**LEARNING AUTOMATA SOLUTIONS TO ENHANCING OPTIMAL SEARCH FOR
UNKNOWN TARGET DISTRIBUTIONS**

by

AMR ELLAITHY, B.Sc.

A thesis submitted to the

Faculty of Graduate Studies and Research

in partial fulfillment of the requirements for the degree of

Master of Computer Science

Department of Computer Science

Carleton University

Ottawa, Ontario

December 2002

©copyright

2002, Amr ELLaithy

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-79764-3

Canada

The undersigned recommend to
the Faculty of Graduate Studies and Research
acceptance of the thesis

**Learning Automata Solutions to Enhancing Optimal Search for Unknown Target
Distributions**

submitted by **Amr Ellaithy, B.Sc.**
in partial fulfillment of the requirements for
the degree of Master of Computer Science

Director, Department of Computer Science

Thesis Supervisor

CARLETON UNIVERSITY

December 2002

ABSTRACT

We consider the problem of locating an object that is hidden in one of N possible locations (or bins). The importance of the field of searching for hidden objects in a specified amount of time is evident in problems involving large databases and in military applications. The intention is to allocate the available resources so as to maximize the probability of locating the object, and also to minimize the cost of search. The locations are identified as $\{C_1, \dots, C_N\}$, with the probability that the object is present at location C_i being $p(i)$.

Unlike the problem addressed by many researchers, we assume that the set of all probabilities, $\{p(i)\}$, referred to as the *Target* distribution, is unknown. Research in this new scenario was recently initiated by Zhu and Oommen, and depends on *assuming* a target distribution, using certain guidelines, which, in turn, could help in allocating the resources and locating the hidden target. In this Thesis we will focus on methods by which we can "guess" this assumed target distribution in real-life scenarios. The methods chosen are efficient, accurate and also valid as per the guidelines laid out in the previous research. To achieve this, we have made use of the field of learning automata, and in particular, of some recent results by Oommen and Raghunath, which have been central for developing the process of guessing the target distribution. The Thesis can thus be seen as a novel contribution that has evolved by merging new results in two completely distinct areas.

Dedicated To My Late Beloved Dad and Mum.

Wish you were with me now to see the final product!

ACKNOWLEDGMENTS

First and foremost, I would like to thank Dr. John Oommen, my Supervisor, for the support he gave me. He was always there to explain and discuss ideas. He has done his best to encourage me, and to guide me in the right directions. Professor Oommen put in extraordinary effort to helping me finish this Thesis, and for that I am deeply thankful to him. I also thank him for the moral support that he was always ready to give, and for being just as good a friend as he was a supportive Supervisor.

Thanks to my late Mother and Father who had to endure many trials as I worked to finish this Thesis. They too were always supportive, and encouraged me to achieve my goals. I only wish they would have been with me now to see the final completed product.

I dedicate this Thesis to them.

I also thank my wife who has consistently encouraged me through many hard times. She has helped me focus on my work. Thanks are also due to my many supportive friends, who assisted me when I needed their help, and to the Examination Committee for their helpful remarks.

TABLE OF CONTENT

TABLE OF CONTENT	VI
LIST OF TABLES	VIII
LIST OF FIGURES	IX
CHAPTER 1 : INTRODUCTION.....	1
1.1 INTRODUCTION.....	1
1.2 MAIN CONTRIBUTIONS	4
1.3 DETAILS OF THE CONTRIBUTION	6
1.4 OVERVIEW OF THE THESIS	8
CHAPTER 2 : THE THEORY OF SEARCH.....	10
2.1 INTRODUCTION.....	10
2.2 KOOPMAN'S TECHNIQUES	11
2.3 ENHANCEMENTS BY COOPER	13
2.4 CHEW'S SEQUENTIAL SEARCH PROCEDURE	14
2.5 DE GUENIN EXTENSION	15
2.6 MINIMIZING COST OF SEARCH	17
2.7 ANOTHER FUNDAMENTAL PROBLEM: WHEREABOUTS SEARCH.....	19
2.8 SEQUENTIAL SEARCH	22
2.9 SEARCH WITH UNKNOWN TARGET DISTRIBUTIONS.....	25
2.9.1 <i>Criteria for Choosing Target Distribution and Error Estimation</i>	27
2.9.2 <i>General cases for Selection Criteria</i>	32
2.9.3 <i>Using Non-Regular Detection Functions</i>	33
2.10 CONCLUSION.....	35
CHAPTER 3 : LEARNING AUTOMATA AND POINT LOCATION	36
3.1 INTRODUCTION.....	36
3.2 LEARNING AUTOMATA.....	37
3.3 FIXED STRUCTURE AUTOMATA.....	41
3.3.1 <i>ϵ-Optimal Schemes</i>	42
3.4 VARIABLE STRUCTURE AUTOMATA	44
3.4.1 <i>Variable structure stochastic automata</i>	45
3.4.2 <i>Linear Schemes</i>	45
3.4.3 <i>Non-Linear scheme</i>	48
3.5 CONTINUOUS POINT LOCATION.....	48
3.5.1 <i>Discrete space learning algorithm</i>	49
3.5.2 <i>Adaptive Tertiary Search Algorithm</i>	55
3.6 CONCLUSION.....	62
CHAPTER 4 : OPTIMAL SEARCH BY EXHAUSTIVE ENUMERATION.....	64

4.1	INTRODUCTION.....	64
4.2	SEARCHING WITH UNKNOWN TARGET DISTRIBUTION.....	66
4.2.1	<i>Optimal search range.....</i>	66
4.2.2	<i>Criteria for choosing target distribution.....</i>	68
4.3	SIMULATION OF THE SEARCH.....	69
4.3.1	<i>Search simulation.....</i>	70
4.3.2	<i>Simulation algorithm.....</i>	75
4.4	EXPERIMENTAL RESULTS BASED ON SIMULATION.....	77
4.4.1	<i>Effect of location size on the search.....</i>	77
4.4.2	<i>Validity of target distribution.....</i>	81
4.4.3	<i>Accuracy of the target distribution.....</i>	82
4.5	CONCLUSION.....	84
CHAPTER 5 OPTIMAL SEARCH USING TERTIARY SEARCH.....		86
5.1	INTRODUCTION.....	86
5.2	LEARNING AUTOMATA AND TERTIARY SEARCH.....	87
5.3	USING THE TERTIARY SEARCH TO LOCATE THE TARGET DISTRIBUTION.....	88
5.3.1	<i>Enhancing The Search by Adaptive Tertiary Search.....</i>	90
5.4	PSEUDO-CODE USING ADAPTIVE TERTIARY SEARCH.....	93
5.5	EXPERIMENTAL RESULTS OF CPLATS IN OPTIMAL SEARCH.....	94
5.5.1	<i>Effect Of the Resolution On The Search.....</i>	95
5.5.2	<i>Validity Of The Target Distribution.....</i>	97
5.5.3	<i>Tertiary Search Versus Exhaustive Enumeration.....</i>	98
5.6	CONCLUSION.....	103
CHAPTER 6 : CONCLUSION		105
6.1	INTRODUCTION.....	105
6.2	FUTURE RESEARCH.....	110

LIST OF TABLES

TABLE 3.1: RESULTS OF SOLVING THE STOCHASTIC POINT LOCATION PROBLEM BY USING THE DISCRETE SPACE METHOD. IN THIS CASE THE VALUE OF λ^* IS 0.9123.....	53
TABLE 3.2: DECISION TABLE TO PRUNE THE SEARCH SPACE BASED ON THE DECISION OF THE INDIVIDUAL AUTOAMTA.....	58
TABLE 4.1: THE RESULTS OF SEARCHING IN THE TWO BINS BY ALLOCATION DIFFERENT AMOUNTS OF TIME TO EACH LOCATION. THE RESULTS ARE PRESENTED FOR BOTH THE CASES WHEN THE INFORMATION IN THE TARGET DISTRIBUTION IS USED AND WHEN IT IS NOT SUED.	80
TABLE 4.2: VARYING THE STEP SIZE AND ITS IMPACT ON THE UPPER AND LOWER BOUNDS OF THE OPTIMAL SEARCH.	82
TABLE 4.3: THE PERCENTAGE OF SUCCESSFUL SEARCHES AS THE STEPS SIZE ' Δ ' IS DECREASED.	83
TABLE 5.1: DECISION TABLE TO PRUNE THE SEARCH SPACE BASED ON THE DECISION OF THE INDIVIDUAL AUTOMATA.....	92
TABLE 5.2: THE RESULTS OF SEARCHING IN THE TWO BINS BY ALLOCATING DIFFERENT AMOUNTS OF TIME TO EACH LOCATION. THE RESULTS ARE PRESENTED FOR BOTH THE CASES WHEN THE INFORMATION IN THE TARGET DISTRIBUTION IS USED, AND WHEN IT IS NOT.	96
TABLE 5.3: A TABLE SHOWING THE TIME REQUIRED FOR EXHAUSTIVE ENUMERATION METHOD AND TERTIARY SEARCH SCHEME ACCURACY WAS SET TO BE 0.1. THESE ARE THE RESULTS OBTAINED FOR 10 INDEPENDENT EXPERIMENTS.	100
TABLE 5.4: A TABLE SHOWING THE TIME REQUIRED FOR EXHAUSTIVE ENUMERATION METHOD AND TERTIARY SEARCH SCHEME ACCURACY WAS SET TO BE 0.01. THESE ARE THE RESULTS OBTAINED FOR 10 INDEPENDENT EXPERIMENTS.	101
TABLE 5.5: A TABLE SHOWING THE TIME REQUIRED FOR EXHAUSTIVE ENUMERATION METHOD AND TERTIARY SEARCH SCHEME ACCURACY WAS SET TO BE 0.001. THESE ARE THE RESULTS OBTAINED FOR 10 INDEPENDENT EXPERIMENTS.	101
TABLE 5.6: A TABLE SHOWING THE TIME REQUIRED FOR EXHAUSTIVE ENUMERATION METHOD AND TERTIARY SEARCH SCHEME ACCURACY WAS SET TO BE 0.0001. THESE ARE THE RESULTS OBTAINED FOR 10 INDEPENDENT EXPERIMENTS.	102

LIST OF FIGURES

FIGURE 3.1: PLOT OF THE ASYMPTOTIC VALUE OF $E[\lambda(\infty)]$ WITH THE RESOLUTION PARAMETER, N FOR VARIOUS VALUES OF P WHEN Λ^* IS 0.9123	54
FIGURE 3.2: PLOT OF THE VARIATION OF THE ESTIMATE $E[\lambda(\infty)]$ OF WITH TIME, N , FOR $N=64$ AND FOR VARIOUS VALUES OF P . IN THIS CASE Λ^* IS 0.9123.	55
FIGURE 3.3: PLOT OF $E[\lambda(N)]$ WITH TIME N , WHEN $P = 0.7$ AND THE L_{RI} PARAMETER Θ VARIES FROM 0.6 TO 0.8. HERE Λ^* IS 0.9123. TIME N IS SHOWN IN EPOCHS OF SIZE 50 UNITS.	61
FIGURE 3.4: PLOT OF THE ASYMPTOTIC VALUE OF $E[\lambda(\infty)]$ WITH P VARYING FROM 0.5 TO 1.0 AND FOR VARIOUS VALUES OF THE L_{RI} REWARD PARAMETER, Θ . THE LATTER INCREASES FROM 0.6 TO 0.8. HERE Λ^* IS 0.9123.	62
FIGURE 4.1 THE PLOT SHOWING THE NUMBER OF SUCCESSFUL SEARCHES AS THE SIZE OF THE LOCATIONS INCREASES.	81
FIGURE 4.2: THE SEARCH ACCURACY AS A FUNCTION OF THE MAGNITUDE OF Δ . THE SCALE OF Δ IS LOGARITHMIC.	84
FIGURE 5.1: GRAPH SHOWING THE RELATIONSHIP BETWEEN THE SIZE OF THE LOCATION AND THE PROBABILITY OF SUCCEEDING IN FINDING THE TARGET.	97
FIGURE 5.2 TIME TAKEN BY EXHAUSTIVE ENUMERATION METHOD AND TERTIARY SEARCH METHOD WITH INCREASING ACCURACY.	102

Chapter 1 : INTRODUCTION

1.1 Introduction

Searching is a basic instinct that is used in everyday life and is performed by all creatures. They search primarily for food, shelter, mates, and maybe for a hiding place in times of danger. For humans, all the types of searches mentioned apply, besides many others. Sometimes the search performed is quite simple, and so the efficiency of the search is not important to the searcher. In other cases, the search is very critical, and consequently, the time taken by the search and the resources consumed are of great concern to the searcher. Crucial to the latter type of search was the field of study called “The Theory of Search” introduced and established during World War II, and it was aimed at obtaining a high probability of quickly detecting the enemy. Later the same techniques were applied to various Search and Rescue operations, and for searching for minerals.

With time, the science of search theory evolved into the field of Optimal Search theory, which, as its name suggests, revolves around methods for finding the most efficient way of locating a hidden or lost object (target).

The authors of [ZO97] identified three basic elements to be present in any solution to the optimal search problem. The first was the probability distribution for the target's location (and motion). This is the information about the target's location at some initial time, and its subsequent motion, and this information is quantified in terms of a probability distribution. The second crucial element was the concept of the detection function. This function relates the amount of effort utilized in searching an area, to the probability of detecting the target given that it is located in that area.

The detection function and the probability distribution of the target's location can be used as follows. For each allocation of the total search effort to the various regions of the search space, the probability of detecting the target with that allocation can be computed. The optimal search problem is then one of finding an allocation of resources, so as to maximize the probability of detection, or to minimize the expected consumption (cost) of resources for detecting the target.

The crucial point is that there is no guarantee that the target would be located even if the target distribution is known. The main reason behind this is that the time allocated for searching is finite, and even if the search was conducted in the correct location, there might not be enough time to find it, and therefore the algorithm would move on to a new location before finding the target.

One important issue to note is that search problems are organized based on different strategies depending on whether the target is stationary or moving, whether the search is

done in discrete or continuous time, and whether the search effort is infinitely divisible or discrete. Also it is important to identify if the target distribution is known to the searcher or not. All of the search problems which were investigated in the literature were studied with the assumption that the target distribution is known *a priori*. This case is not realistic, and therefore, Zhu and Oommen tried to pioneer the study for the scenario in which the target distribution is not known. They achieved the study of a comprehensive theory to find the hidden target with an unknown distribution. Their work proved that when target distribution is unknown, the searcher can learn the true unknown distribution from a set of distributions, and using an algorithm which they provided, also determine whether this distribution was close to the real value or not. They also provided expressions to be used in calculating the error between the assumed and real values. Thus they proved that using an assumed target distribution, the search strategy could result in a possibly optimal solution. This research was very important to the field as it opened the avenues for real life scenarios to be searched optimally. Prior to the work of Zhu and Oommen, the problem of searching for the target would have been limited only to the situations when the target distribution was known, and such situations are rare.

The work of Zhu and Oommen had a limitation; their work did not specify a method by which this assumed distribution could be sampled and learnt in real life scenarios. These problems needed more research. It was essential to develop a method that would help a searcher to *guess* the target distribution itself efficiently in a real life case, and use the results of the Zhu-Oommen algorithm to locate the target. This is the primary work done in this research endeavor.

1.2 Main Contributions

Our research in this thesis is intended to complement the research done by Zhu and Oommen. Admittedly, they have skillfully proven that they can use various assumed values for the target distribution to lead to an optimal search strategy. These are important results because typical real life scenarios obviously have target distributions that are unknown. Our results add to their work by providing the necessary means for guessing the unknown target distribution. We achieve this by simultaneously maintaining the efficiency and accuracy of the processes.

Our research first of all, is concerned with stationary objects searched for by a single searcher in a continuous space, when the target distribution is not known. The research suggests a method by which the searcher would iterate in the interval of possible probabilities. In each iteration a probability is chosen and tested by executing the corresponding search, so as to resolve whether it will lead to an optimal search strategy or not. After going through all possible probabilities, the value that produces the most optimal results leads to the target distribution.

In our research, this procedure is designed, implemented and tested in a computer simulation explicitly created for the task. The software spawns an imaginary target that is “hidden” in one of two possible simulated locations. The simulation then involves our novel procedures, where it is able to *guess* the most accurate target distribution. This target distribution is subsequently utilized to find the hidden target. Our procedure is thus subjected to specific tests to demonstrate its correctness and efficiency. These tests

revolve around demonstrating that the resulting “learned” target distribution is a valid one and close to the optimal value, and also in proving its accuracy. In this context, it is worth mentioning that the characteristics of the environment, in which the search is conducted, are already related to the accuracy of the assumed target distribution, and this relationship is crucial to the learning process.

Since efficiency is our goal, we shall attempt to devise a method that would result in leading to the best value of the estimated target distribution quickly. First of all, we introduce a straightforward method that is a first logical choice to search the space of potential target distributions. But, since efficiency is our main goal, we then attempt to devise an even superior method. In our endeavor to devise an efficient algorithm for finding the unknown target distribution, we use techniques found in the field of learning automata, and in particular, we focus on the innovative and novel algorithm presented in [OR98]. In the latter research, conducted by Oommen and Raghunath, they considered the fundamental problem of determining an unknown value in a bounded domain when the “Oracle” is stochastic. Their results were quite impressive. Their research produced a novel scheme to stochastically locate a point by recursively trisecting the domain, by utilizing the interaction with the environment. Although the research has importance in its own right, we discovered that it could also prove useful in yielding an improved solution to our problem of optimal search when the target distribution is unknown. The new method which combined the results of [ZO97] and [OR98] was expected to be faster and more accurate than the scheme of iterating through the possible values in the search domain. Here too, our tests established the correctness of the estimated target distribution

in an enhanced manner. Also, the effect of computing the different time allocations for the search was considered, and their effect on the accuracy of the search was also investigated.

We believe that the results presented here will be of importance in problems where a search should be conducted with a constraint on the time allocated, as in the case of searching the web. Using our results, a searcher would be able to allocate the time for the search in the best and most efficient way, even though he might not know the probabilities of finding the target in each of the possible locations.

1.3 Details of the Contribution

The research presented in this thesis integrates results from two novel approaches. Zhu and Oommen were the authors of the first research area which presented a pioneering approach that enables a searcher to locate a hidden object in one of N locations, when the target distribution is unknown. They implemented an algorithm that would use a guessed target distribution to find accurately the hidden target. They also designed an algorithm to measure the error estimate between the correct target distribution and the guessed one. This research is of importance to the field of optimal search, yet it was not possible to use it in real-life scenarios, as it did not provide a procedure by which the searcher could guess the target distribution. Our work was to identify such a procedure.

We first started by creating a computer simulation that would simulate the algorithm presented in [ZO97] so that we would have an environment in which we could monitor the performance of the Zhu-Oommen algorithm, while at the same time, we would obtain

an opportunity to enhance it and to compare the differences obtained by incorporating our enhancements.

After confirming that the simulation has created the correct environment, we proposed to complement the algorithm by adding a procedure that would enable us to assume the target distribution needed to find the hidden target.

Our first trial is an exhaustive iteration method, where the algorithm would sequentially examine each value in the space of all possible values for a target distribution. With each value the algorithm tests the optimality of the current target distribution using the algorithms outlined in [ZO97]. Ultimately the optimal value will be chosen. The values obtained from such a technique will be subjected to other tests to check the efficiency and accuracy of the target distribution obtained.

To get a faster and more efficient result we then propose to apply a second technique to find the optimal target distribution. This would be done using an intelligent adaptive tertiary search developed by Raghunath and Oommen. The algorithm divides the search space into three subintervals, and tests each interval's mid-point for optimality. If the optimal target distribution is not one of those mid points, an inaccurate Oracle is "interrogated" for the next move, i.e., whether it is a point to the right or left of the current point. Using the responses from the Oracle the search space can be pruned, and the three subintervals could be reduced to only two or even to one subinterval. This process is done recursively until the value sought, is found. This method is tested to

prove that it renders results as accurate as the first method, yet it does so in a more efficient way, thus consuming less search time.

1.4 Overview of the thesis

The following is the outline of the thesis. In the second Chapter we give an overview of the field of search theory. We will discuss the work of Koopman, who pioneered the work in this field by establishing a model for the theory of search. His work was the basis on which many others built their research. Also, we shall review the research presented by Cooper in which he solved the same problem that Koopman addressed, but in a discrete space. Other studies that were built on Koopman's research will also be reviewed, which includes the work done by Chew, Gilbert, De Guenin, Onaga and Staroverov. We also give a short overview of the area of the whereabouts search problem.

As a part of our introduction to the field of the theory of search, and as a basis to introduce our own contribution, we will present the work done by Zhu and Oommen. In his thesis, Zhu studied the theory of search for stationary targets, when the target distribution is unknown. His findings included methods by which we can obtain the upper and lower bounds for optimal strategies, and he also derived selection criteria for choosing target distributions when two or more cells were searched. In each case, Zhu also derived the error estimate.

Once we have familiarized the reader with the field of search theory, we will present an overview of the field of Learning Automata, and give some of its applications. The

concepts of fixed and variable structure automata will be presented briefly, with special focus on the ϵ -optimal L_{RI} scheme. A background about Learning Automata is essential for understanding the work done by Oommen and Raghunath in which they solved the stochastic search problem, which involved searching for a point on a line, where the Oracle responded with inaccurate responses. We will present all the necessary details required for understanding the methods presented in [OR98].

After laying the foundational background for the topics to be studied, we start presenting our work which complements the work done by Zhu and Oommen. We will present a formal algorithm and a computer simulation that searches for a hidden target using an assumed target distribution. We then add to the simulation a method by which the target distribution can be guessed accurately and efficiently. Such a procedure was not outlined in the Zhu-Oommen algorithm. The procedure involves the most logical way of searching, which is to iterate through all possible values for the target distribution. To demonstrate the efficiency of such a method we introduce some tests to prove our claim. We then proceed to present another procedure to guess the target distribution that incorporates the results of [OR98], and is thus more efficient. This procedure will also be rigorously tested. This is done in Chapter 5.

Chapter 6 concludes the thesis.

Chapter 2 : THE THEORY OF SEARCH

2.1 Introduction

Originally, one of the main applications motivating the theory of search was in military search operations. Finding soldiers, ships, planes, and submarines, as fast and efficiently as possible, is considered as a measure of superiority between battling forces. Not only during wartime does searching with less time and less resources prove to be of high importance, but it is also of great importance during peace. Locating missing ships, or planes are direct applications to the search theory. In his book, *Search and Screening* [K46], Koopman gave numerous applications for the operation of search, other than military applications. He mentioned the search for mineral deposits such as oil, metals, or gems. A second example introduced was the police search for criminals or missing people. An interesting example also, is the search for a pattern among electronic recordings, and has applications in code breaking and information retrieval.

From the different examples mentioned above, we see the diversity of the objects that can be searched. In terms of the nomenclature used in the literature, the object that is searched for will be called the *target*. Collecting as much information about the target is essential to help locate it. Information could include the physical appearance of the target,

including its looks, its shape, and the sounds it emits. Another piece of data that should also be collected about the target is whether it is stationary or moving, and if moving, the behavior of the motion should be known or appropriately modeled. Still with all this information, the search result is not certain; there will always be a probability of failure. Assumptions are made before the search begins as to what the probabilities of finding the target in specific locations is. The different probabilities for each location are referred to as the *target distribution*.

2.2 Koopman's Techniques

The early work of Koopman was mainly directed towards military applications. He submitted three papers on the topic of search. The first article was describing the kinematic bases of search, where he described the formulas governing the relation between the velocity and direction of the target (being in his case a submarine) and the observer. He introduced the concept of the random distribution of the targets which is a description of the following two situations:

1. The presence of a large number of different targets, and what is known of them, is not the value of the search criterion, but rather, contribute to the percentages or the proportions between each of their values.
2. The other situation is when there exists only one target and the values by which it is searched for are not precisely known, but the probabilities that these values assume various possible instantiations are known [K56a] or can be estimated.

One of the simplest and most important cases of random distributions is *uniform distribution* for the location of the targets.

Koopman presented his next paper in his “*Theory of Search*” series, which discussed the target detection. According to him, there are two methods of detection. In the first model, the observer would be making successive quick glimpses, and in the second, the observer is looking continuously.

The last paper in the Koopman series, and the one we care most about, is the one explaining the methods on how to achieve the optimum distribution for the search effort. This means that if it is known that a target resides in one of various possible places and that the amount of search effort, or time, is limited, then we should opt to find the best way to distribute this effort. An optimal way to distribute the effort is to do it in such a way that the chances of finding the target are maximized. The quantitative ideas and methods of this paper need not be confined to the case of search. Indeed, they are valid for numerous fields of application, many of which were listed in the paper.

The paper started by analyzing specific cases, which then are generalized by assuming that the search was done in multiple locations which were not homogenous. In one setting, Koopman assumed that the probability of finding the target depended greatly on *where* it is found, and attempted to maximize this value.

In [A64b], Arkin generalized this paper of Koopman's [K57] by trying to develop strategies for search problems. His work stated that a search strategy has certain characteristics, namely that of being reducible, optimal and uniform. He then showed the existence of such a strategy using a few fairly general assumptions. Since the search can be directed simultaneously in any number of locations, Arkin, elsewhere [A64a] studied techniques to determine the best location to start with. He defined a searching strategy as being the rule which, at any time t , determines in what locations a search is to be carried out.

Koopman's papers on the theory of search were mainly devoted to searching for moving targets. Yet he also devised models for searching for a stationary target. He created a model for searching for a stationary target where a target could be in one of many different locations. This was again one of the earliest efforts to study the optimal distribution for the searching effort for a stationary target. The model was based on assuming an exponential detection function, and on trying to maximize the probability of finding the object, where this probability is defined as the summation of the probabilities of finding the target in any given location.

2.3 Enhancements By Cooper

The basis for the theory of optional search was proposed in the articles of Koopman. He provided a logical basis for how to allocate the effort of search. From these seminal papers, the work was extended in many directions. Cooper and Charnes [CC58] made an

important contribution to this problem as they derived an algorithm to algorithmically solve Koopman's problem on a computer for the discrete case. They were concerned with formulating the optimum allocation of search effort, as a problem in convex programming. The intention was that their optimal solutions would be obtained by the adjacent extreme point methods of linear programming. They directed their attention to the relations between different search theories and the tools used. This was done primarily to show how search theory and linear programming could be combined to attack broader classes of problems in Operation Research. Using convex programming, they formulated a methodology to solve the problem of search in both the discrete and the continuous cases, and of finding the necessary and sufficient conditions for the selection criteria.

2.4 Chew's sequential search procedure

Chew's proposal for a sequential search [C67], was more directed towards the searching problems themselves. Such problems are basically the optimizations that are concerned with the technique, which could provide the best result, regardless of what this result is. Chew considered the case when there was an object hidden in one of many, say n , locations. The probability of finding the object in one of those locations, say the i th location, was assured to be p_i which satisfied

$$\sum_{i=1}^n p_i = 1.$$

When the n locations are searched, the result of the search in each of these locations is independent of the other search operations. A search procedure is the order by which the different locations are to be visited in the search. The goal of the search procedure was to attempt to get an optimal search, which is one that yields a minimal cost of searching. Such a procedure would be able to instruct the searcher to choose the location that yielded the highest ratio of the detection probability.

It is important to note that by determining the optimal one-stage procedure, one could compute the detection probability for n inspections. The use of an optimal procedure thus ensured the greatest chance of finding the object with an optimally allocated number of inspections. Another advantage of using an optimal procedure was that it minimized the expected cost of the search which was the amount of resources needed to achieve the search.

2.5 De Guenin Extension

Earlier it was seen that the basis for a theory of search was initiated in three articles published in the journal *Operations Research* by Professor B.O. Koopman. Building on this work, De Guenin devoted his research to the fundamental problem of search theory, namely, to allocate a given amount of search effort in such a way so as to maximize the over-all probability of discovering an object located in a given space. The problem had already been solved under the assumptions that the probability of detecting the object is a

negative exponential function of the search effort density. While this assumption was fairly realistic in many military applications, it had definite drawbacks in others.

De Guenin worked with the model that an object is known to be located in a given uni-dimensional space. The position of the object can be defined by a single parameter which is a random variable with an associated probability density function. The probability of detecting the target with a given effort when the target is indeed in the space being searched through is called the detection probability or detection function. This detection probability is the probability that one may look directly at an object without actually seeing it. If an observer continues to look towards an object, he will almost certainly locate it. What De Guenin did is that he assumed the detection function $b(\varphi)$ is regular, that is $b'(\cdot)$ is a decreasing function with $b'(0) > 0$ and $b'(\infty) = 0$. His main result was that a necessary condition for $\phi(x)$ to be optimum is that, at any point x , such that $\phi(x) > 0$,

$$g(x) p_{\phi}[\phi(x)] = C,$$

where C is a constant, and $g(x)$ is the probability distribution of the location of the object, and $p[\phi(x)]$ is the probability of detecting the target with an effort $\phi(x)$ when the target is *indeed* at x .

By applying successive approximations to De Guenin's results, we can get a procedure of determining the optimal allocation.

2.6 Minimizing Cost of Search

For a search to be optimal, the searcher must try to maximize the probability of a successful detection with the total cost not exceeding a certain level. To achieve that one may wish to minimize the expected time spent in locating the target. Another solution purposed by Staroverov [S63] assumed a scenario which had n cells, one of them containing the object searched for. The probability that the object is in the cell ' k ' is $p_k > 0$, where $1 \leq k \leq n$, and the p_k 's satisfy:

$$\sum_{k=1}^n p_k = 1.$$

In Staroverov's model, only one cell could be searched in a unit of time. The search could lead to locating the object or it might not. If the object is located in the cell being searched, it can be found with probability q and not found, with a probability of $1 - q$, where $q > 0$. Searching in different cells was assumed to be independent of one another, meaning that if searching in cell number k did not yield a positive search, this gave no information about the presence of the object in any other cell. Therefore, Staroverov [S63] hypothesized that having different sequences of searching might yield different times to spot the object, where a search sequence is the permutation by which the searcher could order the cells to be searched. In other words, in this way the algorithm would tell the searcher which cell to start with, and then which other cells have to be searched, and so on. The goal of the study was to find the optimal sequence, denoted by α^* .

More studies were done on the topic of minimizing the search cost. Gilbert [G59] studied the optimal distribution of the effort for searching which minimized the expected time for detection. According to his results, he showed that in some search problems there is a choice of several different places in which to search. He gave an example where a man searches for a nut to fit an odd-sized bolt. He imagined the sequence by which the man would conduct his search. His assumption was that the man had two bins full of assorted sized nuts mixed together, and that he could search for only one location at a time. However, if he was unsuccessful in obtaining the nut after searching the first location, he could eventually decide that this location did not contain many nuts of the correct size, and thus opt to switch his search to the second location. If he still failed to find the nut in the second location he had the choice to eventually switch back to the first one, and vice versa. To perform this kind of search, the searcher would have to develop a set of rules by which he would decide which location to start with, and when to switch from a location to the next. Gilbert observed that there was always one set of rules that minimized the expected searching time. Gilbert was among the first researchers who considered the case of switching during a search.

Other researches tackled the problem of N-boxes. Among them are Gluss who arranged the boxes linearly and Onaga [O71] who was concerned with the time loss during the search of the N-Box model. Others considered a discrete search model with switching loss, and obtained conditions for the optimal search which minimized the expected detection time.

2.7 Another Fundamental Problem: Whereabouts Search

Within the theory of search there are two fundamental problems: Detection Search and Whereabouts Search. Detection Search is basically the question of how to choose a search strategy that maximizes the probability of finding the object when there is a limit on the available resources. On the other hand, Whereabouts Search involves the case when the objective is to maximize the probability of correctly guessing which box contains the target after performing a search costing no more than a specific amount, and which, in turn, does not exceed the ceiling of the cost. This can be accomplished by either finding the object during the search, or by guessing which box contains the object at the end of an unsuccessful search. Thus a strategy for a whereabouts search consists of a search strategy and a guessing strategy if the object is not found by the searching strategy.

We point out, at this juncture, that an optimal detection search strategy and an optimal whereabouts search strategy are different. To show this fact, suppose that an object is equally likely to be in one of two boxes. Consider the case where only two looks are possible. If we look twice in one box the probability of detecting the object is three quarters if the object is really in the box. If we were wrong in our choice and the object was not in the box then the probability is zero. From this we conclude that the probability of detection is

$$\frac{1}{2} \times \frac{3}{4} + \frac{1}{2} \times 0 = \frac{3}{8}.$$

In the other case, where we take a single look in each box, the probability of detection is

$$\frac{1}{2} \times \frac{1}{2} + \frac{1}{2} \times \frac{1}{2} = \frac{1}{2}.$$

As for the whereabouts search problem, we have to evaluate the amount of information obtained from the two possible search strategies. Suppose we looked twice in box A . If the object is not found, then the *a posteriori* probability that the object is still in the box A is 0.2 and the *a posteriori* probability that the object is in box B is 0.8. If the object is found, these probabilities are 1 and 0 respectively. Using information theory, we know that the amount of information contained in these two situations is given by the functions $I(0.2, 0.8)$ and $I(1, 0)$, where

$$I(p_1, p_2, \dots, p_n) = - \sum_{i=1}^n p_i \log(p_i)$$

Therefore the expected gain in information can be seen to be:

$$\Delta(2, 0) = I(1/2, 1/2) - [3/8I(1, 0) + 5/8I(1/5, 1/5)] = 0.380\dots$$

For the other case, which involves executing a single look per box, the similar calculation would yield the expected gain in information to be 0.347.

From the above, we see that the strategy of using two looks in one box, and which yields a superior information gain, results in the lower detection probability. This proves the hypothesis that the optimal strategies for detection search and whereabouts search are different.

These results were further generalized without compromising the main hypothesis that distinguishes between the optimal detection search and the optimal whereabouts search strategies. One avenue of generalization, primarily involved having N looks within N boxes. In other models, when the problem was generalized to the case of multiple boxes,

the original hypothesis was refuted [K71]. It was shown that the optimal whereabouts search strategy is an optimal detection search strategy.

To visualize the whereabouts search problem, Zhu [Z96], used an example given by Bellman [B57] who assumed that he had the information that a ball is in one of N boxes, with the *a priori* probability p_k that it is in the k -th box. Let t_k be the time consumed in searching the k -th box, and let q_k be the probability that any particular searching of the k -th box will yield no information concerning its contents. When the latter occurs, we continue the search operation with the information already available. Bellman [B57] showed that if we wish to obtain the ball, the optimal (i.e., minimum expected time) policy consisted of searching the box for which $p_k(1-q_k)/t_k$ is largest. On the other hand, if we merely intended to locate the box containing the ball in the minimum expected time, he showed that the box for which the quantity $p_k(1-q_k)/t_k$ is a maximum must be searched first, or not be searched at all.

The only difference between the whereabouts search problem and the previous example is that in the latter, the searcher knows afterwards whether a look has been informative or not. An informative look fails to find the ball only if the ball is not in the box searched. In the detection and whereabouts search problem, however, a look can fail to find the target either because it is not in the box or because it was overlooked, and the searcher would not know which scenario has occurred.

2.8 Sequential Search

In an earlier section we explained Chew's work concerning various sequential search procedures. Here we introduce the problem of sequential search again by briefly explaining the work done by Kadane and Simon [KS83], who studied problem by obtaining results similar to those of the discrete search problems in the case where only finitely many searches have a positive probability of success.

In their paper [KS83], they considered two important sequential problems which have been extensively studied by many other authors. The first sequential problem considered a single object hidden in one of the N boxes. The object is in the k -th box with probability p_k , ($\sum p_k = 1$). A search strategy is a permutation of a subset of the integers $\{1, 2, 3, \dots, n\}$ which tells the searcher where to perform the next search if the object has not yet been found in the previous searches. The cost of searching box k is c_k if the search is unsuccessful, and x_k if it is successful. The first sequential search problem was to determine a search strategy that included each of the boxes, and yet minimized the expected cost of a detection search.

The second sequential problem studied by Kadane and Simon can also be stated as a search problem. In this problem they assumed that there are m objects ($0 \leq m \leq N$) hidden in N boxes. The event E_k , that an object is hidden in the k -th of N boxes, again has probability p_k . E_k was assumed to be independent of E_j for $k \neq j$, as opposed to the case of the first sequential search problem in which the set $\{E_k, k = 1, 2, \dots, n\}$ is disjoint. Again a

search of box k would cost c_k if it was unsuccessful, and x_k if it was successful. The search continued until the first object was found or until all the boxes were searched.

Wegener [W81] later studied the same topic. His goal was to minimize the expected search effort of a successful search assuming that he had a target hidden in one of n boxes. He assumed that the probability of having it in the k -th box was $p(k)$, and that the probability $q(t, k)$ of detecting the object if it was hidden in box k was known. Previously, this problem had been solved only under the assumption that the functions $q(., k)$ are concave. Wegener proved, without concavity assumptions, the existence of an optimal distribution of search effort and gave a procedure for its construction. All that he assumed was that $q(., k)$ was a probability distribution function which meant that it was increasing and right continuous.

It is important to distinguish between the search strategies in which a single person is doing the search, versus the case when the searching is being done by multiple entities. In the latter case, we can specify that some of the parties doing the search can search a specific cell, while others search another cell, and so on. Here we assume that the number of searchers is so large that it is a reasonable approximation to assume that we can divide the search effort in an arbitrary fashion.

A strategy is a function s that belongs to the set of real numbers where $s(t, k)$ specifies how much of the search effort t has to be spent in box k . The function $s(t, k)$ has to satisfy:

$$\sum_{k \in B} s(t, k) = t, \text{ with } s(t, k) \text{ being increasing.}$$

By $Q(s, t)$ we denote the probability that we find the object when we spend search effort t . From the previous arguments, it is obvious that the probability that we never find the object if we use the strategy s is obtained as:

$$Q(s, t) = \sum_{k \in B} p(k) q(s(t, k), k)$$

and

$$Q(s, \infty) = 1 - \lim_{t \rightarrow \infty} Q(s, t)$$

The function $Q(s, \cdot)$ is the probability distribution function of the random variable $Y(s)$ which measures the search effort which is necessary to find the object by using strategy s . Also, $q(t, k)$ is a conditional probability that the searcher finds the object in the place where he expects the object to be after spending time t in the search. The function $q(\cdot, k)$ is an increasing and right-continuous function.

As mentioned earlier, Gilbert [G59] determined the optimal strategies for some definite concave functions, $q(\cdot, k)$. Also Onaga [O71], gave a solution of the problem under the assumption that all the functions $q(\cdot, k)$ were concave.

Wegener [W81] tied these results together as follows. He started by stating the necessary and sufficient conditions for the existence of a strategy with finite expected search effort. Subsequently, he derived the solution of the problem under the assumption that the functions $q(\cdot, k)$ were concave. Finally, he used these results for the solution of

the general problem. By achieving this, he solved a general search problem where the search space was discrete, but where time was continuous, i.e., the object had the potential of being found at any point in time.

A comparison between Wegener's results with the results for some other analogous search problems can be found in Arkin [A64b]. Note that while many authors investigated the discrete search problem in which one may find the object at the end of some definite searches, Wegener was among those who attempted to solve the problem in a continuous setting. The results presented by Wegener were very useful because in the discrete search problem it was not certain that an optimal strategy always existed. As opposed to this, Wegener was able to prove the existence of an optimal strategy in the continuous-time world.

2.9 Search With Unknown Target Distributions

As shown in the previous sections, most of the work done in the field of search theory was done assuming the knowledge of the target distribution, which is essentially the probabilities for the target location. With the target distribution unknown it is more difficult for the searcher to perform his search, as he would have to guess the probability of the target locations and based on this guess he would adopt the most suitable search strategy. The problem of guessing the probability introduces the question of how a meaningful estimate can be made with the information available. Another relevant problem is that of determining the error in the above guess from the real value of the

target distribution. Zhu [Z96] studied this problem in detail. He addressed the problem of searching with unknown target distributions. His results were mainly obtained for discrete systems rather than continuous ones.

Using stochastic systems and optimal control, Zhu tackled the problem by attempting to determine a target distribution that would be close enough to the true one. This closeness implies a smaller error factor between the two distributions. Of course, the error is not easily calculated, as the *true* distribution is not known in advance.

Zhu started his work by assuming a special case when trying to estimate the error between the estimated and *true* target distribution. He assumed an exponential detection function:

$$b(i, z) = 1 - e^{-z}, (i = 1, 2, \dots, N)$$

where b is the detection function, i is the index of the location, and z is the time taken.

For such a detection function he showed that the detection probability $P[f^*]$ of the optimal search strategy f^* would satisfy the following:

$$1 - e^{-K/N} \leq P[f^*] \leq 1 - C_N e^{-K/N},$$

where $C_N = N[p(1) \dots p(N)]^{1/N}$, and $K > 0$ is the total cost constraint which the time allocated for the search.

The above assumption of an exponential detection function was then generalized for the case where the detection function is regular implying that $b(i, z)$ would be continuously

differentiable and that the derivative $b'(i,.)$ is a decreasing function with $b'(i,0) > 0$ and $b'(i,\infty) = 0$.

With this generalization, Zhu went on to derive the upper and lower bounds of optimal strategies. His result involved the usual assumptions of having an object hidden in one of N cells. Those cells are marked $C_1, C_2, \dots, C_N$, with an arbitrary unknown probability distribution, $p(i), (i = 1, 2, \dots, N)$. With the regular detection function, the detection probability $P[f^*]$ of the optimal search strategy f^* was shown to satisfy:

$$b(i,0) + \lambda K \leq P[f^*] \leq b(i,0) + b'(i,0)K$$

By assuming that $b(i,0) = 0$, the previous results were extended as:

$$\lambda K \leq P[f^*] \leq b'(i,0)K$$

where $K > 0$ is the total time constraint, and λ is the parameter of the Lagrangian whose explicit form is given in [ZO97].

2.9.1 Criteria for Choosing Target Distribution and Error Estimation

When assuming a probability distribution for the search problem with an unknown target distribution, it has to be known that there will be a margin of error from the actual target distribution. The main goal in this scenario is to minimize this error. Therefore, Zhu derived a strategy to calculate or estimate the error for the target distribution. After estimating the error Zhu attempted to minimize it. In [Z96], Zhu first, started with the simple case where there are two cells, one of which has the object. He also assumed an

exponential detection function. Later, he extended his work to include the general case of having N cells.

We first consider the two-cell scenario. In this case the probability of the target being in the first cell is assumed to be a , and to be $1-a$ for the second cell. Since this distribution is not known, we expect that the searcher would assume another target distribution, say $(b, 1-b)$. It is logical to expect a higher probability of detection when using the optimal search plan in conjunction with the *true* target distribution, as opposed to using it in conjunction with the *assumed* target distribution. This statement can be expressed by the following equation:

$$P[f_b^*] \leq P[f_a^*],$$

where $b \neq a$.

The proof for this assertion is crucial to our further study and is thus included here. It can be summarized as below. Assuming that the target distribution is $(b, 1-b)$, the Lagrangian would be

$$\ell(i, \lambda, z) = p(i)b(i, z) - \lambda z = p(i)(1 - e^{-z}) - \lambda z.$$

By differentiating the equation we obtain

$$\frac{d\ell(i, \lambda, z)}{dz} = p(i)e^{-z} - \lambda.$$

If we set the right hand side of the equation to zero, we obtain the optimal search plan for the assumed target distribution. This yields:

$$z_{1,b} = \ln \frac{p(1)}{\lambda_b} = \ln \frac{b}{\lambda_b}, \quad z_{2,b} = \ln \frac{p(2)}{\lambda_b} = \ln \frac{1-b}{\lambda_b}$$

By performing the summation of all the probabilities we get

$$\sum_1^2 z_{i,b} = \sum_1^2 \ln \frac{p(i)}{\lambda_b} = K$$

yielding

$$\lambda_b = [b(1-b)]^{1/2} e^{-K/2}$$

By now utilizing the real probability distribution, which is (a, 1-a), the detection probability for the search plan would be

$$\begin{aligned} P[f_b^*] &= \sum_{i=1}^N p(i)(1 - e^{-z_{i,b}}) \\ &= a(1 - e^{-z_{1,b}}) + (1-a)(1 - e^{-z_{2,b}}) \\ &= a(1 - \frac{\lambda_b}{b}) + (1-a)(1 - \frac{\lambda_b}{1-b}) \\ &= 1 - \lambda_b [\frac{a}{b} + \frac{1-a}{1-b}] \\ &= 1 - \frac{a+b-2ab}{\sqrt{b(1-b)}} e^{-K/2} \end{aligned}$$

On the other hand, to get the detection probability for the search plan based on the real probability, we would get

$$\begin{aligned} P[f_a^*] &= a(1 - e^{-z_{1,a}}) + (1-a)(1 - e^{-z_{2,a}}) \\ &= a(1 - \frac{\lambda_a}{a}) + (1-a)(1 - \frac{\lambda_a}{1-a}) \\ &= 1 - 2\lambda_a \\ &= 1 - 2\sqrt{a(1-a)} e^{-K/2} . \end{aligned}$$

Thus the error is calculated as follows:

$$P[f_a^*] - P[f_b^*] = 1 - 2\sqrt{a(1-a)} e^{-K/2} - [1 - \frac{a+b-2ab}{\sqrt{b(1-b)}} e^{-K/2}]$$

$$\begin{aligned}
&= \left[\frac{a+b-2ab}{\sqrt{b(1-b)}} - 2\sqrt{a(1-a)} \right] e^{-K/2} \\
&= \frac{a+b-2ab-2\sqrt{a(1-a)b(1-b)}}{\sqrt{b(1-b)}} e^{-K/2} \\
&= \frac{a+b-2\sqrt{ab}[\sqrt{ab}+\sqrt{(1-a)(1-b)}]}{\sqrt{b(1-b)}} e^{-K/2}
\end{aligned}$$

Considering the numerator in the above equation we find

$$\begin{aligned}
\sqrt{ab} + \sqrt{(1-a)(1-b)} &= \frac{a+b-(\sqrt{a}-\sqrt{b})^2}{2} \\
&+ \frac{(1-a)+(1-b)-(\sqrt{1-a}-\sqrt{1-b})^2}{2} \\
&= \frac{1}{2} [2 - (\sqrt{a}-\sqrt{b})^2 - (\sqrt{1-a}-\sqrt{1-b})^2] \\
&= 1 - \frac{1}{2} [(\sqrt{a}-\sqrt{b})^2 + (\sqrt{1-a}-\sqrt{1-b})^2] \\
&\leq 1.
\end{aligned}$$

Hence

$$\begin{aligned}
P[f_a^*] - P[f_b^*] &\geq \frac{a+b-2\sqrt{ab}}{\sqrt{b(1-b)}} e^{-K/2} \\
&= \frac{(\sqrt{a}-\sqrt{b})^2}{\sqrt{b(1-b)}} e^{-K/2} \\
&\geq 0.
\end{aligned}$$

This shows that if there are two cells and that the exponential detection function is valid, if the real target distribution is $(a, 1-a)$ and the approximated distribution is $(b, 1-b)$, then the error in the detection probability of the optimal search plan is given by

$$E(a, b) = \frac{a+b-2\sqrt{ab}[\sqrt{ab}+\sqrt{(1-a)(1-b)}]}{\sqrt{b(1-b)}} e^{-K/2}.$$

Using the previous two results, Zhu [Z96], introduced two criteria for choosing target distribution. The first criterion was to maximize the following expression

$$P[f_b^*] = 1 - \frac{a + b - 2ab}{\sqrt{b(1-b)}} e^{-K/2}$$

over a priority set, which is the set of those values of a that are the most likely ones. This priority set, A , can be defined mathematically as the set of values which a can take with probability

$$P[a = a_0 \in A] \geq p_0,$$

where p_0 is a fixed number. Usually $0.5 \leq p_0 \leq 1$.

The second criterion for choosing the target distributions was to minimize $E(a,b)$ over a priority set of a , where $E(a,b)$ given above.

It is easy to comprehend that the smaller the priority set is, the more accurate the assignment of the target distribution can be. To explain this mathematically, we could say that the priority set is smaller when $E(a,b)$ is smaller. For this reason, in real life cases, the searchers should always use as much available information as possible so as to reduce the size of priority set, and to thus limit the range of possible target distributions.

The priority set is an interval with $\delta > 0$, where the error $E(a,b)$ tends to zero when δ tends to zero.

2.9.2 General cases for Selection Criteria

The previous section dealt with the special case of having an object in one of two cells, in which an exponential detection function was applicable. This section generalizes the latter results to a case of n cells.

Suppose that in a search problem the target distribution is $(a_1, a_2, \dots, a_N)$, but that this information is unknown to the searcher, and that the searcher chooses $(b_1, b_2, \dots, b_N)$ as the approximation of the target distribution. Again, intuitively we believe that the optimal search plan f_a^* , which is based on the *true* target distribution, should have a superior detection probability than that of the search plan f_b^* , which is based on the assumed target distribution. We prove below that this is true for the case when the detection function is exponential.

In [Z96], Zhu proved that the detection probability of the optimal search plan for the *true* target distribution is greater than the detection probability of the optimal search plan for the *assumed* target distribution. In other words, the expression

$$P[f_b^*] \leq P[f_a^*]$$

is still valid with the new assumptions.

There are two criteria for choosing a target distribution based on the new assumption of the general case. When searching in N cells with the exponential detection function, the first criterion is to maximize the following expression

$$P[f_b^*] = 1 - e^{-K/N} \sum_{i=1}^N \left(\prod_{j=1}^N b_j \right)^{\frac{1}{N}} \frac{a_i}{b_i}$$

over a priority set, which is the set of most likely values for a .

The second criterion for choosing the target distribution is to choose the distribution in a manner so as to minimize $E(a,b)$ the estimated error between the true distribution and the assumed distribution, where $E(a,b)$ is given by

$$E(a,b) = e^{-K/N} \sum_{i=1}^N \frac{\left(\prod_{i=1}^N b_i \right)^{\frac{1}{N}} a_i - \left(\prod_{i=1}^N a_i \right)^{\frac{1}{N}} b_i}{b_i}.$$

2.9.3 Using Non-Regular Detection Functions

The term “regular” detection functions was first used in the book *Theory of Optimal Search* [S75]. These functions were used in the generalized form of Koopman’s results for optimal search. Regular detection functions were characterized by being more generic than the exponential detection function.

A function, $b(i,z)$ is considered regular if $b(i,.)$ is continuously derivable for each i , and the derivative $b'(i,.)$ is a monotone decreasing function with $b'(i,0) > 0$ and $b'(i,\infty) = 0$. These characteristics can also be used to show that a regular function has an invertible derivative. This definition, although more generic than the exponential function, it relaxes the limitations on the applications of search theory.

Generally speaking, a regular detection function implies that if we have N cells, the longer a specific cell is searched, there is a greater chance that the search would lead to the target being detected. Though this is generally true, there are some cases when it is not. Zhu[Z96] gives one such example. Suppose we are searching visually for an airplane in the sky. After a few minutes if we still have not found the target, we are sure that the target does not exist or is not visible, and so any further searching in this direction only yields a detection probability 0. Clearly, in this case, the detection function $b(i, z)$ is not regular, since at first $b(i, z)$ increases and after a point it decreases fairly abruptly to 0. To be more accurate a theory of search would also have to take into account the “bell shaped” functions, such as the normal density function. This would be of great use, because many real life experiences can be approximated by a normal distribution.

To derive the optimal search strategy for a class of Non-Regular detection functions, we consider the discrete search of N cells. Suppose that at any moment only one cell can be reached, and that the probability of detecting the target in cell C_i is given by the following functions

$$b(i, z) = e^{-\alpha_i(z-z_i)^2}, \quad z > 0, \alpha_i > 0, (i = 1, 2, \dots, N)$$

$$b(i, z) = 0, \quad z = 0.$$

The optimal search plan for any given cost constraint for those detection functions, has been studied in detail by Zhu [Z96], and is omitted here as it is not of primary concern for this thesis.

2.10 Conclusion

In this chapter, we have given an overview of some of the known results derived from the theory of search for a stationary target. Section 2.2 concentrated on the pioneering work done by Koopman who formalized a model for the theory of search. His work was first directed towards the military applications, and soon adopted in many other fields. In Section 2.3 we explained the enhancements to the Koopman model, that were done by Cooper. Cooper's main contribution was to solve the problem in a discrete case. In Section 2.4 we introduced the problem of sequential search. This section gave an overview to the work done by Chew, in which he showed that one would invoke a minimal cost in such a search problem. Section 2.5 reviewed the extensions made by De Guenin to Koopman's continuous model so as to get the minimal effort of search. The minimal cost of search was also explained in Section 2.6, where the works of Staroverov, Gilbert, Onaga and others were explained. Section 2.7 contained a brief synopsis on the field of the whereabouts search problem. In Section 2.8, the sequential search problem was revisited to introduce the work done by Wegener.

Section 2.9 forms the basis of this study. In [Z96], Zhu studied the theory of search for stationary targets when the target distribution is unknown. He provided the upper and lower bounds for optimal strategies, and provided selection criteria for choosing target distributions when two or more cells are searched. In each case, Zhu also derived the estimation of error.

Chapter 3 : LEARNING AUTOMATA AND POINT LOCATION

3.1 Introduction

The research presented in this thesis consists of two fields which complement each other. The first, which was discussed in an earlier chapter, involves the theory of search, and the possibility of locating objects when the target distribution of the target is unknown. The second avenue of the research deals with learning mechanisms using stochastic learning automata. In particular, we focus our attention on the research done by Oommen and Raghunath [OR98] in the field of point location. Their research produced a novel scheme to locate a point in the context of a stochastic search as explained below.

We consider a robot or a learning mechanism that is seeking a particular point or location, which in turn is provided guidance by an Environment, referred to as an Oracle. The Oracle assists the robot in identifying which direction is to be followed. First of all, the robot does not know the final location of the point. In addition, we assume that the Oracle gives advice that could be faulty. Because the Oracle does not present correct direction with a finite probability, the problem is considered to be a “Stochastic Point Location Problem”. Indeed, if the answers from the Oracle were guaranteed to be correct, we would have been dealing with a deterministic point location problem.

This difference will be evident later in the chapter.

Although Oommen and Raghunath's research has importance in its own right, it also proves to be very useful in solving optimization problems [OR98]. Such problems, in which the "point" sought for is unknown, are found in image processing, pattern recognition, and neural computing. These examples, as well as some common techniques for solving problems of this type, will be reviewed presently, so that the reader can get a better understanding of the importance of the work, and the results that are available.

In an effort to provide a complete picture of the field of research, we briefly review previous studies done on the problem of stochastic point location. The topic of searching an unbounded region for an object has been tackled in the publications of Baeza-Yates [BCR88, BS92], where he modeled the search in a way that a cost would be imposed on the searcher proportional to the distance traveled between each two successive positions in the search. His research included searching for a point on a line, in a lattice, and in the plane.

3.2 Learning Automata

Research in the field of learning automata was first started in the former Soviet Union. This work was later clearly shaped in India and the western world. The research was motivated by the need for developing well defined deterministic and stochastic learning methodologies. With time, it became more intriguing to work with systems that are more

oriented towards uncertainty. The need for learning, in identification or control problems, is dependant on the amount of data available with respect to the system at hand. In cases where the problem is deterministic, approaches other than those using learning automata would be recommended. Latter types of learning would be more suitable for complex systems, where the decision makers are organized in a hierarchy and are utilizing incomplete information of the system. An example of such systems is a robot that is driven by imprecise information.

Basically, we can describe a Learning Automaton (LA) as a machine capable of choosing one of a finite number of actions offered by a random environment. When an action affected by the environment is chosen by the LA, it would invoke a response, through which the environment would indicate if the action was favorable or not. The goal is to create a response that utilizes the history of actions and responses without invoking the specific details of the environment itself. In other words, the automaton designed would try to learn from past experience. Choosing the actions could be done deterministically or stochastically. For a learning automaton to achieve its goal, it should be capable of converging to the optimal action from the set of allowable actions.

An example for using learning algorithms is given by Lakshmivarahan in [L81], where he assumed M coins, where M is greater than two. Those coins have one of two probabilities when tossed. A coin yielding a head is considered successful, while the ones yielding tails are failures. Let d_i be the probability of success and c_i be the probability of failure for the i th coin, $i = 1, 2, \dots, M$. Its obvious that $c_i = 1 - d_i$, and we assume that

$d_i \neq d_j$ for $i \neq j$ and $0 < d_j < 1$. We further assume that all the d_i 's are unknown, and that without loss of generality $d_1 > d_2 > \dots > d_M$.

The learning algorithm maybe explained as a method by which we increase the probability of choosing the i^{th} coin at stage $(k+1)$ if it was chosen at time k and the toss resulted in a success, while at the same time decreasing the probability of choosing the i^{th} coin if it is chosen at time k if the toss resulted in a failure. The effects of increasing and decreasing are called rewarding and penalizing respectively. The probabilities of choosing all the other coins are adjusted to keep the total probability equal to unity [L81].

The term “environment” used in the context of learning automata does not carry the exact meaning commonly used elsewhere. Usually an environment is thought of as a group of conditions and influences affecting the life and progress of an organism. When used in the field of learning automata, an Environment is mathematically defined by the triple $\{\underline{\alpha}, \underline{c}, \underline{\beta}\}$ where $\underline{\alpha} = \{\alpha_1, \alpha_2, \dots, \alpha_r\}$ is a finite input set, $\underline{\beta} = \{\beta_1, \beta_2\}$ represents a binary output set, and $\underline{c} = \{c_1, c_2, \dots, c_r\}$ is a set of penalty probabilities, where each c_i of $\underline{c}$ corresponds to a particular input action α_i . The input is applied to the environment at a discrete time. The output is, as mentioned earlier, a binary set, and it is represented by ‘1’ and ‘0’ for mathematical convenience. The output ‘1’ represents the unfavorable response, a “Penalty”, while ‘0’ represents the favorable response, a “Reward”. The elements in the set of probabilities, c_i , represent the probability of issuing an unfavorable response [NT89] when the action chosen is α_i .

To better explain the above concept, we consider the example of a robot searching for a goal, where its search includes two paths. The outcome of its choice is random and dependent on its choice. This outcome is that of locating the goal, and is modeled as a random variable, which is the availability of the goal on the corresponding path. The result of the search could be '1' if the goal was not located, or '0' if the goal was found.

Learning schemes applied to an environment then would involve choosing input actions that would lead to the favorable output with the smallest penalty probability, c . Also the output would be used to choose the new action. In real life cases the penalty probability is not known, otherwise there would be no need for learning.

When studying the field of learning automata, one encounters several relevant terms. The *environment* was defined above. The other important concept is that of the automaton. This is quite a general machine that involves various concepts. These include the set of internal states, Φ , the set of input actions, β , the set of outputs, α , $F(.,.): \Phi \times \beta \rightarrow \Phi$ a function that maps the current state and current input into the next state, and $H(.,.): \Phi \times \beta \rightarrow \alpha$ which is a function that maps the current state and input into the current output. Those five components, represented as a quintuple, $\{\Phi, \alpha, \beta, F(.,.), H(.,.)\}$ define the automaton. In this type of automaton the next state and the current output are determined by both the input and the current state. However, if the current output depends only on the current state regardless of the input, then the automaton is referred to as a "state-output automaton". This means that the function

$H(.,.)$ will be replaced by an output function $\vartheta(.): \underline{\Phi} \rightarrow \underline{\alpha}$. If the sets $\underline{\Phi}$, $\underline{\beta}$, and $\underline{\alpha}$ are finite, then the automata are considered finite. State-output automata are also called stochastic sequential machines [NT89].

The automaton is identified as being deterministic or stochastic based on F and ϑ . If both are deterministic mappings then the automaton is a deterministic automaton. Deterministic automata are characterized by having a unique action and state that succeeds the initial state and its input. If F or ϑ is stochastic, the automaton is called a stochastic automaton. In this case, the transitions are stochastic; there will be no certainty associated with the states and actions following the given initial state and input sequence, and only the associated probabilities would be specified.

3.3 Fixed Structure Automata

The work done by Tsetlin was directed towards the study of deterministic automata functioning in random environments. A considerably large part of the initial research was in the same vein. This research led to the advancement in the design of general fixed-structure learning automata.

The first fixed structure learning automaton developed was the Tsetlin 2-State Automaton. This is an automaton that has two states σ_1 and σ_2 , and two outputs denoted by α_1 and α_2 . The inputs to the automaton are the elements of a binary set. Unfavorable responses represented by '1' cause the state to switch, while '0', indicating a favorable response, forces the machine to remain in the current state. In other words, the automaton

will perform the same action with no change as long as the response is good. It will change to the other action immediately upon receiving a negative response.

Changing the transition matrix to reflect stochastic transitions generalized this two-State automaton. In this extension, the automaton chooses the action leading to a favorable response with probability $1-\gamma_1$ and switching to the other action with probability γ_1 . The reverse of this procedure is followed when the output of the environment is unfavorable with probabilities γ_2 and $(1-\gamma_2)$ respectively.

3.3.1 ϵ -Optimal Schemes

The optimal action of the Environment is the action which has the minimum probability of yielding an unfavorable response. To get the minimum expected penalty, one would require a scheme to compute the ranking of the estimates of the penalty probabilities, either implicitly or explicitly, and using this the scheme should be able to choose the superior action. To achieve what is called ϵ -optimality (also called asymptotically optimal schemes in the literature), the optimal action would be chosen with a probability as close to unity as desired, and this closeness to unity would be determined by some parameter of the scheme.

Most of the schemes available for estimating the penalty probabilities, c_i , are simple and on their own, at least lead to expedient automata. The reason behind their impracticality is that the estimation procedure requires infinite time, and furthermore the

intermediate estimates are not used in the choice of new actions. An ϵ -optimal scheme would have the benefit of having all the actions being chosen enough number of times while attempting to have the probability of choosing the optimal action converging to 1 as time $n \rightarrow \infty$. In essence this means that more effort would be spent choosing better actions, and in the limit, the best action would be chosen almost exclusively.

To be more specific, especially in the case of deterministic schemes, ϵ -optimality would be attained as the memory depth tends to ∞ . For simplicity, consider the case of two actions. In such a case, the expected penalty is expressed as $M(c_1, c_2, N)$, and the automation would be ϵ -optimal (asymptotically with respect to the memory N) if

$$\lim_{N \rightarrow \infty} M(c_1, c_2, N) = \min(c_1, c_2).$$

The definition of ϵ -optimality in the context of general automata introduces us to the concept of adjusting a parameter in the updating scheme so that the behavior of the automaton could be made as close to optimal as desired. In the specific case of fixed-structure schemes, the memory depth is chosen as the parameter, leading to the following definition of ϵ -optimality:

A fixed-structure automaton is said to be ϵ -optimal if, for every $\epsilon > 0$, there exists an N_1 such that for $N > N_1$

$$M(c_1, c_2, \dots, c_r, N) \leq \min_i \{c_i\} + \epsilon,$$

where N is the memory depth of the automaton, and $c_i (i = 1, 2, \dots, r)$ lies in the closed interval $[0, 1]$ [NT89].

Another mathematical definition of ϵ -optimality is given in [L81] where the average probability of success at stage k is

$$\eta(k) = \sum_{i=1}^M p_i(k) d_i ,$$

where M is the number of coins being tossed, and d_i is the probability of success for the i^{th} coin. Then, if d_b is the reward probability of the best action, the algorithm is said to be ϵ -optimal if:

$$\limsup_{k \rightarrow \infty} |E[\eta(k)] - d_b| < \epsilon, \epsilon > 0,$$

where $E[.]$ is the expectation operator.

3.4 Variable Structure Automata

The previous section focused on the behavior of fixed-structure automata operating in stationary random environments. A specific characteristic of such automata is their fixed state transition probabilities. In the course of studying fixed-structure automata, the analysis involves the theory of homogeneous Markov chains.

Instead of being restricted to automata whose structure is fixed, one obtains greater flexibility by using reinforcement schemes. The main advantage obtained by such schemes, is their capability of providing more general stochastic systems in which the state transitions or action probabilities themselves are updated at every time instant. Such automata, termed variable-structure stochastic automata (VSSA), are introduced in this section. In [NT89], the authors review various linear VSSA and show how they can be nonlinearized. This study includes the case of 2-action and r-action VSSA. The tools used

to analyze VSSA are, generally speaking, the same tools needed to study discrete time Markov processes.

3.4.1 Variable structure stochastic automata

It was described earlier that a variable-structure stochastic automaton would update one of two, either the transition or the action probabilities, based mainly on the type of input they would get. The research in this area was first directed towards the update of transition probabilities, but it gradually shifted towards updating action probabilities. The reason behind this shift was the simplicity in controlling the action probabilities. The automata would be represented by

$$\{\Phi, \underline{\alpha}, \underline{\beta}, A, G\},$$

where A is the updating algorithm (also known as the reinforcement scheme), Φ is the set of internal states, $\underline{\alpha}$ is the set of outputs, and $\underline{\beta}$ is the set of input actions.

3.4.2 Linear Schemes

Variable structure automata engulf both linear and non-linear schemes. A number of linear schemes were introduced and were of importance due to their simplicity. Mainly, there are three linear schemes, which are the linear reward-penalty scheme ($L_{R,P}$), the linear reward-inaction scheme ($L_{R,I}$), and the linear inaction-penalty scheme ($L_{I,P}$). These three linear schemes could be either two action or multi-action automata [NT89]. In this research we emphasize the linear reward-inaction scheme, and for simplicity only the two-action case will be reviewed.

3.4.2.1 L_{R-P} Scheme

The linear reward-penalty is one of the first linear schemes introduced in the literature of mathematical psychology. The characteristics that identify this scheme are given in terms of the mathematical equations that represent it. Here, we study only the two-action case. The generalizations to the multi-action case are discussed in the literature, but are not needed in our research.

When considering a learning automaton with two actions, we can define the following equations:

$$g_j(\underline{p}(n)) = ap_j(n)$$

and

$$h_j(\underline{p}(n)) = b(1 - p_j(n))$$

where g_j and h_j are continuous and non-negative functions, $\underline{p}(n)$ is an action probability vector, and a and b are reward and penalty parameters with $0 < a < 1$, $0 \leq b < 1$.

The previous equations can be substituted in the equations for updating action probabilities as follows:

$$\begin{aligned} p_j(n+1) &= p_j(n) - g_j[\underline{p}(n)] && \text{when } \beta(n) = 0 \\ p_j(n+1) &= p_j(n) - h_j[\underline{p}(n)] && \text{when } \beta(n) = 1. \end{aligned}$$

The substitution yields the following results:

$$\begin{aligned} &\left. \begin{aligned} p_1(n+1) &= p_1(n) - a(1 - p_1(n)) \\ p_1(n+1) &= (1 - a)p_2(n) \end{aligned} \right\} \alpha(n) = a_1, \beta(n) = 0 \\ &\left. \begin{aligned} p_1(n+1) &= (1 - b)p_1(n) \\ p_2(n+1) &= p_2(n) + b(1 - p_2(n)) \end{aligned} \right\} \alpha(n) = a_1, \beta(n) = 1. \end{aligned}$$

The above equations are known as the general L_{R-P} updating algorithm. From them it follows that if action α_i is chosen in a stage n , the probability $p_j(n) (j \neq i)$ is decreased at stage $n+1$ by an amount proportional to its value at stage n for a favorable response, and increased by an amount proportional to $[1 - p_j(n)]$ for an unfavorable response [NT89].

There would be a symmetry in the equations in the case where a and b are identical. This is called the symmetric linear reward-penalty scheme. In such a case:

$$p_1(\infty) = \frac{c_2}{c_1 + c_2}, \text{ and } p_2(\infty) = \frac{c_1}{c_1 + c_2}$$

and the scheme is expedient.

If $b \ll a$, the scheme tends to be ϵ -optimal [L81].

3.4.2.2 L_{R-I} Scheme

One of the simple linear schemes is the linear reward-inaction scheme. This scheme is based on the more general L_{R-P} scheme. This scheme is best described as a scheme that does not change the probabilities whenever an unfavorable response is received from the environment. Instead, the probability of the action is increased each time a favorable response is received. Scientists first considered the mathematical psychology behind the L_{R-I} schemes, and from there it moved towards the engineering literature.

As mentioned earlier, the L_{R-I} scheme is based on the more general family of L_{R-P} schemes. Thus, to acquire the equations for the L_{R-I} scheme we substitute $b=0$ in the L_{R-P} equations. This would yield the following equations:

$$\begin{array}{lll}
p_1(n+1) = p_1(n) - a(1 - p_1(n)) & \alpha(n) = \alpha_1 & \beta(n) = 0 \\
p_1(n+1) = p_1(n) & \alpha(n) = \alpha_1 & \beta(n) = 1 \\
p_1(n+1) = (1 - a)p_1(n) & \alpha(n) = \alpha_2 & \beta(n) = 0 \\
p_1(n+1) = p_1(n) & \alpha(n) = \alpha_2 & \beta(n) = 1
\end{array}$$

It is evident from the previous equation that the probability $p_1(n)$ is increased when the action α_1 leads to a favorable response. On the other hand, if the response is unfavorable the probability would remain unchanged. It would be decreased only in the case where α_2 is performed and a favorable response is generated [NT89]. The L_{R-1} scheme is ϵ -optimal in all random environments.

3.4.3 Non-Linear scheme

Initially, reinforcement schemes, which were the variable structure automata, were mostly linear schemes. Considerable attention was given to the L_{R-1} scheme in particular, for reasons of mathematical simplicity and ϵ -optimality. This did not prevent efforts to explore the nonlinear schemes as well. These efforts were initially limited to the two-action case, but in the book [NT89], Narendra and Thathachar present all the known nonlinear absolutely expedient schemes and also the necessary and sufficient conditions for such automata.

3.5 Continuous point location

Point location is a foundational concept in many geometric and geographic problems. In many cases it is needed to solve even larger geometric problems. When dealing with a deterministic environment, the problem would be reduced to a deterministic point

location problem. Such problems were discussed earlier. The results available for this problem have been extended to solve for the best way to look for an object in an unbounded region. By knowing the direction to the goal, this would yield to a deterministic solution. On the other hand, in cases where the knowledge available could be incorrect, researchers have resorted to stochastic searching to locate the point. Several attempts have been made towards solving the problem of stochastically locating a point on a line. This section reviews two algorithms that introduced novel approaches in handling the problem. Both the algorithms will be described, and their corresponding results are cataloged.

3.5.1 Discrete space learning algorithm

To search stochastically for the location of a point on a line, we assume that the search algorithm seeks the help of an Environment. The main function of the Environment is to assist the algorithm in selecting the path of the search. This environment is also known as an Oracle, and it is assumed to yield possibly erroneous feedback. This means that it could advise the search algorithm to move to the right, while, in reality, it should move to the left, and vice versa. The probability of error in the Oracle's response transforms this problem into a stochastic point location problem. As mentioned earlier, the problem would have been a deterministic point location problem if the Oracle had been deterministic in its response.

It is crucial to note that apart from the fact that the problem of stochastic point location has importance in its own right, it also has more importance because of its utilization in solving optimization problems. Optimization problems are concerned with minimization

of effort and maximization of benefit. Usually, in optimization problems, the underlying cost function (or benefit function) is known, and the problem would be of minimizing (maximizing) this function. Therefore the main goal would be to find the parameter that optimizes the algorithm being used.

What distinguishes this algorithm from others is that it can be used to be supporting algorithm to augment other algorithms used for optimizations. For example, in a typical hill-climbing algorithm, if the parameter is too small the convergence is sluggish. In contrast, if the parameter is too large, the system could erroneously converge and in some cases it could even oscillate. A stochastic point location can be used to determine the best parameter. Another benefit to the algorithm presented here, is that it does not depend on the computation of any additional derivatives. The Newton's root solving schemes, for example, which depends on second derivative calculations, would diverge, instead of converging, if the starting point is not well chosen.

3.5.1.1 A Discretized Algorithm

The algorithm presented in [O97] is based mainly on discretizing the parameter space and performing a controlled random walk on this space. The primary advantage of a discrete Learning Algorithm (LA) over other LAs is that it uses the limitations of practical implementations to its benefit. Traditional LAs are typically the family of variable structure stochastic automata, which were developed to simplify the analysis of the automata's properties. Among the limitations of those algorithms is the limitation imposed by the fact that, the probability of choosing an action is an element of the finite set of real numbers in the interval $[0,1]$ specified by the precision of the system. A

general approach for solving such a problem is by rendering this probability space discrete, as this would improve the learning properties of the algorithms mentioned earlier. In these new discretized algorithms, the probability changes are made in jumps and not continuously, and thus the speed of the learning process is increased.

The main goal of the stochastic search algorithm presented here is to determine the optimal value of some parameter, λ . We assume that there is an optimal value for λ , which is an unknown value to the algorithm, and denoted to by λ^* . An intelligent environment, $\mathfrak{E}$ (or Oracle), suggests a direction of motion to the algorithm based on the current value of λ . The response of the Oracle is not totally accurate, as it can be faulty. This means that the algorithm could get a response from the Oracle to increase λ , where, in fact, it should decrease it, and vice versa. The probability that the response of the Oracle is correct is p , while q represents the probability of having an erroneous response. To render the problem realistic, we assume $p > 0.5$. If $\lambda < \lambda^*$, the environment could either recommend correctly the increase of λ with probability p , or it could incorrectly advise the algorithm to decrease λ with probability q .

The algorithm assumes the subdivision of the space in a discrete manner such that the unit interval is divided into N steps $\{0, 1/N, 2/N, \dots, (N-1)/N, 1\}$, where N is the resolution of the learning scheme. A larger value of N would intuitively suggest a more accurate result of λ^* . Using this assumption, the algorithm then follows some simple rules in an attempt to learn λ^* . Let $\lambda(n)$ be the value at time step 'n'. Then,

$$\lambda(n+1) := \lambda(n) + 1/N \quad \text{If } \mathfrak{C} \text{ suggests increasing } \lambda \text{ and } 0 \leq \lambda(n) < 1, \quad (1)$$

$$\lambda(n+1) := \lambda(n) - 1/N \quad \text{If } \mathfrak{C} \text{ suggests decreasing } \lambda \text{ and } 0 < \lambda(n) \leq 1. \quad (2)$$

At the end states of the scheme the updating would be:

$$\lambda(n+1) := \lambda(n) \quad \text{If } \lambda(n) = 1 \text{ and } \mathfrak{C} \text{ suggests increasing } \lambda \quad (3)$$

$$\lambda(n+1) := \lambda(n) \quad \text{If } \lambda(n) = 0 \text{ and } \mathfrak{C} \text{ suggests decreasing } \lambda. \quad (4)$$

We again draw the attention of the reader to the fact that, although the above rules are deterministic, yet the scheme is still stochastic because of the possible error in the response of environment.

Based on the rules mentioned above, Oommen proved in [O97] that the parameter learning algorithm, is asymptotically optimal and yields $\lambda(\infty)$ which converges to λ^* with probability 1.

Theorem I

The parameter learning algorithm specified by (1)-(4) is asymptotically optimal.

Proof:

The proof of this theorem is found in [O97].

...

3.5.1.2 Experimental results

To ensure robust results and to obtain an understanding as to how the scheme converges with time, various experiments were conducted. To achieve both goals, the experiments were conducted based on two criteria. The first was to evaluate $E[\lambda(\infty)]$ by computing the closed-form expression of the asymptotic distribution, and as a result evaluate its mean terminal value. The criterion to achieve the second goal, as described in

[O97], would be to conduct a number of experiments in parallel. The exact number of experiments was one hundred. The purpose of conducting such parallel experiments was to evaluate the performance of the algorithm under a variety of constraints and with a large number of experiments, the accuracy of the results would be guaranteed. In each case, the value of the parameter λ^* was assumed unknown to the learning mechanism, and the simulation was carried out for various values of p , the probability of the environment responding correctly. The average of the asymptotic value of λ was recorded for each of the parallel experiments.

The results of the experiments demonstrated the precision of the algorithm presented. Here we give some of the results given in [O97]. Table 3.1, below, provides the recorded true value of $E[\lambda(\infty)]$ for various values of p and for various resolutions when the value of λ^* is 0.9123.

Table 3.1: Results of solving the stochastic point location problem by using the discrete space method. In this case the value of λ^* is 0.9123.

Log₂N	p = 0.70	p = 0.85	p = 0.95
2	0.7470205	0.8341304	0.8644939
3	0.8615779	0.9167632	0.9322447
4	0.885711	0.9035668	0.9060284
5	0.9164335	0.9215552	0.9218668
6	0.9137105	0.914061	0.9140624
7	0.9101543	0.9101563	0.9101564
8	0.9121094	0.9121095	0.9121093
9	0.9130859	0.9130861	0.9130858
10	0.9125975	0.9125977	0.9125975
11	0.9123535	0.9123536	0.9123535
12	0.9122314	0.9122314	0.9122314

In Figure 3.1, a plot of the asymptotic value of $E[\lambda(\infty)]$ is presented, as a function of N for various values of p .

Figure 3.1: Plot of the asymptotic value of $E[\lambda(\infty)]$ with the resolution parameter, N for various values of p when λ^* is 0.9123

The results of the second set of experiments (as reported in [O97]) intended to catalogue the convergence of $E[\lambda(\infty)]$ with time, 'n', is presented in Figure 3.2. With each simulation an average was obtained by performing one hundred parallel experiments. The results presented here report results for specific values of N , although in the actual simulations, the resolution of the mechanism, N , was also maintained as a variable. It shows from Figure 3.2 that the convergence is remarkable, and it is even more rapid with larger values of p .

Figure 3.2: Plot of the variation of the estimate $E[\lambda(\infty)]$ of with time, n , for $N=64$ and for various values of p . In this case λ^* is 0.9123.

3.5.2 Adaptive Tertiary Search Algorithm

The previous background review was necessary to pave the way for understanding the research done by Oommen and Raghunath to solve the stochastic point location problem, which is an integral part of this thesis. In [OR98] these authors used a continuous learning automaton to solve the problem of stochastically searching for a point on a line. The searching algorithm moves in an interval and attempts to locate the unknown point in this interval by recursively splitting the interval. As before, the problem assumes the existence of a learning mechanism (for example, a robot), which attempts to locate this unknown point. Assisted by the Environment, $\mathcal{E}$, this robot moves in a direction that should lead to the point eventually. As before, the Environment is assumed faulty. Thus it

could advise moving to the left, while, in fact, the robot should move to the right, and vice versa.

3.5.2.1 Continuous point location with adaptive tertiary search

The solution presented in [OR98] revolves around systematically exploring an interval for the parameter. This exploration is based on a series of probes that converges towards accuracy. In every stage, the parameter is estimated to be the mid point of the current interval. If the point is not located, the interval is subdivided into three intervals. Each of these intervals would be probed using an “ ϵ -optimal fast converging two-action learning automaton” [OR98], where the two actions represent the decision to go either left or right of the current location. Using the results of exploring each of the three intervals obtained from the Oracle, the algorithm would then determine which interval is to be eliminated, and which interval is to be recursively subdivided to continue the search. The algorithm guarantees that at least one subinterval would be eliminated, as the Oracle might suggest to subdivide only one interval or a combination of two adjacent intervals. At each stage the midpoint of the pruned search subspace, after convergence, is taken to be the estimate of λ^* . The pruning would continue until the search interval is at least as small as a resolution specified by the algorithm.

3.5.2.1.1 Algorithm definitions and pseudo-code

Before detailing the algorithm, the notation is introduced. Let $\Delta = [\sigma, \gamma]$ such that $\sigma \leq \lambda^* \leq \gamma$. When Δ is partitioned, the result is $\Delta^j, j \in \{1, 2, 3\}$. For every partition Δ^j , we define L^j and R^j as its left and right halves respectively, written in the form:

$$L^j = \{x \mid \sigma^j \leq x < \text{mid}(\Delta^j)\}$$

and

$$R^j = \{x \mid \text{mid}(\Delta^j) \leq x < \gamma^j\},$$

where $\text{mid}(\Delta^j)$ is the mid point of the partition Δ^j .

Points x_L^j and x_R^j represent any point x that belongs to L^j and R^j respectively.

It has already been mentioned that the automaton used in each subinterval is a two-action automaton. The two actions are $\alpha_k^j, (k = 0, 1)$, where α_0^j corresponds to selection a point x_L^j , and α_1^j corresponds to selection a point x_R^j . Each of the actions evaluated by the Oracle has a probability, $p_k^j(n)$ which represent the action probabilities at time n . Initially, the automaton chooses as per the assignment $p_k^j(0) = 0.5$, for $k = 0, 1$. At any time n , when the automaton A^j chooses an action α_k^j the Oracle responds with a binary feedback $\beta(n)$ where, $\beta(n) = 1$ implies a penalty and $\beta(n) = 0$ means reward. The probability that the automaton receives the correct feedback from the random environment is p . Thus,

With probability p , $\beta(n) = 1$ given that α_0^j was chosen and $x_L^j < \lambda^*$

With probability p , $\beta(n) = 1$ given that α_1^j was chosen and $x_R^j \geq \lambda^*$

With probability p , $\beta(n) = 0$ given that α_0^j was chosen and $x_L^j \geq \lambda^*$

With probability p , $\beta(n) = 0$ given that α_1^j was chosen and $x_R^j < \lambda^*$.

The scheme is updating by following the L_{RI} rule. Thus, if $\alpha_k^j (k=0,1)$ was rewarded then

$$P_{1-k}^j(n+1) \leftarrow \theta \cdot P_{1-k}^j(n)$$

$$P_1^j(n+1) \leftarrow 1 - \theta \cdot P_{1-k}^j(n)$$

where $0 < \theta < 1$ is the L_{RI} reward parameter.

From the action probabilities, the algorithm deduces the decision of the L_{RI} automaton, after iterating for a known number of times, N_∞ , where N_∞ should be large enough to guarantee convergence. The decision O^j is given by:

$$\begin{aligned} O^j &= \text{Left} && \text{if } P_0^j(N_\infty) \geq 1 - \varepsilon && \text{In this case } \lambda^* \text{ is either to the left of or in } \Delta^j, \\ &= \text{Right} && \text{if } P_1^j(N_\infty) \geq 1 - \varepsilon && \text{In this case } \lambda^* \text{ is either to the right of or in } \Delta^j, \\ &= \text{Inside} && \text{Otherwise} && \text{In this case } \lambda^* \text{ is contained in the interval } \Delta^j. \end{aligned}$$

Pruning of the intervals would take place right after the individual automata for the three sub partitions have made a decision on their guess as to where λ^* is. The new pruned partition for the subsequent learning phase is generated as shown in Table 3.2, and the formal algorithm follows.

Table 3.2: Decision table to prune the search space based on the decision of the individual automata

O^1	O^2	O^3	New Sub-Interval
<i>Inside</i>	<i>Left</i>	<i>Left</i>	Δ^1
<i>Left</i>	<i>Left</i>	<i>Left</i>	Δ^1
<i>Right</i>	<i>Inside</i>	<i>Left</i>	Δ^2
<i>Right</i>	<i>Left</i>	<i>Left</i>	$\Delta^1 \cup \Delta^2$
<i>Right</i>	<i>Right</i>	<i>Inside</i>	Δ^3
<i>Right</i>	<i>Right</i>	<i>Left</i>	$\Delta^2 \cup \Delta^3$
<i>Right</i>	<i>Right</i>	<i>Right</i>	Δ^3

The above table was proved to be complete by the theorems introduced in [OR98]. Based on this, the following theorem was reached:

Theorem II

If the algorithm uses the same L_{RI} scheme at all levels of the recursion with the parameter θ arbitrarily close to unity and N_∞ is sufficiently large, then the unknown λ^ is*

contained in the subinterval encountered in the subsequent invocation of the algorithm with a probability as close to unity as desired, and the algorithm finally converges to a value arbitrarily close to λ^* .

Proof:

The proof of this theorem is found in [O97].

...

Algorithm CPL_ATS(Δ)

Input: Resolution: the size of the smallest signifcant interval containing λ^* . Its magnitude deteremines the accuracy of the final estimate and is used to terminate the recursion.

Output: The estimate of λ^* . It is derived as the mid-point of the final search interval.

Begin

If WidthOfInterval(Δ) < Resolution **Then**

Return (MidPointOfInterval(Δ))

/* Terminate Recursion */

Else

$(\Delta^1, \Delta^2, \Delta^3) \leftarrow \text{PartitionInterval}(\Delta)$

For $j \leftarrow 1$ **to** 3 **Do**

$L_{RI_Output}[j] \leftarrow \text{Execute_L}_{RI}(\Delta^j, j)$

EndFor

$\Delta_{new} \leftarrow \text{ChooseNewSearchInterval}(L_{RI_Output}[])$

CPL_ATS (Δ_{new})

/*Tail Recursion*/

EndIf

End Algorithm CPL_ATS

Procedure Execute_L_{RI} (Δ^j, j)

Input: (i) The Sub_Interval, Δ^j . $j \in \{1, 2, 3\}$ is the index of the Sub_Interval.

(ii) The parameter of the L_{RI} scheme, θ and the convergence error tolerated ϵ .

(iii) We assume that the scheme is interacting with the Oracle which returns the correct response with the probability p .

Output: A decision from the set $\{Left, Right, Inside\}$ representing whether the automaton has determined λ^* to be the left, right or inside the current subinterval Δ^j .

Method

Begin

Initialize **For** $k \leftarrow 0$ **to** 1 $P_k^j \leftarrow 0.5$

For $i \leftarrow 1$ **to** N_∞ **Do**

$k := \text{ChooseAction}(\Delta^j)$

If ($k=0$) **Then**

$x_L^j \leftarrow \text{PickARandomPointIn}(L^j)$

```

         $\beta \leftarrow \text{GetFeedBack}(k, x_L^j)$ 
        If ( $\beta = 0$ ) Then
             $P_1^j \leftarrow \theta \cdot P_1^j$ 
             $P_0^j \leftarrow 1 - P_1^j$ 
        EndIf
    Else
         $x_R^j \leftarrow \text{PickARandomPointIn}(R^j)$ 
         $\beta \leftarrow \text{GetFeedBack}(k, x_R^j)$ 
        If ( $\beta = 0$ ) Then
             $P_0^j \leftarrow \theta \cdot P_0^j$ 
             $P_1^j \leftarrow 1 - P_0^j$ 
        EndIf
    EndIf
EndFor
If ( $P_0^j = 1 - \varepsilon$ ) Then
    Return (left)
Else If ( $P_1^j = 1 - \varepsilon$ ) Then
    Return (Right)
Else
    Rreturn (Inside)
EndIf
EndIf
End Procedure Execute_ $L_{RI}$ 

```

3.5.2.2 Experimental results

Experiments were done using the CPL_ATS algorithm to evaluate its rate of convergence and to verify its validity. While testing for the effectiveness, simulations were done for various values of the reward parameter of the L_{RI} automata, θ . Also various probabilities of the Oracle's correctness were used. In all cases the value of λ^* was assumed unknown. The experimental results were truly conclusive. We present the results for one such experiment, in which the unknown parameter $\lambda^* = 0.9123$. As mentioned earlier those experiments recorded the value of $E[\lambda(n)]$ for various values of p and for various L_{RI} reward factors.

Figure 3.3 and Figure 3.4 chart some of the findings.

Figure 3.3: Plot of $E[\lambda(n)]$ with time n , when $p = 0.7$ and the L_{Rl} parameter θ varies from 0.6 to 0.8. Here λ^* is 0.9123. Time n is shown in epochs of size 50 units.

Figure 3.4: Plot of the asymptotic value of $E[\lambda(\infty)]$ with p varying from 0.5 to 1.0 and for various values of the L_{RI} reward parameter, θ . The latter increases from 0.6 to 0.8. Here λ^* is 0.9123.

3.6 Conclusion

The main goal in this chapter was to introduce the research of Oommen [O97] and Oommen and Raghunath [OR98] to solve the stochastic search problem, which involved searching for a point on a line where the Oracle responded with inaccurate responses.

Since the results of [O97] and [OR98] used the theory from the field of learning automata, we presented a general overview involving both fixed and variable structure automata. Examples for both these families were given, and the examples chosen were the ones used in the algorithm presented in [OR98], and in particular the ϵ -optimal LRI scheme.

Section 3.5 focused on the research done on continuous point location. The first part of the chapter dealt with Oommen's work which was presented in [O97]. In this work Oommen introduced a novel approach to solve the problem of searching stochastically for the location of a point on a line. His work was based on the idea of rendering the search space discrete which is accomplished by subdividing the space into a number of discrete partitions. The algorithm then traverses those partitions based on the guidance from the Environment, which is assumed to be inaccurate.

In Section 3.5.2 the new algorithm due to Oommen and Raghunath, [OR98], was explained. These authors used a continuous learning automaton to solve the problem of stochastically searching for the point on a line. The searching algorithm moves in an interval and attempts to locate the unknown point in this interval by recursively splitting the interval into three subintervals. With the help of inaccurate responses from the Oracle, those subintervals are searched and based on the findings, the search space is pruned to produce the new interval to be searched. The algorithm computes the unknown value of λ^* with an arbitrarily high probability.

The experimental results reported in [OR98] are truly conclusive. In all the experiments the convergence was remarkable. Theoretically, the learning scheme converges ϵ -optimally.

Chapter 4 : OPTIMAL SEARCH BY EXHAUSTIVE ENUMERATION

4.1 Introduction

One of the main goals sought after in search problems is efficiency. A search algorithm or technique that utilizes the resources it has in such a way, so that it would use the minimal amount of effort to locate the target in the fastest possible time, is considered to be both efficient and optimal. This chapter outlines techniques that ensure that the methods of searching being used are characterized by such qualities and that the search is done in an optimal way.

When searching it is expected that an object or a target is located in one of many places and that the search algorithm would search each location till the object is found. This could end up being a very tedious process, and if a specific amount of time is allocated to the overall search, the search could end before covering all the possible locations. Therefore it is reasonable to assign a probability to each location searched, which represents the probability of the target being actually in this location. Considering this probability, the searcher would assign the appropriate amount of time that the search algorithm should spend in that specific location. Typically, in the literature, the set of the location probabilities, called the *target distribution*, are assumed to be known. In reality,

though, this is not the case, and hence we assume that the target distribution is not known and there should be a way to estimate it. This is exactly the scenario studied in this chapter.

The research done by Zhu and Oommen mainly focused on proving the mathematical results for searching for a target while the target distributions is unknown. Although their research was successful in proving the pertinent results, the question of how the algorithms would behave in practical situations was not addressed.

The main focus of our research was to create a simulation that implements such algorithms. The intention was to find a way by which we could obtain strategies for practical scenarios in which the probabilities are, in turn, computed in terms of their estimates, and to compare the results to the findings from the mathematical research done by Zhu and Oommen. Their research did not dictate a process by which the probabilities could be estimated in real life because the events themselves are unobservable, and so this was an unsolved problem. Also such a simulation would be of great importance to serve as an automatic testing mechanism that would support further work in this field. The simulation would be the basis on which modifications could be added to the mathematical model, and the effectiveness of those modifications could be tested. Indeed, although the research introduced by Zhu and Oommen was novel, this present research complements it by proposing how the search times and search accuracies can be efficiently estimated in real life scenarios.

4.2 Searching with Unknown Target Distribution

Since the assumption that the target distribution is known does not apply to many real-life scenarios, Zhu and Oommen initiated research to devise methods to solve the search problem when the target distribution is unknown. The method that solves this problem was the core of the research done by Zhu and Oommen [ZO97]. Rather than assuming the knowledge of the target distribution, they modeled the problem in such a way that the searcher would be required to assume a target distribution based on a priority set of the most likely values.

The strategy, by which this assumed target distribution is to be used, should satisfy two conditions. The first condition is that by using this assumed distribution one should be able to maximize the detection probability of the optimal search. The second condition is that such a distribution should be capable of minimizing the difference between the detection probability of the optimal search plan of the *real* target distribution, and the detection probability of the *assumed* target distribution.

4.2.1 Optimal search range

When the goal is to choose a target distribution that would maximize the efficiency of the search, one must know what the range of possible values is. It was proven in [ZO97] that there is a range of values for the target distribution that would maximize the detection probability.

To identify this range, Zhu and Oommen [ZO97] assumed that the detection function had an exponential form given by

$$b(t) = 1 - e^{-t}.$$

We now formulate the search problem. The searcher is searching for a target found in one of multiple locations identified as $C_1, C_2, \dots, C_N$ with a target distribution $p(i)$, ($i = 1, 2, \dots, N$). The detection probability increases for all i with the time z as per:

$$b(i, z) = 1 - e^{-z}, (i = 1, 2, \dots, N),$$

if the target is indeed in the cell searched.

The detection probability $P[f^*]$ of the optimal search strategy f^* should satisfy the following:

$$1 - e^{-K/N} \leq P[f^*] \leq 1 - C_N e^{-K/N},$$

where $C_N = N[p(1) \dots p(N)]^{1/N}$, and $K > 0$ is the total cost constraint.

Zhu and Oommen relaxed the condition of having an exponential detection function, while maintaining all other assumptions. They generalized the former result for the case when the detection function was a regular function with a derivative $b'(i, \cdot)$ that is decreasing, and satisfying $b'(i, 0) > 0$ and $b'(i, \infty) = 0$. In such a case, the detection probability $P[f^*]$ of the optimal search strategy f^* was shown to satisfy

$$b(i, 0) + \lambda K \leq P[f^*] \leq b(i, 0) + b'(i, 0)K.$$

This defines the upper and lower bounds of the optimal search, and are to be used to calculate the error when deciding on the target distribution.

4.2.2 Criteria for choosing target distribution

In order to actually conduct the search, the first task we achieved was that of calculating the upper and lower bounds of the optimal search. The next task was that of solving the criteria for choosing target distributions.

In [ZO97], the authors outlined two criteria for choosing the unknown target distributions. Before explaining both criteria, they proved that the detection probability of the optimal search strategy when using the *true* target distribution is greater than or equal to the detection probability of the optimal search strategy for any other *assumed* target distribution. This result proven as a theorem is written formally as follows:

$$P[f_b^*] \leq P[f_a^*],$$

where a is the true unknown probability, and b is the assumed one, and the target distribution for the two-celled problem is $[a, 1-a]$.

Using this theorem, the authors of [ZO97] considered the criteria for choosing the target distributions when the search is being done in only two locations. Furthermore, the detection function was assumed to be exponential. The first criterion to determine a was to find the value of a which maximized the expression

$$P[f_b^*] = 1 - \frac{a + b - 2ab}{\sqrt{b(1-b)}} e^{-K/2}$$

over a priority set, which is the set of those values of a that are the most likely ones. This priority set, A , can be defined mathematically as the set of values which a can take with probability

$$P[a = a_0 \in A] \geq p_0,$$

where p_0 is a fixed number.

The second criterion involves the estimation of the error in detection probability of the search plan which utilized a distribution $[b, 1-b]$. This error is given by:

$$E(a,b) = \frac{a + b - 2\sqrt{ab}[\sqrt{ab} + \sqrt{(1-a)(1-b)}]}{\sqrt{b(1-b)}} e^{-k/2}.$$

The second criterion outlined in [ZO97] specifies that the target distribution should be chosen so as to minimize $E(a,b)$ over a priority set of A .

It is worth noting that the smaller the priority set is, the more accurate the assignment of target distribution can be. This means that smaller values $E(a,b)$ can be obtained by using smaller priority sets.

4.3 Simulation of the Search

The research done by Zhu and Oommen, described above, succeeded in proving the mathematical results for the search. This research, however, did not implement or simulate the solution.

The main focus of our research was to create such a simulation. The intention was to have a process that could be used as a supporting tool so as to invoke the formal results by [ZO97]. Those results depended on a priority set, which would be used by the searcher to choose from an assumed target distribution. This is only mathematical and not proven in real-life scenarios. Since the priority set is target dependant, there was a need to simulate it in a generic way to prove the validity of the whole algorithm in practical situations. Furthermore, a simulation would create an automatic testing mechanism that would support further work in this field. Also it would serve as the basis on which modifications could be added to the mathematical model, and the effectiveness of those modifications could be tested.

It should be clearly stated that the simulation is based on the same set of assumptions introduced in [ZO97]. The simulation is, as previously mentioned, aimed at automating these theoretical results. First of all, the algorithm assumes that an object (target) is located in one of two different locations. Both the location and the target distributions are assumed unknown to the simulation package. Finally, the simulation package assumes an exponential detection function.

4.3.1 Search simulation

A simulation is a representation or an imitation of the behavior of some situation or system by utilizing an analogous scenario, typically on a digital computer. In our study, we simulate the problem of searching for an object without knowing the target distribution and the ways by which the target distribution could be assumed. This system is sub-divided into sub-systems each of which is also simulated.

The target, which is the subject of this search, is one of the simulated subsystems. The possible locations at which the target could be found, the real and unknown value of the probability of finding the target in such locations, and the mechanism used to search for the target are the other processes that are also involved.

4.3.1.1 Target and location

As mentioned earlier, a target is the object of the search. This target could range from being an object in one of several possible locations in a room, to a soldier or a group of soldiers lost in some kind of terrain.

The different variations that a target can assume have an impact on the search algorithm, in the sense that it affects the data being collected in order to guess the target distribution. The reason that the data is affected by the change of the target is that this kind of information is mainly related to the behavior and the environment surrounding the target. Therefore it would differ from one target to the other. We earlier mentioned that determining the “best” target distribution is achieved by maximizing the expression

$$P[f_b^*] = 1 - \frac{a + b - 2ab}{\sqrt{b(1-b)}} e^{-K/2}$$

over a priority set. This priority set is based on domain knowledge relevant to the target.

To demonstrate the validity of the above result, we have formally simulated the entire setting. The current simulation makes use of the memory locations that represent the

various entities involved. Thus the target is represented quite simply by an integer random variable as follows.

The target is hidden in one of two possible locations designated by two integer arrays. The target is placed in a specific slot in one of these arrays. The position of the target is maintained as a “bookmark” within the array.

The size of the array is based on the total amount of time allocated for the search. Assuming that searching each slot in a single array would require a single unit of search time, each array has a number of slots equivalent to the total allocated search time. The target would be located in a random position determined by the true but unknown *real* target distribution.

4.3.1.2 Real target distribution

When conducting a search with limited resources, it is more efficient if the target distribution is known. The simulation we embark on assumes that the target distribution is unknown. Still, for the purpose of testing and to be able to compare the results with the real unknown values, we also model the *true* target distribution. Introducing the *real* target distribution in the model does not contradict with our previous assumptions since it will remain unknown to the learning modules of the simulation.

The simulation program would, therefore, use the real target distribution place the target in one of the two bins by involving a random number generator (i.e., the flip of a coin). The “coin is flipped” with a certain probability and depending on which side it falls, it

would be placed in one of two presumed locations. Once the identity of the location is determined, the location within the location is determined by involving another random number generator which “places” the target in the particular array location.

By way of example, if we assume that the coin falls on its “head” for seventy five percent of the flips, then the target would be located at the first location, location ‘A’, with probability 0.75, denoted by ‘a’. Needless to say, the probability of the target being at the other location, location ‘B’, would be (1 - a).

The above simulation would first be repeated for a large number of times, N . The estimate of the real but unknown probability of the target would be based on the location of the target, and its estimated value would be the average probabilities of being assigned to this location.

This estimation is the method by which the simulation can mimic the real life scenario of assuming the value of ‘a’. Such a scenario typically involves choosing a value from a priority set, which represents the set of values that are most likely to be the correct value of ‘a’. This set is defined mathematically by defining the priority set A to be the set of values which ‘a’ can take with probability

$$P[a = a_0 \in A] \geq p_0,$$

where p_0 is a fixed number.

Obviously, the smaller the priority set is, the more accurate the assignment of target distribution can be. Thus, in real life scenarios, the searcher should utilize information from all possible sources to reduce the size of the priority set, and thus limit the range of possible target distributions.

4.3.1.3 Hiding the target

The target is randomly hidden in the one of the two locations using the probability that characterizes the Real Target Distribution. Thus the hiding algorithm would try to randomly hide the target, favoring the locations with the same probabilities specified by the real target distribution.

Observe that once the exact location for hiding the target is selected, another random value would be used to hide the target in one of the slots of the selected array. This would guarantee that the hiding process is totally random and that the search that would be done later would not be biased.

4.3.1.4 Assumed target distribution

Although the real target distribution is used to place the target, the learning algorithm does not know it. Thus, as far as we are concerned, we deal with it as if it is unknown. That means that our algorithm would attempt to learn it by solely assuming a target distribution, from which we could infer the real one. The algorithm would also have to devise a method by which it could obtain different values for the assumed target distributions.

The process, by which the different values of target distribution are tested, is by sequentially probing the range of possible values. With each value, the following equation is applied

$$P[f_b^*] = 1 - \frac{a + b - 2ab}{\sqrt{b(1-b)}} e^{-K/2}$$

and the result is checked. The greater the value, the more accurate the target distribution is, and the closer it is to the real one. The range being scoped is the interval between zero and the unity. Real values are selected from this range, one at a time, and substituted in the equation. The result is the detection probability of the system can be learnt. Values that yield a higher result are more likely to be the correct target distribution. To obtain better accuracy the difference, δ , between each of selected real numbers should be very small. However, using a smaller value of δ comes a price, which is the time taken to go through the desired real numbers in the range. Thus, if $\delta = 0.001$, the simulation will require 1000 attempts to span the entire range.

4.3.2 Simulation algorithm

The algorithm described informally in the previous sections is formally given below.

Algorithm Simulate(K)

Input: This algorithm would simulate the search for an object in one of two bins with unknown target distribution. The search has to be completed within a specific total time, K .

Method

Begin

/* simulating the existence of a target to be searched for */
target $\leftarrow$ CreateTarget()

/* simulating two different locations at which the target can be present */
locationA $\leftarrow$ CreateLocation(K)
locationB $\leftarrow$ CreateLocation(K)

```

/*Probability of 'a', which is the real probability of finding the target in location A*/
realProbability ← GenerateRealProbability()

/*hide the target in one of the locations*/
HideTarget(target)

/*Obtain the assumed probability by getting the maximum Detection function*/
assumedProbability ← AssumeProbability(K, realProbability)

/* Use the assumed probability to search for the target in the more*/
targetFound ← PerformSearch(assumedProbability, K)
End Algorithm Simulate

```

Procedure CreateLocation (K)

```

Input: Each location needs total time  $t$  to be searched fully
Output: Location which can be one of the bins in which the target is suspected to be at.
Method
Begin
    /* Size is one slot * the total time in minutes * 60 to convert it to seconds */
    numberOfSlots ←  $K * 60$ ;
    location ← CreateLocationWithSpecificSize(numberOfSlots)

    For  $i \leftarrow 0$  to numberOfSlots Do
        location.SetAt( $i$ , 0); /*All slots in the location are zero to indicate being empty*/
    EndFor
    Return location
End Procedure CreateLocation

```

Procedure AssumeProbability(K,a)

```

Input: To utilize the expression, introduced by [ZO97], the procedure would use the total time,  $k$ , and
the real probability,  $a$ .
Output: Accurately assumed probability
Method
Begin
    assumedProbability ← 0
     $P[f_b^*] \leftarrow 0$ 
    largest ← 0
    For  $b \leftarrow 0$  To 1 Do
        
$$P[f_b^*] = 1 - \frac{a + b - 2ab}{\sqrt{b(1-b)}} e^{-k/2}$$

        If largest < detectionProbability Then
            /*the Correct probability to find this object is the probability which maximize
            the expression*/
            largest =  $P[f_b^*]$ 
            assumedProbability =  $b$ 
        EndIf
         $b \leftarrow b + 0.005$ 
    EndFor
    Return assumedProbability
End Procedure AssumeProbability

```

4.4 Experimental Results Based on Simulation

Creating a procedure by which we could assume the target distribution that would be used to solve the search problem, described in this chapter, was experimentally simulated to verify the validity of the claims, and to examine how the search proceeds. Besides, we are also interested in the effectiveness of the algorithms introduced in [ZO97]. To verify the power of the scheme and to study its effectiveness for a variety of conditions, the simulation was rigorously tested for various settings and parameters.

4.4.1 Effect of location size on the search

The simulation proposed an array of numbers used to designate the possible locations of the target. The size of the arrays to be searched has direct impact on the search being conducted and on the time taken by the system to locate the target. This is applicable to any type of location, and therefore assumed as a basic parameter in the simulation.

The simulation program assumes that a searcher would require one time unit to go through each slot in the arrays to be searched. Thus, if an array has 10 slots in size, it would require the searcher 10 time units to go through the whole array searching for the target in every location assuming all along that the target was not found before that. This was enforced by using a sleep mechanism in the simulation.

By fixing the amount of time given to execute the search, and by changing the size of the arrays, we could have enough time to search each location thoroughly. Alternatively,

if the time is insufficient, the simulation would have the task of distributing the time carefully.

When the time allocated to search both locations is adequate to go through all the slots in both of them, or at least through one of the locations completely, the need to learn the target distribution is not that important. Once the time allowed is less, and it is not sufficient to completely search one of the two locations, it is essential that we apply a strategy that estimates the target distribution to the search. This conclusion is fully supported by the simulation results. For every simulation, the sizes of the locations are changed, and the target is randomly hidden in one of these locations. The target is searched for by using a sequential search that is not constrained by the target distribution, and also searched for by utilizing another search that depends on the target distribution. In the first scenario, the algorithm would continue searching each location slot by slot until all the time allocated is consumed. Depending on how large the location was, the algorithm might have enough time to go through one of the locations, search it thoroughly, and then move to the next location. In the second scenario, the simulation would assign a probability to each location, and using the results of [ZO97], it would allocate the time based on the *assumed* target distribution.

With locations of smaller size, the search that did not involve time allocation yielded good results. Yet these results deteriorated with the increase of the size of the location. As for the search based on the target distribution, it consistently yielded better results.

The experiments that were conducted required various assumptions. First, we assume that both the locations have the same size. This gives the algorithm a fair chance to search in each location without one location demanding more time than the other, unless this is enforced through the target distribution. The second crucial assumption was that the time needed to search one slot in any of the locations was one time unit of the total allocated time. This means that if the allocated time is 60 units, the algorithm would be able to search 60 slots, regardless of which location the slots were in.

Table 4.1 shows the different sizes used and the result of running the experiment 100 times. In each case both methods of search were applied. In the case of the search that “learns” using a target distribution, the algorithm would scan the range of possible distributions to get the distribution closest to the real distribution. The algorithm assigns probability assignments for the quantity ‘b’ (explained earlier) using a value of δ between consecutive values. In the simulations, δ is assumed to be 0.0001. This value yielded a very high accuracy, and was successful in obtaining a value close to the real target distribution.

Percent of allocated time to search one location	Successful searches using target distribution	Successful searches using sequential search
50%	100	100
60%	100	100
80%	100	100
100%	100	83
150%	64	55
200%	50	40
250%	35	27
300%	39	30
350%	25	20

Table 4.1: The results of searching in the two bins by allocation different amounts of time to each location. The results are presented for both the cases when the information in the target distribution is used and when it is not used.

The results reported in the table above are plotted in Figure 4.1. From the chart it is obvious that the search using a learned knowledge of the target distribution always renders a better result. This is mainly because the allocated search time is allocated in such a way as to dedicate more time to one of the locations based on the access probabilities, and the resulting time distributions explained earlier.

From the results we also see that knowledge of the target distribution is needed especially for the cases when the time allocated is inadequate. As the overall allocated time gets shorter, even the search using the target distribution starts to perform poorly. The reason for this is that the probability of *not* finding the object, although it is in the searched location, increases, because there is not enough time to search the location until the object is found. In other words, the algorithm could search in the location which contains the target, yet it does not find it, as it runs out of time before reaching to the exact location of the target in the location.

Figure 4.1 The plot showing the number of successful searches as the size of the locations increases.

4.4.2 Validity of target distribution

Before using the assumed target distribution in a search operation, it is essential to know how large the error between the assumed and the real target distribution is. Upper and lower bounds for the optimal search were derived in [ZO97], based on the assumption of an exponential detection function. These bounds were shown to satisfy:

$$1 - e^{-K/N} \leq P[f^*] \leq 1 - C_N e^{-K/N},$$

where $C_N = N[p(1)...p(N)]^{1/N}$, and $K > 0$ is the total cost constraint.

To verify these equations, the simulation was run for a very large number of times. Each time the assumed target distribution was tested to verify if the error fell within the appropriate boundaries.

The obtained results verified the claim. In all the cases, the error obtained fell within the expected range. These results are given in Table 4.2

Step Size ' δ ' Log (n)	Percentage of inbound detection probability
-1	0%
-1.7	47%
-2	52%
-2.4	57%
-2.7	62%
-2.9	65%
-3	70%
-3.7	100%
-4	100%

Table 4.2: varying the step size and its impact on the upper and lower bounds of the optimal search.

4.4.3 Accuracy of the target distribution

The simulation program, as mentioned previously, chooses the assumed target distribution by scanning through the range of real numbers. Since it is impossible to try out every value in the range, it is necessary to devise a method that would allow scanning a reasonable number of values. The simplest way to do this is to scan the range sequentially, for a pre-specified value of δ , which is the step size between two consecutive values being scanned. It is clear that a smaller δ leads to a greater accuracy of the assumed target distribution due to the increased resolution of the search space. This, of course, implies a greater cost to search the space.

To verify this, the space was searched using various values of δ as shown in Table 4.3. With each value of δ the simulation program was run one hundred times, and the number of times that the program was able to find the hidden target were recorded.

Step Size ' δ ' $\text{Log}_2(n)$	Percentage of successful searches
2^{-1}	0%
2^{-2}	11%
2^{-3}	35%
2^{-4}	36%
2^{-5}	44%
2^{-6}	47%
2^{-7}	48%
2^{-8}	49%
2^{-9}	59%
2^{-10}	67%
2^{-11}	78%
2^{-12}	83%
2^{-13}	94%
2^{-14}	95%
2^{-15}	97%

Table 4.3: The percentage of successful searches as the steps size ' δ ' is decreased.

The different values used for δ varied logarithmically from 0.5 to 0.00003. The results were unpromising for larger values of δ . With smaller values of δ , the percentage of successful searches started to increase. After attaining a certain upper bound, when $\delta=2^{-9}$, the values remained satisfactorily high. Also, after this threshold, the percentages of successful searches, for different δ , reflected minimal change, indicating that it was not advantageous to continue decreasing the value of δ .

Figure 4.2 graphs the results of this experiment. From it, the accuracy of the search stands out when the set of assumed target distributions is extended to include more values.

Figure 4.2: The search accuracy as a function of the magnitude of δ . The scale of δ is logarithmic.

4.5 Conclusion

In this chapter we presented a simulation for the research results presented by Zhu and Oommen in [ZO97]. The simulation is by no means straightforward because, although the theoretical results are available, the process for the estimation of the probabilities was not suggested in [ZO97]. Our research managed to suggest a process by which the estimation of the target distribution could be done. Through our simulation, we succeeded in demonstrating the exact power of the algorithm introduced in [ZO97]. The results of the experiments prove both the validity of the algorithms, and also show how the search process can be executed in a real life case when the target probabilities are unknown. The simulation software also provides a good testing environment that would help to enhance the theoretical results, and test the potential new results for efficiency considerations.

The simulation provides the searcher with a medium by which he can compare this method of search to other methods, and also to determine the efficiency and the accuracy

of any particular method. Section 4.4.1 contained a comparison between sequential searches that are not assisted with a strategy for learning the target distribution, and those which are. This comparison shows that the sequential search is handicapped when the search is limited by time. It demonstrates that the shorter the time allocated for the search, the harder it is to locate the target. Thus the issue of prioritizing the search and allocating the time based on the target distribution becomes crucial. Indeed, since the target distribution is unknown in real life scenarios, the power of this algorithm to glean information about it, and thus direct the search, becomes evident.

In Section 4.4.3, we showed that the larger the set of values that the searcher can use as assumed target distributions, the more likely it is for him to converge to a “good” target distribution that is close to the real value. Of course, in turn, a learned target distribution close to the real target distribution implies a better chance in finding the target. This large set of values come with a price though, which is the time consumed to consider the values in the set as potential target distributions. The reason for this increased cost in the time is because the values are checked in sequential manner. By attempting a different approach other than a sequential scanning process, we intend to find the most likely target distribution in a faster time, and thus obtain an even more accurate convergence. This is the novel contribution of the next chapter.

Chapter 5: OPTIMAL SEARCH USING TERTIARY SEARCH

5.1 Introduction

In the previous chapter, we presented a simulation to the algorithm presented by Zhu and Oommen. In their research published in [ZO97], they had presented a novel algorithm by which a searcher can locate a hidden target when the target distribution is unknown. The main purpose of the simulation of the previous chapter was to acquire results from practical scenarios and to compare them to the findings of the formal mathematical research in [ZO97]. Also the simulation could be used as a testing ground for enhancements to the algorithm. In this chapter we enhance the simulation to the algorithm with the goal of trying to make it faster.

This was achieved by complementing the algorithm presented in [ZO97] with another algorithm, which was developed in a completely different context, by Raghunath and Oommen. The later algorithm is an intelligent tertiary search algorithm, modeled in such a way so as to use learning automata hierarchically, which would learn from its past experience and therefore be able to reach the parameter searched for in a shorter time.

5.2 Learning Automata and Tertiary search

Before going into the details of how the two algorithms are integrated, we briefly review the tertiary search algorithm, assuming that the reader is reasonably aware of the concepts of learning automata explained in Chapter 3.

As mentioned in an earlier chapter, Oommen and Raghunath considered the fundamental problem of searching for an unknown value (parameter) in a bounded domain. Their results were quite impressive. Their research produced a novel scheme to stochastically locate a point when the interaction with the environment is used. Although the research has importance in its own right, it also proves to be very useful in other domains such as in solving optimization problems, and also, as we shall see, in yielding an enhanced solution to our problem of searching when the target distribution is unknown.

To briefly review the problem, in the adaptive tertiary search algorithm, the aim of the solution is to determine a point that is being searched for in a closed interval. The interval is first subdivided into three equal subintervals. The point of interest is assumed to be the midpoint of each of the subintervals. To decide if this choice is correct or not, the search algorithm consults the environment which serves as an Oracle. The response of the environment is not guaranteed to be correct, thus rendering the problem to be a stochastic search problem. Based on the response that the environment yields for each interval, the algorithm decides whether to continue probing for the required value in the left or the right halves of that interval. The collective response from the three intervals is pruned to

yield a more accurate decision on which interval is more likely to have the target, and on which two of the three subintervals are to be disregarded. In some cases it could happen that only one interval is disregarded, and in this case the remaining two intervals are merged to result into *one* interval for the next iteration. Regardless of how many intervals are discarded, the resulting interval will be subdivided again into three subintervals, and the whole process is repeated again. The process is recursively repeated until the size of the final interval reaches a user-defined minimum, where the parameter searched for is determined.

5.3 Using the Tertiary Search To Locate The Target Distribution

The power of the adaptive tertiary search scheme is quite evident and has been demonstrated in [OR98]. Quite astonishingly, it can also be useful in our present research. During our research for creating a simulation strategy by which the searcher could estimate the unknown target distribution in a search process, it became clear that there were some aspects of the underlying simulation that needed improvement. The most important and vital aspect was the main process of estimating the parameter of the unknown target distribution. Clearly, a sequential search was both time consuming and slow. This forced us to investigate whether there was some other method that would be both faster and more efficient while searching for the *assumed* target distribution.

In the last section (and in Section 3.5.2), we demonstrated that locating a point using the adaptive tertiary search algorithm achieves remarkable results, primarily because it

avoided scanning the entire interval. In fact, the interval is continuously subdivided into three subintervals ensuring that the subsequent search is done on a reduced domain.

In the algorithm outlined in Chapter 4, the assumed probability is acquired by scanning through the unit interval, and selecting values from the interval sequentially. This value was used to create a test scenario to determine if the detection probability for the search plan was maximized at the chosen value. Since the interval searched is the set of real numbers between 0 and 1, an exhaustive search through the interval is impossible. To circumvent this, we resorted to a resolution step-size, δ , which was the width of the subintervals. To achieve a desirable accuracy, the value of δ had to be correspondingly reduced, and this implied more selection, and therefore, more computational time.

Using the adaptive tertiary search, the interval does not have to be scanned sequentially. Rather, the entire interval will be traversed faster, as it will be visited in series of probes. Before an interval is visited, it would have been reduced by at least one third of its previous size. Moreover, the traversal of the interval at each time is not done by examining all the values in the interval, because the algorithm selects a random point and tests it for optimality. If the point proves to be a good choice, then the algorithm proceeds in its direction of hoping to find an even better selection. If, on the other hand, it is not a good selection, the algorithm reverses the direction of the scan and prunes the interval. The advantage of the strategy will soon be clear.

5.3.1 Enhancing The Search by Adaptive Tertiary Search

Since the target distribution is not known, and we have to assume a target distribution, we would like this assumed value to be as close to the real value as possible. The assumed target distribution, based on which the search is conducted, should maximize the detection probability for the search plan.

In order to search for the optimal value in $[0,1]$, we first, at the topmost level, assume that the interval of interest is the entire domain. This interval is passed to the Continuous Pont Location Adaptive Tertiary Search (CPLATS) algorithm explained earlier, which explores the interval for the parameter. The exploration would involve a series of guesses, each one stochastically more focused than the previous one. Thus, since the interval is the entire domain $[0,1]$, the algorithm would search for the estimated target distribution in this interval. CPLATS would proceed by dividing the given search interval into 3 equal parts. The middle point of each interval is chosen to be the prospective assumed target distribution parameters. Each subinterval is investigated, independently, by an L_{RI} learning automata, which as explained earlier is an ϵ -optimal fast-converging two-action learning automaton.

The two potential actions to be chosen by the learning automata correspond to choosing a point in either the left or right half of the interval under investigation. Depending on the choice of the left or right movement, a random point would be chosen from the appropriate part of the interval. Since the main task of the algorithm is to determine the optimal value of the assumed target distribution, the mechanism would be assisted by

having responses from an intelligent “environment”, which is capable of informing it whether the selected point is closer to optimality or not. The environment, (Oracle), would use the following expression obtained from [ZO97] to judge the quality of the selected point:

$$P[f_b^*] = 1 - \frac{a + b - 2ab}{\sqrt{b(1-b)}} e^{-K/2}.$$

If the current selected point maximizes this expression compared to the previous points that were selected in earlier epochs, the action taken is considered a “correct” one and the environment rewards it. Otherwise the action is deemed to be “faulty” and it is penalized. An important issue to be noted here is that the probability of receiving an intelligent response is $p > 0.5$, which reflects on the “effectiveness” of the environment. Thus, whenever a point is selected from the interval, the environment could correctly suggest that it is a more optimal value of the target distribution with probability p , and yet simultaneously incorrectly brand the point as being less optimal with a probability $(1-p)$.

By repeating this process a number of times, the environment creates a probability vector that would reflect whether the point sought for is in the right or the left half of the interval. Because the environment gives its response erroneously, it is easy to see that we are encountering a stochastic point location problem.

Based on the results obtained from searching the three subintervals independently, the subsequent pruning decision is made based on a set of rules outlined in Table 3.2. The resulting subinterval is either one of the previous three subintervals, or is obtained as the

union of two adjacent ones. As mentioned earlier, in all cases the original interval is reduced by at least one third of its original size, resulting in a great enhancement to the speed of search.

O^1	O^2	O^3	New Subinterval
<i>Inside</i>	<i>Left</i>	<i>Left</i>	Δ^1
<i>Left</i>	<i>Left</i>	<i>Left</i>	Δ^1
<i>Right</i>	<i>Inside</i>	<i>Left</i>	Δ^2
<i>Right</i>	<i>Left</i>	<i>Left</i>	$\Delta^1 \cup \Delta^2$
<i>Right</i>	<i>Right</i>	<i>Inside</i>	Δ^3
<i>Right</i>	<i>Right</i>	<i>Left</i>	$\Delta^2 \cup \Delta^3$
<i>Right</i>	<i>Right</i>	<i>Right</i>	Δ^3

Table 5.1: Decision table to prune the search space based on the decision of the individual automata

The new interval would then be passed to the CPLATS process, whence the search for optimal value for the parameter of assumed target distribution continues. Since the search proceeds through the same steps as the search in the parent procedure, it will also be subdivided into three parts, and the environment would evaluate the selection of the mid points so as to cause a further subdivision of the interval.

The process continues until the subdivision of the current interval results in an interval whose size is smaller than the user-defined resolution. Once this level is reached, the mid point of the final interval is reported to be the optimal value for the assumed target distribution.

Once the simulation has chosen the optimal assumed probability, it would then use it to determine the time allocations for searching through the bins. It is easy to see that this strategy for search would be faster, and more likely to succeed, than searching sequentially through all the bins for the hidden target. Of course, there is still a probability that the target would not be found even though the simulation could be searching the location that has the target. The reason for this is that the target could be in the location, but the search time allocated would not be adequate for the process to locate the object. The pseudo-code for the search algorithm follows.

5.4 Pseudo-code Using Adaptive Tertiary Search

Algorithm Simulate(K)

Input: This algorithm would simulate the search for an object in one of two bins with unknown target distribution. The search has to be completed within a specific total time, K .

Method

Begin

/* simulating the existence of a target to be searched for */
target $\leftarrow$ CreateTarget()

/* simulating two different locations at which the target can be present */
locationA $\leftarrow$ CreateLocation(K)
locationB $\leftarrow$ CreateLocation(K)

/*Probability of 'a', which is the real probability of finding the target in location A*/
realProbbability $\leftarrow$ GenerateRealProbability()

/*hide the target in one of the locations*/
HideTarget(target)

/*Obtain the assumed probability by getting the maximum Detection function*/
assumedProbbability $\leftarrow$ AssumeProbabilityUsingLA(K , realProbbability)

/* Use the assumed probability to search for the target in the more */
targetFound $\leftarrow$ PerformSearch(assumedProbability, K)

End Algorithm Simulate

Procedure AssumeProbabilityUsingLA ()

Method

Begin-

```

StartOfInterval  $\leftarrow$  0
EndOfInterval  $\leftarrow$  1
Resolution  $\leftarrow$  0.01 /* Size of smallest signifact interval containing  $\lambda^*$  */

 $\Delta \leftarrow$  CreateInterval(StartOfInterval, EndOfInterval, Resolution)

/* returns the estimate of  $\lambda^*$ . It is derived as the mid-point of the final search interval*/
 $\lambda^* \leftarrow$  CPL_ATS( $\Delta$ ) /*For the CPL_ATS pseudo-code, see Section 3.5.2.1.1*/
assumedProbability  $\leftarrow$   $\lambda^*$ 
Return assumedProbability
End Procedure AssumeProbabilityUsingLA

```

Procedure GetFeedback(k, x, x_0)

Input: k is action to be taken (move left or right). x is the point is a random point in the interval.

x_0 is the mid point of the current interval

Output: returns the feedback on the action passed. Feedback is either a penalty or a reward

Method

Begin

```

 $\beta \leftarrow$  1 /*feedback of 1 is penalty for wrong action*/
 $\lambda^* \leftarrow$   $1 - \frac{a+x-2ax}{\sqrt{x(1-x)}} e^{-K/2}$  /*get detection probability based on random point*/
 $\lambda_0 \leftarrow$   $1 - \frac{a+x_0-2ax_0}{\sqrt{x_0(1-x_0)}} e^{-K/2}$  /*get detection probability based on interval mid point*/
if(  $k = 0 \wedge \lambda_0 > \lambda^*$  ) /*action 0 is to move to left*/
     $\beta \leftarrow$  0
if(  $k = 1 \wedge \lambda_0 > \lambda^*$  ) /*action 1 is to move to the right*/
     $\beta \leftarrow$  0
Return  $\beta$ 

```

End Procedure GetFeedback

5.5 Experimental Results of CPLATS in Optimal Search

To demonstrate the power of the research and the use of CPLATS in optimal search, numerous experiments were conducted using a variety of environments and parameters. Among the issues that we first confirmed, was the fact that the search for the target, which utilizes the target distribution that the learning automata guessed, performs better than the method using a sequential search strategy. We also had to validate that the values that are learned by the learning automata are verified to be within the proposed valid bounds of a target distribution. Once the tertiary search is shown to be capable of

estimating the target distribution accordingly, as in the case of the enumeration method, we directed our attention towards comparing the efficiency of both algorithms.

5.5.1 Effect Of the Resolution On The Search

First of all, it is important that we demonstrate that the tertiary search method helps in converging to the target distribution. Observe that this was one of the facts established for the exhaustive enumeration scheme. To show this, we conducted the same set of experiments that was used in the previous chapter.

To conduct this experiment, we assume that there are two identical bins, with the target hidden in one of them. A search for the target is conducted in the two bins. In the first instance, this search is conducted using a sequential search within each slot in the two bins, where the time allocated is determined by using a target distribution learnt by the tertiary search method. The search process was repeated a number of times, for varying location sizes. Also, as in the previous chapter, we assume that it takes one unit of time to process each slot in the location. If we fix the total allocated time and start increasing the size of the bins, we are effectively reducing the chances of locating the target in the same allocated time. However, if the search utilizes a target distribution, which would be guessed by the tertiary search method, this would increase the chances of finding the hidden target.

We initiated the testing by using a small location size. In this case only half of the allocated time was required to check all the slots in that location. The results obtained, based on this location size, were satisfactory for both scenarios when the information of

the target distribution was specifically used, or when searching sequentially. The results remained acceptable until the size of the bins was large enough that it would require at least twice the time allocated to search a single location thoroughly. At this point the results of the sequential search algorithm started to deteriorate, and the search algorithm that utilized the target distribution showed a superior performance. After this, with the increase in the location size, even the search conducted using the learnt target distribution started to deteriorate. The exact sizes and the ratios of successful searches are tabulated in Table 5.2.

Percent of allocated time to search one location	Successful searches using target distribution	Successful searches using sequential search
50%	100	100
60%	100	100
80%	100	100
100%	100	86
150%	71	66
200%	55	40
250%	55	41
300%	52	38
350%	57	45

Table 5.2: The results of searching in the two bins by allocating different amounts of time to each location. The results are presented for both the cases when the information in the target distribution is used, and when it is not.

The above table is plotted in Figure 5.1. From the figure we notice that using the target distribution during the search process always renders a better result, around thirty percent more, than searching sequentially, indicating that the tertiary search method is efficient in guessing the unknown target distribution.

Figure 5.1: Graph showing the relationship between the size of the location and the probability of succeeding in finding the target.

5.5.2 Validity Of The Target Distribution

In the previous chapter we checked for the validity of the target distribution determined by exhaustive enumeration algorithm. Using the same upper and lower bounds for the optimal search that were derived in [ZO97], we attempted to perform the same verification for the target distribution when using the tertiary search method. Note that those bounds are based on the assumption of an exponential detection function and were shown to satisfy:

$$1 - e^{-K/N} \leq P[f^*] \leq 1 - C_N e^{-K/N},$$

where $C_N = N[p(1)...p(N)]^{1/N}$, and $K > 0$ is the total cost constraint.

By running the simulation we permit the tertiary search method to determine a target distribution, which would be tested using the bounds mentioned earlier. If the target distribution lies within those bounds it is reckoned to be valid. To ensure an accurate result, when working with a stochastic environment, this process was repeated several

times. In all trials the result indicated that the estimated target distribution, which resulted from using the tertiary search algorithm, fell within the correct bounds.

5.5.3 Tertiary Search Versus Exhaustive Enumeration

Since both algorithms, the tertiary search and the exhaustive enumeration, proved to produce valid target distributions, and since the estimated values were useful in the actual search process, the next phase was to see which of the two methods was more efficient. Efficiency in this case was determined in terms of utilizing lesser resources, and the most critical resource in a search problem would be the time taken to do the search. Therefore, to judge which of the two algorithms is more efficient we have to decide which one of them consumes less time to get equivalently accurate values of the estimated target distribution.

The factors used by both algorithms are different. The first factor determines how each of the two algorithms increases the accuracy for the estimated target distribution. For the exhaustive enumeration method, this is accomplished by determining the step size, δ . On the other hand, to increase the accuracy of the estimated target distribution for the tertiary search algorithm, we have to vary the resolution of the interval searched. As mentioned earlier, the tertiary search takes an interval of real numbers and starts subdividing it into three equal partitions. This task will continue as long as the resulting intervals are bigger than a certain resolution. To get a better accuracy, the algorithm should search the interval until it is small enough, and therefore contains fewer values to choose from. This, in turn, increases the probability of choosing the correct value. For the interval to be small, the resolution used should be minimized. Again, as in the exhaustive enumeration

case, if the resolution is minimized to increase accuracy, the speed of the algorithm is compromised.

Tests were done to compare the results of determining the target distribution when using both the exhaustive enumeration and the tertiary search method. In each of those tests the accuracy of the resulting target distribution was different. The tests were in the range of hundreds.

The first test compares the two algorithms when the difference between the real and the estimated values of the target distribution is a user-defined constant. In our experiments this constant was set to be 0.1. For this “large” value of 0.1, the exhaustive enumeration method was much faster than the tertiary search method, since the exhaustive enumeration did not go through many iterations to reach the required target distribution. At the same time, the tertiary search still needed a longer time because of the overhead imposed by the scheme utilizing learning automata. On the other hand, the accuracy obtained was not satisfactory, since the difference between real and estimated values, 0.1, is really too large. To ensure superior results the experiment was repeated to yield second decimal accuracy. In this case, the time taken by the exhaustive enumeration increased by a factor of 10 times its previous value. The tertiary search scheme, on the other hand, took less than double the previously required time. Indeed, if the accuracy required was increased, the tertiary search proved to be even *more* superior to the exhaustive enumeration scheme. This is even more obvious when the difference between the estimated target distribution and the original is set to the third decimal point.

The results of the experiments are tabulated in the following tables showing the time taken by each algorithm, in milliseconds, to reach the estimated target distribution that is closest to the real one.

Exhaustive Enumeration Time (units of time)	Tertiary Search Time (units of time)
90	250
90	240
90	241
90	240
90	240
91	241
90	240
90	250
90	241
90	240

Table 5.3: A table showing the time required for Exhaustive Enumeration method and Tertiary Search scheme accuracy was set to be 0.1. These are the results obtained for 10 independent experiments.

Table 5.4 shows the time by the two algorithms when the difference between the estimated and the real target distribution is 0.01. The increase in time taken by the exhaustive enumeration method is much greater than the increase in time taken by the tertiary search.

Exhaustive Enumeration Time (units of time)	Tertiary Search Time (units of time)
991	390
992	391
991	390
992	391
991	400
1212	391
1161	391
992	390
991	391
992	390

Table 5.4: A table showing the time required for Exhaustive Enumeration method and Tertiary Search scheme accuracy was set to be 0.01. These are the results obtained for 10 independent experiments.

Exhaustive Enumeration Time (units of time)	Tertiary Search Time (units of time)
10294	450
10275	461
10255	451
10175	450
10094	451
10075	461
10024	450
10085	451
10164	451
10295	450

Table 5.5: A table showing the time required for Exhaustive Enumeration method and Tertiary Search scheme accuracy was set to be 0.001. These are the results obtained for 10 independent experiments.

Table 5.6 shows the time by the two algorithms when the difference between the estimated and the real target distribution is 0.0001. Any increased accuracy above this point causes minimal changes in the time taken by the tertiary search method, while the exhaustive enumeration method requires prohibitively large time.

Exhaustive Enumeration Time (units of time)	Tertiary Search Time (units of time)
101677	1322
101616	1332
102006	1312
102638	992
118380	1602
96529	1412
12178	1292
11646	1502
11877	1322
10636	1742

Table 5.6: A table showing the time required for Exhaustive Enumeration method and Tertiary Search scheme accuracy was set to be 0.0001. These are the results obtained for 10 independent experiments.

The following figure shows how the average time increases for both methods. Note that the scale of the accuracy is logarithmic.

Figure 5.2 Time taken by exhaustive enumeration method and tertiary search method with increasing accuracy.

5.6 Conclusion

In Chapter 4 we discussed the work of Zhu and Oommen and showed how we could develop an algorithm that would be capable of searching for a hidden target, in one of two bins, while the target distribution for that target is not known. In that chapter we proposed a scheme by which we could simulate the algorithm of [ZO97], and be able to accurately determine the unknown target distribution. The method discussed required an exhaustive enumeration of the parameter space.

In this chapter, we proposed a new simulation scheme based on the innovative results of Raghunath and Oommen [OR98] earlier discussed in Chapter 3. This algorithm, which used an adaptive tertiary search method, divides the searchable interval into three subintervals and tests the mid point of each subinterval for being the optimal target distribution sought for. Using the result of the tests of the three mid points, a new interval is created, which is either a single interval or the union of two of the original subintervals. The process is repeated recursively until the final interval is smaller than a user-defined width.

In this chapter we have studied the latter simulation strategy and reported the results of conducting several experiments that, utilizing the adaptive tertiary search, yielded the optimal target distribution. The results support our claim. The results also show that for smaller accuracies the exhaustive enumeration method is faster than the tertiary search. However for increased accuracies, the tertiary search has a phenomenal performance and

is significantly (i.e., many orders of magnitude) faster than the exhaustive enumeration method.

Not only do the results obtained support our hypothesis, but we also believe that these simulation methods can be used in many real life search scenarios, such as those which involve searching on the World Wide Web.

Chapter 6 : CONCLUSION

6.1 Introduction

In this thesis we have studied some interesting problems in optimal search theory and contributed some new and innovative results to that field. Our research was concerned with stationary objects searched for by a single searcher in a continuous space when the target distribution is not known. The contents of the thesis can be divided into two parts. In the first part we have considered the problem of locating a hidden target in one of N locations when the target distribution is not known. Although Zhu and Oommen pioneered the study to this topic in their work earlier, we complement their research with a real-life addition. Indeed, we have outlined a process by which the searcher, who is using Zhu and Oommen's algorithm for searching with an unknown distribution, can infer the credibility of an assumed target distribution, which is the main phenomenon that the Zhu-Oommen algorithm depends on. In spite of their contribution, they did not introduce a procedure by which the latter could be estimated in a real life setting.

In the second part of the thesis we developed another method to estimate the unknown target distribution. This approach attempts to produce results as accurate as the first

method, and yet, at the same time, it reduces the time taken to reach the required value. In this way the overall search time would also be reduced.

The results that we worked on were for the case when the object was hidden in one of the two hiding places, (not N as in the original research). Although this was done for the simplicity of the simulation, we believe that the results are applicable for the general case. These are avenues for future research.

Chapters 2 and 3 of the thesis were primarily of a review nature. In Chapter 2 we surveyed the literature that discussed the results derived for the theory of search for a stationary target. The chapter was intended to serve as a background overview to the topic of optimal search with known target distributions, and to lay a basic foundation for the novel work done in this thesis. We concentrated on the early and leading research done in this field by Koopman who, for the first time, established a model for the theory of search. His work was the kernel for many other researches like the work done by Cooper, which we also reviewed. Cooper enhanced the results of Koopman, and solved similar problems even for the discrete space. We also submitted a review of the problem of sequential search, and gave an overview of the work done by Chew, who showed how one could achieve a minimal cost in such a search problem. Among the researchers who have extended the work of Koopman, was De Guenin who demonstrated how one could utilize a minimal search effort when using the continuous model. The minimal cost of search was also explained as part of reviewing the works of Staroverov, Gilbert, Onaga

and others. We also included a brief synopsis on the field of the whereabouts search problem.

Finally in Chapter 2 we also included the basis of this present study. In [Z96], Zhu studied the theory of search for stationary targets when the target distribution is unknown. He provided the upper and lower bounds for optimal strategies, and derived selection criteria for choosing target distributions when two or more cells are searched. In each case, Zhu also derived the error estimate.

In Chapter 3 we presented a literature review on learning automata and some of its applications. This review was done to pave the way for introducing the work done by Oommen and Raghunath in which they solved the stochastic search problem, which involved searching for a point on a line where the Oracle responded with inaccurate responses.

We presented a general overview of the Learning Automata field involving both fixed and variable structure automata. Examples for both of these families were given, with a special emphasis on the ones used in the algorithm presented in [OR98], and in particular the ϵ -optimal L_{RI} scheme.

As mentioned earlier, our research focuses on the research done on continuous point location. This novel work was introduced to solve the problem of searching stochastically for the location of a point on a line. The research was first done by Oommen who

suggested that the search space could be rendered discrete, and he achieved this by subdividing the space into a number of discrete partitions. After the space is discretized, he presented an algorithm that would traverse the partitions thus created. The traversal would be guided by the Environment (Oracle), which is, by definition, inaccurate. The traversal is done using a continuous learning automaton to solve the problem of stochastically searching for the point on a line.

The algorithm introduced by Oommen and Raghunath was then explained in detail. We showed how the searching algorithm moved in an interval, and attempted to locate the unknown point in this interval by recursively splitting it into three subintervals by utilizing the inaccurate responses from the Oracle. The experimental results that were reported in their work [OR98] were truly conclusive, and the convergence was remarkable in all experiments.

The novel and pioneering results presented in this thesis were discussed in Chapters 4 and 5 respectively. In Chapter 4 we started presenting our main topic, which was the attempt to provide a way to estimate the assumed target distribution that Zhu and Oommen mentioned in their work in [ZO97]. Since they did not give a specific technique by which this value could be generated in real life, we decided to complete that phase of the research in this thesis. The first attempt to achieve this goal was to invoke a method that used an exhaustive enumeration technique. The essence of this method is to iterate through all possible values for the target distribution while testing each value for optimality. This technique was used and the results were verified to be accurate. Indeed,

it was found that the exhaustive enumeration scheme could be very accurate in estimating the target distribution provided that the step size, which is the difference between each two consecutive values in the range, is small enough. A small step size implies that a large number of values would be tested, thus increasing the probability of finding the optimal target distribution. The main draw back of this method was the amount of time it consumed.

Since we opt for efficiency, we introduced another method to guess the unknown target distribution. This was done in Chapter 5, where we presented an algorithm that would utilize a tertiary search to reach its goal of determining the optimal target distribution. This algorithm was based on the work done in [OR98] and it initially deals with the entire search space and subdivides it into three equal intervals in each epoch. The algorithm would then search each interval independently with the guidance of the Oracle and the result of the search of the three intervals is used to decide on which interval to concentrate on. This is done recursively until finally the algorithm converges on the value sought after. The value produced by this algorithm was also tested and was shown to be optimal. It was compared with the values obtained from the first sequential search method. The accuracy was the same, but the second method proved to be faster. These results successfully demonstrated that we could efficiently and accurately guess the assumed target distribution, which would be further applied to the algorithm presented in [ZO97], thus adding more value to the solution to this open problem.

6.2 Future Research

As with any field of research there are several areas that may be considered for modification and/or improvement. In [ZO97], the research done by Zhu and Oommen covered two cases for searching. The first, and the simpler, case is the one where the target is hidden in one of two locations (bins). The second case is more generic and involves hiding the target in any of N hiding locations. In the research at hand, the simple case was chosen, and therefore in all the procedures, two hiding places were assumed. For future research in this area, our process could be applied to the case of multiple bins to, hopefully, prove that the generic case would still yield results similar to the special case of searching in only two bins.

REFERENCES

- [A64a] Arkin, V.I., "A Problem of Optimum Distribution of Search Effort", *Theory of Probability and its Application*, Vol.9, pp. 159-160, 1964.
- [A64b] Arkin, V.I., "Uniformly Optimal Strategies in Search Problems", *Theory of Probability and its Application*, Vol.9, pp. 674-680, 1964.
- [A85] Alpern, S., "Search for Point in Interval, with High-Low Feedback", *Mathematical Proceedings of the Cambridge Philosophy Society*, Vol. 98, pp. 569-579, 1985.
- [B57] Bellman, R. S., "Dynamic Programming", Princeton Univ. Press, Princeton, 1957.
- [B64] Beck, Al, "More on the Linear Search Problem", *Israel Journal of Mathematics*, Vol. 2, pp. 221-228, 1964.
- [B80] Brown, S.S., "Optimal Search for Moving Target in discrete Time and Space", *Operations Research*, vol. 28, pp. 1275-1289, 1980.
- [BCR88] Baeza-Yates, R. A., Culberson, J.C., and Rawlins G. J. E., "Searching with uncertainty," in *Proc. Scandinavian Workshop Algorithms Theory*, 1988, SWAT 88, LNCS 318, p. 176-189.
- [BS92] Baeza-Yates, R. A., and Schott, R., "Parallel searching in the plane," in *Proc.1992 Int. Conf. Chilean Computer Society, IC-SCCC'92*, 1992, pp. 269-279.
- [C67] Chew, M.C., "A Sequential Search Procedure", *The Annals of Mathematical Statistics*, vol. 38, pp. 494-502, 1967.

- [CC58] Charness, Al and Cooper, W.W., "The theory of search: Optimum distribution of search effort", *Management Science*, vol. 5, pp. 44-50, 1958.
- [D61] De Guenin, J., "Optimum Distribution of Effort: An Extension of the Koopman Basic Theory", *Operations Research*, Vol. 9, pp. 1-7, 1961.
- [F65] Franck, W., "An Optimal Search Problem", *SIAM Review*, Vol. 7, pp. 503-512, 1965.
- [G59] Gilbert, "Optimal Search Strategies", *SIAM J. Appl. Math.*, Vol. 7, pp. 413-424, 1959.
- [K46] Koopman, B. O., "Search and Screening," OEG Report, no. 56, Center for Naval Analysis, Rosslyn, Va., USA, 1946.
- [K56a] Koopman, B. O., "The Theory of Search, part I: Kinetic Bases", *Operation Research*, Vol. 4, pp. 324-346, 1956.
- [K56b] Koopman, B. O., "The Theory of Search, part II: Target Detection", *Operation Research*, Vol. 4, pp. 503-531, 1956.
- [K57] Koopman, B. O., "The Theory of Search, part III: The Optimum Distribution of Searching Effort", *Operation Research*, Vol. 5, pp. 613-626, 1957.
- [K71] Kadane, J. B., "Optimal Whereabouts Search", *Operations Research*, Vol. 19, pp. 894-904, 1971.
- [KS83] Kadane, J. B. and Simon, H. A., "Correction to Optimal Strategies for a Class of Constrained Sequential Problems", *The Annals of Statistics*, Vol. II, 346, 1983.
- [L81] Lakshmivarahan, S., *Learning Algorithms Theory and Applications*. Berlin, Germany: Springer-Verlag, 1981.

- [NT89] Narendra, K. S., and Thathachar, M. A. L., *Learning Automata*. Englewood Cliffs, NJ: Prentice-Hall, 1989.
- [O71] Onaga, "Optimal Search for Detecting a Hidden Object", *SIAM J. Appl. Math.*, Vol. 20, pp. 298-318, 1971.
- [O97] Oommen, B. J., "Stochastic searching on the line and its applications to parameter learning in nonlinear optimization," *IEEE Trans. Syst., Man, Cybern.*, vol. 27, pp. 733-739, 1997.
- [OR98] Oommen, B. J., and Raghunath, G., "Automata learning and intelligent tertiary searching for stochastic point location," in *Proc.1992 Int. Conf. Chilean Computer Society, IC-SCCC'92*, 1992, pp. 269-279.
- [P89] Plec, A. "Searching with known Error Probability", *Theoretical Computer Science*, Vol. 63, pp.185-202, 1989.
- [S63] Staroverov, O. V. "On a Searching Problem", *Theory of probability and its applications*, Vol. 8, pp. 184-187, 1963.
- [S75] Stone, L. D., "Theory of Optimal Search", *Mathematics in Science and Engineering*, Vol. 118, Academic Press, New York, 1975.
- [W81] Wegener, I., "The Construction of an Optimal Distribution of Search Effort", *Naval Research Logistics Quarterly*, Vol. 28, no. 4, pp. 533-543, 1981.
- [W82] Wegener, I., "The Discrete Search Problem And the Construction of Optimal Allocations", *Naval Research Logistics Quarterly*, Vol. 29, no. 2, pp. 203-212, 1982.
- [Z96] Zhu, Q., "Optimal Search in Discrete and Continuous Spaces," Master's thesis, School of Computer Science, Carleton University, Ottawa, Ontario, 1996.

- [ZO97] Zhu, Q. and Oommen, B.J., "On The Optimal Search Problem: The Case when the Target Distribution is Unknown", *Proceedings of the Seventeenth International Conference of the Chilean Computer Science Society*, Valparaiso, Chile, November 1997, pp. 268-277