maximum of i.i.d. random variables.

$$\mathtt{Price} = \Theta\left(\frac{\mathtt{E}[\max\{X_1, \ldots, X_d\}]}{\mathtt{E}[X_1]}\right) \tag{1}$$

Note that we omitted dependency on the configuration, as by Theorem 2, up to constant factors, it is the same price as in the case where the doors are independent. Let us see a few examples of this value. First:

Lemma 7. *If $X_1, \ldots, X_d$ are i.i.d. random variables taking natural number values, then:*

$$\mathtt{E}\left[\max(X_1, \ldots, X_d)\right] = \Theta\left(\kappa + d\sum_{n=\kappa}^{\infty} \mathtt{Pr}\left[X_i > n\right]\right)$$

Where $\kappa = \min\left\{n \in \mathbb{N} \,|\, \mathtt{Pr}\left[X_1 > n\right] < 1/d\right\}$

Example 8. After the first knock on it, each door opens with probability $1 - 1/d$ and if it doesn't, it will open at its $d+1$'st knock. The expected time to open each door on its own is 2. By Lemma 7, as $\kappa = d + 1$, we get that $\mathtt{Price} = \Omega(\kappa) = \Omega(d)$. By Claim 22, $\mathtt{Price} = \Theta(d)$.

Example 9. If $p(n) = q^n$ for some $1/2 < q < 1$, then $\mathtt{Price} = \Theta(\log(d))$.

Example 10. If for some $c > 0$ and $a > 1$, $p(n) = \min(1, c/n^a)$, then $\mathtt{Price} = \Theta(d^{\frac{1}{a}})$.

Sometimes we know a bound on some moment of the distribution of opening a door. If $\mathtt{E}\left[X_1\right] < M$, then by Claim 22, $\mathbb{T} = O(d^2 M)$. Also,

Example 11. If $\mathtt{E}\left[X_1^a\right] < M$ for some $a > 1$, then $\mathbb{T} = O\left(d^{1+\frac{1}{a}} M^{1/a}(1 + \frac{1}{a-1})\right)$.

For example, if the second moment of the time to open a door on its own is bounded, we get an $O(d^{3/2})$ algorithm.

3 Two Memoryless Cascading Doors

One can say that by Theorem 2 we solved much of the dependent doors problem. There is an equivalence of the independent and cascading models, and we give an up to constant factor optimal algorithm for any situation. However, we still find the question of finding the true optimal sequences for cascading doors to be an interesting one. What is the precise cost of having no feedback, in numbers? Even the simple case of two doors, each opening with probability 1/2 on each knock, turns out to be quite challenging and has a not so intuitive optimal sequence.

In this section, we focus on a very simple yet interesting case of the cascading door problem, and solve it almost exactly. We have two doors. Door 1 opens with probability p_1 each time we knock on it, and door 2 opens with probability p_2. We further extend the setting to consider different durations. Specifically, we assume that a knock on door 1 takes one time unit, and a knock on door 2 takes c time units. Denote $q_1 = 1 - p_1$ and $q_2 = 1 - p_2$. For brevity, we will call a knock on door 1 a *1-knock*, and a knock on door 2 a *2-knock*.

The Semi-Fractional Model. As finding the optimal sequence directly proved to be difficult, we introduce a relaxation of our original model, termed the *semi-fractional model*. In this model, we allow 1-knocks to be of any length. A knock of length t, where t is a non-negative real number, will have probability of $1 - q_1^t$ of opening the door. In this case, a sequence consists of the alternating elements 1^t and 2, where 1^t describes a knock of length t on door 1. We call sequences in the semi-fractional model *semi-fractional sequences*, and to differentiate, we call sequences in the original model *integer sequences*.

As our configuration $\mathcal{C}$ will be clear from context, for a sequence π, we define $\mathtt{E}\left[\pi\right] = \mathbb{T}_{\mathcal{C}}(\pi)$ to be the expected running time of the sequence. Clearly, every integer sequence has a similar semi-fractional sequence with the same expected running time. As we will see, the reverse is not far from being true. That being so, finding the optimal semi-fractional sequence will give an almost optimal integer sequence.

3.1 Equivalence of Models

Theorem 12. *Every semi-fractional sequence π has an integer sequence π', s.t., $\mathtt{E}\left[\pi'\right] \leq \mathtt{E}\left[\pi\right] + 1$.*

For this purpose, in this subsection only, we describe a semi-fractional sequence π as a sequence of non-decreasing non-negative real numbers: $\pi_0, \pi_1, \pi_2, \ldots$, where $\pi_0 = 0$. This sequence describes the following semi-fractional sequence (in our original terms):

$$1^{\pi_1 - \pi_0} \cdot 2 \cdot 1^{\pi_2 - \pi_1} \cdot 2 \cdots$$

This representation simplifies our proofs considerably. Here are some observations:

- 1-knocks can be of length 0, yet we still consider them in our indexing.
- The sequence is an integer sequence iff for all i, $\pi_i \in \mathbb{N}$.
- The i-th 2-knock starts at time $\pi_i + c(i-1)$ and ends at $\pi_i + ci$.
- The probability of door 1 being closed after the completion of the i-th 1-knock is $q_1^{\pi_i}$, and so the probability it opens at 1-knock i is $q_1^{\pi_{i-1}} - q_1^{\pi_i}$

Lemma 13. *For two sequences $\pi = (\pi_0, \pi_1, \ldots)$ and $\pi' = (\pi'_1, \pi'_2, \ldots)$, if for all i, $\pi_i \leq \pi'_i \leq \pi_i + 1$ then $\mathtt{E}[\pi'] \leq \mathtt{E}[\pi] + 1$.*

Lemma 13 is the heart of our theorem. Indeed, once proven, Theorem 12 follows in a straightforward manner. Given a semi-fractional sequence π, define $\pi'_i = \lceil \pi_i \rceil$. Then, π' is an integer sequence, and it satisfies the conditions of the lemma, so we are done. The lemma makes sense, as the sequence π' in which for all $i > 0$, $\pi'_i = \pi_i + 1$, can be thought of as adding a 1-knock of length one in the beginning of the sequence. Even if this added 1-knock did nothing, the running time would increase by at most 1. However, the proof is more involved, since in the lemma, while some of the 2-knocks may have an increased chance of succeeding, some may actually have a lesser chance.

Proof. Given a sequence π and an event X, we denote by $\mathtt{E}[\pi \mid X]$ the expected running time of π given the event X. Let X_i denote the event that door 1 opens at its i-th 1-knock. As already said:

$$\mathtt{Pr}[X_i] = q_1^{\pi_{i-1}} - q_1^{\pi_i} = \int_{\pi_{i-1}}^{\pi_i} q_1^x \ln(q_1)\,\mathrm{d}x$$

Where the last equality comes as no surprise, as it can be seen as modelling door 1 in a continuous fashion, having an exponential distribution fitting its geometrical one. Now:

$$\mathtt{E}[\pi] = \sum_{i=1}^{\infty} \mathtt{Pr}[X_i]\,\mathtt{E}[\pi \mid X_i] = \sum_{i=1}^{\infty} \int_{\pi_{i-1}}^{\pi_i} q_1^x \ln(q_1)\,\mathrm{d}x \cdot \mathtt{E}[\pi \mid X_i] = \int_0^{\infty} q_1^x \ln(q_1) \cdot \mathtt{E}\left[\pi \,\middle|\, X_{i(x)}\right] \mathrm{d}x$$

Where $i(x) = \max_i \{x \geq \pi_{i-1}\}$, that is, the index of the 1-knock that x belongs to when considering only time spent knocking on door 1. Defining X'_i and $i'(x)$ in an analogous way for π', we want to show that for all x,

$$\mathtt{E}\left[\pi' \,\middle|\, X'_{i'(x)}\right] \leq 1 + \mathtt{E}\left[\pi \,\middle|\, X_{i(x)}\right]$$

as using it with the last equality will prove the lemma. We need the following three claims:

1. If $j \leq i$, then $\mathtt{E}[\pi \mid X_j] \leq \mathtt{E}[\pi \mid X_i]$
2. For all x, $i'(x) \leq i(x)$
3. For all i, $\mathtt{E}[\pi' \mid X'_i] \leq 1 + \mathtt{E}[\pi \mid X_i]$

Together they give what we need:

$$\mathtt{E}\left[\pi' \,\middle|\, X'_{i'(x)}\right] \leq 1 + \mathtt{E}\left[\pi \,\middle|\, X_{i'(x)}\right] \leq 1 + \mathtt{E}\left[\pi \,\middle|\, X_{i(x)}\right]$$

The first is actually true trivially for all sequences, as the sooner the first door opens, the better the expected time to finish. For the second, since for all i, $\pi'_i \geq \pi_i$, then $x \geq \pi'_i$ implies that $x \geq \pi_i$, and so:

$$i'(x) = \max_i \left\{x \geq \pi'_{i-1}\right\} \leq \max_i \{x \geq \pi_{i-1}\} = i(x)$$

For the third, denote by Y_j the event that door 2 opens at the j'th 2-knock. Then:

$$\mathtt{E}[\pi \mid X_i] = \sum_{j=i}^{\infty} (\pi_j + cj)\mathtt{Pr}[Y_j \mid X_i]$$

Let us consider this same expression as it occurs in π'. First note that $\mathtt{Pr}[Y_j \mid X_i] = \mathtt{Pr}\left[Y'_j \,\middle|\, X'_i\right]$, as all that matters for its evaluation is $j - i$. Therefore:

$$\mathtt{E}[\pi' \mid X'_i] = \sum_{j=i}^{\infty} (\pi'_j + cj)\mathtt{Pr}\left[Y'_j \,\middle|\, X'_i\right] \leq \sum_{j=i}^{\infty} (\pi_j + 1 + cj)\mathtt{Pr}[Y_j \mid X_i]$$

$$= \mathtt{E}[\pi \mid X_i] + \sum_{j=i}^{\infty} \mathtt{Pr}[Y_j \mid X_i] \leq \mathtt{E}[\pi \mid X_i] + 1$$

□

3.2 The Optimal Semi-Fractional Sequence

A big advantage of the semi-fractional model is that we can find an optimal sequence for it. For that we need some preparation:

Definition 14. For a semi-fractional sequence π, and some $0 \leq x \leq 1$, denote by $\mathtt{E}_x[\pi]$ the expected running time of π when started with door 1 being closed with probability x. In this notation, $\mathtt{E}[\pi] = \mathtt{E}_1[\pi]$.

Lemma 15. *Let $y = x/(q_2 + p_2 x)$. Then:*

$$\mathtt{E}_x\left[1^t \cdot \pi\right] = t + \mathtt{E}_{q_1^t x}[\pi] \qquad\qquad \mathtt{E}_x[2 \cdot \pi] = c + \frac{x}{y}\mathtt{E}_y[\pi]$$

Proof. The first equation is clear, since starting with door 1 being closed with probability x, and then knocking on it for t rounds, the probability that this door is closed is $q_1^t x$.

As for the second equation, if door 1 is closed with probability x, then knocking on door 2, we have a probability of $p_2(1-x)$ of terminating, and so the probability we did not finish is:

$$1 - p_2(1-x) = 1 - p_2 + p_2 x = q_2 + p_2 x = \frac{x}{y}$$

It remains to show that conditioning on the fact that we indeed continue, the probability that door 1 is closed is y. It is the following expression, evaluated after a 2-knock:

$$\frac{\mathtt{Pr}[\text{door 1 is closed}]}{\mathtt{Pr}[\text{door 1 is closed}] + \mathtt{Pr}[\text{door 1 is open but not door 2}]} = \frac{x}{x + (1-x)q_2} = y$$

□

Applying Lemma 15 iteratively on a finite sequence w, we get:

$$\mathtt{E}_x[w\pi] = a(x,w) + b(x,w)\mathtt{E}_{\delta(x,w)}[\pi] \tag{2}$$

Of specific interest is $\delta(x,w)$. It can be thought of as the *state*[3] of our algorithm after running the sequence w, when we started at state x. Lemma 15 and Equation (2) give us the behaviour of $\delta(x,w)$:

$$\delta(x,1^t) = q_1^t x \qquad\qquad \delta(x,2) = \frac{x}{q_2 + p_2 x} \qquad\qquad \delta(x,aw) = \delta(\delta(x,a),w)$$

We start with the state being 1, since we want to calculate $\mathtt{E}_1[\pi]$. Except for this first moment, as we can safely assume any reasonable algorithm will start with a 1-knock, the state will always be in the interval $(0,1)$. A 1-knock will always decrease the state and a 2-knock will increase it.

Our point in all this, is that we wish to exploit the fact that our doors are memoryless, and if we encounter a state we've already been at during the running of the sequence, then we should probably make the same choice now as we did then. The following definition and lemma capture this point.

Definition 16. We say a non-empty finite sequence w is x-invariant, if $\delta(x,w) = x$.

The following Lemma is proved in Appendix D.2, and formalizes our intuition about how an optimal algorithm should behave.

Lemma 17. *If w is x-invariant, and $\mathtt{E}_x[w\pi] \leq \mathtt{E}_x[\pi]$ then $\mathtt{E}_x[w^\infty] \leq \mathtt{E}_x[w\pi]$.*

3.2.1 The Actual Semi-Fractional Sequence

Theorem 18. *There is an optimal semi-fractional sequence $\pi^\star$ of the form $1^s(21^t)^\infty$, for some positive real values s and t, and its running time is:*

$$\mathtt{E}[\pi^\star] = \min_{z \in [0,1]} \left(\log_{q_1}(1-z) + \frac{c + (1-p_2 z)\log_{q_1}(1-p_2 z)}{p_2 z} \right)$$

Proof. Claim 26 of Appendix D.1 says that there is an optimal semi-fractional sequence π. It clearly starts with a non-zero 1-knock, and so we can write $\pi = 1^s 2\pi'$. Intuitively, in terms of its state, this sequence starts at 1, goes down for some time with a 1-knock, and then jumps back up with a 2-knock.

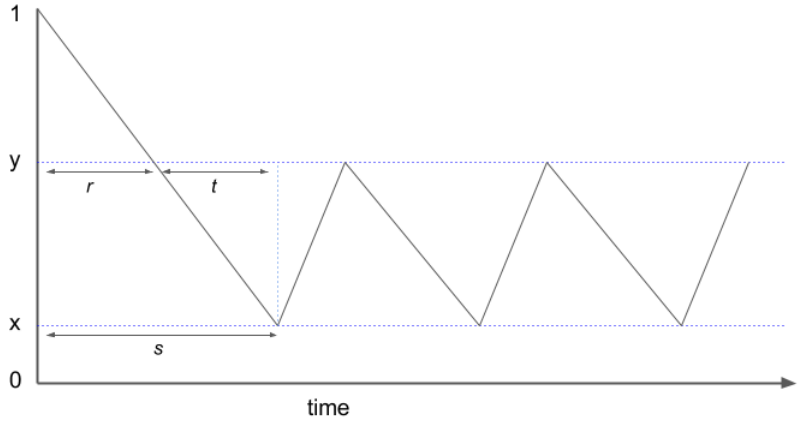

Figure 1: How the state evolves as a function of time. 1-knocks decrease the state, and 2-knocks increase it. Note that $r = \log_{q_1}(y)$ and $s = \log_{q_1}(x)$.

The state it reaches now was already passed through on the first 1-knock, and so as this is an optimal sequence we can assume it will choose the same as it did before, and keep zig-zaging up and down.

We next prove that indeed there is an optimal sequence following the zig-zaging form above. Again, take some optimal π, and write $\pi = 1^s 2\pi'$. Denote $x = \delta(1, 1^s)$ and $y = \delta(1, 1^s 2) = \delta(x, 2) > x$ (see Figure 1). Taking $r = \log_{q_1}(y) < s$, we get $\delta(1, 1^r) = y$. Denoting $t = s - r$, this means that $1^t 2$ is y-invariant. Since π is optimal, then:

$$\mathtt{E}[\pi] = \mathtt{E}\left[1^r(1^t 2)\pi'\right] \leq \mathtt{E}\left[1^r\pi'\right] \qquad \text{which implies:} \qquad \mathtt{E}_y\left[1^t 2\pi'\right] \leq \mathtt{E}_y\left[\pi'\right]$$

So by Lemma 17:

$$\mathtt{E}_y\left[(1^t 2)^\infty\right] \leq \mathtt{E}_y\left[1^t 2\pi'\right] \qquad \text{which implies:} \qquad \mathtt{E}\left[1^r(1^t 2)^\infty\right] \leq \mathtt{E}\left[1^r 1^t 2\pi'\right] = \mathtt{E}\left[\pi\right]$$

Therefore, $1^r(1^t 2)^\infty = 1^s(21^t)^\infty$ is optimal. We denote this sequence $\pi^\star$.

Now for the analysis of the running time of this optimal sequence. We will use Lemma 15 many times in what follows.

$$\mathtt{E}_1\left[1^s(21^t)^\infty\right] = s + \mathtt{E}_x\left[(21^t)^\infty\right]$$

Denote $\alpha = (21^t)^\infty$.

$$\mathtt{E}_x[\alpha] = \mathtt{E}_x\left[21^t\alpha\right] = c + \frac{x}{y}\mathtt{E}_y\left[1^t\alpha\right] = c + \frac{x}{y}(t + \mathtt{E}_x[\alpha])$$

Since $t = s - r = \log_{q_1}(x/y)$:

$$\mathtt{E}_x[\alpha] = \frac{c}{1 - \frac{x}{y}} + \frac{\frac{x}{y}}{1 - \frac{x}{y}}\log_{q_1}(x/y)$$

By Lemma 15, as our y is the state resulting from a 2-knock starting at state x, it follows that $y = x/(q_2 + p_2 x)$. Since $x/y = q_2 + p_2 x$, then $1 - x/y = p_2(1 - x)$ and then we get:

$$\frac{c}{p_2(1-x)} + \frac{q_2 + p_2 x}{p_2(1-x)}\log_{q_1}(q_2 + p_2 x)$$

And in total:

$$\mathtt{E}_1\left[1^s(21^t)^\infty\right] = \log_{q_1}(x) + \frac{c + (q_2 + p_2 x)\log_{q_1}(q_2 + p_2 x)}{p_2(1-x)}$$

Changing variable to $z = 1 - x$, results in $q_2 + p_2 x = 1 - p_2 z$, and we get the expression in the statement of the theorem. □

[3]There is an intuitive meaning behind this. Going through Lemma 15, we can see that $\delta(1, w)$ is actually the probability that after running w, door 1 is closed conditioned on door 2 being closed. Indeed, After running some finite sequence, the only feedback we have is that the algorithm did not finish yet. We can therefore calculate from our previous moves what is the probability that door 1 is closed, and that is the only information we need for our next steps.

3.3 Actual Numbers

Theorem 18 gives the optimal semi-fractional sequence and a formula to calculate its expected running time. This formula can be approximated as accurately as we wish for any specific values of p_1, p_2 and c, but it is difficult to obtain a closed form formula from it. Lemma 27 in Appendix D.3 gives us a pretty good result when $p_1 \approx p_2$, especially when they are small, as by Observation 25, we get $\log(1/(1-p_1)) \approx p_1$, and so the additive mistake in the formula is something like 1.

In general, when p_1 is small, then θ (see Lemma 27) is approximately cp_1/p_2, which is the expected time to open door 2 on its own, divided by the time to open door 1 on its own - a natural measure of the system. Then, ignoring the additive mistake, we get that the lower bound is approximately $\mathcal{F}(\theta)/p_1$, where $\mathcal{F}$ is some function not depending on the parameters of the system. For example $\mathcal{F}(1) = 3.58$. So opening two similar doors without feedback when p is small takes about 3.58 times more time than opening one door as opposed to the case with feedback, where the factor is only 2.

We also note, that when the two doors are independent and similar, it is quite easy to see that the optimal expected running time is at most $3/p$ (see Claim 28 in Appendix D.4). As a last interesting point, in Appendix D.5 we show that if $c = 1$ and $p = p_1 = p_2$ approaches zero, then the ratio between the number of 2-knocks and the number of 1-knocks approaches $\frac{1}{2}(1+\sqrt{5})$, which is the golden ratio.

3.4 Examples

For $p_1 = p_2 = 1/2$ and $c = 1$, the lower bound is 5.747. Simulations show that the best algorithm for this case is slightly more than 5.8, so the lower bound is quite tight, but our upper bound is 6.747 which is pretty far. However, the sequence we get from the upper bound proof starts with:

$$1, 1, 2, 1, 2, 2, 1, 2, 1, 2, 2, 1, 2, 2, 1, 2, 1, 2, 2, 1, 2, 1, 2, 2, 1, 2, 2, 1, 2, 1, 2, \ldots$$

The value it gives is about 5.832, which is very close to optimal.

For $p_1 = p_2 = 1/100$ and $c = 1$, the sequence we get is:

$$1^{97}, 2, 2, 1, 2, 2, 1, 2, 1, 2, 2, 1, 2, 1, 2, 2, 1, 2, 2, 1, 2, 1, 2, 2, 1, 2, 2, 1, 2, 1, 2, \ldots$$

And the value it gives is about 356.756, while the lower bound can be calculated to be approximately 356.754. As we see this is much tighter than the +1 that our upper bound promises.

A "Real Life" Example

The following scenario, much simplified, is the focus of [4]. A *treasure* τ is placed at a leaf of a Δ-regular rooted tree of depth d. A mobile agent starting at the root wishes to find it as fast as possible and is allowed to move along edges. At each node there is an *advice* pointer to one of its neighbours. With probability p the advice is *correct*, *i.e.,* directed towards the treasure, and with probability $q = 1 - p$ it is *incorrect*, *i.e.,* points towards the one of the neighbours uniformly at random. The agent can move between two neighbouring nodes in one unit of time and look at the advice at the current node hosting it. Minimizing the expected running time until finding the treasure turns out to be not trivial, and the crux of the problem is that the advice is permanent, and so cannot be amplified by rechecking it. It is shown in [4] that if $q \gtrsim 1/\sqrt{\Delta}$, then no algorithm has running time which is polynomial in d. Modeling this problem as cascading doors gives a non-trivial solution for the cases where q is smaller.

Consider door i as open once the algorithm visits τ_i, the ancestor of the treasure that is at distance i from the root. The purpose is then to open all doors. At any point in the algorithm, the *candidates* at level i are those unvisited vertices at that level, whose parent is already visited. Also denote the score of a vertex as the number of advice pointers that point towards it in the advice seen so far by the agent. A knock on door i consists of visiting the highest scoring candidate on level i, where symmetry is broken arbitrarily.

The difference in score between two candidates at the same level is affected by the advice on the path between them only, and as the algorithm moves on edges, all of the advice on this path is known. Consider a candidate at level i that is at distance l from τ_i. It will have a score that is at least as high as the treasure if the number of advice pointers on the path connecting them that point towards it is greater than the number of those pointing towards τ_i. The probability that this happens can be viewed as the probability that a random walk of length l sums up to at least 0, where each step is $(-1, 0, 1)$ with respective probabilities $(p + q/\Delta, (1 - 2/\Delta)q, q/\Delta)$. Denote this probability by $\alpha(l)$. It is shown in [4] that $\alpha(l) \approx q^l$ for $q < 1/\sqrt{\Delta}$.

Denote by C_i the number of such candidates at level i that "beat" τ_i. Even assuming all of the vertices at that level are now reachable:

$$\mathtt{E}\left[C_i\right] \leq \sum_{j=1}^{i} \Delta^j \alpha(2j-1) \approx \frac{1}{q} \sum_{j=1}^{i} (\Delta q^2)^j = O(\sqrt{\Delta})$$

This is in fact an upper bound on the expected number of knocks until opening door i, assuming door $i-1$ is already open. By Example 11, $A_{\mathtt{simp}}$ will need an expected $O(d^2\sqrt{\Delta})$ knocks to open all doors and find the treasure. As moving from one candidate to another takes $O(d)$ moves, the running time of this algorithm is at most $O(d^3\sqrt{\Delta})$. If we were able to prove that $\mathtt{E}\left[C_i^2\right]$ is small, then by Example 11 we could have dropped d's exponent to 2.5, but it turns out that this second moment is actually exponential in d and so this approach fails.

Of course, assuming there is no feedback at all in this situation is an over approximation, and while it gives a non-trivial result, using much more sophisticated arguments, it is shown in [4] that there is an $O(\sqrt{\Delta}d)$ algorithm, and that it is in fact optimal.

B General Dependencies

In the main text of the paper we focus on two special cases, that of independent doors and that of cascading doors. In what follows we introduce the possibility of much more general dependencies, and show that in fact, the two cases above are the extreme ones and so proving their equivalence is enough to prove it for all cases. For that we need to revisit our basic definition of doors and knocks.

Acyclic dependencies. We assume that the directed graph of dependencies between doors is *acyclic*. In such cases, the doors can be ordered in a topological order such that a door may depend only on lower index doors. In what follows, w.l.o.g., we shall always assume that doors are ordered in such an order.

Configuration. A configuration $\mathcal{C}$ for d doors indexed $1, \ldots, d$ describes the probabilities of each door opening as a result of knocks on it. It relates a door i with the function:

$$\phi_i^{\mathcal{C}} : \{(X_1, \ldots, X_n) \,|\, n \geq 1, \forall j. X_j \subseteq \{1, \ldots, i-1\}\} \to [0, 1]$$

Which given, for a sequence of n knocks on door i, the set of doors X_j that are open at the time of each of those knocks, returns the probability that door i was opened by one of these knocks. We will omit the superscript $\mathcal{C}$ when it is clear from context.

Monotonicity. The more we knock on a door the better our chances of opening it. More precisely, the *monotonicity property* requires that if $X = (X_1, \ldots, X_n)$, and X' is a sub-sequence of X (possibly a non-consecutive one), then $\phi_i(X') \leq \phi_i(X)$.

Positive Correlation. We focus on the case where the doors are *positively correlated*, namely, a door being open can never decrease the chances of other doors to open. Formally this means that for every i, if for all j, $X'_j \subseteq X_j$, then $\phi_i(X'_1, \ldots, X'_n) \leq \phi_i(X_1, \ldots, X_n)$.

Fundamental Distribution. The *fundamental distribution*[4] of door i in configuration $\mathcal{C}$ is the function $p_i^{\mathcal{C}}$ (again, we will omit the superscript) where $p_i(n)$ denotes the probability, in the best of conditions, *i.e.*, when all doors it depends on are open, that door i remains closed after being knocked on n times. Formally, $p_i(n) = 1 - \phi_i(\{1, \ldots, i-1\}^n)$. So $p_i(0) = 1$ for every door i, and by the monotonicity property p_i is non-increasing. We also denote by $E_i = \sum_{n=0}^{\infty} p_i(n)$ the expected time to open door i assuming all the doors it depends on are already open. We will always assume that for all i, $E_i < \infty$.

Similarity. Two doors are *similar* if they have the same fundamental distribution. Two configurations are *similar* if for every i, door i of the first configuration is similar to door i of the second.

B.1 The Cascading and Independent Configurations

In light of the definitions above we define the two main configurations:

1. *Independent doors.* The distribution associated with a door is independent of whether or not other doors are open. Formally, $\phi_i(X_1, \ldots, X_k) = \phi_i(\{1, \ldots, i-1\}^k)$.
2. *Cascading doors.* Door $i > 1$ cannot open unless door $i-1$ is already open. Only after door $i-1$ opens we start counting knocks on door i. Formally, $\phi_i(X_1, \ldots, X_k) = \phi_i(\{1, \ldots, i-1\}^t)$ where t is the number of X_j's that are equal to $\{1, \ldots, i-1\}$.

Definition 19. For configurations $\mathcal{A}$ and $\mathcal{B}$, we say that $\mathcal{A}$ dominates $\mathcal{B}$, if for every i and every $X = (X_1, \ldots, X_n)$, we have: $\phi_i^{\mathcal{A}}(X) \geq \phi_i^{\mathcal{B}}(X)$.

First:

Claim 20. *For configuration $\mathcal{C}$, similar independent configuration $\mathcal{I}$, and similar cascading configuration $\mathcal{X}$, $\mathcal{I}$ dominates $\mathcal{C}$ and $\mathcal{C}$ dominates $\mathcal{X}$.*

Proof. Denote $n = |X|$, and denote by k the number of elements of X that are equal to $\{1, \ldots, i-1\}$. We get the following series of inequalities:

$$\begin{aligned}\phi_i^{\mathcal{X}}(X) &= 1 - p_i(k) = \phi_i^{\mathcal{C}}(\{1, \ldots, i-1\}^k) \\ &\leq \phi_i^{\mathcal{C}}(X) \leq \phi_i^{\mathcal{C}}(\{1, \ldots, i-1\}^n) = 1 - p_i(n) = \phi_i^{\mathcal{I}}(X)\end{aligned}$$

Where we used, in order: the definition of cascading configuration, the fact that p_i is the fundamental distribution of door i, monotonicity, positive correlation, the fact that p is the fundamental distribution of door i, and the definition of independent configuration. □

An important property of dominance is:

Claim 21. *For any sequence π, if $\mathcal{A}$ dominates $\mathcal{B}$ then $\mathbb{T}_{\mathcal{A}}(\pi) \leq \mathbb{T}_{\mathcal{B}}(\pi)$.*

Proof. A possible way to describe the random process governing the running of an algorithm in a particular configuration, is as follows:

1. For each door i, choose uniformly at random a real number $a_i \in [0, 1]$. Fix this number for the rest of the run.

[4] This is actually not a distribution function, but rather the complement of an accumulative distribution function.

2. Denote by X the history of open doors as usual. Start it as the empty sequence.
3. Go over the knocks in the sequence in order, and when the knock is on door i, check if $\phi_i(X') > a_i$, where X' is part of X that is relevant to calculate ϕ_i (only the indices where there is a knock of door i, and only the information about the doors of $\{1, \ldots, i-1\}$). If it is then consider door i as open from this point on, and start marking it as such in X.

This way of describing the run is a little bizarre, but is in fact very natural, as our doors are described by an accumulative distribution function.

For two histories $X = (X_1, \ldots X_n)$ and $X' = (X'_1, \ldots X'_n)$, we write $X' \preceq X$ if for all j, $X'_j \subseteq X_j$. We note that Definition 19 combined with positive correlation, gives us that if $X' \preceq X$ then $\phi^{\mathcal{A}}(X) \leq \phi^{\mathcal{B}}(X')$.

This fact together with a simple argument finishes the proof: Use the same random coins to run the sequence π in both $\mathcal{A}$ and $\mathcal{B}$. By induction and the fact above, the histories at any point in time satisfy $X_{\mathcal{B}} \preceq X_{\mathcal{A}}$, and so the run on $\mathcal{A}$ will always be at least as fast as the run on $\mathcal{B}$. Since this is true no matter what a_i's we got, it is true in expectation. □

Together these two claims prove the lemma we need for the paper:

Lemma 1. *Consider similar configurations $\mathcal{C}, \mathcal{X}$ and $\mathcal{I}$, where $\mathcal{X}$ is cascading and $\mathcal{I}$ is independent. For every sequence π, $\mathbb{T}_{\mathcal{I}}(\pi) \leq \mathbb{T}_{\mathcal{C}}(\pi) \leq \mathbb{T}_{\mathcal{X}}(\pi)$. This also implies that $\mathbb{T}_{\mathcal{I}} \leq \mathbb{T}_{\mathcal{C}} \leq \mathbb{T}_{\mathcal{X}}$.*

B.2 A Simple Upper Bound on the Price of Lacking Feedback

Claim 22. *For every configuration $\mathcal{C}$, $\texttt{Price}(\mathcal{C}) \leq d$.*

Proof. Denote by $\mathcal{X}$ the cascading configuration that is similar to $\mathcal{C}$. Denote $\pi = (1, 2, \ldots, d)^\infty$. Using Lemma 1,

$$\mathbb{T}_{\mathcal{C}} \leq \mathbb{T}_{\mathcal{C}}(\pi) \leq \mathbb{T}_{\mathcal{X}}(\pi)$$

The behaviour of door i in the cascading case can be described in a simple manner: It doesn't open until all lower index doors are open, and from that time is behaves according to p_i. Hence, the expected number of knocks on door i until it opens when starting the count after all doors $j < i$ are open, is precisely E_i.

In sequence π, it takes dE_i to guarantee that door i was knocked upon E_i times. Therefore, by linearity of expectation, it follows that the expected time until we open all doors in $\mathcal{X}$ is at most $d\sum_{i=1}^{d} E_i$. Dividing by $\sum_{i=1}^{d} E_i$, we get the result. □

B.3 Existence of an Optimal Sequence

Claim 23. *For any configuration $\mathcal{C}$ there is some sequence π such that for every π', $\mathbb{T}_{\mathcal{C}}(\pi) \leq \mathbb{T}_{\mathcal{C}}(\pi')$.*

Proof. Assume there is no optimal sequence. Recall we assume that the fundamental distribution of each door allows it to be opened in finite expected time. It is then easy to see that the sequence $(1, 2, \ldots d)^\infty$ will open all doors in finite time no matter what the configuration is as long as it is acyclic. Therefore, $I = \inf_\pi \mathbb{T}_{\mathcal{C}}(\pi)$ exists.

Take a sequence of sequences $\pi^{(1)}, \pi^{(2)}, \ldots$ where $\lim_{n\to\infty} \mathbb{T}_{\mathcal{C}}(\pi^{(n)}) = I$. W.l.o.g., we can assume that $\pi^{(n+1)}$ agrees with $\pi^{(n)}$ on all the first n places. How so? there is at least one door number that appears as the first knock in infinitely many of the sequences. Take one such number, and erase all sequences that don't have it as a first knock. Of the remaining sequences, take the first one, fix it as $\alpha^{(1)}$, and erase it. Starting with the sequence of sequences that remains, find a number that appears infinitely often in the second place. Erase all sequences not having it as the second knock, and then fix $\alpha^{(2)}$ as the first of the remaining sequences, erase it and continue thus. We get that the $\alpha^{(i)}$'s are a sub-sequence of the $\pi^{(i)}$'s, and satisfy the assumption.

Define $\pi_i = \lim_{n\to\infty} \pi_i^{(n)}$. It is clearly defined for such a sequence of sequences. This is our π. Now:

$$\begin{aligned}
\mathbb{T}_{\mathcal{C}}(\pi) &= \lim_{n\to\infty} \sum_{i=0}^{n} \texttt{Pr}\left[\pi \text{ not finished by time } i\right] \\
&= \lim_{n\to\infty} \sum_{i=0}^{n} \texttt{Pr}\left[\pi^{(n)} \text{ not finished by time } i\right] \\
&\leq \lim_{n\to\infty} \mathbb{T}_{\mathcal{C}}(\pi^{(n)}) = I
\end{aligned}$$

Where the second equality is because $\pi = \pi^{(n)}$ in the first n places. □

C Proofs Related to Section 2

C.1 $A_{\texttt{simp}}$ is Optimal up to a Constant Factor for Identical Independent Doors

Claim 5. *If all doors are similar then* $\mathbb{T}_{\mathcal{I}}(A_{\texttt{simp}}) = \Theta(\mathbb{T}_{\mathcal{I}})$

Proof. By Claim 23, there is some fixed sequence π such that $\mathbb{T}_{\mathcal{I}}(\pi) = \mathbb{T}_{\mathcal{I}}$. Denote by $\pi_i(t)$ the number of times door i has been knocked on by time t in π. Clearly $\sum_i \pi_i(t) = t$.

$$\begin{aligned}\mathbb{T}_{\mathcal{I}} = \mathbb{T}_{\mathcal{I}}(\pi) &= \sum_{t=0}^{\infty} \texttt{Pr}\left[\text{some door is closed at time } t\right] \\ &= \sum_{t=0}^{\infty} 1 - \texttt{Pr}\left[\text{all doors are open at time } t\right] = \sum_{t=0}^{\infty} 1 - \left(\prod_{i=1}^{d}(1-p(\pi_i(t)))\right)\end{aligned}$$

By time t, the number of doors that have been tried more than $2t/d$ is less than $d/2$. So at least half the doors have been tried at most $t' = \lfloor 2t/d \rfloor$ times. Therefore, each such door i satisfies $p(\pi_i(t)) \geq p(t')$. We then have: $\prod_{i=1}^{d}(1-p(\pi_i(t))) \leq (1-p(t'))^{\frac{d}{2}}$. So $\mathbb{T}_{\mathcal{I}}$ is at least:

$$\sum_{t=0}^{\infty} 1 - \left(1 - p\left(\lfloor 2t/d \rfloor\right)\right)^{\frac{d}{2}}$$

In general, for any $x \leq 1$, as t traverses all integers from 0 to infinity, $\lfloor tx \rfloor$ takes every natural value at least $\lfloor 1/x \rfloor$ times. In our case we get:

$$\mathbb{T}_{\mathcal{I}} \geq \left\lfloor \frac{d}{2} \right\rfloor \cdot \sum_{t=0}^{\infty} 1 - (1-p(t))^{\frac{d}{2}} \tag{3}$$

We now turn to analyse the expected running time of $A_{\texttt{simp}}$. By Claim 4, $\mathbb{T}_{\mathcal{I}}(A_{\texttt{simp}}) = O(d \cdot \texttt{E}\left[\max\left(X_1, \ldots, X_d\right)\right])$, where X_i is the number of knocks on door i until it opens. Now:

$$\texttt{E}\left[\max\left(X_1, \ldots, X_d\right)\right] = \sum_{t=0}^{\infty} 1 - \texttt{Pr}\left[X_i \leq t\right]^d = \sum_{t=0}^{\infty} 1 - (1-p(t))^d$$

Denote $x(t) = (1-p(t))^{d/2}$, and then the sum $\sum_{t=0}^{\infty}(1-(1-p(t))^d)$ becomes:

$$\sum_{t=0}^{\infty} 1 - x(t)^2 = \sum_{t=0}^{\infty}(1-x(t))(1+x(t)) \leq 2\sum_{t=0}^{\infty} 1 - x(t) = 2\sum_{t=0}^{\infty} 1 - (1-p(t))^{d/2}$$

Applying this, and then using Equation (3) we get:

$$\mathbb{T}_{\mathcal{I}}(A_{\texttt{simp}}) = O\left(d\sum_{t=0}^{\infty} 1 - (1-p(t))^{d/2}\right) = O(\mathbb{T}_{\mathcal{I}})$$

□

C.2 A Simple Algorithm for General Configurations Where all Doors are Similar

Claim 6. *Denote by* α_n *the sequence* $1^{2^n}, \ldots, d^{2^n}$. *If all doors are similar then for any configuration* $\mathcal{C}$, $\mathbb{T}_{\mathcal{C}}\left(\alpha_0 \cdot \alpha_1 \cdot \alpha_2 \cdots\right) = \Theta(\mathbb{T}_{\mathcal{C}})$.

Proof. Denote $\pi = \alpha_0 \cdot \alpha_1 \cdots$, and note that $|\alpha_n| = 2^n d$. By Lemma 1 we need only consider $\mathbb{T}_{\mathcal{X}}(\pi)$. Taking $a_n = d + 2d + \ldots + 2^n d$, and using the right side of Observation 3 (where indices are shifted to account for the fact that a_0 is the first element and not a_1):

$$\begin{aligned}\mathbb{T}_{\mathcal{X}}(\pi) &\leq d + \sum_{n=0}^{\infty} 2^{n+1} d \cdot \texttt{SC}_{\mathcal{X}}\left(\pi[d + 2d + \ldots + 2^n d)]\right) \leq d + 2\sum_{n=0}^{\infty} 2^n d \cdot \texttt{SC}_{\mathcal{X}}(\alpha_n) \\ &= d + 4\sum_{n=0}^{\infty} 2^{n-1} d \cdot \texttt{SC}_{\mathcal{I}}(A_{\texttt{simp}}[2^n d]) \leq d + 4\mathbb{T}_{\mathcal{I}}(A_{\texttt{simp}})\end{aligned}$$

Where for the last step we used the left side of Observation 3, taking $a_n = 2^n d$. Seeing as all doors start closed, $\mathbb{T}_{\mathcal{I}}(A_{\texttt{simp}}) \geq d$, and we get that:

$$\mathbb{T}_{\mathcal{C}}(\pi) = O(\mathbb{T}_{\mathcal{I}}(A_{\texttt{simp}})) = O(\mathbb{T}_{\mathcal{I}}) = O(\mathbb{T}_{\mathcal{C}})$$

Where for the last two steps, we used Claim 5, and then Theorem 1. □

C.3 Expected Maximum of iid Random Variables

Lemma 7. *If $X_1, \ldots, X_d$ are i.i.d. random variables taking natural number values, then:*

$$\texttt{E}\left[\max(X_1, \ldots, X_d)\right] = \Theta\left(\kappa + d\sum_{n=\kappa}^{\infty} \texttt{Pr}\left[X_i > n\right]\right)$$

Where $\kappa = \min\left\{n \in \mathbb{N} \,|\, \texttt{Pr}\left[X_1 > n\right] < 1/d\right\}$

Proof. Denote $p(n) = \texttt{Pr}\left[X_i > n\right]$. The expectation we are interested in is:

$$\sum_{t=0}^{\infty} 1 - \texttt{Pr}\left[X \leq t\right]^d = \sum_{t=0}^{\infty} 1 - (1-p(t))^d = \sum_{t=0}^{\kappa-1} 1 - (1-p(t))^d + \sum_{t=\kappa}^{\infty} 1 - (1-p(t))^d \tag{4}$$

The first term is at least:

$$\sum_{t=0}^{\kappa-1} 1 - \left(1 - \frac{1}{d}\right)^d \geq \kappa\left(1 - \frac{1}{e}\right)$$

and at most κ, and so is $\Theta(\kappa)$. For the second term, examine $(1-a)^d$ when $a \leq 1/d$. We use $1 + x \leq e^x$ and Observation 24 (see below):

$$(1-a)^d \leq e^{-ad} \leq 1 - \frac{1}{2}ad$$

Hence the second term of (4) is $\Omega(d\sum_{t=\kappa}^{\infty} p(t))$. On the other hand, by the same observation:

$$(1-a)^d \geq e^{-2ad} \geq 1 - 2ad$$

And so the second term of (4) is $O(d\sum_{t=\kappa}^{\infty} p(t))$. □

Observation 24. Every $0 \leq x \leq 1$ satisfies $e^{-x} \leq 1 - \frac{1}{2}x$.

Proof. Define:

$$f(x) = 1 - \frac{1}{2}x - e^{-x}$$

Whenever f is positive the required inequality is satisfied. We note that $f(0) = 0$ and $f(1) = 1 - \frac{1}{2} - \frac{1}{e} > 0$. Now,

$$f'(x) = -\frac{1}{2} + e^{-x}$$

It is positive for $x < \ln(2) < 1$, zero at $\ln(2)$, and negative for larger values. So f starts as 0 at 0, climbs up to reach its maximum at $\ln(2)$ and then decreases. Since $f(1) > 0$, it must be the case that for all $0 \leq x \leq 1$, f is positive, which proves the lemma. □

C.4 Proofs for the Examples of Subsection 2.1.2

Example 9. If $p(n) = q^n$ for some $1/2 < q < 1$, then $\texttt{Price} = \Theta(\log(d))$.

Proof. In this case, $\kappa = \lceil \log_{1/q}(d) \rceil$, and $\texttt{E}\left[X_i\right] = 1/(1-q)$, so by (1):

$$\texttt{Price} = \Theta\left((1-q)\left\lceil \log_{1/q}(d) \right\rceil + d(1-q)\sum_{i=\kappa}^{\infty} q^i\right)$$

The second term inside the brackets is equal to $dq^{\kappa} \leq 1$. The first term is at least:

$$\frac{1-q}{\ln(1/q)}\ln(d) \geq q\ln(d) \geq \frac{1}{2}\ln(d)$$

Where we used Observation 25 below. On the other hand, it is at most:

$$(1-q)\left\lceil \frac{\ln(d)}{1-q} \right\rceil \leq (1-q)\left(1+\frac{\ln(d)}{1-q}\right) \leq 1+\ln(d) \leq 3\ln(d)$$

Since $d \geq 2$ and so $2\ln(d) > 1$. So we get the result. □

Observation 25. For $0 < q < 1$,

$$1-q \leq \ln\left(\frac{1}{q}\right) \leq \frac{1-q}{q}$$

Proof. The following is true for all $x > -1$:

$$\frac{x}{1+x} \leq \ln(1+x) \leq x$$

So for $0 < x < 1$:

$$\frac{-x}{1-x} \leq \ln(1-x) \leq -x$$

Which is:

$$x \leq \ln\left(\frac{1}{1-x}\right) \leq \frac{x}{1-x}$$

Taking $x = 1-q$ we get the first result. □

Example 10. If for some $c > 0$ and $a > 1$, $p(n) = \min(1, c/n^a)$, then $\texttt{Price} = \Theta(d^{\frac{1}{a}})$.

Proof. In this case $\kappa = \left\lceil (dc)^{1/a} \right\rceil$. In this proof we have many approximations (such as dropping the rounding above), and they all go into the constants.

The expected time to open just one door is (we assume $c^{\frac{1}{a}}$ is an integer, again this will only cost a constant factor):

$$\sum_{n=0}^{\infty} p(n) = c^{\frac{1}{a}} + \sum_{n=c^{1/a}}^{\infty} \frac{c}{n^a} \approx c^{\frac{1}{a}} + c\int_{c^{1/a}}^{\infty} \frac{1}{x^a}\,\mathrm{d}x \approx c^{\frac{1}{a}} + \frac{c}{(a-1)c^{1-\frac{1}{a}}} = c^{\frac{1}{a}}\left(1+\frac{1}{a-1}\right)$$

On the other hand, in the terminology of Equation (1):

$$\mathtt{E}\left[\max\left\{X_1,\ldots,X_d\right\}\right] = \Theta\left((dc)^{\frac{1}{a}} + d\sum_{i=\kappa}^{\infty} p(i)\right)$$

We approximate the sum in second term in the brackets by an integral:

$$d\sum_{i=\kappa}^{\infty} \frac{c}{i^a} \approx d\int_{\kappa}^{\infty} \frac{c}{x^a}\,\mathrm{d}x = \frac{dc}{(a-1)\kappa^{a-1}} \approx \frac{dc}{(a-1)(dc)^{1-\frac{1}{a}}} = \frac{d^{\frac{1}{a}}c^{\frac{1}{a}}}{a-1}$$

So the expectation of the maximum is:

$$d^{\frac{1}{a}}c^{\frac{1}{a}}\left(1+\frac{1}{a-1}\right)$$

And we get the result. □

Example 11. If $\mathtt{E}\left[X_1^a\right] < M$ for some $a > 1$, then $\mathbb{T} = O\left(d^{1+\frac{1}{a}}M^{1/a}(1+\frac{1}{a-1})\right)$.

Proof.

$$p(n) = \mathtt{Pr}\left[X_1 > n\right] = \mathtt{Pr}\left[X_1^a > n^a\right] < \frac{\mathtt{E}\left[X_1^a\right]}{n^a} \leq \frac{M}{n^a}$$

So the current configuration dominates the independent door configuration where each door has fundamental distribution $q(n) = M/n^a$, and so by Claim 21 has algorithms with running time at least as good. Following the proof of Example 10, there is such an algorithm with running time $O\left(d^{1+\frac{1}{a}}M^{\frac{1}{a}}\left(1+\frac{1}{a-1}\right)\right)$. □

D Proofs Related to Section 3

D.1 The Existence of an Optimal Semi-Fractional Sequence

Claim 26. *There is an optimal semi-fractional sequence π. That is, for every semi-fractional sequence π', $\mathtt{E}[\pi] \leq \mathtt{E}[\pi']$*

Proof. Assume there is not. But clearly, $I = \inf_\pi(\mathtt{E}[\pi])$ exists. Take a series $\pi^1, \pi^2, \ldots$ where $\lim_{n\to\infty} \mathtt{E}[\pi^n] = I$.

We think of a sequence as its sequence of 1-knock lengths. That is, π^n_i is the length of the i-th 1-knock in π^n. We first show that we can assume that for every i, the set $\{\pi^n_i \mid n \geq 1\}$ is bounded.

For this purpose, we first note that if for some semi-fractional sequence α, $\mathtt{E}[\alpha] < M$, then for every i, $\alpha_i < Mq_2^{i-1}$. That is because with probability at least q_2^{i-1} the algorithm will actually run the i-th 1-knock, and if it's longer than stated, then $\mathtt{E}[\alpha] \geq M$, in contradiction. Since we can assume that for all n, $\mathtt{E}[\pi^n] < 2I$, then by the observation above, we get the boundedness property we were aiming for.

Now, we claim that we can assume that for every i, π^n_i converges as n goes to infinity. For this, start by taking a sub-series of the π^n where π^n_1 converges (it exists, because these values are bounded, as we said). Erase all other π^n. Take the first element of this series and put aside as the new first element. From the rest, take a sub-series where π^n_2 converges, and erase all others. Take the new first element, and put it aside as the new second element. Continuing this, we get an infinite series as required.

Define $\pi_i = \lim_{n\to\infty} \pi^n_i$. We claim that π is optimal.

$$\mathtt{E}[\pi] = \sum_{i=1}^{\infty} \left(\sum_{j=1}^{i} (\pi_j + c) \right) \mathtt{Pr}[\pi \text{ finishes at 2-knock } i]$$

Denoting by X_i the event that π finishes at or after 2-knock i, this is equal to:

$$\sum_{i=1}^{\infty} (\pi_i + c)\mathtt{Pr}[X_i] = \lim_{k\to\infty} \sum_{i=1}^{k} (\pi_i + c)\mathtt{Pr}[X_i]$$

Fix some k. And denote by X^n_i the event that π^n finishes at or after 2-knock i. Since $\mathtt{Pr}[X_i]$ is a continuous function of $\pi_1, \ldots, \pi_i$, we get:

$$\sum_{i=1}^{k} (\pi_i + c)\mathtt{Pr}[X_i] = \lim_{n\to\infty} \sum_{i=1}^{k} (\pi^n_i + c)\mathtt{Pr}[X^n_i] \leq \lim_{n\to\infty} \mathbb{T}_{\mathcal{I}}(\pi^n) = I$$

So $\mathtt{E}[\pi] \leq I$ and we conclude. □

D.2 Memoryless Doors Imply Memoryless Algorithms

Lemma 17. *If w is x-invariant, and $\mathtt{E}_x[w\pi] \leq \mathtt{E}_x[\pi]$ then $\mathtt{E}_x[w^\infty] \leq \mathtt{E}_x[w\pi]$.*

Proof. As in (2), for any sequence α:

$$\mathtt{E}_x[w\alpha] = a + b\mathtt{E}_x[\alpha]$$

Where a and b are functions of x and w. Since w is not empty, and as 1-knocks decrease the state and 2-knocks increase it, there must be at least one 2-knock in w, and thus $b < 1$. So:

$$\mathtt{E}_x[w\pi] = a + b\mathtt{E}_x[\pi] \geq a + b\mathtt{E}_x[w\pi] \quad \Longrightarrow \quad \mathtt{E}_x[w\pi] \geq \frac{a}{1-b}$$

On the other hand:

$$\mathtt{E}_x[w^\infty] = \mathtt{E}_x[ww^\infty] = a + b\mathtt{E}_x[w^\infty] \quad \Longrightarrow \quad \mathtt{E}_x[w^\infty] = \frac{a}{1-b}$$

And we conclude. □

D.3 Approximating the Optimal Semi-Fractional Running Time

Theorem 18 gives a way to calculate the expectation of the best semi-fractional sequence $\pi^\star$ for our configuration. Unfortunately, we were not able to obtain a close formula for this value. The following lemma can be used to approximate it.

Lemma 27. *Denoting $\theta = -c\log(q_1)/p_2$, and $\psi = \frac{1}{2}(\sqrt{\theta^2+4\theta}-\theta)$, we have:*

$$\mathtt{E}\left[\pi^\star\right] \in \frac{1}{\log(1/q_1)}\left(\log\left(\frac{1}{1-\psi}\right)+\frac{\theta}{\psi}+1\right)-\left[0,\frac{p_2}{\log(1/q_1)}\right]$$

Proof. Recall the result of Theorem 18:

$$\mathtt{E}\left[\pi^\star\right] = \min_{z\in[0,1]}\left(\log_{q_1}(1-z)+\frac{c+(1-p_2z)\log_{q_1}(1-p_2z)}{p_2z}\right)$$

By the definition of θ in the statement of the lemma, and denoting:

$$Y = -\frac{(1-p_2z)\log(1-p_2z)}{p_2z}$$

We get:

$$\begin{aligned}\mathtt{E}\left[\pi^\star\right] &= \frac{1}{\log(q_1)}\min_{z\in[0,1]}\left(\log(1-z)-\frac{\theta}{z}-Y\right)\\ &= \frac{1}{\log(1/q_1)}\min_{z\in[0,1]}\left(\log\left(\frac{1}{1-z}\right)+\frac{\theta}{z}+Y\right)\end{aligned}$$

Next, since for $x>-1$,

$$\frac{x}{1+x}\le\log(1+x)\le x$$

Then for $0<x<1$:

$$-x\le\log(1-x)\le-\frac{x}{1-x}$$

Multiplying by $-(1-x)/x$ (a positive number):

$$1-x\le-\frac{(1-x)\log(1-x)}{x}\le 1$$

Therefore, $Y\in[1-p_2z,1]\subseteq[1-p_2,1]$. It follows that:

$$\mathtt{E}\left[\pi^\star\right]\in\frac{1}{\log(1/q_1)}\left(\min_{z\in[0,1]}\left(\log\left(\frac{1}{1-z}\right)+\frac{\theta}{z}\right)+[1-p_2,1]\right) \tag{5}$$

For the minimization, we take the derivative and compare to 0

$$\begin{aligned}&\frac{1}{1-z}-\frac{\theta}{z^2}=0 \quad\Longrightarrow\quad \frac{z^2}{\theta}+z-1=0\\ &\Longrightarrow\quad z=\frac{\sqrt{1+4/\theta}-1}{2/\theta}=\frac{\sqrt{\theta^2+4\theta}-\theta}{2}=\psi\end{aligned}$$

Where we took the root that is in $[0,1]$. Assigning back in (5),

$$\mathtt{E}\left[\pi^\star\right]\in\frac{1}{\log(1/q_1)}\left(\log\left(\frac{1}{1-\psi}\right)+\frac{\theta}{\psi}+1\right)-\left[0,\frac{p_2}{\log(1/q_1)}\right]$$

□

D.4 Similar Independent Memoryless Doors

The following simple claim implies that the expected time to open two similar memoryless doors is at most 3 times the expected time to open one of them. A generalization to d doors can easily be established based on the same idea.

Claim 28. *Consider the configuration $\mathcal{I}$ of two similar doors that open on each knock independently with probability p. Then $\mathbb{T}_{\mathcal{I}}(A_{\mathtt{simp}})=\frac{3}{p}-1$.*

Proof. Until the first door opens (either door 1 or door 2), each knock has probability p to open. Therefore, the first door opens in expected time $1/p$. From that time, every odd knock will be on the other door, and will succeed with probability p. So the expected time to open the second door after the first one has opened is $2/p-1$, and altogether we have expected time $3/p-1$. □

D.5 The Golden Ratio

Returning to the case where $c = 1$, and $p_1 = p_2$ are very small. As we said, θ of Lemma 27 tends to 1, and so ψ there tends to $(\sqrt{5} - 1)/2$. This ψ is actually the value of z that minimizes the expression of Theorem 18. Looking in the proof of the theorem, the length of 1-knocks (except the first), is

$$t = \log_q(x/y) = \log_q(q + px) = \log_q(1 - pz) = \frac{\log(1 - pz)}{\log(1 - p)}$$

For small x, $\log(1 + x) \approx x$ and so, as p goes to zero, the above ratio tends to z, and in our case to ψ. So the length of 1-knocks is ψ, and that of the 2-knocks is 1. In the long run the length of the first 1-knock is insignificant, and the transformation of Theorem 12 will make the ratio of between the number of 2-knocks and the number of 1-knocks approach $1/\psi$, which is the golden ratio.

References

[1] Xiaohui Bei, Ning Chen, and Shengyu Zhang. On the complexity of trial and error. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 31–40, 2013. URL: http://doi.acm.org/10.1145/2488608.2488613, doi:10.1145/2488608.2488613.

[2] David E. Bell. Regret in decision making under uncertainty. *Operations Research*, 30(5):961–981, 1982. URL: http://dx.doi.org/10.1287/opre.30.5.961, doi:10.1287/opre.30.5.961.

[3] Michael Ben-Or and Avinatan Hassidim. The bayesian learner is optimal for noisy binary search (and pretty good for quantum as well). In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS, 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 221–230, 2008. URL: http://dx.doi.org/10.1109/FOCS.2008.58, doi:10.1109/FOCS.2008.58.

[4] Lucas Boczkowski, Amos Korman, and Yoav Rodeh. Searching on trees with noisy memory. *CoRR*, abs/1611.01403, 2016. URL: http://arxiv.org/abs/1611.01403.

[5] Matthias Brand, Christian Laier, Mirko Pawlikowski, and Hans J. Markowitsch. Decision making with and without feedback: The role of intelligence, strategies, executive functions, and cognitive styles. *Journal of Clinical and Experimental Neuropsychology*, 31(8):984–998, 2009. PMID: 19358007. URL: http://dx.doi.org/10.1080/13803390902776860, arXiv:http://dx.doi.org/10.1080/13803390902776860, doi:10.1080/13803390902776860.

[6] Ehsan Emamjomeh-Zadeh, David Kempe, and Vikrant Singhal. Deterministic and probabilistic binary search in graphs. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 519–532, 2016. URL: http://doi.acm.org/10.1145/2897518.2897656, doi:10.1145/2897518.2897656.

[7] Uriel Feige, Prabhakar Raghavan, David Peleg, and Eli Upfal. Computing with noisy information. *SIAM J. Comput.*, 23(5):1001–1018, October 1994. URL: http://dx.doi.org/10.1137/S0097539791195877, doi:10.1137/S0097539791195877.

[8] L.A. Giraldeau and T. Caraco. *Social Foraging Theory*. Monographs in behavior and ecology. Princeton University Press, 2000. URL: https://books.google.co.il/books?id=Q1HDQgAACAAJ.

[9] Richard M. Karp and Robert Kleinberg. Noisy binary search and its applications. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pages 881–890, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=1283383.1283478.

[10] Michael N. Katehakis and Arthur F. Veinott, Jr. The multi-armed bandit problem: Decomposition and computation. *Math. Oper. Res.*, 12(2):262–268, May 1987. URL: http://dx.doi.org/10.1287/moor.12.2.262, doi:10.1287/moor.12.2.262.

[11] Michael J. Kearns and Umesh V. Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, Cambridge, MA, USA, 1994.

[12] Amos Korman, Jean-Sébastien Sereni, and Laurent Viennot. Toward more localized local algorithms: removing assumptions concerning global knowledge. *Distributed Computing*, 26(5-6):289–308, 2013. URL: http://dx.doi.org/10.1007/s00446-012-0174-8, doi:10.1007/s00446-012-0174-8.

[13] Michael Luby. A simple parallel algorithm for the maximal independent set problem. *SIAM J. Comput.*, 15(4):1036–1053, 1986. URL: http://dx.doi.org/10.1137/0215074, doi:10.1137/0215074.

[14] Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.

[15] Douglas C. Montgomery. *Design and Analysis of Experiments*. John Wiley & Sons, 2006.

[16] Andrzej Pelc. Searching games with errors - fifty years of coping with liars. *Theor. Comput. Sci.*, 270(1-2):71–109, 2002. URL: http://dx.doi.org/10.1016/S0304-3975(01)00303-6, doi:10.1016/S0304-3975(01)00303-6.

[17] Richard S. Sutton and Andrew G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998.