Fall 2017

# Quasi-Random Action Selection In Markov Decision Processes

Samuel D. Walker

# QUASI-RANDOM ACTION SELECTION IN MARKOV DECISION PROCESSES

by

SAMUEL DALTON WALKER

(Under the Direction of Stephen Carden)

## ABSTRACT

In Markov decision processes an operator exploits known data regarding the environment it inhabits. The information exploited is learned from random exploration of the state-action space. This paper proposes to optimize exploration through the implementation of quasi-random sequences in both discrete and continuous state-action spaces. For the discrete case a permutation is applied to the indices of the action space to avoid repetitive behavior. In the continuous case sequences of low discrepancy, such as Halton sequences, are utilized to disperse the actions more uniformly.

INDEX WORDS: Statistics, Stochastic Theory, Markov Decision Processes

2009 Mathematics Subject Classification: 90C40 , 60J05

# QUASI-RANDOM ACTION SELECTION IN MARKOV DECISION PROCESSES

by

SAMUEL DALTON WALKER

B.S., Middle Georgia State University, 2015

A Thesis Submitted to the Graduate Faculty of Georgia Southern University in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

STATESBORO, GEORGIA

# QUASI-RANDOM ACTION SELECTION IN MARKOV DECISION PROCESSES

by

SAMUEL DALTON WALKER

| | |
|---|---|
| Major Professor: | Stephen Carden |
| Committee: | Arpita Chatterjee |
| | Emil Iacob |
| | Scott Kersey |

Electronic Version Approved:
December 2017

# DEDICATION

I dedicate this to my fiancée, mother, and friends whose aid made all this possible.

## ACKNOWLEDGMENTS

I wish to acknowledge Dr. Carden for being a good friend and mentor as well as the wonderful faculty at Georgia Southern's Mathematics Department.

# TABLE OF CONTENTS

# LIST OF TABLES

Table Page

# LIST OF FIGURES

Figure Page

# LIST OF SYMBOLS

| | | |
|---|---|---|
| $S_t$ | state of a stochastic process at index $t \in I$ | 10 |
| $\Pr(a \mid b)$ | probability of $a$ given $b$ | 10 |
| $p_{ij}$ | transition probability from state $i$ to state $j$ | 10 |
| $P = (p_{ij})$ | transition probability matrix | 10 |
| $S$ | state space of a Markov decision process | 11 |
| $A_s$ | set of permissible actions from state $s \in S$ | 11 |
| $p_{s,s';a}$ | transition probability from state $s$ to state $s'$ under action $a$ | 11 |
| $\{r(s,a) \mid s \in S, a \in A_s\}$ | set of reward for each state and action | 11 |
| $\pi$ | function which specifies the actions the operator will perform | 11 |
| $\gamma(k)$ | discount factor used to weight future rewards | 12 |
| $V_\pi(s)$ | the expected value of policy $\pi$ over the horizon | 12 |
| $\mathbb{E}[X]$ | expected value of the random variable $X$ | 12 |
| $\pi^*$ | optimal policy | 12 |
| $V^*$ | value under the optimal policy | 12 |
| $Q^*(s,a)$ | state-action value function | 13 |
| $\#$ | cardinality operator | 16 |
| $h(s,t)$ | history random vector | 17 |
| $G_d$ | state space of Grid World | 27 |
| $A(E,N,X)$ | counts the number of the first $N$ indices of $X$ belonging to $E$ | 46 |
| $1_E$ | characteristic function of $E$ | 46 |
| $\Delta$ | discrepancy function | 47 |
| $\lambda$ | Lebesgue measure | 47 |
| $D_N^*(X)$ | star discrepancy of $X$ | 47 |
| $\Psi$ | radical inverse function | 48 |
| $\Phi_n$ | $n$ - dimensional Halton sequence | 50 |
| $\Omega$ | partition of the state space $S$ | 52 |
| $P_i$ | $i^{\text{th}}$ generating prime of a Halton sequence | 53 |

# CHAPTER 1

# INTRODUCTION

A *stochastic process* is a collection of random variables, usually denoted by $\{S_t \mid t \in I\}$, where $t$ is from some indexing set $I$ and $S_t$ is the state of the process at index $t \in I$. It is permissible for the indexing set to be either continuous or discrete. A typical example of a discrete index is when $I$ represents the number of iterations or steps in some process.

Let $\{S_t,\, t \in \mathbb{N} \cup 0\}$ be a stochastic process, where $S_t$ assumes a finite or countable number of possible values. The process is said to be in state $i$ at time $t$ if $S_t = i$. Given that the process is in state $i$ at time $t$, we may describe all possible fixed transition probabilities from said state. If the transitions satisfy the equation

$$\Pr(S_{t+1} = j \mid S_t = i) = \Pr(S_{t+1} = j \mid S_t = i, S_{t-1}, \ldots, S_1, S_0) = p_{ij}.$$

then the process is called a *Markov chain*. The property above is known as the *Markovian property*. We may interpret the Markovian property as saying that the conditional distribution of the next state in a stochastic process depends only on the present state of the process and is conditionally independent of any past state $S_{t-1}, S_{t-2}, \ldots, S_1, S_0$. Supposing there are $N$ possible states for some process, then the transition probabilities may be represented by the matrix $P = (p_{ij}) \in [0,1]^{N\times N}$,

$$P = \begin{bmatrix} p_{11} & p_{12} & p_{13} & \cdots & p_{1N} \\ p_{21} & p_{22} & p_{23} & \cdots & p_{2N} \\ \vdots & & & & \vdots \\ p_{N1} & p_{N2} & p_{N3} & \cdots & p_{NN} \end{bmatrix}$$

where $p_{ij}$ is the probability of beginning in state $S_t = i$ and transitioning to state $S_{t+1} = j$. Each transition probability must satisfy:

1. $p_{ij} \geq 0, \qquad 1 \leq i, j \leq N$

2. $\sum_{j=0}^{\infty} p_{ij} = 1, \quad i \in \{1, \ldots, N\}.$

A *Markov decision process* is a sequential decision model which satisfies the Markov property. At each time step, an operator observes the current state and chooses from a set of possible actions which are permissible to execute from that state. Associated with each state-action pair is a probability distribution on the states, which governs the transition to the next state. We call each time step an *epoch* and at each epoch the process will transition into the next state of the state space based on the transition probabilities from the current state. We will denote the probability of transitioning from state $s$ to state $s'$ under action $a$ as $p_{s,s';a}$. For any state and action, there exists a reward random variable, $r(s,a)$, which assigns a (possibly random) reward to each state-action pair. The reward distribution depends only on the state-action pair; it is stationary in time.

Let $A_s$ denote the set of actions permissible from state $s$. Formally, a *Markov decision process* is a collection of objects $\{I, S, \{A_s \mid s \in S\}, \{p_{s,s';a} \mid s, s' \in S, a \in A_s\}, \{r(s,a) \mid s \in S, a \in A_s\}\}$. If $\#(I) < \infty$, then the process has a *finite horizon*. A *policy* is a mapping from the state space to the action space

$$\pi : S \to A$$

which guides the operator deterministically, specifying the actions of the operator. Note that in general, it is possible for the decision to be *non-stationary*; that is the prescribed action for a state may change with time. However we will restrict our attention only to stationary policies. The goal is to determine the policy which maximize some measure of reward.

Let $\gamma(k)$, for $0 \leq \gamma(k) \leq 1$, be defined as a *discount factor* which is used to weigh future rewards. The *value* of a state $s$ under a policy $\pi$, $V_\pi(s)$, is the expected value of the policy over the horizon:

$$V_\pi(s) = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma(k) r(s_k, \pi(s_k)) \mid s_0 = s\right] \tag{1.1}$$

$$= \mathbb{E}[r(s, \pi(s))] + \gamma \sum_{s' \in S} p_{s,s';a} V_\pi(s'), \quad \forall s \in S. \tag{1.2}$$

Determining an optimal policy using equation (1.1) is difficult because the sum is over all paths. By conditioning on the first state that is transitioned to via the Law of Total Probability, and rearranging into equation (1.2), we arrive at the *Bellman Equation* [13]. If the expected rewards and transition probabilities are known, we can solve the linear system (1.2) to determine the value of all states under a given policy $\pi$. We can improve upon $\pi$ using policy iteration [10], a topic out of the scope of this introduction. Using Bellman's equation we may solve for the optimal policy, denoted $\pi^*$. Then $V^*$, the optimal value, can be determined by the linear system of equations

$$V^*(s) = \max_{a \in A} \left\{ \mathbb{E}[r(s, a)] + \gamma \sum_{s' \in S} p_{s,s';a} V^*(s') \right\}.$$

The associated optimal values are $V^*(s) = V_{\pi^*}(s)$. In many applications the transition probabilities and reward distribution are unknown quantities. Therefore additional constructs are necessary for creating a solution method. In particular the state-value function is generalized to the state-action value function,

$$Q^*(s, a) = \mathbb{E}[r(s, a)] + \gamma \sum_{s' \in S} p_{s,s';a} V^*(s').$$

This function is the basis for *reinforcement learning*, which are methods for finding approximately optimal policies from observations of the system.

These methods can be classified into one of two categories. First, *online learning* algorithms explore the state-action space and learn the value function simultaneously. The operator must choose between selecting actions which have not been tried in order to learn their value and choosing actions which are known to lead to high reward in order to accumulate as much reward as possible. This choice is known as the *exploration-exploitation* dilemma. Examples of online reinforcement learning algorithms are Q-learning, presented in Algorithm 1 [17].

**Algorithm 1** Q-learning

```
procedure MAIN
    initial $Q(s,a), \forall s \in S, \forall a \in A_S, \alpha, \gamma$
    for iteration do
        Initialize $s$
        $a \leftarrow$ Selected via policy derived from $Q$, i.e. uniform or $\epsilon$-greedy
        while $s$ non-terminal do
            Take action $a$, observe reward $r$ and state $s'$.
            $Q(s,a) \leftarrow Q(s,a) + \alpha \left[ r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right]$
            $s \leftarrow s'$
        end while
    end for
end procedure
```

and SARSA [16]. One of the conditions for Q-learning and the like to converge to an optimal policy is that every state and action pair is visited an infinite number of times.

Second, *offline learning* algorithms separate exploration of the environment and learning about the state-action value function into separate tasks. The first phase of an offline algorithm is concerned only with exploration. At the end of the exploration phase the result is a set of observations which come as a collection of four tuples, $\{(s_i, a_i, u_i, r_i) \mid i = 1, \ldots, m\}$, where $s$ is the state of the system, $a$ is the action chosen by the operator, $u$ is the state the system transitions to, and $r$ is the reward received. An example of an offline algorithm is fitted Q-iteration [15], which takes a collection as input and the output is an approximately optimal policy.

In practice the success of these algorithms hinges on the thorough exploration of the state-action space. The prototypical implementations of the above algorithms employ random action selection. There is significant literature on how to improve on uniform action selection. X. Zhang and H. Gao consider a Markov decision process in the maintenance of infrastructure. The operator selects maintenance actions depending on the state of disrepair of each piece of infrastructure. Given a sub-grouping of infrastructure all with a common state, the operator will select a maintenance action randomly (uniform) to enact [19].

The $\epsilon$-greedy strategy is another action selection method. Given an epsilon $\epsilon \in [0, 1]$, the *epsilon-greedy strategy* dictates that the action with the highest known expected return be selected for a proportion of $1 - \epsilon$ of the trials, and a randomly selected action is used for a proportion of $\epsilon$ of the trials. Xia and Zhao utilizes an epsilon-greedy action selection strategy in their online reinforcement learning algorithm called Bayesian SARSA which uses Bayesian inference to update the action value function to solve the issue of policy evaluation [18]. In their implementation of an $\epsilon$-greedy strategy the actions are distributed uniformly. Boltzmann exploration [4], also known as soft-max exploration, seeks to alleviate the exploration-exploitation dilemma by weighing the value of taking action $a$ in comparison to other possible actions via the equation:

$$\text{Pr}(a) = \frac{e^{Q(a)/\tau}}{\displaystyle\sum_{i=1}^{n} e^{Q(i)/\tau}}$$

where $\tau$ is a parameter governing the degree of exploration.

Asadi and Littman demonstrated that implementing Bayesian SARSA with a Boltzmann soft-max exploration policy can lead to problematic behavior, introducing instabilities for certain Markov decision processes [1]. Convergence of SARSA in a tabular setting using $\epsilon$-greedy strategies is guaranteed [12], but modifying this strategy using the Boltzmann method does not guarantee convergence. Asadi and Littman demonstrated a counter example in the form of a simple two-state Markov decision process.

Another action selection strategy is the *optimism in the face of uncertainty* maxim. This maxim lets the operator act according to an overly optimistic model of the Markov decision process, assuming all unused actions are better than those used thus far. Optimism encourages the operator to explore the environment in search of higher expected return. While optimism incentivizes exploration, this behavior does not guarantee convergence to an optimal policy [14]. The problem lies in the algorithm's inability to distinguish between a small probability to successfully transition to a state with a higher expected reward and its impossibility. This problem can be circumvented via algorithms such as R-max [2] which only considers states which have been visited a sufficient number of times.

Hester and Stone's TEXPLORE algorithm [9] utilizes a variant of UCT (upper confidence bounds applied to trees) to select actions using the assignment

$$a \leftarrow \operatorname*{argmin}_{a'} \left( Q(s,a') + 2\frac{r_{\text{max}}}{1-\gamma}\sqrt{\frac{\log c(s)}{c(s,a')}} \right)$$

where $c(s)$ counts the number of visits to state $s$, $c(s,a)$ counts the number of times action $a$ is attempted from state $s$, $r_{\text{max}}$ is the maximum return, and $\gamma$ is the discount factor.

The goal of this paper is to introduce an alternative to these methods. Our methodology uses low discrepancy action selection to reduce redundancies encountered during standard random action selection. In the case of a discrete set of actions we do this by recording a history of the state-action pair and then choosing the action with the smallest number of attempts from a given state $s$, preferring to select the minimal $\{a \mid c(s,a)\}$ instead.

Markov decision process with continuous - valued actions are less common, but there do exists RL algorithms to deal with them [3].

In the continuous case, we use so-called *quasi-random sequences*, a technique popular in Monte-Carlo integration. The next section will describe each of these methods in more detail.

# CHAPTER 2

# DISCRETE MDP

The concept of discrepancy is related to the idea of similarity. If two features are dissimilar then we say that the feature space exhibits low discrepancy.

In this chapter the theory of low discrepancy will be applied to exploration in a discrete state Markov decision process, i.e. $\#(S) = N, \#(A) = M$ for some $N, M < \infty$. A proof that the expected hitting time for a two-state system under low discrepancy action selection is less than or equal to the expected hitting time under random action selection is presented. The theory of selecting actions in finite domains is given, illustrated by an application of the selection theory.

To begin we will build the theory needed to measure the increase in efficiency low discrepancy action selection has over random action selection. The basis of our theory rests on the simple fact that when selecting a sequence of actions randomly from a discrete set, it is inevitable for repetitions to occur. By reducing repetitions we decrease the expected hitting time for each state. We will examine the theory for a simple two state system, then empirically perform experiments on processes with larger state spaces.

A simple methodology for selecting actions from a list of $M$ actions is to take a random permutation of the indices of the $M$ actions and execute each action as their index appears. Once the end of the permutation has been reached, re-permutate the indices and repeat the process. Under this method we will pick each action once, selecting the same action again only once every other action has been attempted.

Let $h : S \times T \to \mathbb{N}^M$ be a history random vector which takes a state and time pair and maps to a vector $h(s,t) \in \mathbb{N}^M$ whose $i^{\text{th}}$ coordinates is the number of times action $a_i$ has been attempted from state $s$ by time $t$.

Under low discrepancy action selection, when selecting an action to perform from state $s$ at time $t$ we choose an action uniformly from the set of actions whose $i^{\text{th}}$ coordinate of $h(s,t-1)$ is less than or equal to every other coordinate of $h(s,t-1)$.

For instance suppose that we have a six state system $S = \{s_1, s_2, \ldots, s_6\}$ and 10 possible actions $A = \{a_1, a_2, \ldots, a_{10}\}$. Then a possible history for state $s_1$ at time epoch $t = 20$ would be

$$h(s_1, 20) = (2, 2, 2, 1, 2, 2, 2, 2, 1, 2, 1).$$

Under low discrepancy action selection we would choose either of actions $a_4$, $a_8$, or $a_{10}$ to perform since their action history from this state is minimal. From those actions which are admissible, we sample uniformly.

When selecting actions in this manner we classify all actions into two categories:

- **Class 1**: Members of class 1 have been attempted before in this current round of attempts. Note that when the 1-norm of $h$ is equivalent to an integer multiple of the length of $h$, $|h|_1 = k \cdot M, k \in \mathbb{Z}$ then all actions have been attempted $k$ times and the slate is reset.
- **Class 0**: Members of class 0 have not been attempted in this round of attempts.

To illustrate this classification scheme, consider the previous history vector:

$$h(s_1, 20) = (2, 2, 2, 1, 2, 2, 2, 2, 1, 2, 1).$$

We will write a 1 for when an action is a member of class 1 and a 0 when it is a member of class 0.

$$h(s_1, 20) \mod k - 1 = (1, 1, 1, 0, 1, 1, 1, 0, 1, 0).$$

By this classification scheme, all history vectors are binary strings. Since members of class 1 have been attempted previously, we wish to select uniformly from class 0 when choosing an action during the $t^{\text{th}}$ time epoch. Observe that $t = |h|_1$ under the two state model. The count of class 1 for state $i$ is equivalent to $\#(\text{Class } 1) = |h(i, t)|_1 \mod M$ where $M$ is the cardinality of the action space.

The probability of selecting an action and the current time epoch are intertwined. For every increase in time, the probability of selecting an action from the admissible class grows as the size of the admissible actions diminishes. Therefore, when calculating the probability of selecting an action, we introduce $t$ as an independent variable.

## 2.1 Two State System

Consider the simple Markov decision process which contains the finite state space $S = \{1, 2\}$ and a finite action space $A = \{a_k \mid k \in \mathbb{N}_M\}$ where $\mathbb{N}_M = \{1, 2, \ldots, M\}$. Let state 1 to be the initial state of the system.

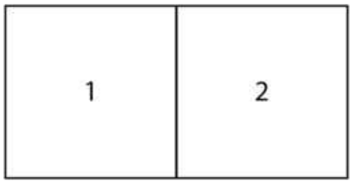

Figure 2.1: Basic discrete problem for comparing hitting times under different action selection protocols.

When selecting from the action space uniformly, we denote the probability of transitioning from state 1 to state 2 under action $a_i$ by $p_{1,2;a_i}$. The probability of transitioning from state 1 to state 2 is given by the Law of Total Probability as

$$\Pr(2 \mid 1) = \sum_{\forall a_i, i \in \mathbb{N}_M} p_{1,2;a_i} \Pr(a_i). \tag{2.1}$$

Since $a_i \sim$ unif, $\Pr(a_i) = 1/\#(A) = M^{-1}$. Equation 3.1 may then be written as

$$\sum_{\forall a_i, i \in \mathbb{N}_M} \frac{p_{1,2;a_i}}{M} := \overline{p}. \tag{2.2}$$

Observe that $\overline{p}$ is the average of the probabilities of transitioning from state 1 to state 2 under random action selection. We are mainly interested in the number of epochs until the first successful transition. Let $T_{RAS}$ denote the number of time epochs until the first successful transition under random action selection, i.e. the hitting time for state 2, and $T_{LDAS}$ the low discrepancy analog. Then we want to formally compare $\mathbb{E}[T_{\text{RAS}}]$ to $\mathbb{E}[T_{\text{LDAS}}]$. To compute $\mathbb{E}[T_{\text{RAS}}]$ we first need to determine the probability distribution for $T_{\text{RAS}}$. For $t$ to be the hitting time it implies that the first $t-1$ attempts have failed. Thus

$$\text{Pr}(T_{\text{RAS}} = t) = (1 - \overline{p})^{t-1}\overline{p}$$

which is the probability mass function for a geometric random variable,

$$T_{\text{RAS}} \sim \text{Geometric}(\overline{p}).$$

The expected value for a geometric random variable is

$$\mathbb{E}[T_{\text{RAS}}] = 1/\bar{p}.$$

The main result of this section will be showing the average hitting time under low discrepancy action selection, $\mathbb{E}[T_{\text{LDAS}}]$ is smaller than the average hitting time under random action selection, $\mathbb{E}[T_{\text{RAS}}]$.

Before we can present a proof, we must state the *Maclaurin inequality* which is a generalization of a famous mathematical inequality, the *arithmetic-geometric mean inequality*:

Arithmetic-Geometric Mean Inequality. For every positive integer $n$ and real numbers $x_1, x_2, \ldots, x_n > 0$, the inequality

$$\frac{x_1 + x_2 + \cdots + x_n}{n} \geq \sqrt[n]{x_1 x_2 \cdots x_n}$$

holds with strict inequality unless all of the $x_i$'s are equal.

Before we can state the Maclaurin inequality we must first define the *elementary symmetric polynomials* in $x_1, \ldots, x_n$, which are

$$e_k(x_1,\ldots,x_n) = \sum_{1\le i_1<i_2<\cdots<i_k\le n} x_{i_1}x_{i_2}\cdots x_{i_k} = \sum_{\substack{i\subset\{1,\ldots,n\}\\ \#(I)=k}} \prod_{i\in I} x_i$$

for $1 \le k \le n$. It is the sum of $\binom{n}{k}$ terms, where each term is the product of an unordered sample without replacement of size $k$. The first and last of these polynomials are the sum of the terms and the product of the terms respectively, $e_1 = \sum x_i$ and $e_2 = \prod x_i$. When $n = 3$ we get

$$\begin{aligned} &e_1(x,y,z) = x+y+z, \\ &e_2(x,y,z) = xy+xz+yz, \quad \text{and} \\ &e_3(x,y,z) = xyz. \end{aligned}$$

Each elementary symmetric polynomial is the sum of $\binom{n}{k}$ terms. We define its average as

$$E_k(x_1,\ldots,x_n) := \frac{e_k(x_1,\ldots,x_n)}{e_k(1,\ldots,1)} = \frac{e_k(x_1,\ldots,x_n)}{\binom{n}{k}}.$$

$E_k(x_1,\ldots,x_n)$ is called the $k$th *elementary symmetric mean* of $x_1,\ldots,x_n$. With this language we can now state the Maclaurin inequality:

Maclaurin Inequality. For $x_1,\ldots,x_n > 0$,

$$E_1(x_1,\ldots,x_n) \ge \sqrt{E_2(x_1,\ldots,x_n)} \ge \cdots \sqrt[k]{E_k(x_1,\ldots,x_n)} \cdots \ge \sqrt[n]{E_n(x_1,\ldots,x_n)}, \text{ or}$$

$$\frac{x_1 + \cdots + x_n}{n} \geq \sqrt{\frac{\sum\limits_{1 \leq i < j \leq n} x_i x_j}{\binom{n}{2}}} \geq \sqrt[3]{\frac{\sum\limits_{1 \leq i < j < k \leq n} x_i x_j x_k}{\binom{n}{3}}} \geq \cdots \geq \sqrt[n]{x_1 x_2 \cdots x_n}$$

with strict inequality unless all $x_i$'s are equal [6]. The main result of this section can now be stated with proof.

**Theorem 2.1.** *For a two state Markov system,*

$$\mathbb{E}[T_{LDAS}] \leq \mathbb{E}[T_{RAS}].$$

*Proof.* Let $A$ be the set of actions of a Markov decision process and $\#(A) = M$. Under low discrepancy actions selection (LDAS),

$$\mathbb{E}[T_{\text{LDAS}} \mid T_{\text{LDAS}} > M] = M + \mathbb{E}[T_{\text{LDAS}}]$$

since $T_{\text{LDAS}} > M$ implies there have been $M$ failures and hence the process probabilistically restarts.

Then

$$\mathbb{E}[T_{\text{LDAS}}] = \mathbb{E}[T_{\text{LDAS}} \mid T_{\text{LDAS}} \leq M]\text{Pr}(T_{\text{LDAS}} \leq M)$$
$$+ (M + \mathbb{E}[T_{\text{LDAS}}])\text{Pr}(T_{\text{LDAS}} > M).$$

The last term contains $\mathbb{E}[T_{\text{LDAS}}]$. Isolating this term to the left hand side gives

$$\begin{aligned}
\mathbb{E}[T_{\text{LDAS}}](1 - \text{Pr}(T_{\text{LDAS}} > M)) &= \mathbb{E}[T_{\text{LDAS}} \mid T_{\text{LDAS}} \leq M]\text{Pr}(T_{\text{LDAS}} \leq M) \\
&+ M\text{Pr}(T_{\text{LDAS}} > M) \\
\mathbb{E}[T_{\text{LDAS}}]\text{Pr}(T_{\text{LDAS}} \leq M) &= \mathbb{E}[T_{\text{LDAS}} \mid T_{\text{LDAS}} \leq M]\text{Pr}(T_{\text{LDAS}} \leq M) + M \\
&- M\text{Pr}(T_{\text{LDAS}} \leq M) \\
\mathbb{E}[T_{\text{LDAS}}] &= \mathbb{E}[T_{\text{LDAS}} \mid T_{\text{LDAS}} \leq M] - M + \frac{M}{\text{Pr}(T_{\text{LDAS}} \leq M)}.
\end{aligned}$$

The same logic is used to show

$$\mathbb{E}[T_{\text{RAS}}] = \mathbb{E}[T_{\text{RAS}} \mid T_{\text{RAS}} \leq M] - M + \frac{M}{\text{Pr}(T_{\text{RAS}} \leq M)}.$$

Since

$$\begin{aligned}
\text{Pr}(T_{\text{LDAS}} \leq M) &= 1 - \text{Pr}(T_{\text{LDAS}} > M) \\
&= 1 - E_M(q_1, \ldots, q_M)^M \\
&\geq 1 - E_1(q_1, \ldots, q_M)^M \\
&= 1 - \text{Pr}(T_{\text{RAS}} > M) \\
&= \text{Pr}(T_{\text{RAS}} \leq M)
\end{aligned}$$

we have

$$\frac{M}{\text{Pr}(T_{\text{LDAS}} \leq M)} \leq \frac{M}{\text{Pr}(T_{\text{RAS}} \leq M)}. \tag{2.3}$$

To finish the proof we will show that the difference $\mathbb{E}[T_{\text{LDAS}}] - \mathbb{E}[T_{\text{RAS}}] \leq 0$. To that end,

$$\begin{aligned}
\mathbb{E}[T_{\text{LDAS}}] - \mathbb{E}[T_{\text{RAS}}] &= \left(\mathbb{E}[T_{\text{LDAS}} \mid T_{\text{LDAS}} \leq M] - M + \frac{M}{\text{Pr}(T_{\text{LDAS}} \leq M)}\right) \\
&\quad - \left(\mathbb{E}[T_{\text{RAS}} \mid T_{\text{RAS}} \leq M] - M + \frac{M}{\text{Pr}(T_{\text{RAS}} \leq M)}\right) \\
&= \mathbb{E}[T_{\text{LDAS}} \mid T_{\text{LDAS}} \leq M] - \mathbb{E}[T_{\text{RAS}} \mid T_{\text{RAS}} \leq M] \\
&\quad + \left(\frac{M}{\text{Pr}(T_{\text{LDAS}} \leq M)} - \frac{M}{\text{Pr}(T_{\text{RAS}} \leq M)}\right)
\end{aligned}$$

where

$$\frac{M}{\text{Pr}(T_{\text{LDAS}} \leq M)} - \frac{M}{\text{Pr}(T_{\text{RAS}} \leq M)} \leq 0$$

by Inequality 2.3. To prove the inequality it is sufficient to show that

$$\mathbb{E}[T_{\text{LDAS}} \mid T_{\text{LDAS}} \leq M] - \mathbb{E}[T_{\text{RAS}} \mid T_{\text{RAS}} \leq M] \leq 0.$$

$$\begin{aligned}
\mathbb{E}[T_{\text{LDAS}} \mid T_{\text{LDAS}} \leq M] &= \sum_{k=1}^{M} k\text{Pr}(T_{\text{LDAS}} = k \mid T_{\text{LDAS}} \leq M) \\
&= \sum_{k=1}^{M} k\frac{\text{Pr}(T_{\text{LDAS}} = k)}{\text{Pr}(T_{\text{LDAS}} \leq M)} \\
&= \frac{1}{\text{Pr}(T_{\text{LDAS}} \leq M)} \sum_{k=1}^{M} k\text{Pr}(T_{\text{LDAS}} = k) \\
&= \frac{1}{\text{Pr}(T_{\text{LDAS}} \leq M)} \sum_{k=1}^{M} \text{Pr}(T_{\text{LDAS}} > k) \\
&\leq \frac{1}{\text{Pr}(T_{\text{RAS}} \leq M)} \sum_{k=1}^{M} \text{Pr}(T_{\text{LDAS}} > k)
\end{aligned}$$

Showing that $\Pr(T_{\text{LDAS}} > k) \leq \Pr(T_{\text{RAS}} > k)$ will prove

$$\begin{aligned}\mathbb{E}[T_{\text{LDAS}} \mid T_{\text{LDAS}} \leq M] &= \frac{1}{\Pr(T_{\text{RAS}} \leq M)} \sum_{k=1}^{M} \Pr(T_{\text{LDAS}} > k) \\ &\leq \frac{1}{\Pr(T_{\text{RAS}} \leq M)} \sum_{k=1}^{M} \Pr(T_{\text{RAS}} > k) \\ &= \mathbb{E}[T_{\text{RAS}} \mid T_{\text{RAS}} \leq M]\end{aligned}$$

completing the proof. The probability $\Pr(T_{\text{LDAS}} > k)$ is the sum of products of permutations of $k$ actions from the action space divided by $\binom{M}{k}$, the number of ways we may choose those $k$ actions. The use of Maclaurin's inequality will complete the proof. To wit,

$$\begin{aligned}\Pr(T_{\text{LDAS}} > k) &= \binom{M}{k}^{-1} \sum_{\substack{I \subset \{1,2,\ldots,M\} \\ \#(I)=k}} \prod_{i \in I} q_i \\ &= [E_k(q_1, \ldots, q_M)]^k \\ &\leq [E_1(q_1, \ldots, q_M)]^k && \text{Maclaurin's Inequality} \\ &= \left(\sum_{i=1}^{M} \frac{q_i}{M}\right)^k = \bar{q}^k \\ &= \Pr(T_{\text{RAS}} > k).\end{aligned}$$

□

This offers theoretical support for the superiority of selecting actions in a low discrepancy manner over purely random action selection. In the next section we will gather empirical data via experiments conducted on a simple Markov process.

## 2.2 Discrete System Dynamics

Grid World is a discrete state Markov decision environment we will use as a testing platform for comparing low discrepancy action selection to random action selection. Before we labor toward that task, the system dynamics must be explained.

We denote the set of possible states of the discrete Grid World by

$$G_d = \{1, 2, \ldots, 25\}.$$

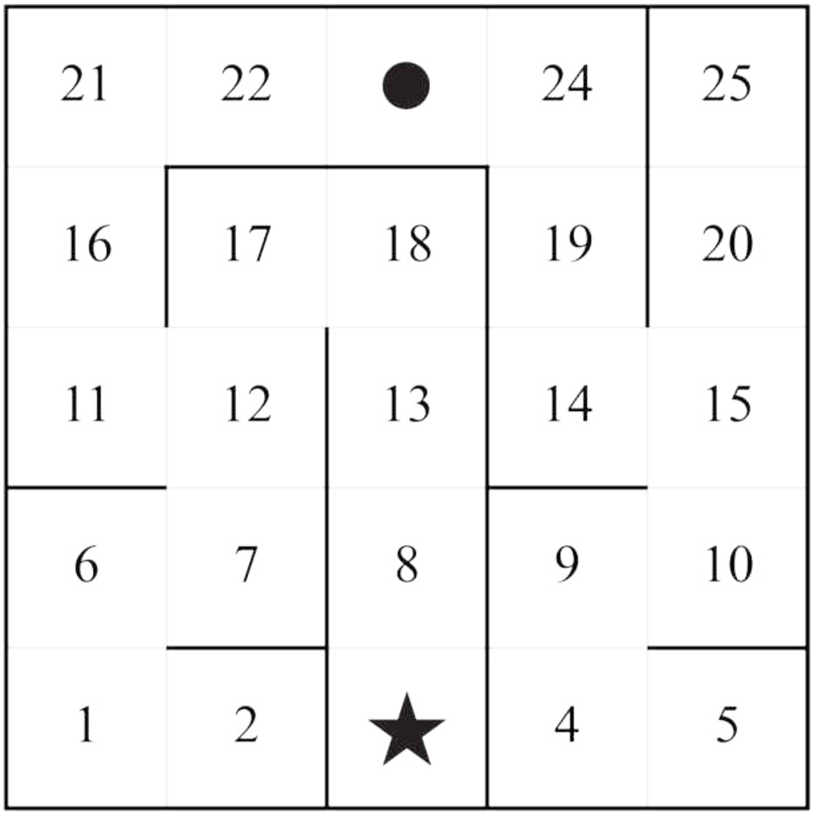

Figure 2.2: Grid World is a 5x5 grid maze which is simple enough that we can test our theory quickly, but complicated enough that we avoid trivial solutions. A circle denotes the starting location of the operator and a star denotes the goal state.

When we are limited to a discrete number of actions we consider the most obvious case where the four directions of motion are the cardinal directions. Thus let $a_1$ = up, $a_2$ = down, $a_3$ = right, and $a_4$ = left. Then the discrete action space is

$$A_d = \{a_1, a_2, a_3, a_4\}.$$

When the operator transitions from state to state, he does so via a transition probability. In the discrete action/discrete state Grid World process we may only move in one of four directions: up, down, left, and right. The probabilities for each are given by matrices in Appendix A.

Under random action selection we select actions from a uniform distribution, therefore the average of the above matrices will govern the movement of the operator throughout Grid World:

$$P = \frac{\text{Pr(up)} + \text{Pr(down)} + \text{Pr(left)} + \text{Pr(right)}}{4}.$$

When the operator chooses a direction, he has an 80% chance of successfully moving in that respective direction and a 10% chance to move in either adjacent directions. For instance if action up is selected, then he has an 80% chance of going up, a 10% chance of going right, and a 10% chance of going left. If a barrier is encountered, then the operator remains in his starting position.

## 2.3 Hitting Time Distribution

To begin, we will define a *hitting time*. A hitting time is a random variable representing the number of epochs until the operator first encounters a specific state.

The probability calculations for moving between states in more than one time epoch are straightforward. Suppose we are interested in the probability of transitioning from state 1 to state 4 in two time epochs. This may be calculated using the one-step transition probabilities:

$$\begin{aligned} p^{(2)}_{1,4} &= \sum_{\forall j \in G_d} p_{1,j} p_{j,4} \\ &= p_{1,1}p_{1,4} + p_{1,2}p_{2,4} + \cdots + p_{1,23}p_{23,4} = (p_{1,j})_{j \in G_d} (p_{j,4})_{j \in G_d}. \end{aligned}$$

Of course it is impossible to move from state 1 to state 4 in merely two time epochs, but it is feasible to move there in three time epochs:

$$\begin{aligned} p^{(3)}_{1,4} &= p_{1,1}P^{(2)}_{1,4} + p_{1,2}P^{(2)}_{2,4} + \cdots + p_{1,25}P^{(2)}_{25,4} \\ &= p_{1,1}\left(\sum_{\forall j \in G_d} p_{1,j}p_{j,4}\right) + p_{1,2}\left(\sum_{\forall j \in G_d} p_{2,j}p_{j,4}\right) + \cdots + p_{1,25}\left(\sum_{\forall j \in G_d} p_{25,j}p_{j,4}\right) \\ &= \sum_{\forall k \in G_d} p_{1,k}\left(\sum_{\forall j \in G_d} p_{k,j}p_{j,4}\right) \\ &= \sum_{\forall k \in G_d} \sum_{\forall j \in G_d} p_{1,k}p_{k,j}p_{j,4} = (p_{1,k})(p_{k,j})(p_{j,4}) \mid \{k, j\} \subset G_d \end{aligned}$$

In general, given $n$ time epochs, the probability of starting at state $i \in G_d$ and ending at state $j \in G_d$ can be calculated via

$$p_{i,j}^{(n)} = \sum_{\forall k_1 \in G_d} \sum_{\forall k_2 \in G_d} \cdots \sum_{\forall k_{n-1} \in G_d} p_{i,k_1} p_{k_1,k_2} \cdots p_{k_{n-2},k_{n-1}} p_{k_{n-1},j}$$

where $k_1, k_2, \ldots, k_{n-1}$ range from 1 to the number of allowable states. Here we have calculated the probability of arriving in state $j$ from state $i$ after $n$ time epochs.

We are going to derive the exact distribution for the random variable $T_{\text{RAS}}$, the hitting time for an arbitrary state. Normally this would depend on two states $i$ and $j$, but for ease of notation we will suppress that dependence and specify $i$, the initial state, and $j$, the goal state, only when necessary. To that end, we will first derive a general method for calculating tail probabilities.

**Proposition 1**. Let $S'$ be a set of states to avoid. The probability of starting in state $i \notin S'$ and avoiding all members of $S'$ for $t$ time epochs is the $i^{\text{th}}$ column and $n^{\text{th}}$ row of the probability matrix $Q$ where $Q$ is defined recursively as

$$\begin{cases} Q_{1,i} := \sum\limits_{k \notin S'} p_{i,k}, \\ Q_{n,i} := \sum\limits_{k \notin S'} p_{i,k} Q_{n-1,k}. \end{cases}$$

*Proof.* The initial probability, $Q_{1,i}$, is found by summing the probabilities of transitioning to a state $k \notin S'$.

$$Q_{1,i} := \sum_{k \notin S'} p_{i,k}$$

Regard this as a base case and assume the induction hypothesis; the probability of avoiding states $s \in S'$ for $n-1$ time epochs is given by $Q_{n-1,i}$. To avoid states in $s \in S'$ for $n$ epochs, one must first transition to a state not in $S'$, then continue to avoid $s \in S'$ for $n-1$ epochs. Conditioning on the first transition, we have

$$Q_{n,i} := \sum_{k \notin S'} p_{i,k} Q_{t-1,k}.$$

□

Suppose we have a single state, $s'$, for which we wish to calculate a hitting time. It easiest to think in terms of tail probabilities, Pr$(T_{\text{RAS}} > n)$, which is precisely the probability that the state has been avoided for $n$ time epochs. This probability may be found in column $s'$ of the $Q$ matrix. The probability of arriving in $s'$ by time epoch $n$ is the compliment of the probability of avoiding said state by time epoch $n$. The elements of $Q$ are tail probabilities; that is Pr$(T_{\text{RAS}} > t)$ for some integer $t > 0$. We can now calculate an exact hitting time distribution for reaching a state $j \in G$ beginning in state $i \in G$ by

$$\begin{aligned}
\Pr(T_{\text{RAS}} = k) &= \Pr(T_{\text{RAS}} > k-1) - \Pr(T_{\text{RAS}} > k) \\
&= (1 - Q_{k-1,i}) - (1 - Q_{k,i}) \\
&= Q_{k,i} - Q_{k-1,i}.
\end{aligned}$$

For instance, if we are interested in the probability distribution for the hitting times of state 3, we would calculate

$$\Pr(T_{RAS} = k) = Q_{k,3} - Q_{k-1,3}.$$

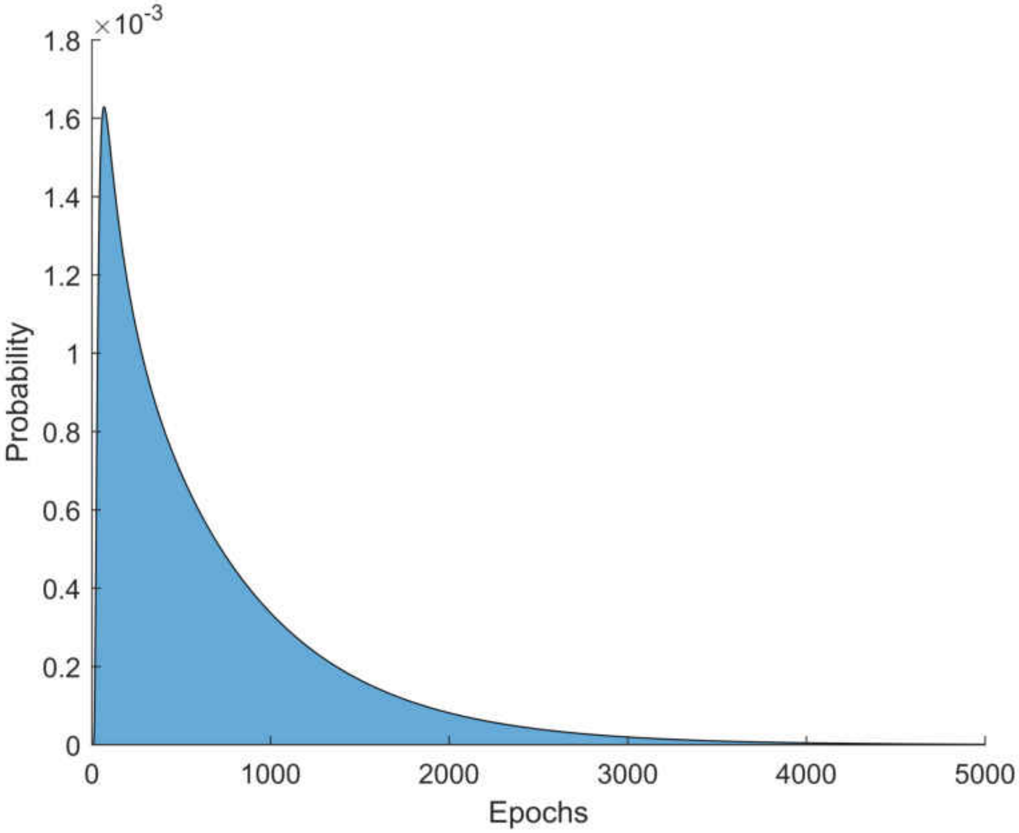

Figure 2.3: Hitting time distribution for state 3 under random action selection.

This distribution will be shown empirically later in the chapter when we conduct experiments utilizing the above theory. The cumulative distribution of reaching the goal state can be calculated by summing column 3 of $Q_{k,3} - Q_{k-1,3}$:

$$\begin{aligned}\Pr(T_{\text{RAS}} \leq k) &= \sum_{v=1}^{k} \Pr(T_{\text{RAS}} = v) \\ &= \sum_{v=1}^{k} Q_{v,3} - Q_{v-1,3}.\end{aligned}$$

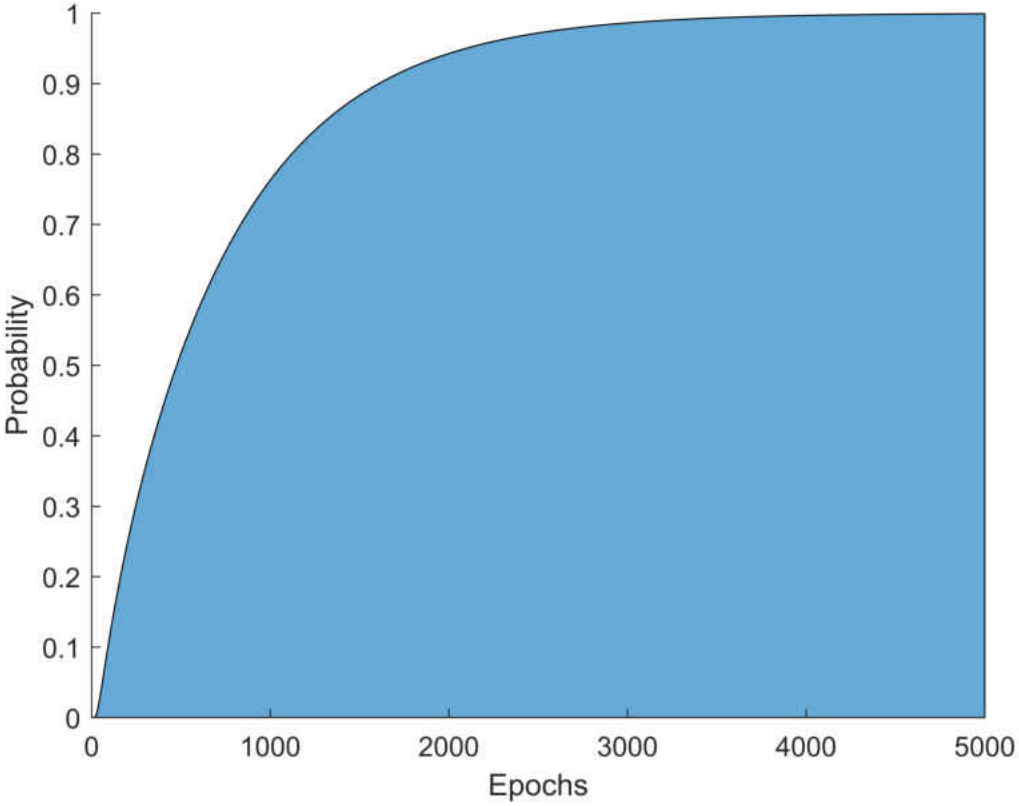

Figure 2.4: Cumulative hitting time probabilities for state 3 from state 23.

The quantity $1 - \Pr(T_{\text{RAS}} \leq k) < \epsilon$ for $\epsilon \approx 0.0001$ after roughly 4,000 time epochs. This is true for all states as can be seen in Figure 2.4.

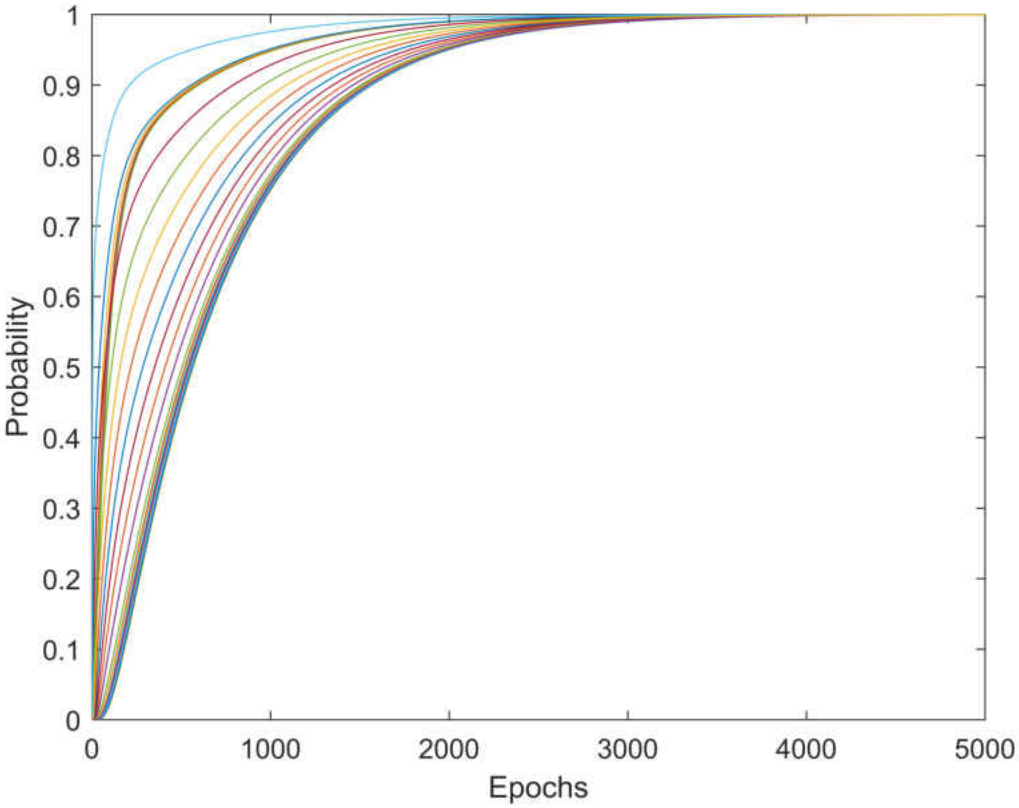

Figure 2.5: Cumulative hitting time probabilities for state 3. Each line represents the cumulative distribution for different initial states.

The recursive method used for the above probabilities is due to the Markov nature of the problem; we may build the probabilities based only on the current state of the system.

## 2.4 Empirical Data

For random action selection we were able to derive an exact hitting distribution. Under low discrepancy action selection expressions for the exact hitting time distribution become intractable empirical evidence must be supplied. We will use MATLAB to simulate the operators exploration of Grid World. The following algorithm will facilitate our needs:

**Algorithm 2** Discrete/Discrete Grid World

```
procedure MAIN
    INPUT ← state $i$,    initial state
    Initialize state-action history vector
    Initialize hitting time storage
    while $\exists$ states unexplored do
        $action \sim$ uniform;    if RAS
        $action \leftarrow$ minimum count of state-action history vector;    if LDAS
        Update state-action history
    end while
end procedure
```

Above we developed a recursive formula for finding avoidance probabilities. We also proved that the expected hitting time in two state system is smaller under low discrepancy action selection. Now we will present some empirical data. We will simulate the exploration phase of a Markov decision in the familiar Grid World environment. The experiment will be ran for 10,000 trials and then data from the experiment will be summarized. We will start by analyzing the number of epochs required by the operator to reach state 3 from state 23 under random action selection and low discrepancy action selection.

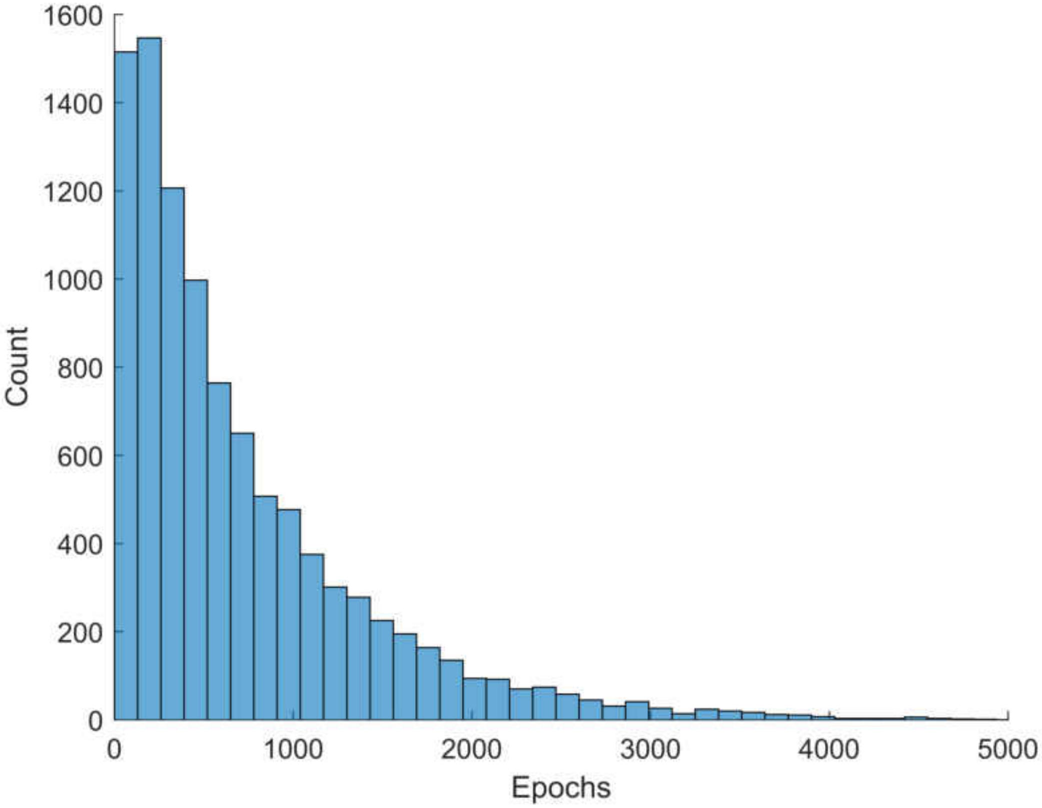

Figure 2.6: Histogram of required epochs until state 3 is reached from state 23 under RAS.

The shape of the histogram in Figure 2.6 mimics the shape of our hitting time distribution shown in Figure 2.3 as expected. We chose a sample size of 10,000 for this simulation and for future simulations because the difference in statistical data for 10,000 trials and 100,000 trials was minimal.

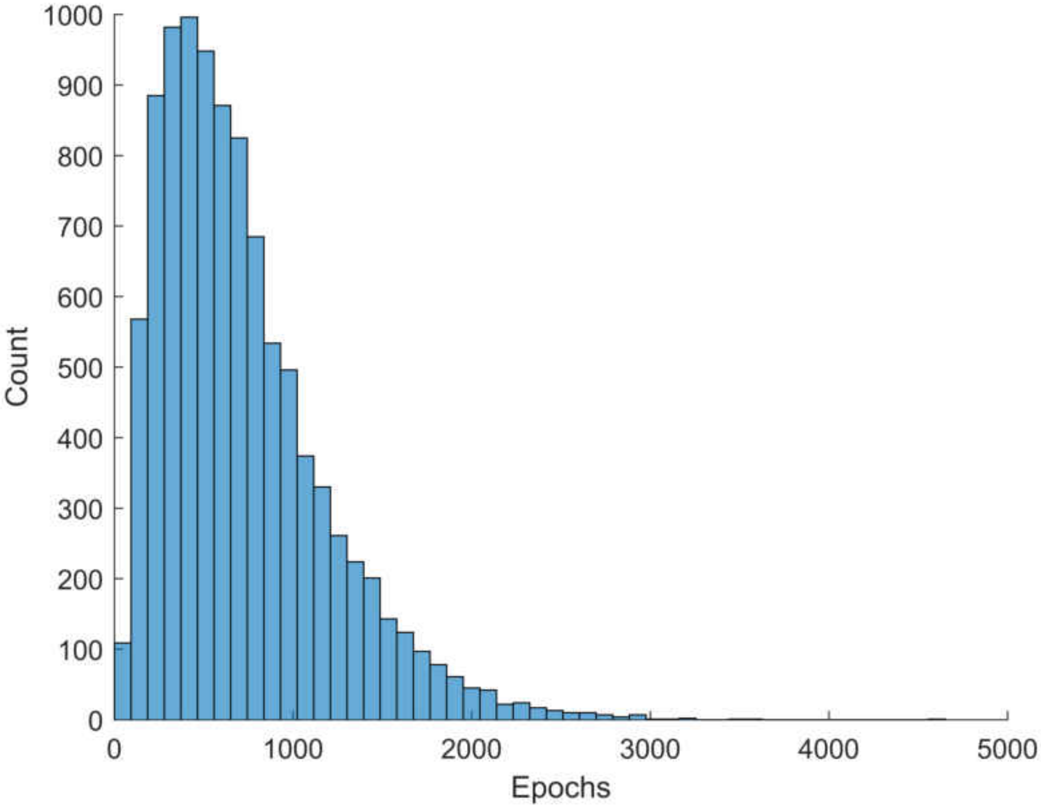

Figure 2.7: Histogram of required epochs until state 3 is reached from state 23 under LDAS.

Table 2.1: Statistics For Goal Stop Criterion (Discrete)

| Action Selection Protocol | Mean | Variance | Standard Deviation | Standard Error |
|---|---|---|---|---|
| RAS | 698.2 | 493,464.5 | 702.5 | 7.0 |
| LDAS | 460.4 | 152,119.5 | 390.0 | 3.9 |

Sample Size: 10,000. STATE = 3 stop criterion.

By all statistical metrics low discrepancy action selection is superior to random action selection. The mean number of epochs is lower and the standard deviation is much smaller for low discrepancy action selection indicating a tighter distribution.

The expected value for the hitting time for state 3 can be calculated via $Q$. Since the entries in column $i$ and row $j$ of $Q$ are tail probabilities, $\text{Pr}(T_{\text{RAS}} > j)$, for reaching state 3 from state i, we may take the sum over the number of trials to find the expected value:

$$\mathbb{E}[T_{\text{RAS}}] \approx \sum_{j=1}^{10{,}000} \text{Pr}(T_{\text{RAS}} > j) = 699$$

which is nearly identical to the empirical result.

The advantage of low discrepancy action selection is in the exploration of the state space, thus we will rerun the previous experiment, this time stopping only once the state space has been fully explored.

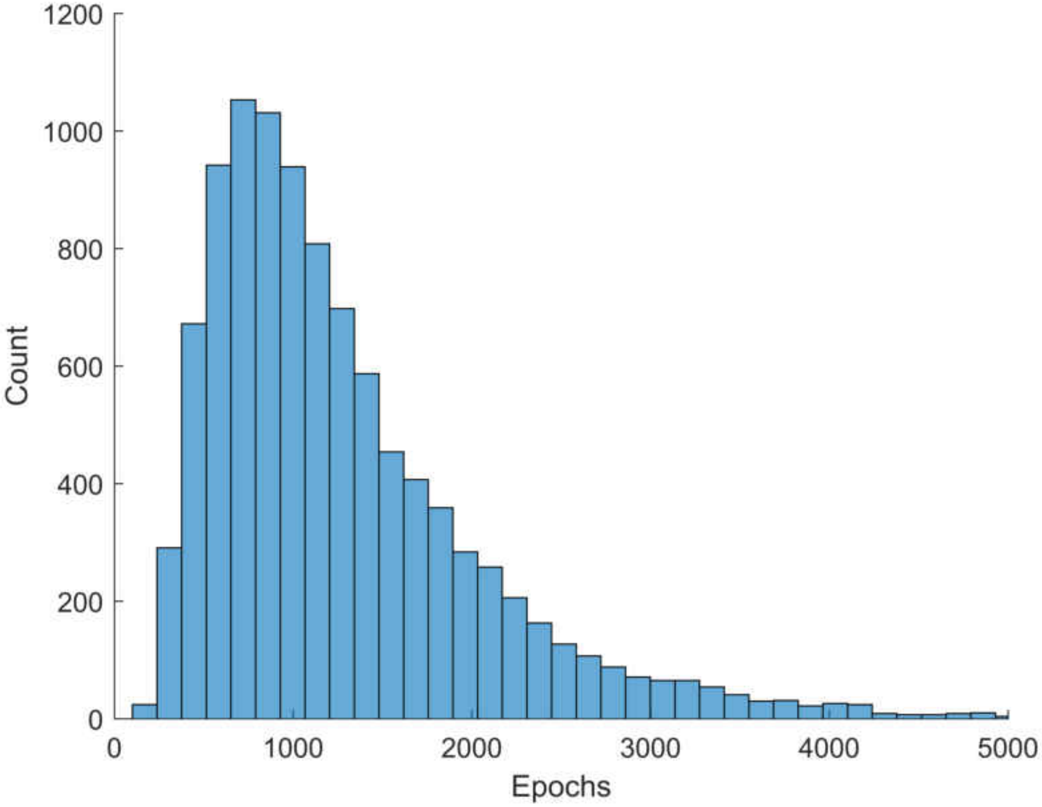

Figure 2.8: Histogram of required epochs until the state space has been fully explored under RAS.

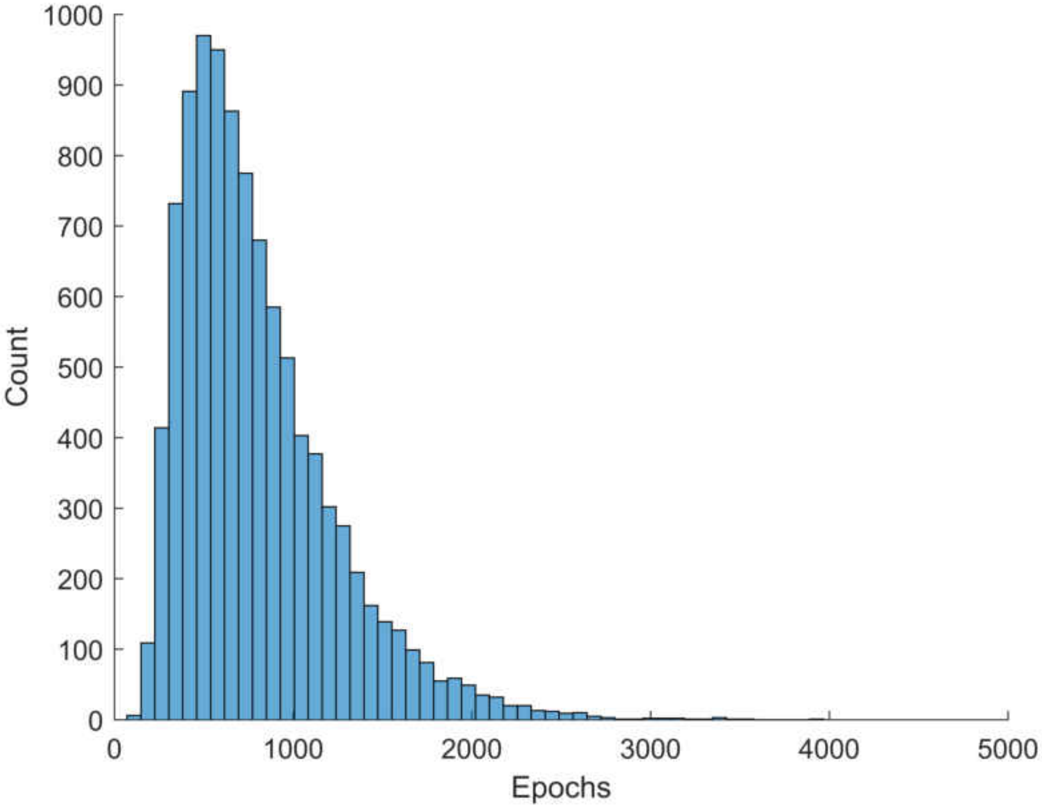

Figure 2.9: Histogram of required epochs until the state space has been fully explored under LDAS.

Table 2.2: Statistics For Explore Stop Criterion (Discrete)

| Action Selection Protocol | Mean | Variance | Standard Deviation | Standard Error |
|---|---|---|---|---|
| RAS | 1220.7 | 636,113.9 | 797.6 | 8.0 |
| LDAS | 780.5 | 198,132.5 | 445.1 | 4.5 |

Sample Size: 10,000

The mean and standard error reveal that random action selection is significantly worse at exploring. The tendency of random action selection to lead the operator back to the same state space is what low discrepancy helps avoid. Not only is the state space explored much quicker under low discrepancy, but the hitting times for all states are lower:

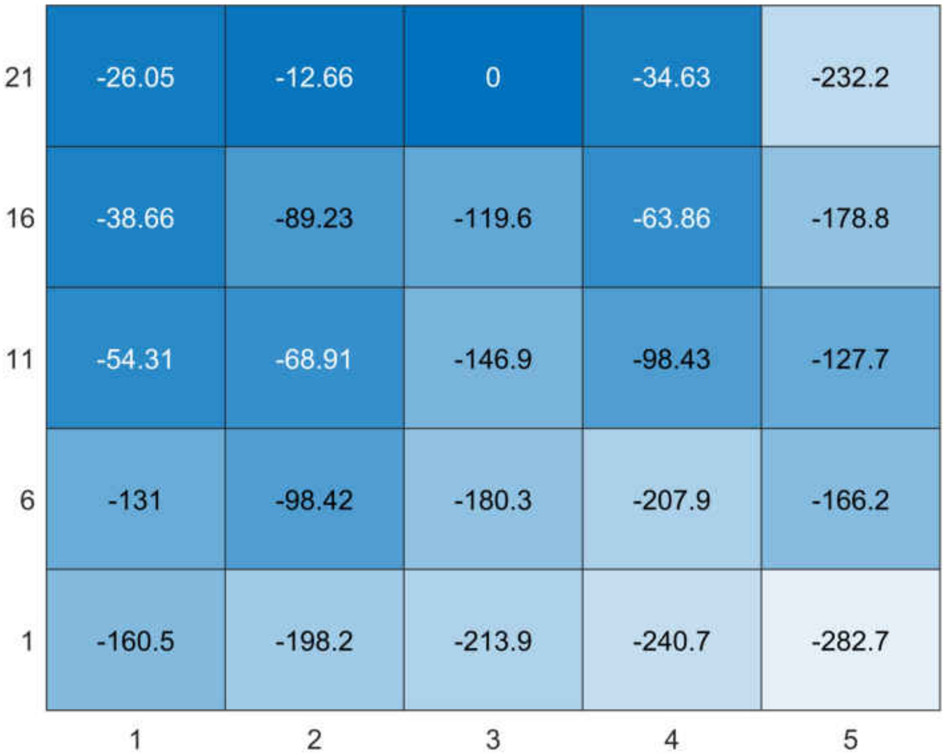

Figure 2.10: A heatmap of the difference of hitting times for each state (LDAS - RAS). The operator initializes in state 23, hence both action selection protocols have identical hitting times for that particular state.

Figure 2.10 is a heatmap for the difference in average hitting time for low discrepancy action selection versus random action selection. Negative values indicate the operator's hitting time for that state was on average superior under low discrepancy action selection. As we expected all of the states have negative values expect for the initial state.

To explore is beneficial to the exploitation phase of the decision process - the faster we traverse the environment the faster we can exploit that data to develop optimal strategies. For this reason it is important that the operator not only reach his goal, but cover as much *new* ground as possible on the path to the goal state. If the operator selects his action randomly, he may arrive at the goal state reliably, but only travel through a subset of the state space. This has the effect of limiting the amount of knowledge to be exploited due to the full state space not being known.

This chapter presented theory on action selection in Markov decision processes that aimed at optimizing exploration. A proof of the hitting times for the two-state scenario was successfully presented. Following that empirical data was gathered for the discrete case. In our experiments, low discrepancy action selection outperformed random action selection by a considerable margin. Exhibiting tighter margins of error in estimating the mean number of iterations required for successful exploration of the state space.

# CHAPTER 3

# CONTINUOUS MDP

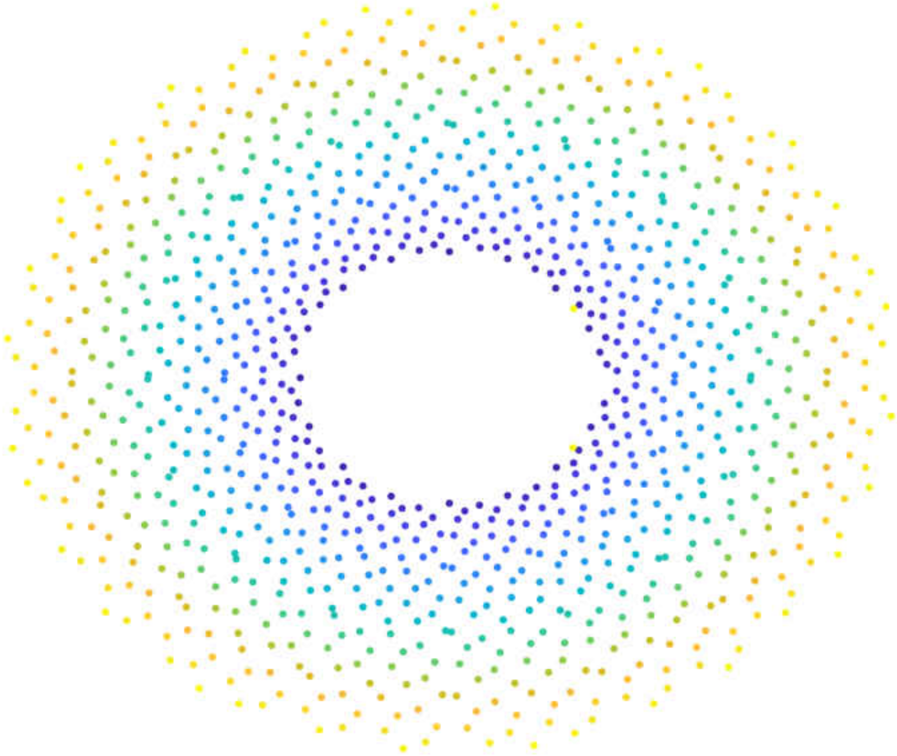

Figure 3.1: The modified Halton sequence given in polar coordinates for primes 2 and 7907.

The previous chapter introduced the theory of low discrepancy as it applied to discrete state-action spaces of a Markov decision process. In this chapter we will consider continuous state-action spaces and develop new mechanisms for the selection of low discrepancy actions from a continuous space. To wit, we will first review the Halton sequence: a multidimensional sequence of real numbers which exhibits low geometric discrepancy. Halton sequences will then be used on a partition of the state space from which we pool common histories, allowing us to avoid redundant behavior. Finally, empirical evidence will be collected from 10,000 trials ran on a continuous analog of Grid World.

## 3.1 Halton Sequence

The concept of a *low discrepancy* sequence is best explained geometrically. Intuitively, given a sequence of points $\boldsymbol{X} = \{\boldsymbol{X}_i = (x_{i1}, \ldots, x_{in}) \in \mathbb{R}^n \mid i \in I\}$, we say that the sequence is of low geometric discrepancy if the sequence emits minimal clustering while being distributed as uniformly as possible. Figure 3.1 is an example of such a sequence. In this chapter we will introduce some theory related to discrepancy and construct two commonly utilized sequences of low discrepancy.

Let $E$ be an arbitrary set, $E \in \mathbb{R}^n$, and $N$ an integer. Let $A(E, N, \boldsymbol{X})$ count the number of the first $N$ indices of $\boldsymbol{X}$ belonging to $E$ [11]. Define $1_E$ to be the characteristic function of $E$. Then we may write $A(E, N, \boldsymbol{X})$ as

$$A(E, N, \boldsymbol{X}) = \sum_{k=1}^{N} 1_E(\boldsymbol{X}_k).$$

The function $A(E, N, \boldsymbol{X})$ measures the proportion of the first $N$ elements of $\boldsymbol{X}$ contained in $E$. With $A(E, N, \boldsymbol{X})$ we can measure the deviation of a sequence from being uniform via the discrepancy function,

$$\Delta_{(X)}(E,N) := \frac{A(E,N,\boldsymbol{X})}{N} - \lambda_n(E)$$

where $\lambda(E)$ is the Lebesgue measure of $E$.

A sequence $\boldsymbol{X} = \{\boldsymbol{X}_i \mid i \in I\}$ in the $n$-dimensional unit cube is said to be *uniformly distributed modulo one* if

$$\lim_{N\to\infty} \frac{A(E,N,\boldsymbol{X})}{N} = \lambda_n(E), \quad \forall E \subseteq [0,1]^n.$$

The deviation of a given sequence from being uniform is not an easy thing to calculate, if it can even be calculated analytically. We *can* place an upper-bound on the deviation though. The star discrepancy, $D_N^*$, is the sup-norm of $\Delta_{(X)}(X,N)$,

$$D_N^*(\boldsymbol{X}) := \sup_{X\in[0,1]^n} |\Delta_{(X)}(X,N)|.$$

A simple, one-dimensional, sequence of low discrepancy is the *van der Corput* [11] sequence. To construct a van der Corput sequence we must first express an integer via its $b$-ary representation, i.e.

$$n = \sum_{k=0}^{L-1} d_k(n) b^k, \quad 0 \leq d_k(n) < b$$

where $b$ is the base for which the number is represented, $L$ is the length of the representation, and $d_k(n)$ is the $k^{\text{th}}$ digit in the $b$-ary expansion of $n$. From this, the van der Corput sequence may be constructed. The $n^{\text{th}}$ term of the van der Corput sequence is given by the *radical inverse function*, $\Psi : \mathbb{N} \to [0,1)$, defined by

$$\Psi_b(n) = \sum_{k=0}^{L-1} d_k(n) b^{-k-1}.$$

An example of a van der Corput sequence for $b = 2$ would be

$$\left\{ \frac{1}{2}, \frac{1}{4}, \frac{3}{4}, \frac{1}{8}, \frac{5}{8}, \frac{3}{8}, \frac{7}{8}, \frac{1}{16}, \frac{9}{16}, \frac{5}{16}, \frac{13}{16}, \frac{3}{16}, \frac{11}{16}, \frac{7}{16}, \frac{15}{16}, \dots \right\}.$$

The base in which the above sequence is constructed is immaterial. For example, we could have constructed the sequence is base 2,

$$\{0.1_2, 0.01_2, 0.11_2, 0.001_2, 0.101_2, 0.011_2, 0.111_2, 0.0001_2, 0.1001_2, \dots\}.$$

The terms of the van der Corput sequence are deterministic with no randomness involved, therefore it is called a *quasirandom sequence*, although they do mimic some properties of uniformly distributed random sequences.

A generalization of the van der Corput sequence to higher dimensions is achieved by the Halton sequence (Figure 3.1 was created via a modified version of the van der Corput sequence) [11]. To construct the Halton sequence of dimesnion $n$ we choose $n$ coprime numbers as bases, then we construct the Van der Corput sequence corresponding to each prime. This procedure generates $n$ sequences, one for each dimension. If we choose the coprimes 2 and 3, then

$$\boldsymbol{X}_2 = \left\{\frac{1}{2}, \frac{1}{4}, \frac{3}{4}, \frac{1}{8}, \frac{5}{8}, \frac{3}{8}, \frac{7}{8}, \frac{1}{16}, \frac{9}{16}, \dots\right\}, \quad \text{and}$$

$$\boldsymbol{Y}_3 = \left\{\frac{1}{3}, \frac{2}{3}, \frac{1}{9}, \frac{4}{9}, \frac{7}{9}, \frac{2}{9}, \frac{5}{9}, \frac{8}{9}, \frac{1}{27}, \dots\right\}.$$

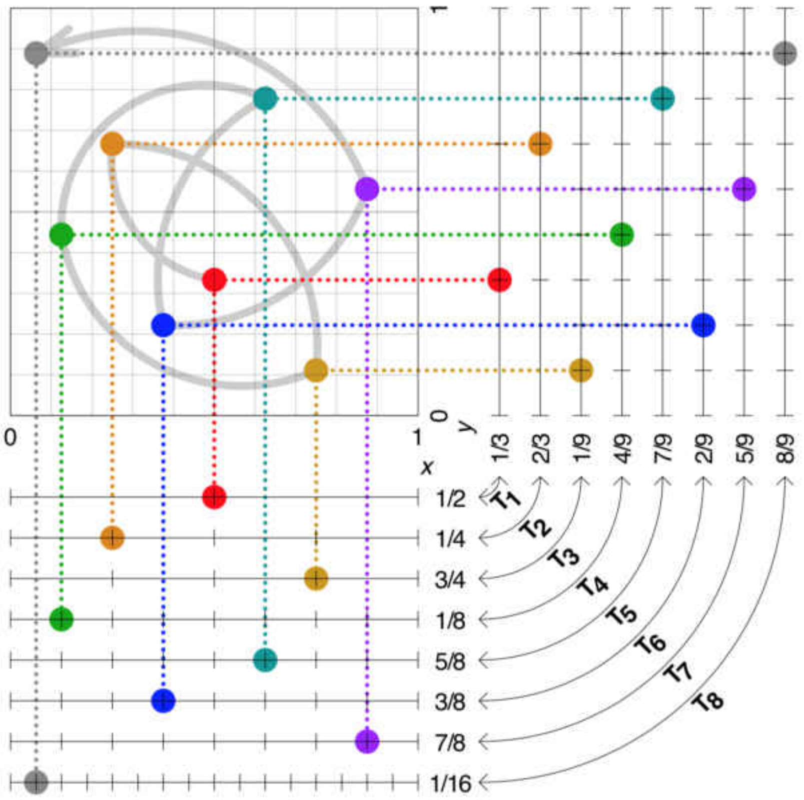

Figure 3.2: The Halton sequence with coprimes 2 and 3.

`https://en.wikipedia.org/wiki/Halton_sequence`

The result of this division is a sequence which has low geometric discrepancy. To construct an $n$ - dimensional Halton sequence, all we require are $n$ coprime numbers, $b_1, b_2, \dots, b_n$ and the radical inverse function $\Psi$. Then the $n$-dimensional Halton sequence is defined as

$$\Phi_n := \{(\Psi_{b_1}(k), \Psi_{b_2}(k), \ldots, \Psi_{b_n}(k))\}_{k \in \mathbb{N} \cup 0}.$$

**Theorem 2.1** [7] *Let $b_1, b_2, \ldots, b_n$ be coprime integers and $\boldsymbol{X}$ a Halton sequence. Then for $N \geq 2$, the star discrepancy is bounded above by*

$$D_N^*(\boldsymbol{X}) \leq \frac{1}{Nn!} \prod_{i=1}^{n} \left( \frac{\lfloor b_i/2 \rfloor \log(N)}{\log b_i} + n \right) + \sum_{k=0}^{s-1} \frac{b_{k+1}}{k!} \prod_{i=1}^{k} \left( \frac{\lfloor b_i/2 \rfloor \log(N)}{\log b_i} + k \right).$$

The proof of this theorem is too involved for this paper. A proper handling of the subject, as well as the proof of the above theorem (on pg. 91), may be found in *Digital Nets and Sequences* by Josef Dick and Friedrich Pillichshammer. Note that the Halton sequence is noteworthy because it has the lowest known bound of $D_N^*(X)$ [8].

## 3.2 Continuous System Dynamics

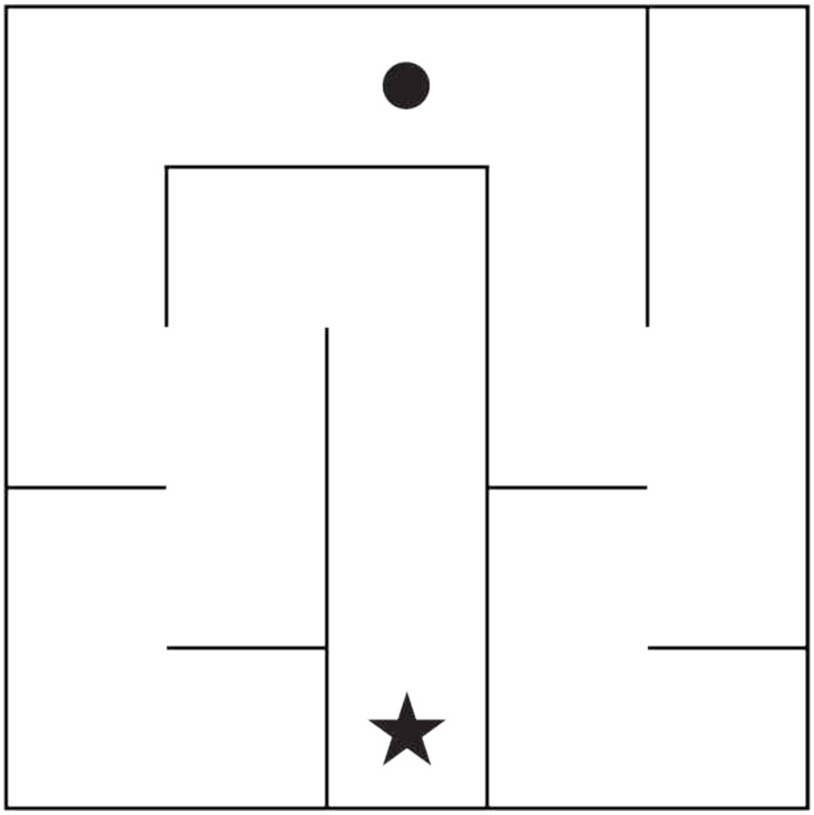

Figure 3.3: Continuous analog of Grid World.

The operator inhabits a world much akin to the discrete Grid World of the previous chapter. But unlike the previous chapter where the operator inhabited a finite set of states, the state space in the continuous case is an uncountable pair of points within the square $[0, 5] \times [0, 5]$. The barriers and starting regions are analogous to the discrete case. The operator initializes at $(2.5, 4.5)$ which corresponds to the center of the discrete state 23 from the previous chapter.

The actions are real valued vectors in the square $[-1, 1] \times [-1, 1]$. Under random action selection two numbers, $r_x$ and $r_y$, both distributed as uniform($-1, 1$), are chosen as the respective horizontal and vertical distances the operator is to travel. Once an action is selected, we calculate a Euclidean distance between the origin and the selected action. The operator will follow the trajectory dictated by the selected action until an obstacle is met or until the distance is completed. If the operator encounters a barrier its exploration does not terminate. Instead, the operator will ricochet off the surface of the boundary, whose mechanics are described in Appendix C, and complete its distance. No friction is introduced either on the surface of the barrier or along the operator's trajectory.

After an action is selected we introduce a small amount of noise. A random variable $e$, distributed as $e \sim \text{Normal}(0, 0.001)$, is added to both coordinates of the given action to generate a small amount of randomness. This mimics the 80-10-10 distribution for the discrete state-action instance.

Assigning a Halton sequence to every point within the state space will cause the operator to select the first member of the Halton sequence as his action for every state. This phenomenon occurs because every member of $S$ has a Lebesgue measure of zero and hence has a zero probability of being traversed multiple times. To avoid this conundrum we will instead partition the state space into neighborhoods which share a common action history. Let $\Omega$ define a partition of the continuous state space $S$ and $I$ some indexing set, then

$$\Omega := \left\{ S_i \;\middle|\; \bigcap_{i \in I} S_i = \emptyset \wedge \forall i, j \in I : S_i \neq S_j \wedge \bigcup_{i \in I} S_i = S \right\}.$$

We call each $S_i \in \Omega$ a cell. Given $s \in S, \exists! i \in I$ such that $s \in S_i$.

An example of a simple partition is the discrete state space encountered in the previous chapter. For the two-dimensional continuous space $S$, we may define a partition $\Omega$ which covers $S$ in a family of cells $S_1, S_2, \ldots, S_{25}$ of equal Lebesgue measure. These partitions are equivalent to the discrete state spaces previously encountered. Figure 3.4 re-illustrates this space.

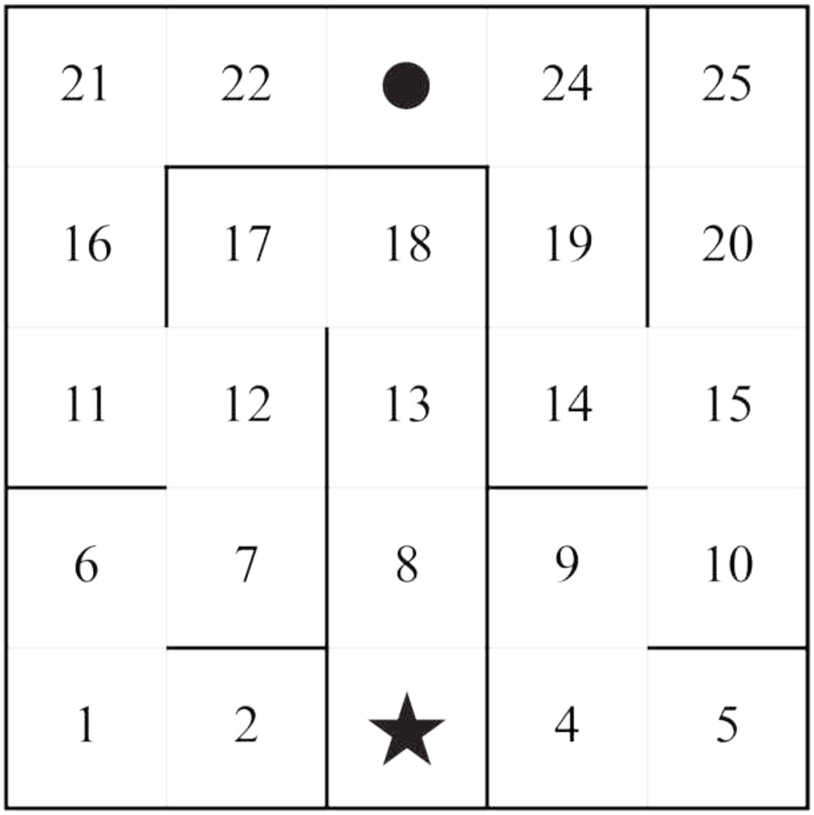

Figure 3.4: The partition $\Omega := \{S_1, S_2, S_3, \ldots, S_{25}\}$ of the continuous Grid World state space $S$.

For each member $S_i \in \Omega$ we define a random variable $h_i : S_i \to \mathbb{N}$ which counts the number of times the operator has been observed in cell $S_i$. Under low discrepancy action selection we choose primes $P_1, \ldots, P_j$ to generate a Halton sequence $\Phi$ of dimension $j$. When the operator is observed in cell $S_i$ we will choose element $h_i$ of $\Phi$ to perform as an action. Under random action selection, the operator selects his action uniformly. Other random action selection protocols will not be considered in this paper.

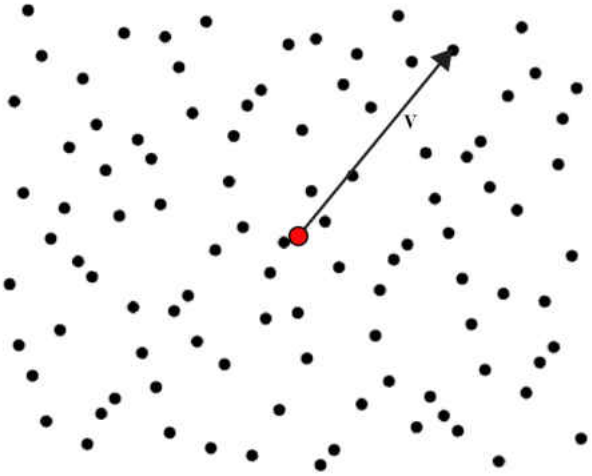

Figure 3.5: First 100 elements of the Halton Sequence with co-prime base 2 and 3. Operator $i$ is in red, the velocity he chooses is element $h_i$ of the Halton sequence.

The state space may be partitioned into cells which have a common history. We may increase the refinement of the partition to an arbitrary degree. For simplicity, and since we are familiar with the discretization of the state space previously exhibited, we will stick to 25 partitions. No claim is made that this is superior to any other refinement, this is just a single realization of the problem.

If the operator is in bin $i$, then his next action is given by the $h_i^{\text{th}}$ element of Halton$(P_1, P_2)$. We may denote the coordinates of this action by $l_x$ and $l_y$. Since the Halton sequence produces points which lie entirely in the first quadrant, the linear transformation $L(x) = 2x - 1$ is applied to both coordinates to center the distribution at zero.

$$a_{\text{LDAS}} \leftarrow (2l_x - 1, 2l_y - 1).$$

Under random action selection the operator's actions are uniformly distributed over the same convex set $\{(r_x, r_y) \mid r_x, r_y \sim \text{unif}(-1, 1)\}$.

## 3.3 Empirical Data

Unlike in Chapter 2 where we successfully derived a hitting time distribution for random action selection, deriving an exact hitting time distribution for both random action selection and low discrepancy action selection is intractable. Therefore only empirical evidence can be supplied to support the utilization of Halton sequences in an effort to optimize exploration. To this end, we will use MATLAB to simulate the operators exploration of Grid World. The following algorithm will facilitate our needs:

**Algorithm 3** Continuous/Continuous Grid World

```
procedure MAIN
    INPUT ← state $i$,    initial state
    Initialize state-action history vector
    Initialize hitting time storage
    Define boundary matrix
    while ∃ states unexplored do
        $action(i) \sim$ uniform$(-1, 1)$;    if RAS
        $n$ ← action history for current state
        action ← Halton($n$) sequence using action history;    if LDAS
        Update state-action history
        Calculate collisions
    end while
end procedure
```

First, a comparison between the number of epochs required by the operator under RAS and LDAS will be made. Our goal is to reach any state in partition 3 starting in a state within partition 23. We will perform this experiment for multiple pairs of primes over 10,000 trials. The selection of primes was arbitrary.

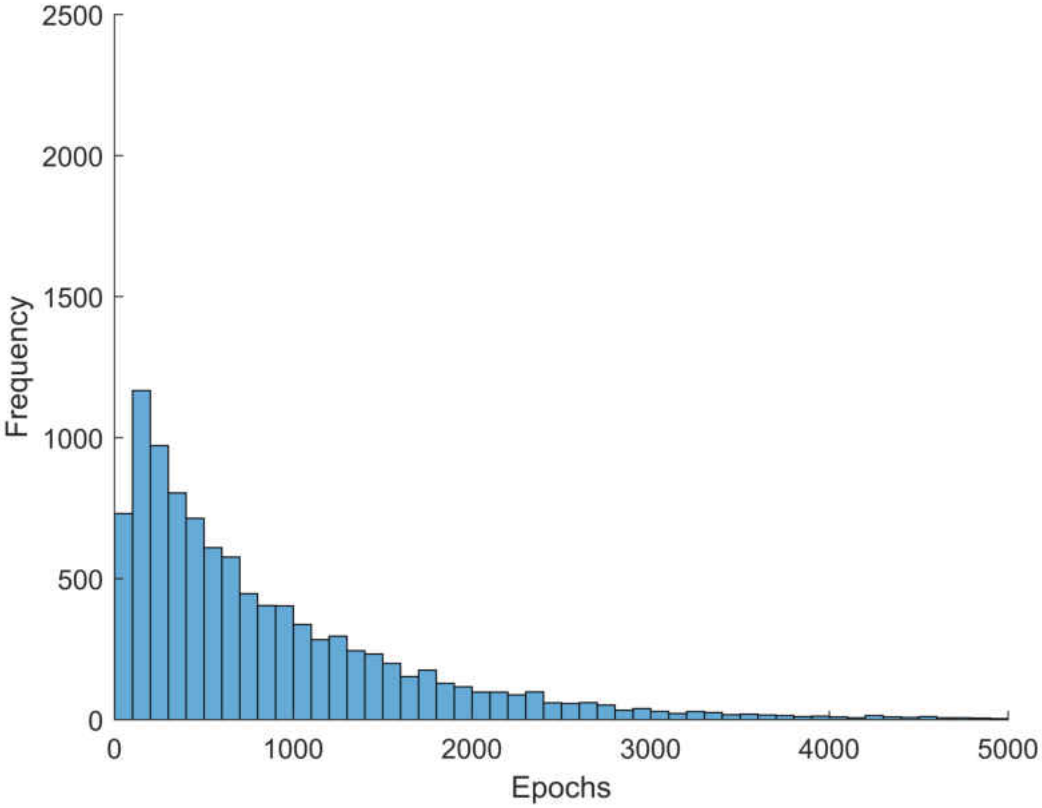

Figure 3.6: Histogram of required epochs until a state within partition 3 is reached from the initial state under RAS.

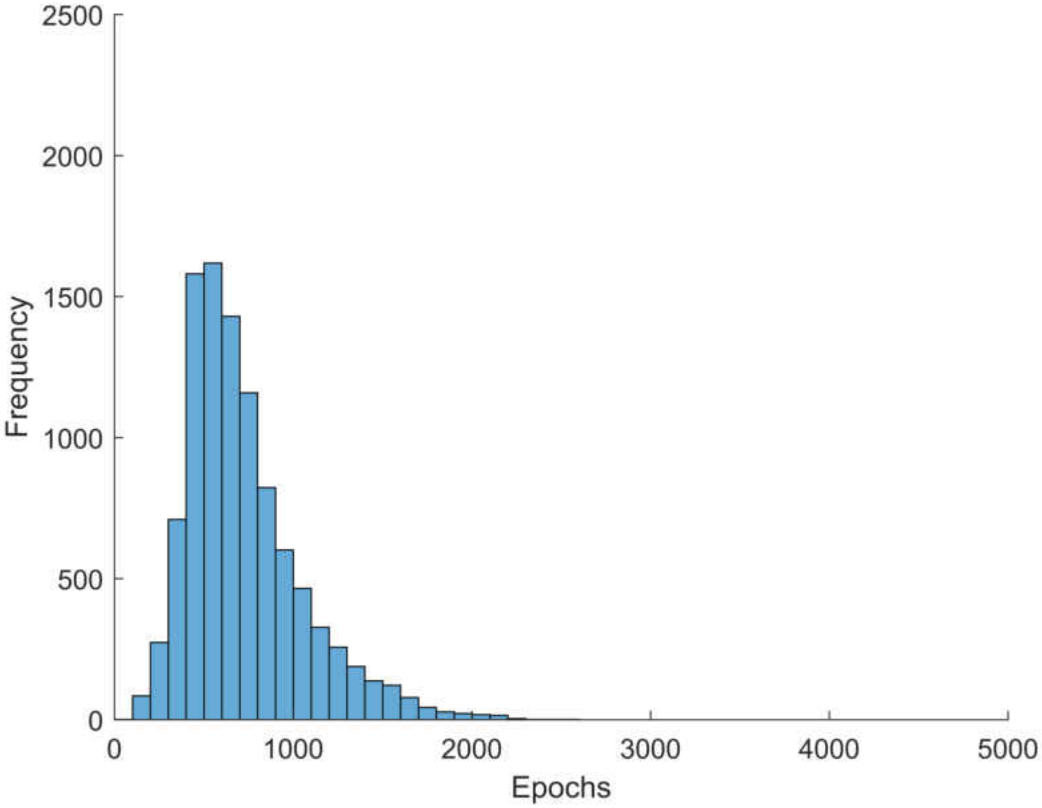

Figure 3.7: Histogram of required epochs until a state within partition 3 is reached from the initial state under LDAS with primes 2 and 3.

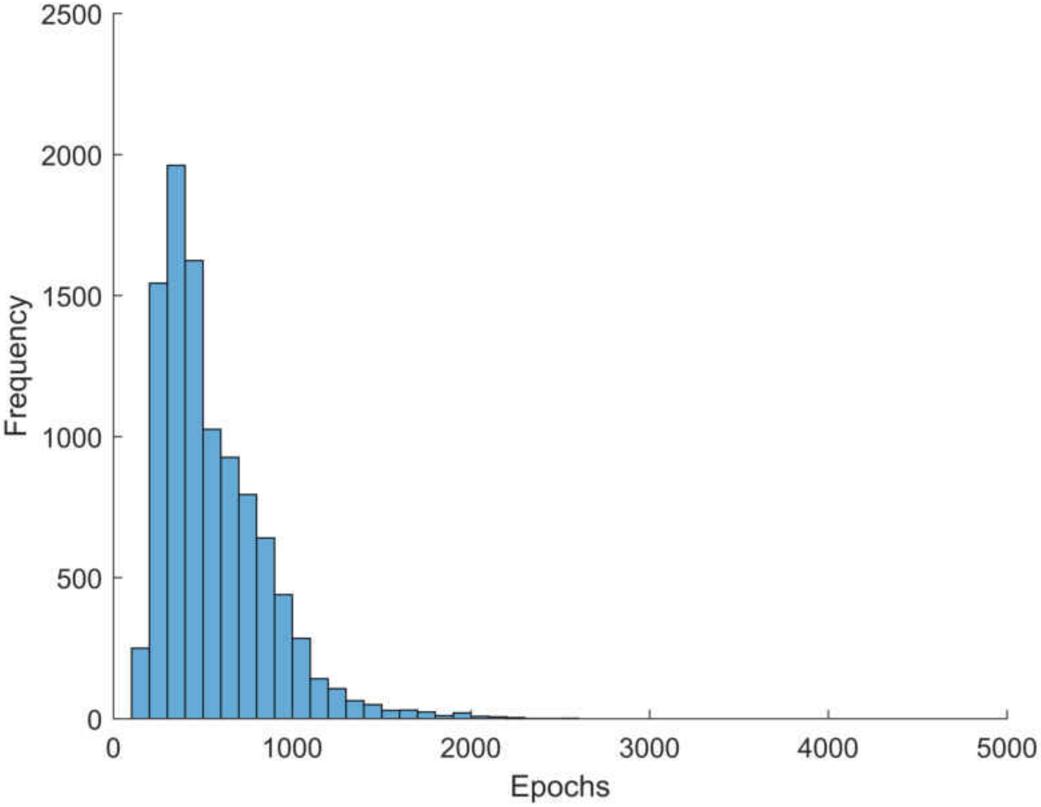

Figure 3.8: Histogram of required epochs until a state within partition 3 is reached from the initial state under LDAS with primes 3 and 5.

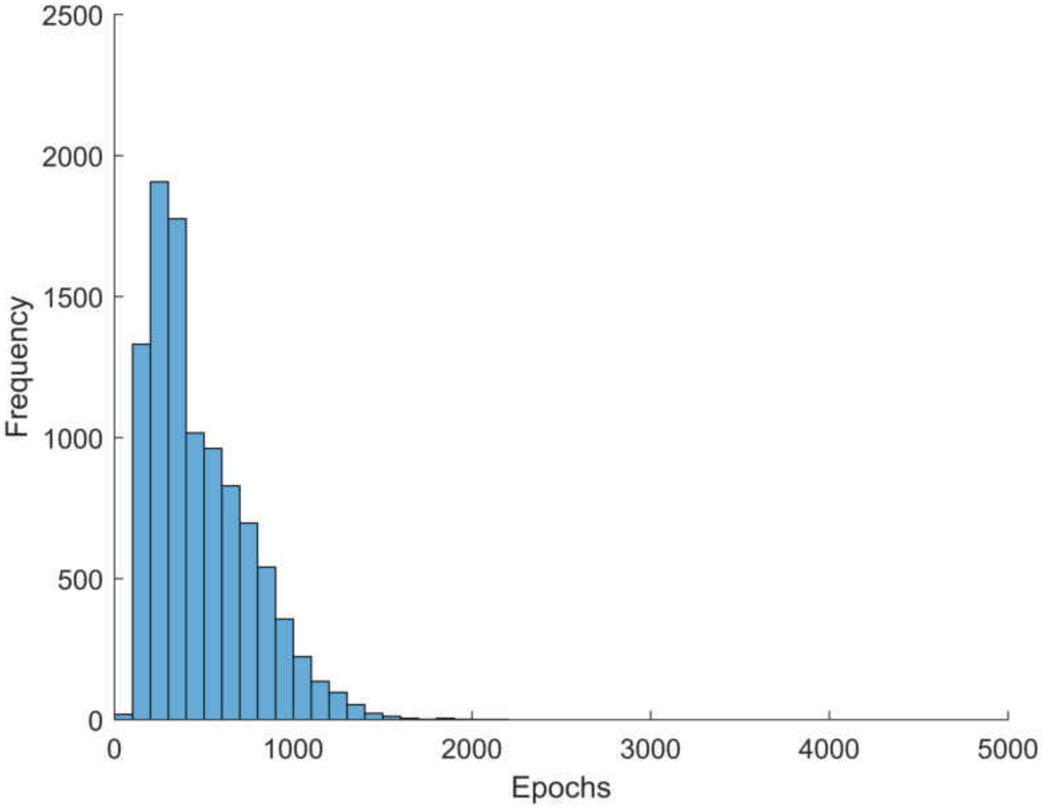

Figure 3.9: Histogram of required epochs until a state within partition 3 is reached from the initial state under LDAS with primes 5 and 13.

Selecting actions randomly resulted in a right skewed distribution with higher variance and a higher mean. The histograms for LDAS under all three pairs of primes exhibited a tighter grouping indicating lower variance. This data is summarized below:

Table 3.1: Statistics For Goal Stop Criterion (Continuous)

| **Action Selection Protocol** | **Mean** | **Variance** | **Standard Deviation** | **Standard Error** |
|---|---|---|---|---|
| RAS | 876.8 | 753,952.8 | 868.3 | 8.7 |
| LDAS(2,3) | 723.9 | 107,896.8 | 328.5 | 3.3 |
| LDAS(3,5) | 558.2 | 92,000.8 | 303.3 | 3.0 |
| LDAS(5,13) | 483.4 | 77,935.8 | 279.2 | 2.8 |

Sample Size: 10,000. Goal state: 3.

The standard error for random action selection is considerably higher than either of the corresponding standard errors for low discrepancy action selection.

The selection of the goal state was arbitrary. We could have selected any of the remaining 24 partitions as goal states and repeated the above experiment. The main result of this section is that exploration of the state space is faster under low discrepancy action selection. Therefore choosing any other goal state should result in higher performance for low discrepancy action selection. The same experiment as the one above was conducted again for 10,000 more trials with the condition that the operator explores until all 25 partitions have been visited a minimum of once. The action selection protocol and the starting states are identical to the previous goal-oriented experiment.

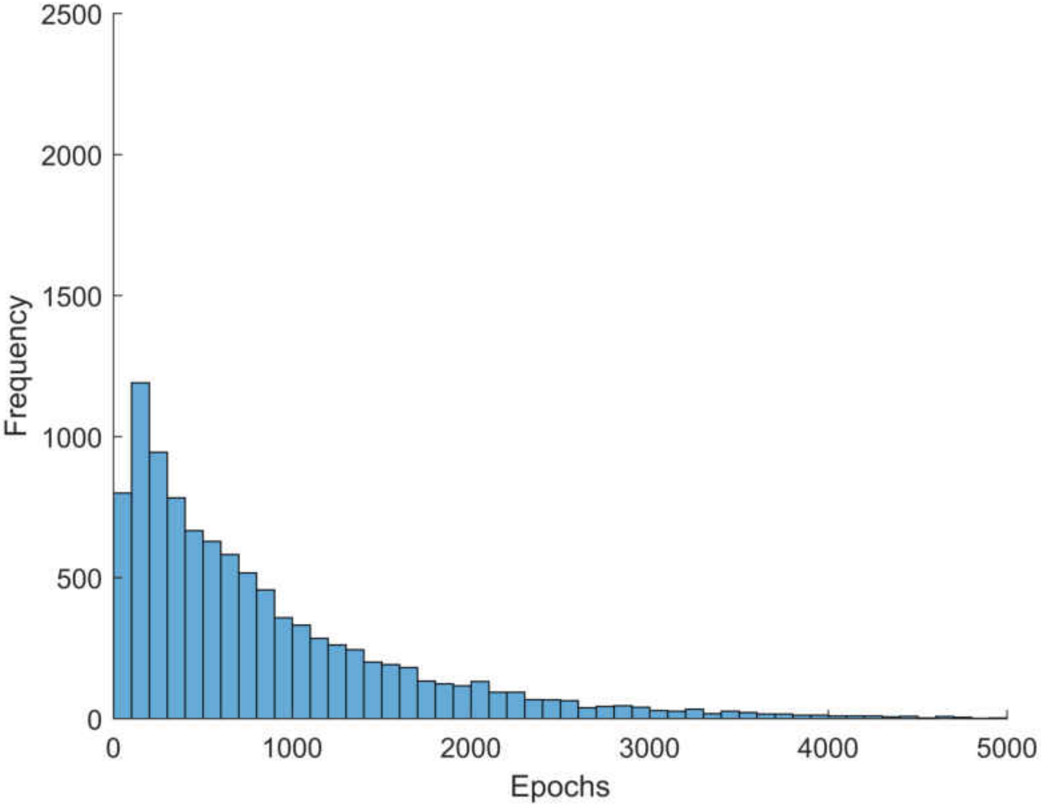

Figure 3.10: Histogram of required epochs until sufficient exploration reached under random action selection.

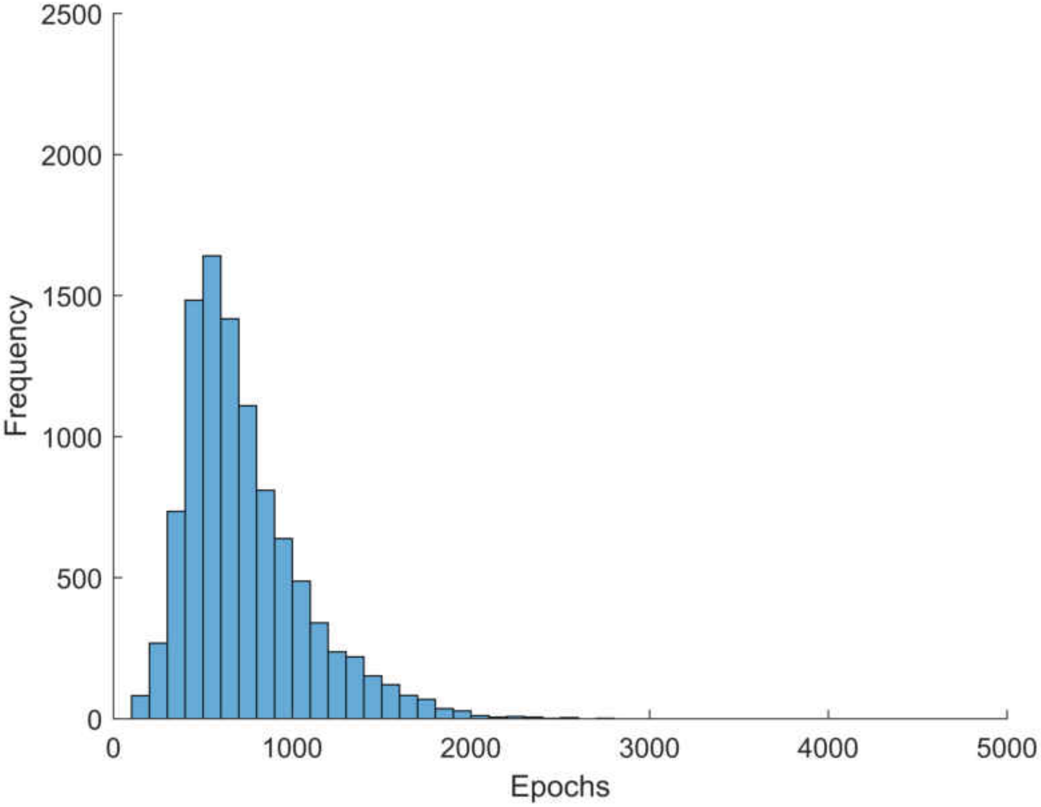

Figure 3.11: Histogram of required epochs until sufficient exploration reached under low discrepancy action selection with primes 2 and 3.

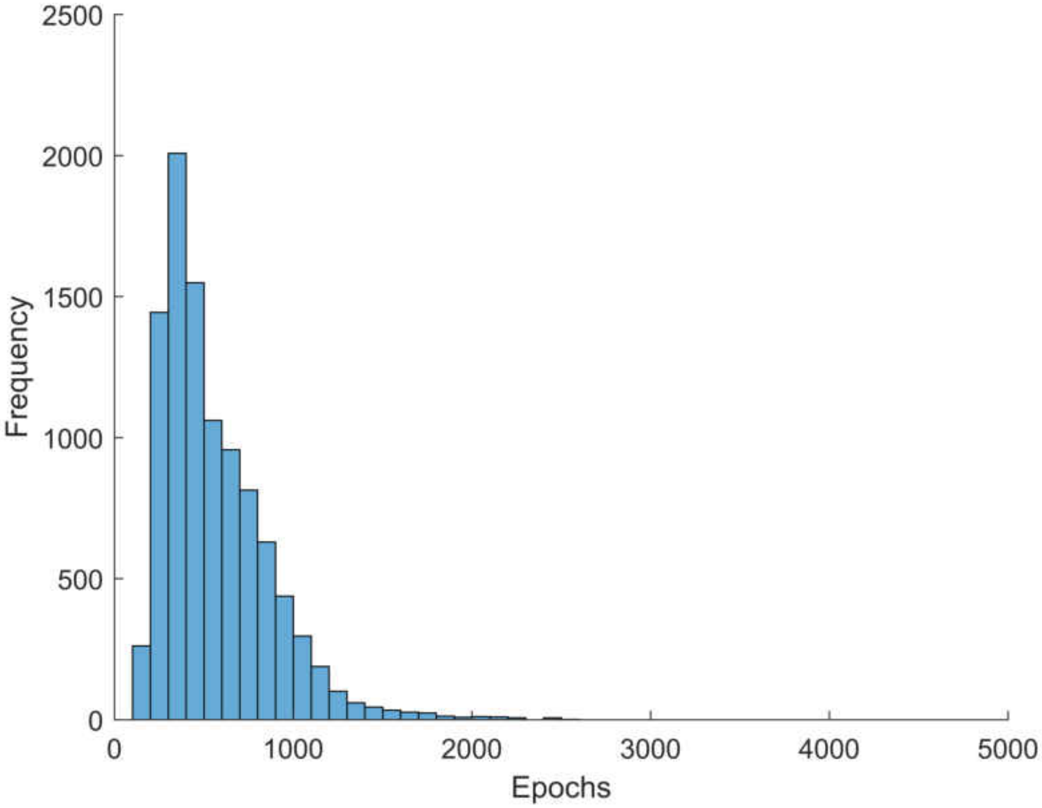

Figure 3.12: Histogram of required epochs until sufficient exploration reached under low discrepancy action selection with primes 3 and 5.

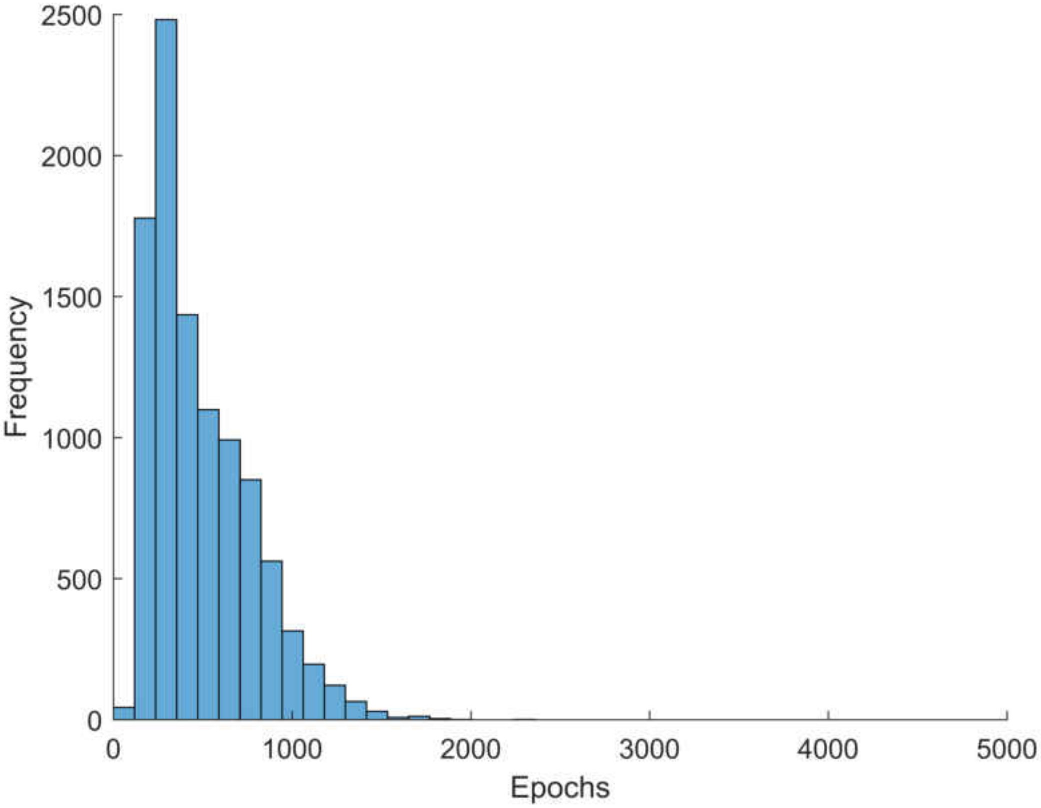

Figure 3.13: Histogram of required epochs until sufficient exploration reached under low discrepancy action selection with primes 5 and 13.

Again, selecting actions randomly resulted in a right skewed distribution with higher variance and a higher mean. The histograms for low discrepancy action selection under all three pairs of primes exhibited a tighter grouping indicating lower variance. This data is summarized below:

Table 3.2: Statistics For Explore Stop Criterion (Continuous)

| **Action Selection Protocol** | **Mean** | **Variance** | **Standard Deviation** | **Standard Error** |
|---|---|---|---|---|
| RAS | 861.6 | 717,798.3 | 847.2 | 8.5 |
| LDAS(2,3) | 564.1 | 93,947.6 | 306.5 | 3.1 |
| LDAS(3,5) | 733.3 | 115,169.0 | 339.4 | 3.4 |
| LDAS(5,13) | 483.4 | 77,935.8 | 279.2 | 2.8 |

Sample size: 10,000

The summary data bears a remarkable resemblance to the data obtained in the goal-oriented experiment. The standard error for LDAS is considerably lower than the RAS analog.

This chapter focused on the use of quasi-random sequences to facilitate low discrepancy action selection in continuous state-action spaces. A partition over the state space allows for all actions from members of the partition to share a common history. Then when the operator encounters that cell of the partition at a later epoch he will choose an action from a Halton sequence ensuring minimal discrepancy from previously attempted actions.

# CHAPTER 4

# CONCLUSION

A logical question to ask is what happens as the partition over the state space becomes arbitrarily refined? This is a question which deserves further investigation. As the partition becomes more and more refined less histories are shared. The benefits of refining the state space in this manner may not outweigh the computational penalties associated with the increased number of cells.

The theory of low discrepancy sequences as they apply to action selection in Markov decision processes is far from complete. The two state proof coupled with copious empirical evidence warrants further investigation into this area. The possibility of scrambling a Halton sequence to produce a sequence with even lower geometric discrepancy via permutating the indices of the sequence was investigated by Mascagni and Chi [5]. Their use in our action selection theory presents an avenue of further research. The obvious problem of our theory lies in the selection of primes. While we believe that on average any pair of primes chosen will outperform random action selection due to the nature of Halton sequences, it would be prudent to ensure such behavior. The *Generalized Halton Sequence* devised by Mascagni and Chi is promising in that regard, but further investigation is not presented in this paper.

# REFERENCES

[1] K. Asadi and M. L. Littman. A new softmax operator for reinforcement learning. *CoRR*, abs/1612.05628, 2016.

[2] R. I. Brafman and M. Tennenholtz. R-max - a general polynomial time algorithm for near-optimal reinforcement learning. *J. Mach. Learn. Res.*, 3:213–231, Mar. 2003.

[3] S. W. Carden. Convergence of a q-learning variant for continuous states and actions. *Journal of Artificial Intelligence Research*, 49:705–731, 2014.

[4] N. Cesa-Bianchi, C. Gentile, G. Lugosi, and G. Neu. Boltzmann exploration done right. *CoRR*, abs/1705.10257, 2017.

[5] H. Chi, M. Mascagni, and T. Warnock. On the optimal halton sequence. *Math. Comput. Simul.*, 70(1):9–21, Sept. 2005.

[6] Z. Cvetkovski. *Newton's Inequality, Maclaurin's Inequality*, pages 117–119. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[7] J. Dick and F. Pillichshammer. *Digital Nets and Sequences: Discrepancy Theory and Quasi–Monte Carlo Integration*. Cambridge University Press, 2010.

[8] H. Faure. *Discrepancy lower bound in two dimensions*, pages 198–204. Springer New York, New York, NY, 1995.

[9] T. Hester and P. Stone. TEXPLORE: Real-time sample-efficient reinforcement learning for robots. *Machine Learning*, 90(3), 2013.

[10] R. Howard. *Dynamic Programming and Markov Processes*. Published jointly by the Technology Press of the Massachusetts Institute of Technology and, 1960.

[11] L. Kuipers and H. Niederreiter. *Uniform Distribution of Sequences*. Dover Books on Mathematics. Dover Publications, 2012.

[12] M. L. Littman and C. Szepesva. A generalized reinforcement-learning model: Convergence and applications. Technical report, Providence, RI, USA, 1996.

[13] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning*. Adaptive computation and machine learning series. MIT Press, 2012.

[14] R. Ortner. Optimism in the face of uncertainty should be refutable. *Minds and Machines*, 18(4):521–526, Dec 2008.

[15] M. Riedmiller. *Neural Fitted Q Iteration – First Experiences with a Data Efficient Neural Reinforcement Learning Method*, pages 317–328. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[16] G. Rummery and M. Niranjan. *On-line Q-learning Using Connectionist Systems*. CUED/F-INFENG/TR. University of Cambridge, Department of Engineering, 1994.

[17] C. J. C. H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3):279–292, May 1992.

[18] Z. Xia and D. Zhao. Online reinforcement learning by bayesian inference. *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6, 2015.

[19] X. Zhang and H. Gao. Road maintenance optimization through a discrete-time semi-markov decision process. *Reliability Engineering and System Safety*, 103(Supplement C):110 – 119, 2012.

# Appendix A

# PROBABILITY MATRICES

Table A.1: Pr(down)

| $i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **.9** | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | **.1** | **.9** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | **.9** | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | **.1** | **.9** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | **.8** | 0 | 0 | 0 | 0 | **.1** | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | **.1** | **.9** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.2** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.1** | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | **.9** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.9** | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 12 | 0 | 0 | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | **.1** | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.2** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.9** | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | **.1** | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.2** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 17 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.1** | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | **.1** | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 19 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.2** | 0 | 0 | 0 | 0 | 0 | 0 |
| 20 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.2** | 0 | 0 | 0 | 0 | 0 |
| 21 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.1** | **.1** | 0 | 0 | 0 |
| 22 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | **.8** | **.1** | 0 | 0 |
| 23 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | **.8** | **.1** | 0 |
| 24 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | **.1** | **.1** | 0 |
| 25 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.2** |

Table A.2: Pr(left)

| $i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **.9** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | **.8** | **.2** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | **.9** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | **.9** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | **.8** | **.2** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | **.1** | 0 | 0 | 0 | 0 | **.9** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | **.8** | **.1** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.9** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.8** | **.1** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.9** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 12 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.9** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 |
| 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 |
| 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 |
| 17 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.9** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | **.8** | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 19 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.1** | 0 |
| 20 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.1** |
| 21 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.9** | 0 | 0 | 0 | 0 |
| 22 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.8** | **.2** | 0 | 0 | 0 |
| 23 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.8** | **.2** | 0 | 0 |
| 24 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | **.8** | **.1** | 0 |
| 25 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.9** |

Table A.3: Pr(right)

| $i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **.1** | **.8** | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | **.9** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | **.1** | **.8** | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | **.1** | 0 | 0 | 0 | 0 | **.1** | **.8** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | **.9** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.1** | **.8** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.9** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | **.8** | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 12 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | **.8** | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 | 0 |
| 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | 0 |
| 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 |
| 17 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.1** | **.8** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.9** | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 19 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.1** | 0 |
| 20 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.8** | 0 | 0 | 0 | 0 | **.1** |
| 21 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.1** | **.8** | 0 | 0 | 0 |
| 22 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.2** | **.8** | 0 | 0 |
| 23 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.2** | **.8** | 0 |
| 24 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.9** | 0 |
| 25 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **.1** | 0 | 0 | 0 | 0 | **.9** |

Table A.4: Pr(up)

| $i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | .1 | .1 | 0 | 0 | 0 | .8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | .1 | .9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | .2 | 0 | 0 | 0 | 0 | .8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | .1 | .1 | 0 | 0 | 0 | .8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | .1 | .9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | .9 | .1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | .1 | .1 | 0 | 0 | 0 | 0 | .8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .2 | 0 | 0 | 0 | 0 | .8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .9 | .1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .1 | .1 | 0 | 0 | 0 | 0 | .8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 11 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .1 | .1 | 0 | 0 | 0 | .8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .1 | .1 | 0 | 0 | 0 | 0 | .8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .2 | 0 | 0 | 0 | 0 | .8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .1 | .1 | 0 | 0 | 0 | .8 | 0 | 0 | 0 | 0 | 0 | 0 |
| 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .1 | .1 | 0 | 0 | 0 | 0 | .8 | 0 | 0 | 0 | 0 | 0 |
| 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .2 | 0 | 0 | 0 | 0 | .8 | 0 | 0 | 0 | 0 |
| 17 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .9 | .1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .1 | .9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 19 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .2 | 0 | 0 | 0 | 0 | .8 | 0 |
| 20 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .2 | 0 | 0 | 0 | 0 | .8 |
| 21 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .9 | .1 | 0 | 0 | 0 |
| 22 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .1 | .8 | .1 | 0 | 0 |
| 23 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .1 | .8 | .1 | 0 |
| 24 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | .1 | .9 | 0 |
| 25 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

# Appendix B

## HYPERCUBE

*hypercube* is an $n$-dimensional generalization of a regular three dimensional cube. Let $\boldsymbol{X}$ be the set of vertices of a hypercube. It may be of interest to select from this set in such a manner that we produce a low discrepancy set via a permutation on $\boldsymbol{X}$. In other words, we wish to define a permutation $\sigma : \boldsymbol{X} \to \boldsymbol{X}$, which produces a low discrepancy set of points. It is important to note that in this instance, we define discrepancy via a new definition. We define the discrepancy of a set to be the maximum distance between two adjacent members of said set. For instance, if we are given a set of vertices $\boldsymbol{X}$, then we aim to find a permutation $\sigma$, for which $\sigma(\boldsymbol{X}_i)$ and $\sigma(\boldsymbol{X}_{i+1})$ are sufficiently far from each other. Thus, given a vertex $i$ on some hypercube, we would like to avoid any permutations which place any of the neighboring vertices adjacent to $i$ in the sequence $(\sigma(\boldsymbol{X}_1), \sigma(\boldsymbol{X}_2), \ldots, \sigma(\boldsymbol{X}_n))$.

To illustrate this, consider the three dimensional hypercube with vertices

$$\boldsymbol{X} = \{000_2, 001_2, 010_2, 011_2, 100_2, 101_2, 110_2, 111_2\}.$$

We may deconstruct the hypercube in the following manner:

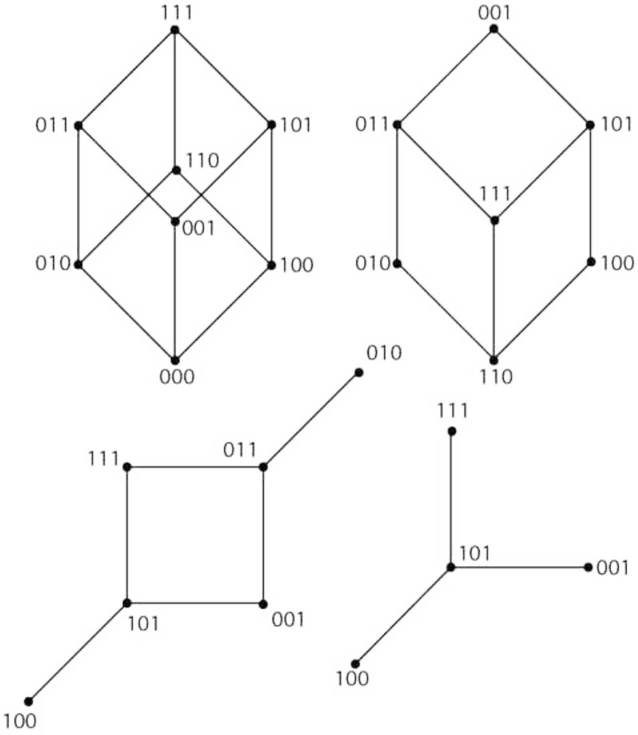

Figure B.1: Deconstructing the hypercube

We begin with the most obvious vertex, $000_2$, and then select a vertex which has no adjacent vertices with $000_2$, but whose removal leaves the most connected subgraph. We continue in this manner, selecting the $k^{\text{th}}$ vertex by inspecting the previous $k-1$ vertices, finding the vertex who is not adjacent to either of the $k-1$ previous vertices, and whose removal leaves the most connected subgraph.

**Algorithm 4** Hypercube Solver

```
procedure MAIN
    $A \leftarrow$ hypercube of dimension $j$
    $n \leftarrow$ Number of columns of A
    $v \leftarrow$ Stores the degree of each vertex
    $u \leftarrow$ Low discrepancy sequence
    $k \leftarrow$ Position index
    for $i = 1 : n$ do
        $v(i) \leftarrow sum(A(i, :))$
    end for
    while $max(v) > 0$ do
        $i \leftarrow max(v)$
        $u(k) \leftarrow i$
        $A \leftarrow A \setminus A(i, i)$
        for $j = 1 : n$ do
            $v(i) \leftarrow sum(A(i, :));$
        end for
        $n \leftarrow size(A, 2)$
        $k \leftarrow k + 1$
    end while
    $u((sum(u > 0) + 1) : length(u)) = setdiff(S, u);$
end procedure
```

Hypercube Solver takes a hypercube (defined as an adjacency matrix) and produces the low discrepancy sequence starting at position 1. The solver will safely take hypercubes up to 22 dimensions. If the dimension exceeds 22 the amount of required RAM the program needs to operate is out of the capabilities of most computers.

The main purpose of this section is to expand on the concept of low discrepancy. A geometric interpretation is not restricted to Euclidean space exclusively. Our goal is to apply these concepts to Markov Decision Processes in an effort to optimize the exploration phase of the process. The dynamics of the environment will vary from problem to problem, thus it is important that the idea of low discrepancy is robust and feasible for a large body of problems.

# Appendix C

## COLLISION MECHANICS

For the continuous action scenario we will allow for perfectly elastic collisions with the environment. Suppose the actors initial position is $P_i = (x_i, y_i)$ and the velocity chosen at this time epoch is $v$. Then, assuming no collisions occur, the final position, $P_f = (x_f, y_f)$ is $P_f = P_i + v$. To determine if there has been a collision, we need to encode information about the environment into a matrix:

$$\text{B} = \begin{pmatrix} 0 & 0 & 4 & 0 \\ 4 & 0 & 4 & 3 \\ 4 & 3 & 0 & 3 \\ 0 & 3 & 0 & 0 \\ 1 & 1 & 2 & 1 \\ 2 & 1 & 2 & 2 \\ 2 & 2 & 1 & 2 \\ 1 & 2 & 1 & 1 \end{pmatrix}$$

Each row of $B$ is a barrier of Grid World with the first two columns being the starting point and the last two columns being the ending point. For instance the equation of the bottom barrier of Grid World may be calculated by

$$\begin{aligned}\text{Eq1} &= (B(1,1), B(1,2)) + t[(B(1,3), B(1,4) - (B(1,1), B(1,2))]\\ &= (0,0) + t(4,0)\end{aligned}$$

We can do a similar calculation for the actor's path: $P_i + t(P_f - P_i)$. Note that $0 \leq t \leq 1$. To find the point of intersection we will solve

$$(B(1,1), B(1,2)) + t[(B(1,3), B(1,4) - (B(1,1), B(1,2))] = P_i + t(P_f - P_i)$$

Separating the equations by components gives

$$\begin{aligned}B(1,1) + t_1(B(1,3) - B(1,1)) &= x_i + t_2(x_f - x_i)\\ B(1,2) + t_1(B(1,4) - B(1,2)) &= y_i + t_2(y_f - y_i)\end{aligned}$$

isolating the terms with $t_1$ and $t_2$ on the left hand side yields

$$\begin{aligned}t_1(B(1,3) - B(1,1)) - t_2(x_f - x_i) &= x_i - B(1,1)\\ t_1(B(1,4) - B(1,2)) - t_2(y_f - y_i) &= y_i - B(1,2)\end{aligned}$$

which may be turned into the matrix equation

$$\begin{pmatrix} B(1,3) - B(1,1) & x_i - x_f \\ B(1,4) - B(1,2) & y_i - y_f \end{pmatrix} \begin{pmatrix} t_1 \\ t_2 \end{pmatrix} = \begin{pmatrix} x_i - B(1,1) \\ y_i - B(1,2) \end{pmatrix}$$

which may be solved by taking the inverse of the first matrix (we assume it is not singular which would only happen if the path of travel is parallel to either of the boundaries, but since we are sampling from a continuous space for action selection the probability of that occurrence is null).

$$\begin{pmatrix} t_1 \\ t_2 \end{pmatrix} = \begin{pmatrix} B(1,3) - B(1,1) & x_i - x_f \\ B(1,4) - B(1,2) & y_i - y_f \end{pmatrix}^{-1} \begin{pmatrix} x_i - B(1,1) \\ y_i - B(1,2) \end{pmatrix}.$$

If both $t_1$ and $t_2$ are less than zero then a collision has occurred. To find out what the result of the elastic collision would be we need to need calculate the reflection about the normal of the barrier surface. Figure 3.8 shows the operator traveling with velocity $v$ colliding with a barrier. The normal vector, $n$, is normal to the surface of the barrier and emanates from the point of collision. The result of this collision is a new velocity, $v'$, which is to be determined. First we must quantify what portion of $v$ travels in the direction of $n$, or rather what portion of $v$ is parallel to the direction of $n$. To do this we will take the dot product of $v$ and $n$, then multiply by $n$ to give it the direction we desire:

$$u = (v \cdot n)n$$

then we will subtract two times $u$ from $v$ to give us $v'$,

$$v' = v - 2u = v - 2(v \cdot n)n.$$

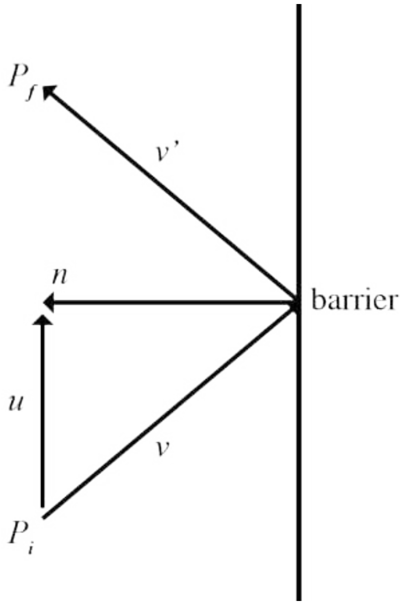

Figure C.1: Operator’s collision with the environment