Contributions to the Theory and Applications of Genetic Algorithms

Mauro Manela

Thesis submitted for the degree of
Doctor of Philosophy
of the University of London.

December 1993

Department of Computer Science
University College London

ABSTRACT

The thesis consists of a series of studies around the central theme of Genetic Algorithms (GAs), with particular emphasis on its application in the domain of distributed artificial intelligence (DAI).

The theoretical part of the thesis starts from results derived by A. D. Bethke for binary alphabets, and from the suggestion that *epistasis variance* (borrowed from population genetics) is a reasonable measure to judge the suitability of a given representation for exploration by genetic search. It is suggested that these seemingly unrelated topics can be tied together with the help of abstract group theory. This provides a general framework within which further theoretical generalisations may be derived. Within this framework results derived for binary strings are extended to any non-binary alphabet. Also, issues of representation and different encodings are discussed and studied, and suggestions about how to modify simple genetic algorithms are reported. By means of a practical application to fermentation-process data, it is illustrated that a suitable change of representation can overcome the problem of dealing with search spaces that have non-simple boundaries determined by constraints. Also, it is demonstrated that optimisation over parameters that are not obviously suitable for "classical" treatments of optimisation can be conducted simultaneously with exploitation of the classical variables.

The design of autonomous agents and a framework to discuss basic design issues within the scope of applicability of GAs are introduced. The usefulness of GAs as tools to help DAI designers is demonstrated by a systematic exploration of the domain of pursuit games, traditionally used as testbeds for comparing alternative architectures. A GA is used both to optimise low-level architectural features of agents and to help evaluate the quality of pursuit games as testbeds for DAI. As a consequence of this exploration, new features (e.g. "boredom", "listening rate") of potential value for general agent architectures are introduced.

At his elbow now his father halted, with a word to the wise: ...Work out a plan of action in your mind, dear son, don't let the prize slip through your fingers. Astuteness makes a forester, not brawn, and by astuteness on the open sea a helmsman holds a ship on the right course though roughed by winds.

The Iliad

To the memory of my father, **Bernardo Samuel Manela,**

and to all the lessons he taught me that took me so long to understand.

ACKNOWLEDGEMENTS

Contrary to a widely held view, a PhD work depends on a good deal of collaboration, consultation and support. The thesis presented here is no exception and I consider myself particularly fortunate to have had the privilege to interact with several different individuals who contributed in one way or another to shape my current ideas on the subjects addressed in this work.

First and foremost I would like to thank my supervisor Professor J. A. Campbell for his encouragement and guidance received before and throughout the period under his supervision. His insights in several different areas of knowledge were invaluable and I am also thankful to him for having introduced me to the topic of distributed artificial intelligence and for being a continual source of encouragement as the work progressed.

Nina Thornhill of the Department of Electronic and Electrical Engineering deserves a special acknowledgment. I have learned a lot from her cooperation, constructive criticisms and her enthusiasm about using a novel technique to explore a well-defined problem which nonetheless has several important applications (and implications) in biochemical engineering.

Dr. Charles Leedham-Green from the Department of Mathematics at Queen Mary and Westfield College was a source of mathematical acumen in group theory and I gratefully acknowledge his timely comments, expert advice and inspiration. I was very honoured to meet Professor Cedric A. Smith from the Department of Genetics at University College London and Professor Philip Holgate from the Mathematics and Statistics Department at Birkbeck College. They both pointed me towards interesting topics in Genetics which I was completely unaware of. They also showed an enormous supply of patience and kindness in listening to my experiences and attempts to be more systematic in an area where the empirical material still outweighs the theoretical substance by a large amount. Dr. Mark L. Roberts from the Department of Mathematics at University College London was always ready to help me with sound comments and remarks and I have learned much from our conversations.

I am particularly grateful to Aggeliki Balou for numerous comments and suggestions on the written work. I would also like to acknowledge the expert assistance from Deborah Pope, Rae Harbird, Steve Davey and Jonathan Crompton from the

Systems Groups in the Computer Science Department at University College London who on so many occasions helped me get through my programming problems.

I would like to express my gratitude to CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) - Brazil - for the financial support without which this thesis could not have been accomplished.

Thanks also to my mother, Anita Manela, and family for their encouragement and support throughout the duration of the work. Finally, and most importantly, I thank my wife Mônica for her patience and assistance and for being so unfailingly understanding of the inevitable disruptions and stresses a project like this can cause on family life. If it were not for her caring help and endurance I would certainly have not succeeded.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1
Overview

1.1 Foreword

The work presented in this thesis consists of a series of studies around the central theme of Genetic Algorithms (GAs). The thesis can be divided into two distinct parts. The first one reviews and clarifies the theoretical tools often referred to in the GA literature, in terms of more fundamental concepts. This was motivated by the need to get further insights into the domains of applicability of GAs and to extend the analysis often applied to binary alphabets to other domains. A second motivation was to put the results in the GA literature in perspective with other results already known elsewhere, as a means of not only contributing new results unknown to the GA literature but also investigating a more fundamental and general formulation of the basic concepts. It was hoped that the ensuing formulation would lead to further progress in the mathematical treatment of the domains of applicability of GAs. The basic questions that this first part of the thesis investigates are:

- is there a simple index to gauge the difficulty a given problem poses for genetic search ?
- given a problem, can one find a suitable transformation (via a GA approach) on the parameter's encoding such that it makes the problem easy for a GA to explore?

My opinion of the achievements of this first part are mixed. On one hand it has succeeded in providing a general framework where several questions of interest in the literature can be addressed in an objective way. Among these are the extension of the known results of binary strings to any non-binary alphabet (the use of which is still being debated in the literature), the relation between epistasis variance and Walsh function analysis and some discussions of possible extensions of the basic algorithm and the difficulties of finding an adaptive approach for encoding the representation itself. On the other hand I could not provide any *new* fundamental insights beyond those already known in some way or another. (But the value of the work is in its clarification of some existing issues that needed clarification). As it stands, this first part looks somewhat pessimistic because most of the results stated here imply negative answers to the questions posed above. But they may also indicate the very nature of the difficulties one should encounter when trying to apply the techniques described to diverse problems - which is a contribution to improved understanding in an area in need of such understanding.

The second part of the thesis is more synthetic and tries to examine some applications which might profit from the power of genetic search. This was attempted not only because of the limitation of the results in the first part, but also due to my recognition of the difficulties in extending that analysis to address more fundamental questions related to the dynamical behaviour of GAs. However, the reason behind these attempts is more general than a quick browse through them may reveal. One can distinguish a gradation of problems in terms of the effectiveness with which they might be treated by a genetic algorithm approach. The simplest problems are those that require optimisation of a function given in "closed form" by combinations of simple expressions such as $x^2 + 1$, $\sqrt{y}$ or $\frac{x^2}{\sqrt{|y+z|+\sin(z)}}$. The most relevant fact here is that parametrisation of the search space is immediate and hence, at least in principle, a GA used as a function optimiser may offer an alternative when no other approach proves to be effective. At the other extreme stand problems for which no parametrisation may exist and hence that may be impossible to model via a GA approach. In between these extremes are those problems that do not appear to be amenable to a GA treatment and yet through careful modelling can harness at least some of the power of GAs as search algorithms. The work reported in this thesis concentrates on problems of this class. We focus on the use of GAs as experimental tools to assist the *design* of different systems as part of an exploratory process. The interest is the search for potential interesting design dimensions in domains for which no other approach can guarantee to make good design choices. I believe that Distributed Artificial Intelligence is one area that can profit significantly from this approach. But there is a problem, as one needs to face the traditional maladies associated with the use of GAs as function optimisers and at the same time build systems that achieve reasonably robust behaviour so that the conclusions from the experiments can be regarded consistent throughout a series of different scenarios.

One of the important aspects that distinguishes the work done here from mainstream applications usually found in the literature is the fact that in the latter case the function to be optimised (and the corresponding parameter set) is known in advance.

Hence, the application of a GA becomes an optimisation exercise where discussions of convergence to the optimum, and modifications of the basic algorithms, are central. In contrast, the problem considered here has no definite functional form to start with, nor is the parameter space specified from the outset. In fact, if it were so, much progress could have already been accomplished by traditional techniques. At the same time, the success of the experimental exploration conducted in this thesis may point to a new application area of GAs for which they can be used with reasonable performance. As ever more sophisticated systems are designed to deal with complex industrial needs, there is a real demand to examine their quality in providing good solutions in highly demanding applications. I believe the techniques discussed in this thesis can be extended to deal with some of these domains.

The organisation of the chapters follows basically the chronological steps in which the work was undertaken, and in this way can serve as an indicator of how the ideas presented developed and the difficulties I faced in their exploration. Hierarchically speaking though, the DAI application should be regarded as the one with greater significance and the main focus of the thesis.

1.2 Thesis overview

The central application of this thesis deals with the problem of how to best accomplish the design of autonomous agents using a genetic algorithm as a means of finding suitable testbeds for comparing alternative Distributed Artificial Intelligence (DAI) architectures. There are many motivations for attempting to apply natural metaphors (and in particular GAs) as adaptive techniques in DAI. Perhaps the most obvious (and possibly the most difficult to achieve) is the generation of agents that can adapt to a changing environment and react accordingly in much the same way that natural organisms seem to evolve. Understanding the phenomena that render this possible for natural organisms may provide important clues and directions of research in the design of more adaptive and powerful architectures. GAs have an appeal for their simplicity and elegance as algorithms. They also draw inspiration from the mechanisms of genetic variation and natural selection. Hence, at this stage they offer an interesting alternative with which to explore the above issues.

Secondly, in many circumstances design and optimality issues are intimately related. The choice of a particular architecture may depend on its capacity to deliver better performance than competing ones. In general, the design of a particular architecture depends on a large number of different and interacting factors (i.e. *dimensions*). Therefore, searching for the best mix of factors can be regarded as a search in a difficult high-dimensional space. Hence, the desirability for an algorithm that is able to discover good solutions for difficult high-dimensional problems. At the same time, the complexity of a given architecture and the environment in which it is embedded may demand a large amount of computation to assess the performance of a particular set of factors. Exercising all the possible alternatives may be an infeasible task. Thus the necessity of an algorithm that has the power to discover good solutions rapidly.

Thirdly, there is no established theoretical framework that can explicate the many possible approaches in dealing with agents' interactions. This has resulted in many distinct and sometimes competing theories of social interactions for which no clear winner has yet been established. A simple approach that is able to generate interesting experimental systems and results may offer theoreticians further insights for developing more refined theories of social interactions.

Additionally, it is not clear whether a DAI architectural idea developed and shown to be useful for a given project is **generally** good and useful, because of the difficulty of comparing results obtained in different environments. This may result in a "Tower of Babel" curse, in which different groups use different set of examples (which may be tailored to their particular architectures and devised solutions) to measure and compare the performance of their solutions. A common set of problems is therefore needed to assess and compare different approaches to the distribution of knowledge and information.

Now, one of the difficulties in devising a good set of problems (**testbeds**) is associated with the suitability of this set for exploration of the desired issues. In many circumstances a given set of problems is devised in an *ad hoc* manner and used to measure the performance of a proposed system. There are at least two main drawbacks with such an approach. The first and most obvious is related to the fact that a given problem may appear interesting in principle and that the proposed solution, once validated on it, may be applied to other different and difficult circumstances. The general literature is full of examples of such systems and their failure to yield the promised performance levels when confronted with more mundane situations. The second drawback is related to the fact that the problem by itself may not possess (contrary to the initial subjective expectations) enough complexity and that the results obtained could have been accomplished using a much simpler approach.

One response to these drawbacks is to justify, probably by theoretical means, the properties that a given set of problems should have in order to be used as a good testbed. In principle this is a difficult task, as an answer to it would convey with itself important explanations about the architectural issues it was intended to explore and settle in the first place. Another alternative is to search for an accepted minimal architecture and use its performance on the suite of problems as a baseline of comparison with more sophisticated solutions. A third alternative, bearing the previous idea in mind, is to find and parametrise a class of problems and search for a suitable subset within it that has the intended characteristics.

GAs have been applied successfully to many different problems, but there have been equally many others where they have failed badly to yield the expected performance levels for which they were devised in the first place. One reason for this effect, is related to the fact that (despite their simplicity and elegance) there is often little detailed understanding of why they have been successful or failed in a given problem. This has only been exacerbated by the fact that the GA literature seems to have embarked on its own independent path from other areas of knowledge. This may have obscured the relation between itself and other bodies of knowledge, and as a result may have hindered further progress in understanding the domains in which GAs can be applied successfully. Bridging this gap is a desirable issue, as it may point to already-established results that can be used in a straightforward way. As a consequence, further insights may be gained that are likely to carry over to more general contexts. Also, the limitations faced by a proposed theoretical approach may be envisaged in a more effective way, by reference to the already-established results.

Given the above, it is also important to note that even a limited theoretical exploration may be useful, as it may establish some guidelines for applying GAs to an ever wider range of problems for which their application may not be apparent from the outset.

The primary focus of the "applications" part of this thesis is the application of GAs to help the search for meaningful testbeds for DAI systems. In doing so, it examines all the issues discussed above and proposes a framework with which the design of general DAI systems can be approached using genetic search. Since to our knowledge such an attempt has never been tried, it is difficult to put it in perspective with other known methods. The results reported in this thesis, though, indicate in my opinion that much progress can be gained by the combination of these two seemingly unrelated themes.

1.3 Organisation

Chapter 2 discusses genetic algorithms as usually formulated. The discussion is fairly general and is intended to bring out the most salient aspects which will be addressed in later chapters. In particular, the question of what are the conditions that make a given problem most appropriate to genetic search.

Chapter 3 extends the discussion introduced in chapter 2 and puts the concepts developed in the GA literature in perspective with respect to already-established ones. In this way, it both "rediscovers" the results derived by Bethke [8] and presents them with respect to a more unified framework, namely of **Harmonic Analysis**. Several results unknown in the GA literature are presented here in a natural way, and alternative approaches explored elsewhere are criticised using this framework. In appendix A the results presented for binary alphabets are extended to any non-binary alphabet. In this way this appendix is intended to provide the theoretical tools with which to examine the still ongoing debate of which alphabet is more appropriate to a particular problem. It also furnishes a class of functions to test implementations of GAs that are specifically tailored to deal with alphabets of high cardinality.

Chapter 4 examines the concept of epistasis, which is often referred to in the literature but is difficult to quantify for use in an objective way. Use is made of the framework developed in previous chapters to address the concept of epistasis variance as suggested by Davidor [16]. It is shown that epistasis variance is not capable of complete discrimination between functions that are easy and difficult for genetic optimisation.

Chapter 5 examines several important aspects related to the "representation problem" and the importance of finding representations which allow schemata to capture regularities and correlations that can be explored successfully by a GA. It exploits the possibility of subjecting the representation itself to adaptation. The discussion is restricted to the set of linear transformations but may point to several problems to be faced when a similar scheme is attempted in general. It also presents computational experiments on an alternative approach based on ideas derived from genetics, which may point to future viable extensions of the basic algorithm.

Chapter 6 gives a fairly general introduction and discussion of distributed systems, with particular emphasis on the choice of meaningful testbeds for comparing alternative distributed architectures. The attention is focused on the **Pursuit Problem,** the currently most popular testbed for comparing alternative DAI systems. It also brings together the issues of designing a minimal DAI architecture and the application of genetic techniques for that purpose.

Chapter 7 explores the ideas proposed in chapter 6 with respect to the Pursuit Problem. Using the GA as a search algorithm for suitable architectures, it comprises a series of studies around the theme of searching systematically the space of test problems for suitable testbeds. In addition, by focusing attention on this class of games the chapter explores some basic dimensions in agents' architecture that are potentially valid for many different DAI applications. Within this activity, the issues introduced through the concept of boredom are the most novel. The chapter also discusses experiments conducted on the concepts of "listening rate", thresholds and attenuation of communication as additional simple mechanisms that provide further insights into the architecture of problem-solvers.

Chapter 8 illustrates the benefits one can obtain from the discipline of looking at non-standard dimensions for genetic optimisation suggested in chapter 7 in a more traditional domain for which the issue of optimisation is central. The application deals with the problem of fitting noisy data with spline functions.

Chapter 9 presents directions of future research on the work discussed in chapters 7 and 8. With regard to the DAI problem, it discusses the extension of the basic DAI architecture with features that were observed through the exploration of the test problems devised in chapter 7. I believe that they will contribute to further insights on how coordination and cooperation can arise among collections of different problem-solvers. The application considered in chapter 8 is put into perspective with respect to a larger class of important practical problems where the techniques explored in the thesis may be extended as a means to advance the understanding of these important domains.

1.4 Main contributions

The contributions of the thesis are distributed along several lines. First, it puts the analysis undertaken by Bethke within a more general framework that seems to have passed unnoticed. In doing so, it serves as a critical appraisal and also shows for the first time how the Bethke-type analysis for binary alphabets can be extended systematically to any non-binary one.

Parts of the work reported in chapters 3 and 4 have already appeared in the Proceedings of the Second Conference on Parallel Problem Solving From Nature, Brussels 1991 [55]. The application treated in chapter 8 is discussed in the Proceedings of the Fifth International Conference on Genetic Algorithms, Urbana-Champaign 1993 [56]. It has several immediate industrial applications. Because it belongs to a more general class of problems (i.e. **regularisation problems**) it may point to areas where the application of GAs may be most appropriate in comparison to other existing alternatives.

The discussion of the Pursuit Problem and a systematic search for sets of good testbed problems has never been attempted before. In fact, it examines and suggests problems with a complexity of at least an order of magnitude greater than the currently accepted testbed. It also shows by an experimental approach that the latter is in fact not a suitable problem for use as a good testbed. Given the fact that this problem has been used without critical assessment for the last six years, I believe that the contribution provided here will be of great interest to researchers of DAI. Some of the results reported in chapter 7 appeared in the Fifth European Workshop on Modelling Autonomous Agents in a Multi-Agent World, Neuchâtel, 1993.

The work presented here offers other contributions beyond those described above. At the current time no definite theoretical framework exists to examine whether a given domain is suitable for genetic search. The explorations conducted in this thesis point to the difficulties one is faced with in the current most-discussed theoretical framework. Up to now this is basically the only one that can at least provide some definite clues for recognition of possible easy and difficult problems for a GA. It thus serves as a practical guideline for designing systems with the assistance of GAs. The application problems examined in this thesis are non-trivial problems with far-reaching consequences. The application of splines for curve fitting indicates a very advantageous way to exploit GAs in this problem, which is an important and active topic of research. The results reported here are of interest, for example, for biochemical engineers. With regard to the DAI application, designers of DAI systems may meet many future instances of similar problems, of a form that guarantees that success of a technique of solution in one instance strongly implies success in the others. The design of suitable testbeds deserves to be considered an important topic in industrial applications where proposed systems should be validated on meaningful problems. It is hoped that this thesis serves also as an experimental exercise that shows how extensions to the basic framework can be useful in the design of ever more powerful architectures of problem-solvers.

Chapter 2
Genetic Algorithms

This chapter describes the main features of Genetic Algorithms, a class of heuristic search algorithms based on the mechanics of evolution and population genetics. The description is meant to introduce the basic concepts and to make apparent the most important topics that will be of interest for later chapters.

2.1 Introduction

The problem of function optimisation can be stated as follows. Given a function $f : D \mapsto R$, find the global maximum (or minimum) of f. f is called the **objective function**, D is the **domain** or the **search space** and R is the **range** which will be assumed to be a subset of the real numbers $\Re$. In general, the search space consists of n real scalar **independent variables** denoted by $x_1, x_2, \ldots, x_n$, but it may also be a vector space over a finite field, a cross-product of different subspaces, a non-numerical space [8] or even a cross-product of all the component spaces. In practice many values of the independent variables are not allowed (e.g. due to economic or physical considerations) and thus the problem formulation should include some relations (usually in the form of inequalities) which must be satisfied by any member of the domain. Whenever this occurs the problem is known as a **constrained optimisation** problem and the points satisfying the given relations are said to form the problem's **feasible region**. When this does not happen the problem is known as an **unconstrained optimisation** problem.

Function-optimisation problems arise in many different areas such as system identification, pattern recognition, design of logic circuits and on-line control for industrial processes. In several circumstances the function to be optimised can be described explicitly in a form that is amenable to mathematical treatment, allowing an explicit solution for the problem to be formulated or the investigation of whether optimal solutions exist at all.

On the other hand, in several situations one is confronted with a system for which there is no complete detailed knowledge. In other cases no particular interest exists in obtaining all the information about the system itself. Nevertheless one may still be interested to determine what settings of the independent variables will yield the optimal value of a given criterion.

It often happens that the total number of experiments (either numerical or practical) to be performed on the system is limited and hence the planning of the placement of the experiments (and corresponding variable settings) should be considered carefully. Any set of prescriptions for the placement of the experiments is called a **search plan** and any investigation seeking the optimal value of an unknown function is called a **search problem** [95]. Given the restriction on the number of experiments to be conducted, it is of interest to find among all possible search plans, the one which looks for the optimum in an optimal manner or to use Wilde's words ([95] (p.2)):

> ... we are not only trying to optimise the *function*; we are also optimising the *optimisation procedure.*

The above discussion begs the question on how one should measure the effectiveness of any search plan. It is also important to note that finding the optimum of an arbitrary real-valued function is a difficult problem. Hence if any mathematical treatment is at all possible, some simplifying assumption about the search space should be made, e.g. by the introduction of concepts such as unimodality, smoothness, convexity and similarity [95] or by assuming that the objective function is differentiable (or even twice differentiable) [8]. The latter motivates the introduction of gradient descent algorithms while the idea of unimodality in closed intervals of unidimensional real-valued spaces gives rise to very efficient search plans such as the minimax methods [95]. It turns out that fortunately in this case an a priori criterion, which depends on the placement of the experiments, but not on their outcomes, can be defined precisely and then optimised[1] [95].

Multivariable problems, however, have a structure entirely different from that of a single-variable problem and the extension of acceptable assumptions that may hold for single-variable methods may not be valid when carried over to multivariable problems [95]. For example, even if the idea of unimodality may be a plausible hypothesis for some unidimensional search spaces it is quite unlikely that it should hold as the number of dimensions increases [95]. Moreover, as argued by Wilde, due to the multidimensionality of the search space one is unable to find a measure of

[1]Here the **interval of uncertainty after k measurements** can be shown to be a reasonable measure of the success of a particular assignment of k experimental trials.

search effectiveness that does not depend on the experimenter's luck even when the assumption of unimodality is accepted.

2.2 Random search

The recognition of the limitations of extending successful techniques from unidimensional to multidimensional problems has led to proposals of using random-search techniques as viable alternatives to the traditional methods. For example, as early as 1958 Brooks [12] proposed the use of a random method when the number of dimensions was large. Central to his analysis was the consideration of the two factors that should be decided a priori by the experimenter, the first being the definition of what constituted a successful maximum-seeking experiment and the second related to the probability of conducting such a successful experiment.

His considerations led to the definition of success as the discovery of a combination of the independent variables (or factors) such that not more than a predetermined proportion of the search space would contain combinations having associated higher values. This proportion was denoted by a. The other factor, namely the probability of success, was related to the probability of finding at least one combination in the best fraction a of the search space and was designated as S. It is easy to see that these quantities are related by the following relation

$$S = 1 - (1 - a)^t$$

where t is the number of trials conducted and $(1 - a)$ is the probability that a single trial will fail to be made inside the desired region. Solving for t gives

$$t = \frac{\log(1 - S)}{\log(1 - a)} \qquad (1)$$

Since t does not depend on the number of dimensions, this result may appear rather surprising, especially when the number of dimensions is very large. This was observed by Hooke and Jeeves [38] who considered as an example a problem of 50 independent variables x_i such that $-1 \leq x_i \leq 1$ for $i = 1, 2, \ldots, 50$. The size of the total search space is 2^{50} and 5% of it is a subspace almost as intimidating as the original space itself. As pointed out by Wilde [95], this is due to the application of a first-degree measure, i.e. percentage, to a high-degree quantity, namely volume and its multidimensional generalisation. Accordingly, even if the method guarantees to cut down the region of uncertainty to a fixed fraction of the original search space, this percentage would have to be extremely small before the ranges of the individual independent variables would be reduced significantly.

Brooks' ideas were more general, though. He argued that the random method need not be applied in a sequential manner but rather all the trials could be conducted simultaneously. He also suggested two different versions of the random method. The first one, which he called **the Stratified Random Method**, was based on the idea of dividing the search space into n equal sub-regions. (Brooks also proposed rejecting points that were within a prescribed distance of previously-selected points). The second one (and for the purposes of this work, the more noteworthy of the two) was referred to as **the Creeping Random Method**. This technique involved making an initial guess for a starting point and then conducting trials at random in the vicinity of the first estimate. The location of the best of these random trials was made the centre of a second random exploration. Brooks suggested the use of a normal distribution whose variance was diminished at each iteration. This was motivated by the desire to cluster the values around the best currently-available combination.

That he was anticipating major breakthroughs to come can only be attested by his final comments on [12] with which we end this section:

> *There are some rather intriguing analogies that can be made between the creeping random method and evolution.*

2.3 Genetic Algorithms

In the previous sections it has been argued that one of the central problems with devising a search plan on a multidimensional space is related to the difficulty of finding an optimal strategy for allocating trials over the search space. Random search was introduced to cope with the limitations faced by search plans that operated on spaces of a large number of dimensions. However, even in the latter case multidimensionality forces one to seek regions of uncertainty that are a tiny fraction of the original space before the ranges of the independent variables are reduced significantly [95]. Given this, one could still use a random search method in the following effective way. Consider the case of an algorithm composed of two routines. The first places n random samples on a prescribed subspace of the entire space exactly as Brooks' random-search method. Given the observed values of the sampled points and a precise knowledge of the subspace being sampled at that iteration (probably by using some special bookkeeping routines), a decision module would specify a more restricted subspace on which to sample the next n points in the next iteration.

Suppose also that the function being searched has enough redundant information so that the decision module reduces (with a high probability of being correct) the search space by say 30% at each iteration. Thus even with a huge initial search space of size 2^{50}, the algorithm would be able to converge to the global optimum after approximately 100 such iterations. For example, if $n = 40$ this would correspond to approximately 4000 trials, which is a minute fraction of the original space.

It is obvious that the assumptions in the last example are somewhat over optimistic and that in general the allocation of trials is a nontrivial and difficult problem. Fortunately, there is a class of relatively simple algorithms which allocate trials in a nearly optimal way[2]. These algorithms, generally known as **Genetic Algorithms** (GAs), were introduced by John Holland [37] and are inspired by the observation of how adaptation takes place in evolving natural systems. Holland proposed searching a general space using **reproductive plans** which would selectively manipulate and reproduce a collection (**population**) of candidate solutions (**individuals**) to generate new individuals on the most promising areas of the search space.

In the simplest form of the GA, candidate solutions to some problem are encoded in binary strings which play the role of artificial **chromosomes**, while individual bits play the role of **genes** and their respective positions are referred to as **loci**. The different values that a gene can assume are known as its **alleles**.

Each **individual** therefore comprises a candidate solution to a specific problem and has associated to it a corresponding **fitness** or **utility**, i.e. a nonnegative figure of merit that measures the adequacy of the related solution. As discussed above, one of the difficulties of any search plan working in a high-dimensional space is to decide where one should place the next set of trial experiments subject to the information provided by the current set of alternatives. In Holland's reproductive plans, individuals with higher fitness values have a higher probability of being selected for producing multiple copies of higher-fitness individuals for the next **generation**. This by itself would not produce any new individuals but merely change the number of copies of each chromosome. Hence, the **selection** of high-fitness individuals is combined

[2]A precise meaning will be given in section 2.4.

with **genetic operators** such as **mutation** (flipping individual bits) and **crossover** (exchanging substrings of two parents to obtain two offspring), all applied in a probabilistic way to produce a new generation of individuals. The GA is judged to be successful if it can evolve a population of highly fit individuals as a result of iterating this procedure through successive generations. Figure 2.1 illustrates a typical reproductive plan.

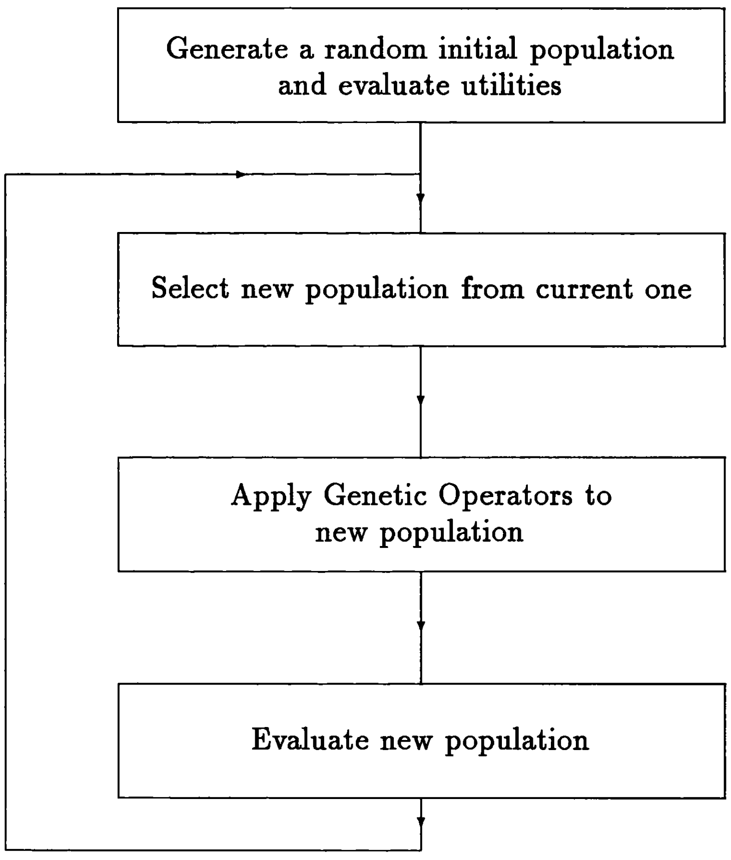

Generate a random initial population and evaluate utilities
Select new population from current one
Apply Genetic Operators to new population
Evaluate new population

Figure 2.1: A typical reproductive plan

2.4 Selection, schemata and utilities

It is apparent from the previous discussions that if a search plan is to be successful in a high-dimensional space some restriction about the structure of the problem should be assumed. A GA is no exception. The simplifying assumption made on the function domain, typically of high dimension, is that it should contain regions of good objective-function values and regions of poor values. Any useful search algorithm must decide concurrently which regions are potentially interesting to be explored while collecting information about the function itself. This trade-off is a crucial point in any adaptive system and is usually referred to as the trade-off "between the exploration for knowledge and the exploitation of that knowledge" [28].

In order to make a proper analysis of the class of functions that should be searched efficiently using a GA, the notion of schema is necessary. A **schema** corresponds to a partition of the space of chromosomes with each partition (subspace) being characterised by the sharing of some specified genes by its elements [70].

More formally[3], if we consider the space of chromosomes of length l a chromosome will be represented by a string $s = s_1 s_2 \dots s_l$ where $s_i \in \{0,1\}$ in the binary case. A schema will be represented by a string $h = h_1 h_2 \dots h_n$ where $h_l \in \{0,1,\#\}$ and a string s will be said to instantiate h if $h_i = s_i$ or $h_i = \#$ for $i = 1,2,\dots,l$ [8]. For example, if $l = 3$ then the schema 10# corresponds to the set $\{100, 101\}$ and could be described as the set of binary strings of length 3 that "begins with 10".

It is easy to see that every chromosome instantiates 2^l schemata. Following [70] let s denote a string whose corresponding utility is given by $u(s)$. Let ζ be a schema instantiated by s. The utility of ζ is defined as

$$\mu(\zeta) \triangleq \frac{1}{\mid \zeta \mid} \sum_{s \in \zeta} u(s) \tag{2}$$

where $\mid \zeta \mid$ is the number of chromosomes instantiating the schema ζ.

Holland observed that subject to the fact that there are some correlations between the utilities of different instances of schemata, each evaluation of a chromosome provides statistical information about the utility of *each* of the 2^l schemata instantiated by that chromosome. Hence, although the GA does not manipulate schemata directly, it does that in an implicit way, a phenomenon often referred to as *implicit parallelism*. In doing so, the analysis is shifted from the individual chromosomes to the schemata that they instantiate.

[3]We follow Bethke's notation [8] in this section.

Each evaluation of a chromosome can thus be regarded as a statistical sampling event that yields information about the utility of the schemata that it instantiates. In general, due to the restriction on the number of trials, one cannot sample enough points in the search space so as to provide completely accurate information about the utility of each schema. Hence each schema is assumed to be a random variable with an associated distribution of utilities [8]. The more this sampling reflects the utility of the schemata instantiated, the better this information will be used to guide the algorithm into more promising areas of the search space.

Holland [37] suggested that the GA behaviour could be understood in terms of optimising a sequential decision process where uncertainty was present in several different factors such as lack of a priori knowledge, noisy feedback and a time-varying pay-off function [41]. In terms of this interpretation, an optimal allocation of trials among competing alternatives would give exponentially increasing trials to the best observed alternative. As we shall point out in what follows, this is nearly realised in a GA if one allows the selection of chromosomes to the next population to be based on a probability that is proportional to their corresponding utilities.

To see that, first note that in an actual implementation one can only refer to the observed schemata utilities which may be considered as a restriction of (2) to the actual population of chromosomes. However, it is immediate to see that if the probability of selecting a given chromosome is equal to the ratio of its utility to the average utility of the population at that iteration then the number of examples of a particular schema h in a given population grows as the ratio of the average utility of that schema to the average utility of the population [28] at that iteration.

The ideas discussed above are usually explored with the help of the following concepts which will be used later on.

Definition 1 *(Bethke 1981 [8]) The set of* **defining positions** *of a schema h is the set of indices of the ones and zeroes in h.*

Definition 2 *(Bethke 1981 [8]) Two distinct schemata, h and h' are said to be* **competing schemata** *if they have the same set of defining positions.*

The **defining length** of a schema h, denoted as $d_l(h)$, is the distance between the first and last defining positions in it. The **order** of a schema h, denoted as $o(h)$, is equal to the number of defined positions in h

Holland's interpretation of the behaviour of a GA is to regard it as allocating (among a set of competing schemata) exponentially increasing trials to the best observed one [28]. In terms of the discussion presented in section 2.2, this action corresponds to the decision module providing some guidance about the subspace that should be sampled in the next iteration. What is remarkable about a GA is the simplicity with which this is done, as this subspace is only given implicitly by the different number of individuals in the next population.

This simplicity is respected in the design of the mechanisms to generate new members from the current population (primarily the crossover operator), which are discussed in the next section.

2.5 Genetic operators: crossover and mutation

A simple crossover works in the following way (figure 2.2). Two individuals (*parents*) are chosen in the selection step. Then, an integer position k is chosen randomly in the interval $[1, l-1]$ and two new strings (*offspring*) are created by swapping all string elements between positions $k+1$ and l inclusively [28].

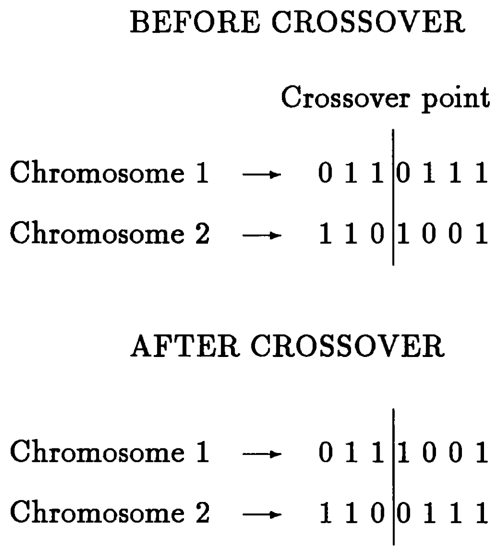

BEFORE CROSSOVER
Crossover point
Chromosome 1 → 0 1 1 | 0 1 1 1
Chromosome 2 → 1 1 0 | 1 0 0 1
AFTER CROSSOVER
Chromosome 1 → 0 1 1 | 1 0 0 1
Chromosome 2 → 1 1 0 | 0 1 1 1

Figure 2.2: A single point crossover shows the partial exchange of two chromosomes at a cross site chosen in a random way.

Alternatively, a two-point crossover proceeds by selecting two integer points p and q lying in the range $[1, l-1]$ and with $p > q$. Two new offspring are formed from the parent chromosomes by swapping all string elements that lie between p and q. Other extensions are also possible and are discussed in [28].

A simple crossover operator biases the search by disrupting those schemata with long defining length. This can be expressed by the crossover survival probability which is bounded below by the expression $1 - p_c \frac{d_l(h)}{l-1}$, where p_c is the probability of performing a crossover on two selected strings [28].

Crossover is often understood as the most important genetic recombinant operator and much of the power of GAs results from the interaction of the selection process and the crossover mechanism. In fact, the interplay of these mechanism is often considered the heart of the balance of exploring and exploiting information present in a given population of chromosomes.

Mutation is often considered to play a secondary role in the simple GA and its basic action is to prevent premature loss of important information from the population[4]. This is due to the fact that crossover never introduces new alleles in a given population while the dynamics of the selection process allows deletion of particular alleles as the algorithm proceeds. Accordingly, the probability of applying mutation is very small when compared to that of crossover. In the same as above, one may calculate the probability of a schema surviving mutation (applied with a probability p_m) is bounded below by the expression $1 - o(h)p_m$.

The interaction of the selection and reproduction phases of a simple GA is often described by the following fundamental theorem due to Holland [37].

Theorem 2.1 *The Fundamental Theorem of Genetic Algorithms. Let $\eta(t)$ denote the number of instances of a schema h at iteration t. Let $\hat{u}_h(t)$ denote the sample utility for the schema h at that iteration and $\bar{u}(t)$ denote the mean utility of the population at that iteration. Then the expected number of instances of h at iteration $t+1$ is governed by the following expression:*

$$\eta(t+1) \geq \eta(t)\frac{\hat{u}_h(t)}{\bar{u}(t)}\left[1 - \frac{d_l(h)}{l-1}p_c - o(h)p_m\right]. \tag{3}$$

The implications of theorem 2.1 are far reaching. Its most common interpretation states that short, low-order, and highly fit schemata are sampled, recombined and

[4]Mutation is processed in binary strings, choosing a randomly selected integer point in the string and changing the corresponding bit to its complement value.

resampled to form strings of potentially higher fitness [28]. This interpretation is so vital that it remains the cornerstone on which most implementations of GAs rely, and is often referred to as the **Building Block Hypothesis**. It is remarkable how equation (3) can incorporate in a simple way the rather complex dynamics of a simple GA. In fact, many of the problems faced by many applications of GAs may have resulted from the deceptively simple belief that *any* implementation that follows the guidelines suggested by theorem 2.1 will perform in a successfully efficient way.

2.6 Discussion

In the previous subsections we reviewed the basic formalism developed by Holland [37]. The fundamental ideas there were the optimal allocation of trials among competing alternatives and the action of crossover to combine good building blocks in order to generate strings of potentially large fitness.

In what follows we shall review briefly several problems that often occur in applications involving GAs and that will be of interest for this work.

2.6.1 Choice of fitness function

Although the formalism offered by Holland is elegant and aesthetically pleasing, it leaves the choice of the fitness function open. However, the rate at which trials are allocated to different individuals depends on their observed utility function. Radcliffe [70] observed that given any utility function, its composition with an arbitrary (monotone) increasing function will yield another valid utility function. He concluded that there was scope for slowing the rate of convergence of the algorithm by making a suitable choice of utility function. Note however, that Holland's formalism implicitly requires that the sampling process should give reasonable statistical evidence about the utilities of the schemata instantiated by a given chromosome. This certainly restricts the number of possible choices.

Note also that in applications such as those examined in this thesis an explicit form of the fitness function does not exist. One is left therefore with the difficult and time-consuming task of examining combinations of functions and parameters that make genetic search viable, a process that may be computationally very undesirable and which we would like to avoid.

2.6.2 Premature convergence

One of the problems most often cited in applications of GAs is related to the loss of diversity in the population before the global optimum is found. This may happen as a result of some individual (whose fitness is much larger than any other individuals of the population) being allocated a very large number of trials. Under this condition many of its genes can dominate the population and become fixed. This may be undesirable, particularly if this individual does not represent a point close to the global optimum. On the other hand, in some circumstances the fitness of individuals in a population does not differ significantly and the selection process becomes almost uniform among the population, thus losing its main advantage over random search [28].

Several mechanisms have been suggested to improve these problems, most noticeably the scaling of the fitness function (e.g. linear scaling) [28]. Other techniques are discussed in [70]. Unfortunately no specific mechanism has emerged as a winner. As a result, the decision on which one should be used may itself be a difficult one especially in problems where the search space is very large and where information about properties of the utility function is unavailable.

2.6.3 Codings and good parametrisation

Since GAs work with a coding of the parameter set rather than the parameters themselves, a natural question is how this encoding can take place. The success of earlier experiments may have led researchers to place much hope in the adaptive capabilities of GAs as a very powerful mechanism that could perform well under *very* different encodings. However, the formalism presented assumes implicitly that there should be some correlation among the individuals comprising a given population and that this correlation can be explored successfully by the GA. Some encodings may be effective, others may render the GA completely unsuccessful. Holland [37] proposed subjecting the representation itself to adaptation but this has remained an open problem and one that the author believes (as discussed in chapter 5) may be rather difficult to tackle. Several genetic operators (e.g. inversion) can be regarded as changing the representation as the algorithm progresses. However, the reported evidence has placed some doubts on their effectiveness (e.g. see [70]). As previously argued, the sheer size (and dimensionality) of some problem spaces leads to many difficult issues, and one is obliged to restrict the class of problems being explored to a set that falls under some generally acceptable assumptions.

2.6.4 Good analytical tools

Even when restricted to the theory presented in this chapter one is often left with the question of trying to characterise the domains where a GA can be applied successfully. The most used tool of analysis follows from Bethke's explorations with Walsh functions [8] on binary representations. Other attempts have been made (e.g. by Liepins and Vose [52]) but do not offer much insight beyond the one provided by Bethke.

Also, most studies of the behaviour of genetic algorithms (GAs) refer to schemata that are expressed in a binary alphabet. In some instances where the efficiency of GAs is unsatisfactory, there is a view that the problem should be overcome or reduced if an alphabet with more characters is used. This is still an ongoing debate (e.g. see [58, 75]). For the non-binary case, to the author's knowledge the only reported analysis was conducted by Mason [58] following completely different lines from the one used by Bethke. It is difficult to see how the two can be put in correspondence with each other. It would therefore be interesting to find a unified framework in which questions of interest could be discussed in a more general way.

The difficulty in understanding the wider implications of Bethke's approach has resulted in several different attempts to characterise the domains of applicability of GAs, and different measures to decide on the suitability of a particular problem to genetic search has been suggested. It has often led researchers to focus attention on certain problems that are difficult for a GA to explore but whose intrinsic nature is not completely understood (e.g. [29, 21]). Part of the contribution of this thesis is to put all these rather different approaches into a common framework where the above concepts can be discussed, criticised and extended objectively.

2.7 Summary

In this chapter we have motivated and discussed the main features of GAs. The basic formalism was reviewed and some problems stemming from lack of a detailed description of the dynamics of GAs were also presented. Although GAs have shown very promising results in some areas, in others they have performed in a poor way due to the lack of a correct interpretation of the basic ideas behind their design. Thus identifying the techniques and analytical tools behind their analysis is a desirable issue to consider. Also, one is faced with the dilemma of deciding whether the possible solution of a given high-dimensional problem can be suggested in an effective way by genetic search. Finding suitable indices, for that matter, is also important. Finally, if the latter approach proves to be difficult to achieve, it is then important to find meaningful problems which can be explored effectively by GAs. All the above issues form the body of the work explored in this thesis.

Chapter 3
Walsh Series Analysis

This chapter examines schema analysis from the broader viewpoint provided by abstract harmonic analysis. This offers a more general and natural framework for characterising the domains of applicability of GAs and provides new insights and motivation for extending schemata and genetic algorithms through the use of more abstract concepts. Previous results obtained by Bethke are embedded in a more general framework, namely of group theory, where the concepts discussed above can be understood by reference to certain linear subspaces and their respective nullspaces.

3.1 Motivations

In chapter 2 a simple Genetic Algorithm, a non-standard technique for function optimisation, was described. The basic motivation for its introduction was the desire to explore problems for which conventional techniques such as gradient methods perform poorly. In general these problems are characterised by functions that are not everywhere differentiable, not unimodal or such that the dimension of the search space is large.

Given a certain algorithm it is important to know the class of problems for which it has a good chance of performing well. For GAs, some of the relevant conditions were established by Bethke [8] who introduced Walsh functions and the Walsh transform to analyse the behaviour of GAs when the search space is considered to consist of binary strings. It has remained the most often used technique for discussing issues of interest in genetic optimisation. His conditions are stated in the transform space of the Walsh coefficients, and are difficult and not intuitive to understand.

The idea that a similar analysis could be extended to the non-binary case may have occurred to several authors (for example, through the obvious similarity of Walsh and Fourier functions [28, 21]). However to the author's knowledge the only investigation on non-binary alphabets so far was conducted by Mason [58] along completely different lines.

Walsh and Fourier functions have long been exploited in other application areas (e.g. communication [31], switching theory [46], statistical analysis [67] and signal processing [57]).

In spite of the above, the GA literature seemed to have embarked on its own independent path, and many concepts and results already established elsewhere were defined or derived again in the context of GAs in ways that are sometimes more difficult to understand and to use (e.g. the concept of **nullspaces** and their translations, which is helpful for clarity, as we shall see below, has nevertheless not been used previously in the GA field). This has only enlarged the gap between itself and the general available literature, and in the opinion of the author this may have hindered further theoretical progress. Consider for example the Walsh-Schema transform that is so central in many of the theoretical analysis of GAs. This is in fact fully equivalent to a well-established result in group theory known as the **Poisson Summation Formula** (see e.g. [88, 46]) and whose group-theoretic analogue plays a major role in the theory of group characters [46][1].

The main goal of this chapter is to review Bethke's work and to embed it in the more general framework of group theory[2], where the concepts discussed above can be understood by reference to certain linear subspaces and their respective nullspaces.

It is often the case that (in such circumstances) ascribing originality of ideas is a difficult task, since what may appear to be original in one area may already have been known for a long time in another one. For example, the idea that schemata and hyperplanes were intimately related was already established for a long time in the GA culture. However the fact that these concepts could be expressed by reference to more fundamental entities, such as **cosets**, was not appreciated until very recently (e.g. [6, 52, 93]), although they were already well known in basic mathematical literature (e.g. see [10] for a thorough discussion of these concepts) and also in the communication area. In fact, as long ago as in 1972, Harmuth [31] showed explicitly how these concepts from group theory could be used in the design of multiplex systems while in 1971 Lechner [46] used the above ideas in exploring an algorithm to detect prime implicants for logic functions. To the knowledge of the author, the results

[1]In the theory of data communication Shannon's famous **Sampling Theorem** follows as a straightforward consequence of it [88].

[2]The interested reader should refer to chapter 5 for a more detailed presentation of the basic concepts of group theory.

derived in the GA literature were never put into a clear perspective with those from related work in other areas. One of the purposes of this chapter is to attempt to correct this situation. In doing so, we shall follow Lechner [46] due to historical reasons and clarity of exposition. Most of the results presented in this chapter will be stated without proof, as the appendix provides complete proofs of their extension to any non-binary alphabets.

3.2 Notation and representational issues

Our main purpose in this section is to review the concepts introduced elsewhere [28] in a more abstract way, and to provide the groundwork for future generalisations that may be useful with respect to genetic algorithms. We shall try to avoid GA terminology, referring to it only if necessary.

Traditionally GAs operate on binary strings known as chromosomes. Formally we define Z as the 2-element field $\{0,\ 1\}$ (or alternatively the additive group of integers modulo 2), and Z^n as the n-dimensional vector space over the field Z (or alternatively as the product group of Z). This corresponds to the domain of our search space.

The range of the function is usually considered to be the set of real positive numbers, but occasionally one may also be interested in functions over Z. Therefore we place no restrictions on the space of functions, $\mathcal{F}$, which will be clear according to the context. X will denote the set Z^n of all binary n-tuples when it is the domain for the space $\mathcal{F}$. An element $x \in X$ will be represented by the row vector $(x_1, \ldots, x_n)$, and its transpose x^T, will be represented by a corresponding column vector. Elements of X are indexed in the natural ordering, defined by the correspondence between a number and its radix-two expansion on binary n-tuples.

The **weight** of a given vector $x \in X$, denoted $|x|$, is defined as the number of coordinates in x with value 1. Alternatively, following Bethke [8], we shall refer to it as the **order** of the binary string x.

3.2.1 Schemata and subcubes

Z^n is usually pictured geometrically as an n-dimensional unit cube (or n-cube) having the 2^n binary n-tuples which represent elements of Z^n as vertex coordinates. We next consider subsets of Z^n as sets of n-tuples.

If in a given subset the i-th coordinate in the n-tuple x has a fixed value we call it a **bound variable**, otherwise we shall call it a **free variable** [46]. Bounded variables are normally referred to in the GA literature as **fixed positions** [26]. A **subcube** of Z^n is a subset with bound and free variables. The **dimension** of a subcube is the number of free variables it contains. Therefore a subcube of dimension k includes 2^k points of Z^n, since each free variable can assume two possible values i.e., 0 or 1 . Following Lechner [46] we shall refer to a k-dimensional subcube of Z^n as a k-cell or k-cube.

A **schema** is a subcube denoted by a ternary n-tuple

$$h = (a_1, a_2, \ldots, a_n), \ , a_i \in \{0, 1, *\}$$

where the symbol '*' stands for 'don't care' and is assigned to the k components a_i for which x_i is a free variable. Whenever $a_i = *$, it is understood that the corresponding x_i assumes both values 0 and 1.

In this way a schema is a subcube, but certain subcubes cannot be expressed uniquely by a schema. For example, $\{(0,0), (1,1)\}$ is a subcube of Z^2 with two free variables, but there is no schema that can describe it in a unique way. However, any subcube is always contained in some schema, since we can consider a schema whose i-th entry is '*' if the i-th variable in the subcube is free.

A schema with k free variables is often referred to as a **k-order** schema, but for convenience and consistency with the above notation we shall refer to it as **k-schema**.

Since each a_i independently can assume one of 3 possible distinct values, it follows that there are 3^n ternary n-tuples corresponding to 3^n possible schemata. Also, since there are $C^n_k = \frac{n!}{k!(n-k)!}$ ways to select k free variables among the n coordinates of Z^n, and $2^{(n-k)}$ ways to assign values to the $(n-k)$ bound variables, the total number of k-schemata in Z^n is $2^{(n-k)} \times C^n_k$.

We note that Z^n is an additive group with bitwise addition modulo 2, also denoted as $\oplus$. Therefore it is natural to consider the possibility of embedding schemata in a group-theoretic description. First we note that any subspace of Z^n is a subgroup and therefore contains the zero vector and is closed under addition. Thus any schema with the bound variables equal to 0 is a subspace of Z^n. Let a denote one such k-dimensional subspace. Following [46], we denote the subspace defined by a as V_β where β is the binary n-tuple obtained from a by replacing the symbol '*' by 1. Note that V_β can be generated by the unit vectors whose sum is β. For example, if $a = (*, 0, *)$, then $\beta = (1, 0, 1)$ and $V_\beta = \{(0,0,0),(0,0,1),(1,0,0),(1,0,1)\}$. More formally, V_β is referred to as the **linear span** of the unit vectors whose sum is β. The action of β can be understood as one of pointing to the free variables in a.

Since V_β is a subgroup its cosets, which are denoted by $V_\beta + c$, $c \in Z^n$, partition Z^n [47]. The notation $V + c$ denotes the set $\{x \oplus c \mid x \in V\}$ (where $x \oplus c$ denotes the bitwise addition modulo 2 between x and c) and is referred to as a **translation** of V_β by c.

Note that V_β is itself a coset. Also two cosets $V_\beta + x$ and $V_\beta + y$ are identical if and only if $x \oplus (-y) \in V_\beta$, where $-y$ is the inverse element of y; otherwise they are disjoint. Since for any $x \in Z^n$ $x \oplus x = 0$, each element is its own inverse, i.e, $-y = y$, and the last condition implies that $x \oplus y \in V_\beta$. Since V_β is closed under addition, it follows that 2^k different vectors $x \oplus c$ will translate V_β into the single coset $(V_\beta + c)$. However, only one of these vectors has zero values for each of the free variables in V_β. From now on, the symbol c will be used to identify this vector uniquely.

The unique correspondence between a k-schema a and a coset $V_\beta + c$ as defined above is reproduced in the following table, which lists the corresponding components of a, β and c:

a	β	c
0	0	0
1	0	1
*	1	0

From the above table we note that components of c that correspond to bound variables in a always agree in value with the corresponding components of a, whereas components that correspond to the free variables are always 0. Thus, except for the vector c itself, every vector in the coset V_β is the translation of a nonzero element x of V_β which has unit value for at least one of the free variables in V_β. Therefore $|x \oplus c| > |c|$ for every nonzero x in V_β, or in other words c is the unique vector of minimum weight (or order) in the coset $V_\beta + c$. It is usually referred to as the **coset leader** [46].

Note that if $|\beta| = k$ then V_β has k unit vectors and 2^k elements. Since V_β is the linear span of these unit vectors, we see that no other vector element in V_β has a weight greater than or equal to β. Therefore β is the unique vector of maximum weight in the linear subspace V_β. The above facts can be denoted conveniently by using the following partial ordering relation [46]:

Definition 3 *For fixed $\beta \in Z^n$, the notation $x \leq \beta$ means $x_i \leq \beta_i$ for $1 \leq i \leq n$.*

In this way, V_β can be represented as $V_\beta = \{x \mid x \leq \beta\}$.

We shall be interested in subspaces with the following property:

Definition 4 *Let V be a k-dimensional subspace of Z^n (i.e., an additive subgroup of order 2^k). The set $\{y \mid yx^T = 0\}$ for all $x \in V$ is called the complementary subspace or* **nullspace** *V' of V.*

Consider now the set $V_{\bar\beta} = \{y \mid y \leq \bar\beta\}$, where $\bar\beta$ is the logical complement of β. This set is also a subspace, the sum of whose unit basis vectors is $\bar\beta$. Note that if $x \in V_\beta$ and $y \in V_{\bar\beta}$, then $xy^T = 0$. $V_{\bar\beta}$ is clearly the nullspace of V_β (and vice versa).

It is a well-known fact from linear algebra that Z^n is the direct sum of V_β and $V_{\bar\beta}$[3]. In other words, every $x \in Z^n$ has a unique decomposition $x = x_1 \oplus x_2$, where $x_1 \in V_\beta$ and $x_2 \in V_{\bar\beta}$ and are called the **projections** of x in the corresponding subspaces. Also, since V_β and $V_{\bar\beta}$ partition Z^n, the dimension of $V_{\bar\beta}$ is given by $(n-k)$ where k is the dimension of V_β.

As will be shown shortly, the relation between V_β and its nullspace will be very important for the computation of schema average fitness. We therefore study these subspaces in more detail. First note that if H is a subgroup of a group G, the **index** of H in G is the number of distinct right cosets [47] and is denoted by

$$[G : H].$$

The **order** of a finite group G, denoted by $order(G)$ is the number of elements of G. It is a basic fact in group theory that

$$[G : H] = \frac{order(G)}{order(H)}.$$

[3]It is important to note as pointed out by Lechner [46] that unless V_β has a basis of unit vectors, $V_{\bar\beta}$ cannot always be identified with the set of coset leaders and Z^n is not necessarily the direct sum of V_β and $V_{\bar\beta}$.

Translated in terms of the entities with which we are dealing, it follows that

$$[Z^n : V_\beta] = \frac{2^n}{2^k} = 2^{(n-k)}$$

which is the order of $V_{\bar{\beta}}$. Furthermore, any coset leader in V_β is a linear combination of the unit vectors of $V_{\bar{\beta}}$, since any coset leader has zero values on the components corresponding to the 1 values in β. In this way $V_{\bar{\beta}}$ can be identified with the coset leaders of V_β. Note in the above arguments the importance of V_β having a unit basis. For example, the subspace $\{(0,0),(1,1)\} \subset Z^2$ is its own nullspace and its only non-trivial coset is $\{(0,1),(1,0)\}$ [46]. The way schemata are defined guarantees that each schema is a coset of a subspace which has a basis of unit vectors.

EXAMPLE 1

To fix all the ideas discussed above let us consider the schema $a = (*,0,0,*)$. Its corresponding β is given by $\beta = (1,0,0,1)$ and V_β is $\{(0,0,0,0),(0,0,0,1),(1,0,0,0),(1,0,0,1)\}$. Accordingly $\bar{\beta} = (0,1,1,0)$ and $V_{\bar{\beta}}$ is given by $\{(0,0,0,0),(0,0,1,0),(0,1,0,0),(0,1,1,0)\}$. There are four distinct cosets of V_β. Each of them corresponds to a translation of V_β by an element of $V_{\bar{\beta}}$.

In the GA literature $\bar{\beta}$ is associated with the **partition number** of a schema h, while the coset leader corresponding to h is associated with the β **function** [26].

As already argued, in a GA the schemata give an idea of the direction of the genetic search. With regard to that, it is of interest to understand the relation between a given schema and any schema contained in it. Given two different subspaces, the one with higher cardinality has lower index on Z^n. When one subspace is contained in another it is said to be a *refinement* of the latter, or alternately we say that it *implies* the one with higher cardinality. The next lemma shows the implication this fact has for the corresponding nullspaces.

Lemma 3.1 *Let $\beta' \in V_\beta$. Then $V_{\beta'} \subset V_\beta$ and $V_{\bar{\beta}} \subset V_{\bar{\beta}'}$.*

Proof of Lemma 3.1 *By the alternative representation of V_β, $\beta' \leq \beta$. Therefore, β has components with unit values whenever β' has. Thus any unit vector of $V_{\beta'}$ is a unit vector of V_β. This proves the first part of the lemma. To prove the second part we note that the hypothesis of the lemma implies that $\beta'_i \leq \beta_i$ for every $i, 1 \leq i \leq n$. Thus, $1-\beta'_i \geq 1-\beta_i$ for every $i, 1 \leq i \leq n$. Since each component can only assume the* *values* 0 *and* 1*, the last inequality leads immediately to* $\bar{\beta}'_i \geq \bar{\beta}_i$ *for every* $i, 1 \leq i \leq n$*, i.e.,* $\bar{\beta}' \geq \bar{\beta}$*, and using the same arguments of the first half the proof is complete.*

Lemma 3.1 has an interesting consequence. Suppose we examine two schemata, one of which is contained in the other. The more *specific schema* i.e., the one with fewer free variables, is associated with a subspace that is contained in the subspace associated with the less specific one. On the other hand the corresponding nullspaces obey an opposite relation. As we shall show below, when we consider the expansion of schema average fitness in terms of Walsh coefficients the only Walsh coefficients that will appear in the expansion are those elements that belong to the nullspace. So Lemma 3.1 implies that the more general a schema, the shorter its expansion in terms of Walsh coefficients.

Before concluding this section it is instructive to introduce the following definition:

Definition 5 *A schema is a coset of a linear subspace of* Z^n *which has a basis of unit vectors.*

3.3 Summary of properties of Walsh-function analysis

The following definition and properties of Walsh functions will be of interest and will be stated without proof. The interested reader should refer to [31, 46, 67] for detailed discussions.

Definition 6 *The k-th order Walsh function is defined as* $\psi_k(x) = (-1)^{\sum_{i=1}^{n} k_i x_i}$*, where* k_i *and* x_i *correspond to the ith entries in the binary representation of* k *and* x*.*

Transform pair: If $f(x)$ is an arbitrary function of x, then the Walsh transform of $f(x)$ denoted by $f^*(\omega)$ is given by $f^*(\omega) = 2^{-n} \sum_x \psi_\omega(x) f(x)$. The inverse transform of $f^*(\omega)$ is given by $f(x) = \sum_\omega \psi_\omega(x) f^*(\omega)$. In both sums it is understood that the indices vary from 0 to $2^n - 1$.

Parseval Theorem: Let $f(x)$ and $g(x)$ be two arbitrary functions of x and $f^*(\omega)$ and $g^*(\omega)$ their respective transformations. It follows that

$$\sum_x f(x)g(x) = 2^n \sum_\omega f^*(\omega)g^*(\omega)$$

Symmetry: From definition 6 it is easy to see that $\psi_k(x) = \psi_x(k)$, i.e. that x and k are completely interchangeable in any relation involving Walsh functions or Walsh transforms.

Group Character: Multiplication of two Walsh functions of orders k and k' results in another Walsh function of order $k \oplus k'$ i.e.,

$$\psi_k(x)\ \psi_{k'}(x) = \psi_{k \oplus k'}(x)$$

Formally, the Walsh functions are characters of an Abelian group, the group elements being the binary vectors of length n, and the group operation being addition modulo 2 [67].

3.4 Classical properties

In this section we examine three important properties of Fourier transforms focusing attention on the domain of Walsh transforms. We shall show that the Walsh-Schema transform [21, 26] obtained by Bethke [8] is nothing more than a restatement of a fundamental result in Fourier analysis in groups, namely the Poisson Summation Theorem (or Poisson Summation formula) which can be derived as a natural consequence of two other important results known as the Convolution theorem and the Quotient Character theorem. The basic interest of this result for this work is that it provides a way to determine schema averages in terms of the Walsh coefficients. The exposition follows closely [46].

3.5 Convolution Theorem

As pointed out in [46] the convolution theorem is one of the most powerful tools of linear functional analysis. Examples of areas of applications are in linear filter theory and stochastic processes.

A good motivation for understanding the Convolution Theorem is the following: we know that the Fourier transform can be regarded as a linear operation between two isomorphic spaces, namely X and Ω. If f and g are two functions with domain X, and f^* and g^* are their corresponding Fourier transforms then the transform of $f+g$

is given by $f^* + g^*$. Since the other operation in the vector spaces is multiplication, we might be interested to investigate which function forms a transform pair with a given product of two Fourier transforms. Such a function is known as the convolution sum of f and g.

Definition 7 *Let X be an n-dimensional vector space over the 2-element field $\{0,1\}$. Given two real-valued (or integer-valued) functions $f * g$ we form the sum*

$$(f * g)_c = (f * g)(c) = \sum_{x \in X} f(x \oplus c) g(x)$$

and refer to it as the **convolution sum** *of f and g.*

The next theorem asserts that the convolution sum is precisely the function we are looking for.

Theorem 3.1 *(Convolution Theorem) Let f and g be any two integer, real or complex-valued functions on Z^n, $(f * g)_c$ their convolution sum evaluated at $x = c$ and $(f^* g^*)$ the product of their Fourier transforms. Then, for each c in Z^n, $(f * g)_c = 2^n (f^* g^*)^*$ evaluated at the point $x = c$, where $(f^* g^*)^*$ is to be understood as the inverse Fourier transform of the product of the transform of f and g.*

We shall also be interested in the Convolution Theorem in later chapters when we use it to construct functions whose Walsh decomposition involves high-order coefficients and yet may still be easy for the GA to optimise.

3.6 Quotient Group Character Theorem

Suppose V is a k-dimensional subspace of Z^n (i.e. an additive subgroup of order 2^k). The theorem to be stated in this section is a classical result of group theory, and basically says that any function that is constant on each coset of V can be expanded in terms of a subset of $2^{(n-k)}$ Fourier basis functions [46]. It will be useful for our purposes since as an an immediate result of it we can calculate the Fourier transform of the characteristic function of a schema. Also, a corollary of the theorem provides a way to identify functions that depend on fewer than n linear combinations of its arguments [46].

Theorem 3.2 *(Quotient Group Characters) Let V be a subspace of Z^n. Then the subset of Fourier basis functions $\Psi_V = \{\psi_\omega(x) \mid x\omega^T = 0, for\ all\ x \in V\}$ is a complete* *orthogonal basis for the subspace of $\mathcal{F}$ consisting of functions that are constant on cosets of V. No other functions are generated by this basis. If V has dimension k, then Ψ_V consists of just those $2^{(n-k)}$ $\psi_\omega(x)$ for which ω is orthogonal to all $x \in V$. If V has a basis consisting of k unit vectors, then $\psi_\omega \in \Psi_V$ iff ω is in the space generated by that subset of $(n-k)$ unit vectors which are not in the basis for V.*

Note that according to our previous discussion, a schema is a coset of a subspace with a basis of unit vectors. Therefore the last part of theorem 3.2 applies immediately to it. A characteristic function of a schema is an example of a constant function as described in theorem 3.2. The following corollaries [46] are immediate consequences of the last theorem and describe the Fourier transform of the former.

Lemma 3.2 *(Corollary 1) Let $V+c$ be any coset of any k-dimensional subspace V of Z^n, with c belonging to the nullspace of V, denoted V'. Then the characteristic function $\chi_c(x)$ (which is equal to 1 on $V+c$ and 0 elsewhere) has the Fourier transform*

$$\chi_c^*(w) = 2^{k-n}\delta_{\omega_{2}0}(-1)^{\omega_1 c^T}$$

where $\delta_{\omega_{2}0}$ is 0 unless $\omega_2 = 0$.

If we let $c = 0$ in the above corollary we have immediately

Lemma 3.3 *(Corollary 2) The characteristic function $\chi_V(x)$ of a subspace V of dimension k has the Fourier transform $\chi_V^*(w) = 2^{k-n}$ on V' and 0 elsewhere.*

The results so far presented stress the importance of the relationship between k-dimensional subspaces, their corresponding nullspaces and their translation by elements of the latter. Corollary 2 is especially interesting for our purposes since it gives the Fourier transform of the characteristic function of a schema. Based on the above results one can express (see e.g. [46] for details) the sum of $f(x)$ over a coset $V+c$ of V in terms of Walsh coefficients. Before embarking on more detailed discussions on this issue, let us use the results explored in this section to shed some light on the structure of some functions discussed in [21], which experimental evidence suggests are difficult for optimisation by a GA.

EXAMPLE 2

Consider the following function described in [21]:

$$f(x) = \psi_{11110}(x) + 2\psi_{11101}(x) - 3\psi_{11011}(x)$$

This function (in fact, a Walsh polynomial) has two special properties: (1) all the non-zero terms in the polynomial are of the same order; and (2) there exists a global optimum x' whose fitness receives a positive contribution from each term in the polynomial [21].

Let V' be the subspace generated by $\{\omega \mid f^*(\omega) \neq 0\}$. It is easy to see that the set $\{(1,1,1,1,0),(1,1,1,0,1),(1,1,0,1,1)\}$ is of rank 3 (i.e. the 3 vectors are linearly independent). V' is given by

$$\{(0,0,0,0,0),(0,0,0,1,1),(0,0,1,0,1),(0,0,1,1,0),$$
$$(1,1,0,0,0),(1,1,1,1,0),(1,1,1,0,1),(1,1,0,1,1)\}$$

Note that the vectors $(1,1,1,1,1)$ and $(0,0,1,1,1)$ are orthogonal to every element of V'. The subspace generated by these vectors is

$$V = \{(0,0,0,0,0),(0,0,1,1,1),(1,1,0,0,0),(1,1,1,1,1)\}$$

V being a subspace partitions Z^5 into 8 cosets. Theorem 3.2 states that all functions generated by the orthogonal set $\{\psi_\omega(x) \mid \omega \in V'\}$ are constant on V-cosets. Consider the coset of $V + c$ where $c = (0,1,0,1,1)$ and given by $\{(0,1,0,1,1),(0,1,1,0,0),(1,0,0,1,1),(1,0,1,0,0)\}$. An immediate calculation shows that $f(x)$ attains the value 6 for all the elements of this set. Incidentally, these are all global optima, implying that $f(x)$ has at least 4 of them.

3.7 Poisson Summation Theorem

According to Lechner [46] the Poisson Summation theorem is another important property of Fourier transforms and plays a major role in the theory of group characters. This theorem states that the average of a function's values over a subgroup H of the domain G may be computed by summing its Fourier transform over a convenient group with suitable normalisation. It provides a convenient way to express schema averages in terms of the Walsh coefficients. As pointed out by Lechner it can be applied in more abstract domains than the one considered in this chapter.

Theorem 3.3 *Poisson Summation Theorem (generalisation). If $V + c$ is a translation of any subspace V of Z^n of dimension 2^k, whose nullspace is V', and f is a real- or complex-valued function on Z^n, then*

$$\sum_{x \in V} f(x \oplus c) = 2^k \sum_{\omega \in V'} f^*(\omega)(-1)^{\omega c^T} \tag{1}$$

The Poisson summation theorem establishes the relationship between sums on subspaces of Z^n and those in the corresponding nullspaces. Note that the effect of the translation of the subspace is to multiply some of the Walsh coefficients by -1. This happens whenever the number of 1s covering the 1s in c is odd.

In his thesis Bethke [8] derived the above result, namely the Poisson Summation Theorem, by following a different approach. He restricted his analysis to the case when $V + c$ was a schema. Our purpose here is to show that this result can be embedded in a more general framework. By doing so we gain the immediate power of generalisation. As pointed out by Lechner [46], the above result can be extended to a direct sum of arbitrary cyclic groups. Thus this broadened foundation should be capable of providing further extensions of the simple GA and new theoretical insights about the domains for which their use should be most efficient. As an immediate application, we mention the analysis of deception (i.e. the study of the factors that cause a GA to converge to a suboptimal solution) for non-binary alphabets, to which the results provided by the analysis of deception on binary strings may be extended in a straightforward way.

In the next section we review schema analysis in the light of the concepts discussed above.

3.8 Applications

Recall that the order of a schema h, denoted as $o(h)$, is the number of 1s and 0s in h and that the definition length of h, which we shall denote as $d_l(h)$, is the distance between the first and last **defining positions** of h, i.e., those positions associated with bound variables of h. Analogous definitions were given by Bethke [8] for a binary string in the following way: the order of a binary string s, denoted as $o(s)$, is the number of 1s in s and the definition length, denoted as $d_l(s)$, is the distance between the first and last 1s in s.

His theorem 3.3.9 is a restatement of the generalisation of the Poisson Summation Theorem (where the function f is replaced by u, the schema utility), but the lack of a more geometrical insight provided by subspaces and their corresponding nullspaces makes the interpretation of his results less intuitive. As a consequence of this theorem he proved two corollaries with important consequences. As an application of the concepts discussed so far, we shall derive these results in what follows.

Lemma 3.4 *Corollary 3.3.10 (Bethke 1981). Let j be a binary string and h be any schema. If the order of j exceeds the order of h, then $f^*(j)$ does not affect $f(h)$.*

Proof of Lemma 3.4 *Note that $o(h) = n - o(\beta) = o(\bar{\beta})$. Therefore, the hypothesis forces $o(j) > o(\bar{\beta})$ which implies that $j \notin V_{\bar{\beta}}$ since it cannot be spanned by the unit vectors of $V_{\bar{\beta}}$. Consequently $f^*(j)$ does not contribute to the sum expressed in equation (1).*

Lemma 3.5 *Corollary 3.3.11 (Bethke 1981). Let j be a binary string and h be any schema. If the definition length of j exceeds the definition length of h, then $f(h)$ does not depend on $f^*(j)$.*

Proof of Lemma 3.5 *Let $d_{\bar{l}}(j)$ be the distance between the first and last 0s in the string j. Note that $d_l(\bar{j}) = d_{\bar{l}}(j)$ for any string j and $d_l(h) = d_{\bar{l}}(\beta)$. It follows by the hypothesis that $d_l(j) > d_l(h) = d_{\bar{l}}(\beta) = d_l(\bar{\beta})$; in other words, j cannot be spanned by the unit vectors of $V_{\bar{\beta}}$ that is, $j \notin V_{\bar{\beta}}$. Therefore $f^*(j)$ does not contribute to $f(h)$.*

Based on the above corollaries Bethke [8] suggested that if the Walsh coefficients $f^*(j)$ decrease rapidly with increasing order and definition length, then the location of the optimum may be predicted by those schemata with definition length no greater than m, and hence the function may possibly be easy for the GA to optimise.

The above results are very important because they provide some conditions that a function should have in order to have at least the chance of being searched successfully by a GA. One is therefore led to consider classes of functions that obey the conditions expressed in the above corollaries. The following theorem and example (quoted from [43]) illustrate the rapid convergence of Walsh expansions of smooth analytic functions and gives an idea of how fast the Walsh coefficients decrease for smooth analytical functions.

Theorem 3.4 *For any function $f(x)$, $x \in [0,1]$, having a continuous q-th derivative, the coefficients $f^*(\omega)$ of its expansion in a series of Walsh functions with index weight q satisfy the inequality*

$$|f^*(\omega)| \leq (2^{q^2+3q})^{-1/2} |max_{x\in[0,1]} f^{(q)}(x)|$$

where $f^{(q)}(x)$ denotes the q-th derivative of $f(x)$.

EXAMPLE 3

Consider $f(x) = \sin(2\pi x)$ $(x \in [0,1])$. The maximum coefficient of index weight 10 is given by

$$max_{x\in[0,1]} |sin^{10} 2\pi x| = (2\pi)^{10},$$

and hence by theorem 3.4,

$$max_{o(\omega)=10} |f^*(\omega)| \leq 2^{-65} \times (2\pi)^{10} < 2 \times 10^{-10} \ .$$

We hasten to point out that theorem 3.4 and the example by no means guarantee the complete success of a GA search on a smooth analytical function (a fact already recognised by Bethke [8])[4]. A high-order schema will receive contributions from a very large number of Walsh coefficients (as can be seen from equation (1)). From the point of view of the analysis considered in this thesis one should (whenever possible) establish conditions on a given class of functions that are ultimately checked by the Poisson Summation Formula in order to have some guarantee of success for the optimisation exercise. Unfortunately, this may itself be a very difficult task due to the calculations involved.

[4]It would be interesting to have a similar result relating the derivatives of $f(x)$ and Walsh coefficients with a given definition length. However, to the author's knowledge no such results exist.

3.9 Deception and difficult problems for a GA

There is a variety of reasons of why a GA can fail to converge to a global optimum. Most prominent among them are poorly-chosen parameter values, premature convergence, sampling noise and the function being such that the algorithm converges to wrong areas of the search space in the initial iterations (for further details see [51]).

In much the same way as above, one may be interested to understand what are the characteristics of a function that make this function difficult for genetic optimisation. As argued by Bethke [8], if the GA fixes the wrong genes early in the search, it is likely to concentrate its exploration efforts in the wrong part of the search space and miss the optima. Bethke used the Walsh-Schema transform to construct functions that mislead the GA. The rationale behind his approach was to choose Walsh coefficients so that short, low-order schemata had relatively low average fitness. He then proceeded to choose other coefficients in order to force the global optimum to be contained in them. Liepins and Vose [51] have extended Bethke's results by showing how to construct fully deceptive functions for binary alphabets on strings of arbitrary length.

Much of the discussions of difficult problems for a GA has been centred on the notion of deception. Recall from definition 2 in chapter 2 that two distinct schemata h and $h^{'}$ are said to be **competing schemata** if they have the same set of defining positions. Alternatively, two schemata are said to be competing if they have the same order, with *'s in the same positions and different fixed bits [51]. Using this definition one can sometimes state some conditions on the Walsh coefficients to check whether a given function is easy (or deceptive) for a GA. We give an illustration of this in the next example, which was suggested in [51].

EXAMPLE 4

Following Liepins and Vose, let h and $h^{'}$ be two competing schemata such that for all bounded variables i, $h_i^{'} = 0 \Rightarrow h_i = 0$. This implies that the coset leader associated with h, i.e, c, has a zero entry whenever the corresponding entry in the coset leader associated with $h^{'}$, i.e., $c^{'}$ is 0. Since we suppose that the schemata are competing, there must be at least one zero entry in c whose corresponding entry in

c' is 1. The difference between the corresponding schema averages is [using (1)]

$$f(h) - f(h') = 2^k \sum_{\omega \in V'} f^*(\omega)[(-1)^{\omega c^T} - (-1)^{\omega c'^T}]$$

If we suppose further that $f^*(\omega) = 0$ when $1 < o(\omega) < o(h)$ and if $f^*(\omega) > 0$ when $o(\omega) = 1$, then the above argument guarantees that $f(h) > f(h')$. Therefore $[(-1)^{\omega c^T} - (-1)^{\omega c'^T}]$ will be positive for at least one ω, and $f(h) > f(h')$.

Note that competing schemata are different cosets of the same subspace. For optimisation purposes it is important that the GA will be led eventually to explore areas that contain an optimum. When this does not happen the GA is misled and the function (or problem) is called deceptive for obvious reasons.

Definition 8 *[51] Let f be a function with global optima at $\{x^*, \ldots\}$. f is* **deceptive** *of order m iff there exists $x \notin \{x^*, ...\}$ such that when h and h' are competing schemata of order not greater than m, $x \in h \Rightarrow f(h) > f(h')$.*

We should stress that the characterisation of f always includes the particular encoding of the domain space. In this sense the above definitions should be used with caution, since a function can be deceptive with one particular encoding while it need not be so with a different one.

The issue of how difficult or hard a problem is for a GA has become increasingly important as GAs are applied to ever more diverse types of problems [21]. Some researchers such as Goldberg [29] have constructed high-order deceptive functions from low-order Walsh coefficients. Both his work and that of Forrest and Mitchell [21] were motivated by surprising results found by R. Tanese in her dissertation (as reported by Forrest and Mitchell) when she examined a class of functions that seemed to contradict Bethke's analysis and which, therefore, might have cast doubts on the assumptions of the Schema Theorem (from which Bethke's analysis is developed). Goldberg's results, however, rely on the use of Walsh coefficients of arbitrary definition length. Hence, they do not respect Corollary 3.3.11. With regard to the functions studied by Forrest and Mitchell [21] we have shown in example 2 that they may possess a peculiar nature, i.e. with the search space possibly partitioned into several sets on each one of which the function attains a constant value. Also, the low-order coefficients are all zero and hence by Bethke's argument the GA may fail to concentrate its efforts in good areas of the search space early in the run, or even fail to use any information or correct correlation to guide the search. This explanation is in line with Forrest and Mitchell's conclusions, where they identified another major cause for the GA's difficulty on Tanese's functions, namely the high degree of overlap among significant loci, which may cause most of them to be correlated [21]. In this particular set of functions such correlations appear to be detrimental to the GA performance.

One may enquire to what extent do these functions serve as good models for functions occurring in real-world applications. But the experiments serve to emphasise the point that, without a careful analysis about the characteristics of a given problem domain, a GA may fail consistently when used to explore it. The main focus of this thesis is to design systems in such a way that they can be explored efficiently by a GA. All other things being equal, Bethke's results currently remain the best guidelines with which to pursue this goal. One should also aim (when possible) to have a design that is balanced, i.e. where no particular feature can be dominant in the overall performance. Otherwise some candidate solutions might receive a much higher fitness and could take over the entire population in just a few generations, often with many undesirable genes "hitchhiking" along with those that do contribute effectively to the overall performance. In general, however, the choice of features for a proposed system depends on the designer and hence their appropriateness may only be checked at an experimental level.

As a final remark, note that the Poisson Summation Theorem applies to any subspace V of Z^n. Given a function and its corresponding encoding, it is natural to enquire (in case it proves to be difficult for a GA search) what sort of transformations on the domain would render it fully easy. Since linear transformations map linear subspaces to linear subspaces, they form a nice candidate set of transformations with which to examine this issue. The topic is explored in more detail in chapter 5.

3.10 Summary

We have examined schema analysis from a broader viewpoint provided by group theoretic concepts, and have shown the equivalence of the results provided by this framework with results derived elsewhere that are regarded as significant in GA research. As pointed out in [46], the above proofs could easily be extended to a direct sum of arbitrary cyclic groups. Thus this broadened foundation should be capable of providing further extensions of the simple GA and new theoretical insights concerning the domains for which their use should be most efficient. In the appendix we extend the results discussed in this chapter to any non-binary alphabet.

Chapter 4
Epistasis and Genetic Algorithms

This chapter examines the concept of epistasis which is often used in informal discussions about the behaviour of GAs. As a means to quantify it in a suitable measure and relate to Walsh analysis, we use the framework developed in previous chapters and address the concept of epistasis variance as suggested by Davidor. We show that this concept is not capable of complete discrimination between functions that are easy and difficult for genetic optimisation. As a result of this exploration we describe explicitly a whole class of functions that occur in some signal processing applications, which can exhibit high degrees of nonlinearity and yet belong to the class of fully easy functions.

4.1 Motivations

In previous chapters we saw how the connection between the Walsh transform and schemata enabled Bethke to state some conditions under which a function would be easy for a GA to explore. Despite the promising nature of the overall approach, Bethke's analysis was not intended as a practical tool for use in deciding whether any given problem would be hard or easy for the GA. This follows from the observation that the computation of the Walsh coefficients requires evaluating the function being explored in its argument space - which, in general, is an infeasible operation (and it would be completely ineffective if the sole purpose of genetic search were to search for the global optimum!). Moreover, the analysis is restricted to representations of fixed length, which prevents its application to domains where more flexible representations are adopted [16].

The above discussion brings to light several important issues:

1. Can we find a simple alternative to assess quickly the possible difficulty a GA will face when confronted with a particular problem?
2. Can we state different conditions from those already explored by Bethke for the suitability of a particular domain for genetic search?
3. Can we devise a simple measure to discuss the above issues?

As GAs draw inspiration from the principles of genetic variation and natural selection, it is also of interest to examine the theoretical results derived by Bethke with respect to those already explored in the fields of genetics and population genetics. This might offer some insights to guide the eventual understanding of why evolution takes place in natural organisms.

On the other hand, it is quite natural to expect that concepts derived in classical genetics should have an immediate appeal for application in the analysis of GAs as well, even if that is done in a limited way.

4.2 Epistasis

One of the concepts most often referred to in discussions of convergence issues for GAs is that of **epistasis**. According to Kempthorne [44] the term epistasis was introduced by Bateson in 1908 to describe the situation when there are two loci and the genes of one locus suppress the effect of genes at the other locus. Rieger et al. [73] define epistasis as "a form of gene interaction whereby one gene interferes with the phenotype[1] expression of another nonallelic gene (or genes), so that the phenotype is determined effectively by the former and not by the latter when both genes occur together in the genotype". They point out that in population and quantitative genetics, the term epistasis is sometimes used to refer to *all* nonallelic gene interactions. Davidor [16] argues that epistasis is used to describe the situation where one gene pair masks or modifies the expression of another gene pair. According to him, when the epistasis of a chromosome is said to be high, it means that the expression of many genes is dependent on the presence (or absence) of other genes.

It is a known fact in population genetics (see for example [15]) that changes in the allele frequencies either by random drift or isolation can lead to local differentiation of populations. In both these cases gene frequencies are usually changed in an erratic way. It is thus of interest to consider the systematic changes brought about by selection mechanisms. Here it is natural to consider the changes in the proportion of the alleles in successive generations.

[1]According to the Oxford Dictionary a phenotype is a set of observable characteristics of an individual or group as determined by its genotype and environment.

The rate at which selection changes any property of a population depends on how much genetic variability for this property exists in the given population [15]. To simplify the discussion let us suppose that this is caused by the presence of a given allele. The difference between the average fitness of the population having a given property, and average fitness of the entire population is an important parameter, and is known as the **average excess** of the corresponding allele. Note that the average excess is the amount by which the weighted fitness of individuals containing a given allele exceeds that of the population as a whole [15]. Therefore one should expect that the rate of change of a given allele should be proportional to its average excess. In a remarkable result Fisher showed that if one considered the variance of the average excesses (otherwise known as the **additive genetic variance**) it would be possible to predict the increase in mean fitness in any generation. In this way, he showed that natural selection picks out the additive genetic component and changes the population mean in proportion to the variance of this component[2][15]. This is rendered even more remarkable by the fact that natural selection is acting on phenotypic differences of the individuals rather than on the genes themselves.

The above discussion motivates the idea that interactions between different loci that are additive can be explored efficiently by natural selection. Whenever nonadditive effects between loci occur, the general pattern is much more complicated to predict. The variance attributable to interlocus nonadditivity is called the **epistatic variance** and plays an important role in the analysis of quantitative traits in quantitative genetics. Therefore, it seems to be a natural candidate to help in further understanding of how GAs work.

Note how the mere introduction of the concept of epistasis shifts part of the emphasis of the optimisation exercise from the question of whether a particular domain is suitable for genetic search to one that examines whether a given representation of a problem promotes an efficient search by a GA [16]. Despite its importance and its widespread use in informal discussions, epistasis has proven to be rather elusive to quantify.

Recently, however, Davidor [16] has suggested the use of epistasis variance as a simple and computationally effective measure to discriminate between functions that are easy or difficult for genetic search. The rationale behind his idea is that GAs seem

[2]It is important to note that the average excess of an allele is not a constant, since it depends on the relative frequency of individuals in a given population.

to be a worthwhile alternative to other methods such as gradient methods when the search spaces are complex and nonlinear enough to preclude the applicability of the latter[3].

According to this interpretation, only certain degrees of nonlinearity enable a GA search to exhibit relatively efficient behaviour. In some cases the amount of nonlinearity even diminishes their performance. As discussed above, a key point in the GA search is the representation issue since GAs do not "see" the problem domain. Because of this, the question of whether a certain representation of a problem promotes an efficient GA is a highly important one [16]. The amount of interdependency among the representation elements is an important source of information for that issue. To help in reasoning about such interactions, the concept of epistasis is borrowed from natural genetics.

Basically, in our context, epistasis measures the mutual interaction of different elements of a representation. In some circumstances this can lead to better alternatives in the search space. In others it can produce less fit alternatives resulting from the combination of alternatives with higher fitness. An important point to bear in mind is that, according to the above argument, if we know the representation's epistasis then we may possibly be able to predict the efficiency of a GA search in that domain. For example, if a representation contains very little or no epistasis, then any individual element is almost or entirely unaffected by the value of the remaining elements. At the other extreme, if the representation is highly epistatic, then unless a uniquely fit combination of values is found in some single GA step, no substantial improvements in fitness can be expected. Here one's general intuition may suggest that neither extreme is natural territory for GAs.

These considerations raise the question of the problem domains for which a GA search is most suitable. Any clarification of this question or the mathematical tools with which it can be attacked is therefore likely to be very helpful in improving our understanding of the applicability of GA methods. As suggested by Davidor [16],

[3]The term "linear" deserves a more precise explanation. Consider the subset of Walsh functions with indices of order 1. These functions are also known as **Rademacher functions** [43]. It is possible to prove that any Walsh function can be expressed as a product of Rademacher functions. In other words, the set of Walsh functions is the multiplicative closure of the set of Rademacher functions [43]. Throughout this thesis a function will be regarded as linear if it can be expanded as a linear combination of Rademacher functions. In a sense, linearity here is related to the simplest decomposition of a given function in terms of its argument variables (see e.g. Example 1 in this chapter).

if "epistasis could be calculated for a given representation, then it would offer an important yardstick of its suitability to a GA".

Davidor [16] points to the connection between epistasis, Fisher's Theorem (to be discussed briefly in section 4.6.1), and the use of a linear decomposition as a means to measure the amount of nonlinearity (in terms of gene interaction) embedded in a given representation. However, Davidor fails to put his results in perspective to those derived by Bethke by suggesting (p. 118):

> There are possibly other methods besides the Walsh functions by means of which the suitability of a representation can be assessed. We propose ... a different approach to the suitability and applicability aspect.

The aim of this chapter is first to address all the above issues partly by showing explicitly how several quantities of interest in Davidor's proposal are related and by expressing them in terms of the framework provided by Walsh-function analysis. In this way we provide a bridge from Davidor's work to Bethke's results. Also we show explicitly that contrary to Davidor's assumptions: (*i*) there exist functions that are highly epistatic with respect to the proposed epistasis measure and are easy for GA exploration, and (*ii*) there exist functions that possess a large degree of nonlinearity and yet are easy for genetic search. It is important to emphasise that these functions are non-trivial and not artificial, e.g. they could well occur in some pattern-recognition problems.

In order to give maximum clarity to our observations, we use mathematical rather than intuitive arguments, with appropriate notation. This approach seems unavoidable if we are to obtain a framework where the above concepts can be discussed, criticised and extended objectively. However, to improve the readability of the chapter we transfer as much technical detail as is practicable to the final section (4.8).

4.3 Epistasis variance

In this section we review epistasis variance [16] in terms of the concepts developed in previous chapters. We begin this process by calculating schema averages for low-order schemata. As a means to provide a common language in which to compare epistasis variance and schema analysis, we recall the definitions provided by Davidor [16].

Let s be a string of length n. The ith position in s is referred to as the ith **locus** of s. The **allele** at locus i is the value that the ith element in s assumes. The **average allele value** for allele b at locus i, denoted $A_i(b)$, is the average value of $f(x)$ when $s_i = b$.

Let $h_{b_i} = (* * * \ldots * b * \ldots *)$ be a schema where b at position i, $i \in \{1, \ldots, n\}$, is the only fixed variable (which can assume either the value 0 or 1). It follows that

$$A_i(b) = 1/2^{n-1} \sum_{x \in V} f(x \oplus c),$$

where $V = \{x \in Z^n : x_i = 0\}$ and $c = (0, 0, \ldots, b, 0, \ldots, 0)$. It is an immediate consequence of Poisson Summation Formula that

$$A_i(b) = (f^*(0) + (-1)^b f^*(2^{i-1})) \tag{1}$$

It is easy to see from the definition of the Walsh transform pair that $f^*(0)$ is equal to $\bar{f}$, the average fitness value. Substituting this fact in (1) gives

$$f^*(2^{i-1}) = \frac{A_i(b) - \bar{f}}{(-1)^b}$$

Davidor [16] defines the **excess allele value** for allele b at locus i as $X_i(b) = A_i(b) - \bar{f}$. From the above discussion we see that

Lemma 4.1 *The excess allele value for allele $b = 0$ at locus i is given by the Walsh coefficient $f^*(2^{i-1})$* .

The excess genic value, the genic value, and the epistasis of a string s are given by the following expressions:

$$X(s) = \sum_{i=1}^{n} X_i(s_i), \quad A(s) = X(s) + \bar{f}, \quad \epsilon(s) = v(s) - A(s).$$

The epistasis variance, σ_ϵ^2, the fitness variance, σ_v^2, and the genic variance, σ_a^2 are defined as:

$$\sigma_\epsilon^2 = 1/2^n \sum_{s \in Z^n} \epsilon^2(s), \quad \sigma_v^2 = 1/2^n \sum_{s \in Z^n} (f(s) - \bar{f})^2, \quad \sigma_a^2 = 1/2^n \sum_{s \in Z^n} X^2(s) .$$

A simple calculation shows that $X_i(0) = -X_i(1)$ from which it follows that

$$X(s) = \sum_{i=1}^{n} (-1)^{s_i} X_i(0) \tag{2}$$

We use this relation in the following results:

Lemma 4.2 *The genic variance is given by* $\sigma_a^2 = \sum_{j=1}^{n} X_j^2(0)$.

Lemma 4.3

$$\sigma_a^2 = 1/2^n \sum_{s \in Z^n} f(s)X(s) \ .$$

With the above results and definitions it is easy to see that the difference between the fitness variance and the genic variance is the epistasis variance. We state this result in the following way:

Theorem 4.1 *Let* σ_v^2 *be the fitness variance,* σ_a^2 *the genic variance and* σ_ϵ^2 *the epistasis variance. Then,* $\sigma_v^2 = \sigma_a^2 + \sigma_\epsilon^2$.

The above result shows that the variance of fitness can be accounted for by two possible causes, namely the genic and the epistasis variances. Our main interest is to find a suitable measure for prediction of possible difficulty a genetic search would have on a given problem. Also, we are interested in comparing performances of a given genetic algorithm on different examples. The following definition therefore provides a normalised way to compare epistasis in different functions.

Definition 9 *The proportion of epistasis* P_ϵ *is given by* $P_\epsilon = \sigma_\epsilon^2 / \sigma_v^2$.

Lemma 4.1 and Lemma 4.2 give us a way to express the genic variance in terms of the Walsh coefficients of $f(x)$. To describe the fitness variance in terms of the Walsh coefficients we first expand the fitness variance in the following way:

$$\sigma_v^2 = 1/2^n \sum_{x=0}^{2^n-1} (f(x) - \bar{f})^2 = 1/2^n \sum_{x=0}^{2^n-1} (f^2(x) - 2f(x)\bar{f} + \bar{f}^2) = 1/2^n \sum_{x=0}^{2^n-1} f^2(x) - \bar{f}^2$$

Using **Parseval's Theorem** and noting that $f^*(0) = \bar{f}$, we can rewrite the above expression as

$$\sigma_v^2 = \sum_{\omega} f^{*2}(\omega) - f^{*2}(0) = \sum_{\omega=1}^{2^n-1} f^{*2}(\omega) \ .$$

From these results the following fact is easy to derive:

Theorem 4.2 *Let $f(x)$ be a given fitness function making up a transform pair with $f^*(\omega)$. Then the proportion of epistasis of $f(x)$ is established by the expression*

$$P_\epsilon = 1 - \frac{\sum_{\omega=1}^{n} \left(f^*(2^{\omega-1})\right)^2}{\sum_{\omega=1}^{2^n-1} \left(f^*(\omega)\right)^2} \tag{3}$$

4.3.1 Interpretation

If we consider a given function as a signal, equation (3) shows that if most of the power of the signal is in the linear components then the proportion of epistasis is low. Hence, according to the above discussion, it would follow that such a function should be easy for a GA to optimise. It also offers a possible alternative to deal with the cases when this does not happen. Consider for instance the case where most of the power of a signal is in a set of nonlinear components. If it would be possible to find some transformation on the search space such that it would induce a permutation of the Walsh coefficients of highest values and the linear Walsh coefficients, then the proportion of epistasis would be reduced and an easier problem for genetic search would possibly result. This motivates the search for transformations that permute the Walsh coefficients, a topic which will be addressed in chapter 5.

The discussions presented in this and previous chapters seem to indicate that a general pattern is emerging with regard to the design of architectures suitable for a GA to explore efficiently. One can be tempted to find a set of parameters, encode them in any way that seems convenient, run the GA and "see" the results. If the results are promising (according to whatever means one uses to judge them), they may well be pointing to interesting unforeseen possibilities. If they are not, the designer may reassess his encoding, try new genetic operators (which would probably be tailored to his particular application) or new explanations based on epistatic effects of the representation he has employed. The latter approach would be difficult in practice, since a good description of the search would be needed. The previous paragraph may offer an interesting alternative if it can be accomplished. Here one would require the GA to search also for a good encoding of the parameters.

A third alternative is to base the design on a conservative approach that would increase the chances of producing good architectures without too many risks. Recall that all alternative theoretical approaches currently available seem to agree that if a function can be decomposed into a sum of independent functions such that each can be explored efficiently through a GA, then the chances are that the original function will also be explored efficiently. This is by no means an absolute rule of success, since other effects such as premature convergence could lead the algorithm to a local optimum. However, since the same phenomena could occur with other different parametrisations, it stands as the best available design guideline to follow and is the alternative that we shall try to explore in this thesis. We next illustrate the ideas discussed in the previous sections with a few examples.

EXAMPLE 1

Consider an arbitrary l-bit, linear function, as given in Goldberg (1989b):

$$f(x) = b + \sum_{i=1}^{l} a_i x_i$$

where $x_i \in [0,1]$. The Walsh coefficients are given (according to Goldberg) by the following expression:

$$f^*(j) = \begin{cases} b + 1/2\sum_{i=1}^{l} a_i & \text{if } j = 0 \\ \frac{a_j}{2} & \text{if } j \in \{2^{i-1}, i = 1, 2, \ldots, l\} \\ 0 & \text{otherwise} \end{cases}$$

It is immediate to see that the substitution of the above values in equation (3) results in $P_\epsilon = 0$ as expected.

EXAMPLE 2

Consider next the *minimal deceptive problem* as given in [28][4]. As shown by Goldberg, although conceived to be deceptive a minimal deceptive problem can still converge to the correct optimum and is therefore not considered as hard for GAs. The following conditions express the relation between the Walsh coefficients for a minimal deceptive problem [28] (where for clarity of exposition we define $\omega_i \triangleq f^*(i)$):

[4]The minimal deceptive problem is the smallest possible deceptive problem that can exist. The distinction between type-I and type-II is related to whether $f(01) > f(00)$ or $f(00) \geq f(01)$ and can be appreciated better with a geometrical picture of the respective functions as shown in [28].

$$\begin{cases} \omega_1 > 0 & \text{and} \\ \omega_2 + \omega_3 > 0 & \text{for a type-I minimal deceptive problem or,} \\ \omega_2 + \omega_3 < 0 & \text{for a type-II minimal deceptive problem} \end{cases}$$

In both the above cases, the proportion of epistasis is given by the following equation:

$$P_\epsilon = \frac{\omega_3^2}{\omega_1^2 + \omega_2^2 + \omega_3^2}$$

It is easy to see from the above conditions that $P_\epsilon < 1$.

EXAMPLE 3

Finally, consider the following functions (given in [93]) where the first is a type-II minimal deceptive problem and the second a type-I minimal deceptive problem:

type-II		type-I	
string	Fitness	string	Fitness
00	4	00	4
01	3	01	3
10	1	10	5
11	5	11	1

The proportion of epistasis for the first case is 71.5% while for the second it is 25.7%, which is a much easier problem than the first one.

Although simple, the above examples may suggest that P_ϵ is indeed a good index to measure whether a given problem is difficult for genetic optimisation. That this is often not the case is shown in the next sections.

4.4 High epistasis does not imply GA-Difficult

Consider the following function defined in a $2n$-dimensional space:

$$f(x) = \sum_{i=0}^{n} c_i(1 - x_{2i} \oplus x_{2i+1}).$$

where all operations except $\oplus$ are done in real arithmetic. Evidently $f(x)$ has its highest values whenever pairs of bits are equal. We want to obtain the Walsh expression of $f(x)$. First note that if $x, y \in \{0,1\}, x \oplus y = x + y - 2xy$, where the operations on the right-hand side involve real arithmetic. Also note that Walsh expansions can be obtained from polynomial expansions over $\{0,1\}$ by the substitution:

$$y_i = 1 - 2x_i,$$

where $y_i \in \{-1, +1\}$. Using these facts, f can be expanded in the following way:

$$f(y) = 1/2 \sum_{i=0}^{n} c_i + 1/2 \sum_{i=0}^{n} c_i y_{2i} y_{2i+1} \tag{4}$$

We remark that (4) contains no terms with Walsh functions whose indices are powers of i. Thus it follows immediately from (3) that $f(x)$ is fully epistatic, i.e., $P_\epsilon = 1$. If we make the substitution $z_i = y_{2i} y_{2i+1}$, then f is expressed as a linear function of the components of z - an n-dimensional vector. It is a known fact [51] that such functions are easy for genetic search. The point is that crossover can easily combine the best pairs, and thus make $f(x)$ easy for genetic search. As pointed out by Davidor [16], the limitation of epistasis variance is that it cannot differentiate between different orders of nonlinearity. As a final comment we should like to add that $f(x)$ is not a particularly abstract example or one with little practical relevance, since it could well be an embedded component of some simple pattern-recognition problem.

4.5 Nonlinear problems can be fully easy

The results presented in the previous sections indicate that there are levels of interaction that cannot be discriminated by the simple epistasis measure. Unfortunately, a generalisation of the ideas discussed in this chapter to encompass all possible undesirable epistatic effects seems to be much more complicated.

From most of the relevant discussions of GA-based computations in the literature (e.g. see [8] and [51]) it is possible to gain the impression that problems that are fully easy are basically linear or have dominant features that are linear (although, to be fair, we have to mention that Liepins and Vose [51] also consider monotone polynomials, i.e. polynomials having all coefficients of like sign). For people interested in problem-solving techniques as manifested in nature, this may be read as a pessimistic message about GAs.

It seems somewhat ironic that the only reported results of easy problems explored by a technique that was introduced exactly to explore complex search spaces are related to linear or very smooth functions. It would be desirable therefore to examine whether there exist examples of easy functions that are intrinsically nonlinear.

We subsume all the above points in the following questions:

1. Is the degree of nonlinearity present in a given representation a necessary condition for the possible inefficiency of a GA in processing a particular application?
2. Can we find examples of functions that are highly nonlinear and yet are easy for genetic search?

Note that the building-block hypothesis (discussed in section 2.5) requires only that the low-order schemata should be able to point towards the optimum. Provided that this happens, a function may have a very large amount of nonlinear components and yet may be easy for a GA to optimise.

Consider for example a function such that **all** Walsh coefficients in its corresponding expansion are of like sign, which we assume for simplicity to be positive[5]. Let h_1 and h_2 be two competing schemata each of dimension 2^k with $0 \in h_1$. Then

$$f(h_1) - f(h_2) = 2^k \sum_{\omega \in V'} (1 - (-1)^{\omega c_2^T}) f^*(\omega) \quad ,$$

and since c_2 has at least one entry not equal to zero the last expression is always greater than zero. Thus f should be easy for a GA to optimise.

Next note that the Convolution Theorem (described in section 3.5) states that the Walsh coefficients of the convolution of two functions are equal to the product of the coefficients of the individual expansions. Let $f(.)$ be a function whose Walsh expansion does not include any linear coefficients equal to zero. Let its **dyadic autocorrelation** function be defined as $D_f(x) = \sum_z f(z)f(z \oplus x)$, where z ranges

[5]We can in principle relax the hypothesis by requiring that all Walsh coefficients corresponding to the linear components (i.e. with $o(\omega) = 1$) should be greater than zero and $f^*(\omega) \geq 0$ otherwise.

over the entire domain of the search space[6]. It follows from the above discussion that all the linear Walsh coefficients of D_f are positive. We express the consequence as follows:

Conjecture 4.1 *Dyadic autocorrelations may be easy for genetic optimisation.*

It is important to note that the dyadic autocorrelation can exhibit wide oscillations and be very nonlinear. Hence they offer a counter-example to the question posed at the beginning of this section. Moreover, by the same arguments as above, if two functions are such that corresponding Walsh coefficients have exactly the same sign and all linear coefficients are nonzero, then their convolution should be easy for genetic search. The above discussion may illuminate some recent comments by Schaffer et al. in [80] (p. 443), who in connection with the design of multiplierless filters point out:

> ... that GAs can solve these cascaded designs at all is somewhat surprising given the extreme amount of epistasis that must be introduced by the convolution of the coefficients (genes) (Davidor 1991).

4.6 Discussion

It would be interesting at this stage to contrast the results obtained above with those explored in population genetics. This comparison not only illuminates the motivation for introducing the concepts discussed in this chapter but aims to put them (and those obtained by Bethke [8]) in perspective to those already known in genetics. It must be said, however, that the analysis pursued by the latter always considers a very large (sometimes infinite) population, as opposed to the practical consideration of relying on finite and small populations in the application of GAs.

4.6.1 The Fundamental Theorem of Natural Selection

With no recombination and under selection, the Fundamental Theorem of Natural Selection (also known as Fisher's Theorem) states that the average fitness of a

[6]The use of autocorrelation functions within the engineering area is fairly standard, and Harmuth [31] presents a typical discussion of some applications.

population increases from generation to generation (see [36] for details). A standard interpretation for this theorem (following [15]) is:

> The relative (geometric) rate of increase in mean fitness in any generation is approximately proportional to the additive genetic variance of fitness at that time.

Fisher's Theorem not only provides a good motivation for evolutionary techniques such as GAs, but also gives some suggestion on how to monitor the progress of mean fitness in a given population. The latter fact serves as a good motivation for the introduction of the concepts explored in this chapter. However, when the recombination rate is not zero the above theorem does not hold [36]. In fact if the fitness function depends on more than one locus, the Fundamental Theorem of Natural Selection need not be valid. For some special cases, though, it is possible to prove that the average fitness does increase. This happens, for instance, if the fitness depends additively upon the contribution of the loci [36]. Therefore, the idea that linear functions (or linear-dominated functions) should be easy for genetic search has some support from the mathematical treatments in classical genetics.

It is particularly pleasing to see how a seemingly different approach such as Walsh analysis provides the same conclusions. So it is natural to advocate the use of techniques similar to those applied in classical genetics (e.g. epistasis variance) to help in the discrimination of the domains suitable for GAs.

Some words of caution are necessary. Fisher's Theorem is applicable on a generation-to-generation basis. Hence, it does not follow that the additive genetic variance at the first generation will give all the necessary information to assess the increase in mean fitness in successive generations.

4.6.2 The Fundamental Theorem of Genetic Algorithms and static analysis

In the same way as with Fisher's Theorem, the Schema Theorem is only directly applicable to a single generational cycle [30] and hence should be used with great care. In fact, Grefenstette [30] emphasises strongly the recommendation that efforts should be made to divert the emphasis from static analysis of functions to the dynamics of GAs. Recent steps in this direction have appeared, most notably in the work of Vose and coworkers (e.g. [63, 91]). These studies establish an exact model for real GAs in the form of a Markov chain and relate the trajectory followed by finite populations to the evolutionary path predicted by an infinite-population model. They also show that at least in the infinite population case, GAs should not be thought of as global optimisers, due to the fact that populations can get trapped in some local basin of attraction. For large finite populations, Vose and Nix were able to prove that the evolutionary path of a real GA follows very closely, with large probability and for a long period of time, the path that is predicted by the infinite-population model. Moreover in [91] it is proved that a real GA will (with large probability) asymptotically arrive at that local minimum having the largest basin of attraction.

Although we welcome and recognise the broad implication of these results, an analysis such as the one above is out of the scope of this thesis, and in the opinion of the author it may seem difficult at this stage to envisage immediate practical gains to be made through it.

One of the purposes of this thesis is to build structures that can be tailored to efficient exploration by a GA. It is of particular interest in this work to use the current best theoretical results as guidelines (even in an informal way) for such a task. It is important to emphasise that we are not trying to solve *any problem* to optimality using a GA approach. We would be happy if we could. What we are aiming for is to explore design issues for general agent architectures in distributed artificial intelligence (DAI) such that they "do a good job" and serve as an initial framework for building more complex architectures. We also want to use this approach to advance the understanding of important issues related to DAI, in particular with respect to the Pursuit Problem (as introduced in section 6.5).

It is gratifying to note that this has been accomplished (as we shall show in later chapters) by basing the design guidelines on the "culture" or "general sense" provided by the theoretical investigations. Unfortunately we cannot claim to have any formal technical derivations to prove that they would work in general. However, the fact that the informal use of these ideas proved to be successful for the problems studied in this thesis may demonstrate experimentally that DAI can profit from the application of GAs as experimental tools to assist the design of different architectures.

4.7 Summary

We have explored epistasis variance as suggested by Davidor [16], using the concepts developed in previous chapters. We have shown in section 4.4 that it is not a fine discriminator of functions that are easy or difficult for genetic search, by exhibiting a function that is indeed easy for GAs to explore and which exhibits a high degree of epistasis according to the epistasis measure. In section 4.5 we have described explicitly a whole class of functions that are of relevance in some signal-processing applications, which can exhibit high degrees of nonlinearity and yet belong to the class of fully easy functions.

Several points about our analysis should be stressed. We have used the field's standard practice of evaluating a schema by averaging the payoff-function (fitness) value of strings that are contained in the schema. We recognise the limitations of this approach. For example, in practical applications the population is finite and this introduces a large amount of noise in the sampling process and therefore in the relations between the different variances. Further, our analysis does not take into account those cases where the fitness function varies because the environment in which the "genes" exist is not static and changes with time. However, the usual practice in theoretical genetics (e.g. [44]) is to express such changes by adding other variance terms to those that are given above. (This would complicate the formal treatment in Theorem 4.1 etc., but leave the basic line of reasoning unchanged).

Our intention is not to study the variance of fitness of individual schemata, but to explain the variance of a given function in terms of variances related to a linear regression on that original function, i.e. to adopt a point of view that is more intuitively related to typical statements of optimisation problems in nature. However, the exploration conducted in this chapter suggests that this is not the only class of problems suitable for GA optimisation. Essential non-linearity of a problem, whether in nature or as a theoretical exercise, does not automatically rule out GAs as potentially valuable tools for optimisation.

4.8 Proofs

Proof of Lemma 4.2.

$$\begin{aligned}\sigma_a^2 &= 1/2^n \textstyle\sum_{s\in Z^n} X^2(s) = 1/2^n\{\sum_{s\in Z^n}\sum_{i,j=1}^n(-1)^{s_i+s_j}X_i(0)X_j(0)\}\\ &= 1/2^n\{\textstyle\sum_{s\in Z^n}\sum_{i=1}^n X_i^2(0) + \sum_{s\in Z^n}\sum_{i\neq j}^n(-1)^{s_i+s_j}X_i(0)X_j(0)\}\end{aligned}$$

The last term on the right-hand side is equal to zero since for a particular pair i, j where $i \neq j$, the number of positive contributions is equal to the number of negative ones. The first term is equal to $2^n\sum_{i=1}^n X_i^2(0)$ (q.e.d.). □

Proof of Lemma 4.3.

$$\begin{aligned}\textstyle\sum_{s\in Z^n} f(s)X(s) &= \textstyle\sum_{s\in Z^n} f(s)\sum_{j=1}^n X_j(s_j)\\ &= \textstyle\sum_{j=1}^n\{X_j(0)\sum_{s\in Z^n, s_j=0} f(s) + X_j(1)\sum_{s\in Z^n, s_j=1} f(s)\}\\ &= \textstyle\sum_{j=1}^n X_j(0)\{\sum_{s\in Z^n, s_j=0} f(s) - \sum_{s\in Z^n, s_j=1} f(s)\}\\ &= \textstyle\sum_{j=1}^n\{X_j(0)\{(X_j(0)+\bar{f})2^n/2 - (\bar{f}2^n - (X_j(0)+\bar{f})2^n/2)\}\\ &= \textstyle\sum_{j=1}^n X_j^2(0)2^n = \sigma_a^2 2^n\end{aligned}$$

where in the above derivation we have made use of equation 2 and Lemma 4.2. □

Proof of Theorem 4.1.

Note that $f(s) - \bar{f} = \epsilon(s) + X(s)$. Therefore,

$$\begin{aligned}\sigma_v^2 &= 1/2^n \textstyle\sum_{s\in Z^n}(f(s)-\bar{f})^2 = 1/2^n\sum_{s\in Z^n}(\epsilon(s)+X(s))^2\\ &= 1/2^n\textstyle\sum_{s\in Z^n}\{\epsilon^2(s) + 2\epsilon(s)X(s) + X^2(s)\} = \sigma_\epsilon^2 + \sigma_a^2 + 2/2^n\sum_{s\in Z^n}\epsilon(s)X(s).\end{aligned}$$

By noting that $\epsilon(s)X(s) = f(s)X(s) - X^2(s) + \bar{f}X(s)$, and $\sum_{s\in Z^n} X(s) = 0$, substituting these facts in the last expression and using Lemma 4.2 we obtain the desired result immediately (q.e.d.). □

Chapter 5
Representation Issues and Linear Transformations

The description of the previous chapters emphasises the "representation problem" and the importance of finding representations which allow schemata to capture regularities and correlations that can be explored successfully by a GA. This chapter exploits the possibility of subjecting the representation itself to adaptation. The discussion is restricted to the set of linear transformations but may point to several problems to be faced when a similar approach is attempted in general. It also presents experiments with an alternative approach based on ideas derived from genetics, which may suggest future viable extensions of the basic algorithm.

5.1 Introduction

Since GAs have met with considerable success in many reported applications, one may wonder whether the failures reported (or merely hinted at) in others could not be due to an unfortunate choice of representation. If one considers the space of representations as a search space where one particular point corresponds to a particular chosen encoding, it is natural to conceive the possibility of using a GA to search that space for a good representation for the problem in hand. This could offer an extremely attractive possibility to explore difficult and yet poorly-understood problems. So far, this area has remained basically unpopulated with theoretical results, probably because of lack of a systematic approach.

One may wonder about the possibility of making something more systematic and useful that sheds some light on the few approaches undertaken so far. This chapter is a record of my experience in this area. What the chapter shows is that it is possible to be more systematic on these issues. The chapter also presents some small contribution to the technical picture. The overall message is that the problem itself is far from trivial and may prove to be rather difficult to tackle in its full generality. On the brighter side, it at least attempts to make that view clearer by trying to give a more technical picture of why this is so.

The chapter is laid out as follows. Section 5.2 reviews the problem of finding suitable representations, and presents previous approaches in the GA literature for dealing with the problem. Section 5.3 provides historical motivations for the use of linear transformations and reviews their use in other areas of knowledge as a means to put the results in the GA literature into perspective with them. Since (as shown in previous chapters) the theoretical progress that has occurred with respect to identifying the domains of applicability of GAs has relied basically on concepts of group theory, it is natural enough to use the latter in a first attempt to understand the structure of the space of transformations considered in this chapter. Section 5.4 therefore presents a quick review of the basic concepts and results in group theory. As it is known that every finite group is isomorphic to a permutation group, section 5.4.1 gives an overview of this important group-theoretic concept. It makes explicit the action of Gray codes that are so often cited in the GA literature, and it also lays the foundations for showing the relation between Walsh-Hadamard matrices, Walsh functions (section 5.4.2) and linear transformations (section 5.5). This is in my opinion an aesthetically pleasing result, since it shows that the Hadamard matrices encapsulate all the necessary information to generate the General Linear Group on the field of two elements (5.5.1). From a practical point of view, it suggests an explicit way to generate different sets of linear transformations that act on the search space. Section 5.6 explains how the idea of equivalence relations and the equivalence classes generated by them can be mapped to concepts of group theory, while section 5.7 offers some technical results upon which I conjecture that the hope that this space of transformations can in general be searched effectively via a GA may be very difficult (if at all possible) to achieve.

The modifier-gene idea has been used in biology for some time (e.g. see [42]). Informally speaking, a **modifier gene** is a gene that does not contribute directly to the evaluation of the fitness function but may influence the performance of the GA by conveying within itself information about important parameters for the algorithm, such as mutation or crossover rates. Section 5.8 reports on some experiment to assess the extent to which these ideas might be beneficial for future extensions of the traditional GA approach.

5.2 Motivations

In previous chapters we have emphasised the importance of finding representations which allow schemata to capture regularities and correlations that can be explored successfully by a GA. Although Holland [37] has suggested subjecting the representation itself to adaptation, there is still little theory surrounding good representations for genetic algorithms [70]. Certainly there exist implementations that alter the representation in the course of the search, e.g. the ARGOT Strategy [83], Delta Coding [59] and Dynamic Parameter Encoding (DPE) [81]. These techniques share some similarities among themselves by explicitly restricting search to promising areas in the search space [59]. DPE for example uses convergence statistics derived from the GA population to control the mapping from fixed-length binary genes to real values adaptively [81][1]. The way it works is as follows: for every Gray-coded parameter, convergence statistics are collected on the schemata $0\#\#\ldots, 1\#\#\ldots$, and $\#1\#\ldots$. Note that these define three overlapping "target intervals", each half as big as the original range of the parameter. Whenever the population converges on one of these target intervals, all the genomes are changed essentially via a leftward bit shift on them. The parameter range is changed as well, such that half the search space (the target interval) is searched at twice the resolution from that stage onwards.

Although these techniques may offer empirical evidence of improved efficiency in a set of given problems, they do not operate in a self-adapting manner as suggested by Holland. It is also difficult to foresee the sort of theoretical treatment to which they are amenable.

On the other hand, not all problems find natural expression as binary or k-ary strings [16, 70]. It would therefore seem desirable to abstract the basic notions and conventional understanding of GAs and extend them to non-string representations. One such approach was undertaken by Radcliffe [70] who considered the idea of arbitrary equivalence relations and the replacement of schemata by their equivalence

[1]The ARGOT strategy works in a similar fashion. It consists basically of a substructure roughly equivalent to a simple GA, plus numerous triggered operators which modify the intermediate mapping that translates the chromosomes into trial parameters. These triggered operators are controlled through measurements performed upon the chromosome population or the population of trial solutions [70].

classes[2]. In doing so, he hoped that a significant generalisation of the formalism underpinning GAs would be accomplished, and insights into general principles for designing representations and operators for new problem domains could be gained. The rationale behind his idea is based on the recognition that schemata can be identified as the equivalence classes of a set of equivalence relations and that the Schema Theorem is equally valid under the given interpretation.

Although promising, the above approach does not clarify how to accomplish Holland's goal. The idea of constructing new representations that are more suitable for a given domain is appealing. However, this formulation is subject to the same problems already observed in conventional binary representations [70]. In the latter case the problem is not a product of an unfortunate selection of a particular mapping. Any conceivable mapping from a 2^n dimensional space onto the reals would suffer from the same deficiency, since no mapping exists that conserves *Hamming distances* [67]. We may therefore conjecture that the same problem would occur with any similar approach.

Radcliffe [70] has applied the above approach to a well-known domain (the De Jong standard test suite of functions [28] which are used throughout the GA literature to compare and test the performance of different implementations of GAs). He reported very promising results with this approach when compared to the standard GA. Here he has used the concept of *locality* as a guiding principle to establish suitable equivalence relations.

With regard to the above, there are some important points to be stressed. First, the above approach does not shed any light on how to tackle the problems observed with binary representations that, as pointed out above, are likely to occur with it as well.

Also, it assumes some knowledge about the problem domain being explored. Part of the appeal of GAs comes from their simplicity and elegance as algorithms and their power for solving complex problems in poorly-understood multidimensional spaces. If those spaces are well-understood and contain structure that can be exploited, it is conceivable that the basic GA will perform less efficiently than special-purpose techniques that exploit this structure [21] or even implementations of GAs that take

[2]It is a well-known fact [10] that every equivalence relation on a set S determines a partition of this set, and conversely each partition of S yields an equivalence relation.

advantage of this information. GAs are basically an interesting exploratory tool exactly when this knowledge is not available from the outset.

Besides, in general the idea of devising the necessary equivalence relations and suitable operators that act on the partitioned space might by itself be very difficult to implement. It is obvious that any such attempt implicitly assumes some knowledge about the search space and the equivalence relation that should relate chromosomes with correlated performance. There are many different ways with which a given search space can be partitioned. It is far from obvious how this can be accomplished effectively in a systematic way via the above approach.

At the same time, by looking at the GA as a rather simplistic version of natural evolutionary organisms we can understand how nature can cope with evolution when confronted with suboptimal convergence. It is therefore of interest to examine how the basic algorithm can be extended in such a way that the basic framework of adaptation remains unchanged.

The idea of using different transformations to encode the parameters was recognised in the the early days of GA research. Most prevalent among these encoding schemes were Gray codes. According to Goldberg [28], the reason for their early adoption is related to the fact that in a Gray code adjacent integers differ by a single bit (in other words, their Hamming distance is equal to 1) and in this way they would be able to avoid the well-known problem of *Hamming Cliffs.* Also, it was argued that the perturbation caused by many single mutations is usually small.

Given the above, it is conceivable that the recognition of the fact that no particular mapping could be successful across a whole spectrum of diverse problems might have led researchers to consider the possibility of subjecting the encodings themselves to adaptation. Gray codes belong to the class of linear transformations over the dyadic field. Also, we noted in chapter 3 that the Poisson Summation Theorem applies to any subspace V of Z^n. It is reasonable to enquire about transformations that can map subspaces into schemata. Linear transformations do map linear subspaces into linear subspaces and hence it is natural to examine the effect of such transformations on the search space.

One of the aims of this chapter is to investigate the possibility of finding suitable equivalence classes on the space of linear transformations as a means to search for a "natural encoding" for them in line with Radcliffe's proposal. The discussion here will focus on results from Group Theory because it furnishes at least some possibilities to start with. Moreover, the explorations of the previous chapters relied basically on concepts furnished by Group Theory, so it seems natural enough to attempt to base further extensions on them. Lastly, the characterisation obtained via this approach refers to fundamental aspects of the underlying structures and as such, in the opinion of the author, may explain the possible difficulties in attempting to submit this particular space to an efficient genetic search.

The next section briefly discusses applications of linear transformations (encodings) in other areas, in order to put the results presented here in perspective with those already in use elsewhere.

5.3 Historical background

The study of linear transformations (and the corresponding systems described by them) is of great interest in several different areas such as speech communication, data communication, acoustics [64], the study of random processes [65] and in several interrelated problems of estimation and modulation theory in the context of data communication [90].

The concept of linear operators describing linear systems is of particular importance in connection with complete systems of orthogonal functions [31]. Orthogonal functions such as Walsh and Fourier functions have been exploited in communications since its very beginning [31], and in the design of digital devices and Threshold Logic Synthesis [20, 43, 46, 60]. In the latter case spectral methods are used for certain optimisation problems such as the **linearisation problem**. This refers to the problem of representing a given system of logical functions as the superposition of a system of linear logical functions and a residual nonlinear part of minimal complexity [43]. The idea here is to transform the logical variables via the application of a suitable linear transformation such that a given complexity criterion is minimised and hence the corresponding circuit design can be simplified[3].

The idea of exchanging one set of variables for another, the variables of either set being linear homogeneous functions of the variables of the other set, is often of importance in analysis [61]. The study of groups of transformations that leave a function's output invariant is also of interest in the design and synthesis of logic

[3]It is interesting to note the similarity of this approach and the ideas discussed in chapter 4.

circuits [46]. Here, the problem of analysis of Boolean functions is to decide whether a given function belongs to some standard class for which special-purpose methods exist that are usually more efficient (e.g. in the complexity of the systems being synthesised) than universal methods of synthesis [43][4]. This parallels the study of the "simplification" of matrices, polynomial quadratic forms and various geometrical figures through suitable linear transformations [10] for which the notion of canonical forms and invariants is of central importance. The study of canonical forms and a complete set of invariants finds immediate application in the identification of linear time-invariant systems [69], where the search space is greatly restricted by focusing on representatives (which possess a "simple form") of the resulting equivalence classes under the given group of transformations.

It is tempting to apply the above ideas to the problem of subjecting the representation to adaptation. Unfortunately, such a task is not so straightforward here, since (in principle) the search space is completely arbitrary and the function being optimised is not known in advance. In addition, one can only hope to learn something about the search space as the search itself takes place.

The above begs some of the questions about what sort of equivalence relations one should consider when attempting to encode the space of linear invertible transformations. Birkhoff and MacLane [10] list some equivalence relations on the set of matrices over some field that arise naturally from various interpretations of a matrix. For example, they consider the relation of *similarity* of two matrices A and B where A is said to be similar to B if $B = PAP^{-1}$, P being non-singular. This relation appears in connection with studies of various matrix representations of a linear transformation of a vector space into itself. Note however, that in principle two transformations might be similar and yet produce very different permutations of the search space. In this way, one might make the search more effective for a GA while the other one might have the opposite behaviour. The question therefore is: is it possible to partition the space of linear transformations in a way that enables a GA to search efficiently through it?

[4]Within this context it is relevant to quote Lechner in [46] (p. 123):
The most important theoretical contributions of harmonic analysis to switching theory depend on the intimate connection between Fourier transforms and linear or affine operators (encoding transformations) on the domain and range of switching functions.

To make the above discussion more clear, it is convenient at this stage to discuss the action of linear transformations on the search space and the corresponding action on the space of Walsh coefficients using Gray Codes as an example. Before doing so, we introduce in the next sections some concepts that will be important in future discussions. We shall assume an understanding of group theory, and only state the definitions and results that are of interest. The reader can find more information in references [9, 47, 61, 34].

5.4 Groups

A **group** is a set G together with a law of composition (an associative binary operation) for which the following conditions are satisfied [47]:

1. **Closure:** for every ordered pair a, b of G there exists a unique element c of G, called the **product** of a and b and such that
$$c = ab$$
2. **Associative law:** if a, b, c are any three elements of G, then
$$(ab)c = a(bc)$$
3. **Unit element:** G contains an element $\mathbf{1}$, called the **unit element** such that for every element a of G
$$a\mathbf{1} = \mathbf{1}a = a$$
4. **Inverse element:** for every element a of G, there exists in G an element a^{-1} such that
$$aa^{-1} = a^{-1}a = 1$$

We shall be interested in those subsets of a group G that obey the group postulates. A **subgroup** H is a nonempty subset of a group G such that (*i*) it is closed under the associative law, (*ii*) the identity of G belongs to H and (*iii*) for each element $x \in H$, $x^{-1} \in H$. We shall write $H \leq G$ to indicate that H is a subgroup of G. If G consists of a finite number of elements, then this number is called the **order** of G; otherwise G will be said to be of infinite order. In both cases the order of G is denoted by $|G|$.

Let $H \leq G$. For $x \in G$ we shall write $Hx = \{hx : h \in H\}$ and $xH = \{xh : h \in H\}$. Hx and xH are called the **cosets** of H in G. Hx is a **right coset** and xH is called a left coset. G/H denotes the set of all (right) cosets of H in G and is also known as the **coset space** of H in G. The number of distinct right cosets of G, is called the **index** of H in G and is denoted by $[G : H]$. The importance of the concept of cosets is furnished by the following theorem [47]:

Theorem 5.1 *(Lagrange's Theorem):*

Let G be a finite group or order g. If H is a subgroup of order h, then

1. *h divides g (i.e. $g = nh$), and*
2. *n is equal to the index $[G : H]$, so that there exist decompositions of G into right and left cosets given by*

$$G = \bigcup_{i=1}^{n} Ht_i, \qquad G = \bigcup_{i=1}^{n} s_i H$$

The collections $\{t_i\}$, $\{s_i\}$ are called the right (left) **representatives** (also known as the right (left) **transversal** of H in G). Theorem 5.1 establishes a natural partition of the group G in terms of cosets of one of its subgroups. It is a known fact (see [47] for details) that this partition is associated with an equivalence relation defined on G. Whenever this occurs the given set can be expressed as the union of the distinct equivalence classes. In the present case, the equivalence classes are the right (or left) cosets of G. Theorem 5.1 therefore furnishes a natural partition of a given set in terms of its group structure.

The above discussion provides a method of dividing a group G into equivalence classes relative to a subgroup of G, but in general the set of right and left cosets may be different. When these sets are the same, the subgroup that defines them possesses an important group theoretical structure: a normal subgroup. To state the concept in full generality consider, in any group G, the element $a^{-1}xa$. $a^{-1}xa$ is known as the **conjugate** of x under the operation of **conjugation** by a. It is possible to prove [47] that conjugation establishes an equivalence relation on a group G, and hence that the latter can be partitioned into disjoint classes. The class of all elements that are equivalent to a given one under the operation of conjugation is known as a **conjugacy class**.

Let $a^{-1}Ha$ denote the set of all products $a^{-1}ha, h \in H$.

Definition 10 *A subgroup H of G is* **normal** *if $a^{-1}Ha = H$ for every $a \in G$ (i.e. it contains, along with any element, all its conjugates). We shall denote $N \lhd G$ to indicate that a subgroup N is normal in G.*

Normal subgroups are most interesting from a group-theoretic point of view due to the fact that the collection of a normal subgroup can be endowed with a group structure. If $N \lhd G$ we define a multiplication of the cosets Nx and Ny as the coset containing the set $NxNy$ of all products $uv(u \in Nx, v \in Ny)$ [10]. It is possible to show that this coset is given by Nxy. The group of cosets of N is called the **quotient-group** (or **factor group**) of G by N.

It is easy to see that in every group G, the unit subgroup $\{1\}$ and G itself are normal subgroups of G. If a group of order greater than unity does not possess any normal subgroups other than these trivial ones it is said to be **simple**. Informally, a group G is simple if its structure cannot be taken apart by normal subgroups.

Normal subgroups are intimately related to single-valued transformations from a given group G into another group G' when these mappings preserve the group operations. Whenever this occurs, the transformation is known as an **homomorphism**. The following theorem [10] shows that the knowledge of all the normal subgroups of a given group G provides the necessary information about all the possible homomorphisms that can be generated on G.

Theorem 5.2 *The homomorphs of a given abstract group G are the quotient groups G/N by its different normal subgroups N.*

It is convenient at this stage to summarise the main results of this section. If a set G is endowed with a group structure then this set can be partitioned into equivalence classes via a natural group-theoretic approach in at least two different ways. The first one is provided by the cosets of any of G's subgroups while the other one emerges through the examination of the conjugacy classes. A most important concept here is the notion of a normal subgroup. The cosets of a normal subgroup of G partition G in such a way that they "preserve" group structure. A group G is simple if it does not possess nontrivial subgroups and hence, its structure cannot be examined by resolving it into simpler components.

Recall that the main interest here is to examine how to partition a given set (or search space). It is conceivable that if a given set possess a group structure, then one may take advantage of "canonical ways" to partition it using basic group concepts. This should not be taken to imply that these are the only alternatives to partition a given set nor that the partitions defined by these methods should be "optimal" to devise an efficient representation for exploration by a GA. However, the above methods may point to the difficulty of conceiving good equivalence relations on a given set precisely when these approaches prove to be ineffective.

Having said so, it is also important to stress that, as argued by Radcliffe [70], some of the equivalence relations must relate chromosomes with correlated performance, and that it should be possible to gather meaningful information on the performance of a given equivalence class by sampling chromosomes that instantiate it. In other words, even if a set can be partitioned into equivalence classes through the use of the above methods, one should check whether elements of a given equivalence class have correlated performance.

In order to gain further insights into the way a given transformation acts on the elements of a group, we review in the next section the concept of permutations which is central in the theory of groups.

5.4.1 Permutations

We shall be interested to study maps that act on a finite set Σ of objects. Following [47] the objects of Σ will be denoted by the integers $1, 2, \ldots, n$ and a map of Σ onto itself is called a **permutation** of **degree** n. It is described explicitly by the symbol

$$\pi = \begin{pmatrix} 1 & 2 & \cdots & j & \cdots & n \\ a_1 & a_2 & \cdots & a_j & \cdots & a_n \end{pmatrix}$$

where $a_j = j\pi$ is the image of j under π. A permutation that interchanges m objects cyclically is called a **cycle** of degree m. For example, if the set of objects $1, 2, \cdots, m$ are interchanged in a cyclical way, the corresponding permutation is described by

$$\gamma = \begin{pmatrix} 1 & 2 & \cdots & m-1 & m \\ 2 & 3 & \cdots & m & 1 \end{pmatrix}$$

To make the notation more compact it is customary to describe a cycle in a contracted notation. In this way, the above permutation is described by

$$\gamma = (\ 1\ 2\ \cdots\ m\ )$$

Two permutations will be called disjoint if they act on disjoint sets of objects. The set S_n of all permutations of n objects forms a group of order $n!$, called the **symmetric group** of degree n, the law of composition being that for maps of the objects onto themselves [47]. It is a known fact that every permutation α can be expressed as a product of disjoint cycles in an essentially unique manner, i.e.

$$\alpha = \gamma_1 \gamma_2 \cdots \gamma_r$$

where $\gamma_1, \gamma_2, \ldots, \gamma_r$ are disjoint cycles involving $m_1, m_2, \ldots, m_r$ objects respectively. The set of these integers is known as the **cycle pattern** of α. The following important fact links the cycle patterns with the conjugacy classes of S_n (for a detailed proof see [47]):

Theorem 5.3 *Two permutations are conjugate in S_n if and only if they have the same cycle pattern.*

Finally, we shall follow the convention that an object which remains fixed under a given permutation need not be mentioned explicitly in the representation of the given permutation.

5.4.2 Permutations on Hadamard matrices

A Hadamard matrix is an orthogonal matrix with elements +1 and −1 only [31]. Certain Hadamard matrices of rank 2^n are related to Walsh functions [31]. We shall be particularly interested in Hadamard matrices generated by the following process. The Hadamard matrices of rank 1 and 2 are:

$$H_1 = [1]\ ,\ H_2 = \begin{bmatrix} +1 & +1 \\ +1 & -1 \end{bmatrix}$$

Given a Hadamard matrix H_{2^n} of rank 2^n we generate a Hadamard matrix of rank 2^{n+1} as follows:

$$H_{2^{n+1}} = \begin{bmatrix} H_{2^n} & H_{2^n} \\ H_{2^n} & -H_{2^n} \end{bmatrix}$$

For example, the Hadamard matrices of rank 4 and 8 are

$$H_4 = \begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \end{bmatrix}, H_8 = \begin{bmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 & +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 & +1 & -1 & -1 & +1 \\ +1 & +1 & +1 & +1 & -1 & -1 & -1 & -1 \\ +1 & -1 & +1 & -1 & -1 & +1 & -1 & +1 \\ +1 & +1 & -1 & -1 & -1 & -1 & +1 & +1 \\ +1 & -1 & -1 & +1 & -1 & +1 & +1 & -1 \end{bmatrix}$$

Note that the number of positive and negative entries are equal, and that this number is constant on all lines (except of course the first one) and is equal to $2^{(n-1)}$. Given a Hadamard matrix of rank 2^n, consider the permutation by means of which each of its lines (except the first one) can be obtained from the line $2^{(n-1)}+1$ (i.e., the one where all the entries $+1$ precede all the entries -1) through column permutations such that the i-th positive (or negative) entry of line $2^{(n-1)}+1$ is mapped into the corresponding i-th positive (or negative) entry of the relevant line[5].

For example, line 3 is mapped onto line 2 of the Hadamard matrix H_4 by the permutation of columns 1 and 2. This permutation is expressed in cycle notation by $(1,2)$. In the same way, line 4 in H_4 is obtained from line 3 through the permutation $(1,3,2)$. Note that column 0 is always fixed by the above permutations since the first column of any such Hadamard matrix starts with a positive entry. Clearly, line 3 is mapped onto itself through the identity transformation.

It can be shown [54] that the above permutations form a group under the law of composition of mappings. Additionally, to any such permutation acting on the columns of H_{2^n} there exists a corresponding permutation of the rows of the resulting matrix that maps it back to H_{2^n}. It is easy to see that any permutation of the columns of H_{2^n} can be obtained by postmultiplying H_{2^n} by a permutation matrix Q. In the same way, any permutation of the rows of H_{2^n} can be obtained by premultiplying H_{2^n} by a permutation matrix P. Hence the following relation holds: $PH_{2^n} = H_{2^n}Q$.

[5]Columns are numbered from 0 to $2^n - 1$ and the line $2^{(n-1)}+1$ is scanned from left to right first for all positive entries and then for all negative ones.

5.4.3 Gray codes

There are a number of alternative ways to define Gray codes. The primary motivation for their use is related to the fact that in a Gray code adjacent integers differ by a single bit. For our purposes the following table of a 3-bit Gray code will be sufficient:

Integer	Binary	Gray
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

Table 5.1: A 3-bit Gray Code

We note from Table 5.1 that the 3-bit Gray code can be regarded as a $1:1$ function of $\{0, 1, ..., 7\}$ onto itself, so it acts as a permutation of the first 8 positive integers. Also 0 is mapped onto 0, so the permutation fixes 0 (the same holds for 1). Writing the 3-bit Gray code as a permutation in cycle notation we have the following:

$$(2,3)\ (4,6,5,7)$$

Since the above Gray code belongs to the class of linear transformations, it can be represented by a matrix over the dyadic field. To do so, one needs only know how the given transformation acts on the unit vectors. From the above table it is seen that:

$$\begin{aligned} 0\,0\,1 &\mapsto 0\,0\,1 \\ 0\,1\,0 &\mapsto 0\,1\,1 \\ 1\,0\,0 &\mapsto 1\,1\,0 \end{aligned}$$

Hence if an integer in the interval $[0, 7]$ is encoded in binary notation as a row vector x, its corresponding Gray code is $G(x) = xT$ where T is given by the following matrix:

$$T = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

We are interested to understand how the above permutation affects the Walsh coefficients. One way to see that is by reference to Table 5.1 where the first column corresponds to the Walsh function with index 3, while the second column corresponds to the Walsh function with index 6 and the third to the Walsh function with index 4. Therefore it is easy to see that if $f^*(\omega_i)$ denotes the i-th Walsh coefficient of a function $f(.)$ and $g^*(\omega_i)$ denotes the i-th Walsh coefficient of a function $f(G(.))$ the following relation holds:

$$\begin{aligned} g^*(\omega_1) &= f^*(\omega_3) \\ g^*(\omega_2) &= f^*(\omega_6) \\ g^*(\omega_4) &= f^*(\omega_4) \end{aligned}$$

From the discussions of chapter 4, if a given function is such that the Walsh coefficients of largest magnitude are $f^*(\omega_3)$ and $f^*(\omega_6)$ then encoding the domain through a Gray code will result in a function with larger linear coefficients and hence with a lower proportion of epistasis. Although (as seen in previous chapters) nonlinearity is not a necessary condition for hard GA problems, in general this "linearisation" of the search space may render the genetic search more effective.

Since Gray codes are associated with linear transformation it is natural to extend the above analysis to include all linear codes, and to examine the problems one may face when confronted with the problem of devising an algorithm that subjects the representation itself to adaptation.

5.5 Linear transformations

It is a well-established fact [10] that all non-singular linear transformations of an n-dimensional vector space form a group also known as the **General Linear Group**. Translations form another important group. An **affine** transformation can be viewed as a linear transformation followed by a translation.

One element belonging to this group is described by $xA + b$, where A is a matrix and b a vector with entries in the 2-element field $\{0, 1\}$ and all operations are done over this field[6]. If we look at the expression of schema averages in terms of the

[6]In this case the General Linear Group is denoted as $GL(n, 2)$.

Walsh coefficients, we see that we might be able to transform a difficult problem into one that could be manageable by the genetic search if we could permute the Walsh coefficients suitably. Hence, it would be interesting to study how a given linear transformation in the domain (referred to as $X = Z^n$) affects the corresponding Walsh coefficients. With these preliminaries we state an important theorem related to the Walsh Transform of affine transformations (see [46] for details of the proof).

Theorem 5.4 *(Domain Encodings).*

Let $g(x)$ be a known function from X into $\Re$. If $f(x)$ is defined for all x, by $f(x) = g(xA \oplus c)$, then the Walsh Transforms of f and g are related as follows:

$$\begin{aligned} f^*(\omega) &= (-1)^{\omega d^T} g^*(\omega\ A^{-T}) \\ g^*(\omega) &= (-1)^{\omega c^T} f^*(\omega\ A^T) \end{aligned}$$

where A^{-T} is the transposed inverse of A and $d = cA^{-1}$.

Following [46], we can examine the effect of the affine transformation on the Walsh coefficients by separating the action of the linear transformation and the vector translation by c. The action of vector addition by c is to permute elements of X in pairs, i.e., x is permuted with $x \oplus c$. The effect on $f^*(\omega)$ of this transformation is to change the signs of those transform coefficients $f^*(\omega)$ for which ωc^T has odd parity. When A permutes X, its transposed inverse A^{-T} permutes the Walsh coefficients. It is a well-known fact from group theory that A and A^{-T} have the same cycle structure [46]. Next, consider the effect when A is restricted to be a permutation matrix P. It is easy to see that any subset of weight k (i.e., exactly k bits are equal to 1 while the remaining ones are equal to 0) is closed under variable permutations $x \mapsto xP$, but not under variable complementations ($x \mapsto x \oplus c$) or symmetry transformations ($x \mapsto xP \oplus c$). On the other hand, the corresponding set of points of weight k in the transform space of Walsh coefficients remains closed under general symmetry transformations [46].

Goldberg [29] has shown how to construct fully deceptive functions with coefficients no higher than third order (i.e., with weight less than or equal to 3). This is done by requiring that all order-i Walsh coefficients should be equal. Using this fact it is immediate to see from the discussion in the previous paragraph that symmetry transformations are not powerful enough to overcome deception. An alternative proof is given by Liepins and Vose in [51] following a different approach. In that work, they pointed out that affine transformations are sufficiently powerful to transform selected deceptive problems into easy ones when the encoding is binary, but noted that addition of a constant vector does not overcome deceptiveness (since competing schemata are cosets of the same subspace and one is obtained from the other by a proper translation)[7].

Specifically, Liepins and Vose in [51] considered the following class of functions. Let $\pi(n)$ denote the n-fold Cartesian product of the integers modulo 2 (otherwise known as the dyadic group). For $n > 2$, define a fully deceptive function f as follows:

$$f(x) = \begin{cases} 1 - \frac{1}{2n} & \mathit{for}\ x : o(x) = 0 \\ 1 - \frac{1+o(x)}{n} & \mathit{for}\ x : o(x) \neq 0, o(x) \neq n \\ 1 & \mathit{for}\ x : o(x) = n \end{cases}$$

They showed that the above function could be transformed into a fully easy function g by means of an affine transformation $g(x) = f((1 \ldots 1) \oplus xM)$ where M is a zero-one invertible matrix defined by:

$$m_{ij} = \begin{cases} 0 & \mathit{for}\ i = j;\ i, j = 1,\ \ldots\ ,\ n-1, \\ 1 & \mathit{otherwise.} \end{cases}$$

What is even more remarkable is the fact that the inverse of M also transforms the original function into a fully easy one as well! We do not have a formal proof to substantiate this claim, but a simple check shows that this holds for the case $n = 3$, while computer simulations with $n = 32$ suggest that it may also hold for other (if not all) values of n. Note that M establishes a permutation of the points of the search space and the action of M^{-1} can be derived easily from the action of M. If M fixes a given point of the hypercube (i.e. $xM = x$ for a given x), then so does M^{-1}. Moreover, it is easy to see that that if $\gamma = (a_1 a_2 \ldots a_m)$ is a given permutation, its inverse γ^{-1} has the same cycle pattern of γ. Since any permutation can be resolved into a product of disjoint cycles, theorem 5.3 guarantees that M^{-1} is conjugate in S_n to M. In other words, there exists a nonsingular matrix P such that $M^{-1} = P^{-1}MP$.

Although restricted to a small class of deceptive problems, Liepins and Vose [51] suggested that similar results might also be obtained for other classes of difficult functions. The above result was derived with a complete knowledge of the fully

[7]As a result of that, if deceptiveness is to be overcome by an affine transformation this should happen due to the action of an element of the group of invertible linear transformations.

deceptive function, so it is natural to ask whether this could be obtained via some adaptive technique. They also discussed the possibility of implementing a metalevel search with respect to inversion operators. Therefore, although not explicitly stated, their analysis may have left the impression that such a goal could be accomplished on the set of linear invertible transformations. One may argue that only a small set of linear transformations should be necessary, as the number of possible linear transformations is very large. The question that remains to be answered is how one should decide on the choice of this particular set. The next section shows that the transformations examined in section 5.4.2 provide at least one alternative to start with.

5.5.1 A set of generators for the General Linear group $(GL(n,2))$

As shown in [47], every finite group is isomorphic to a permutation group. In section 5.4.2 we have produced a set of transformations that are intrinsically related to certain classes of Hadamard matrices (and therefore to the corresponding set of Walsh functions). These permutations form a group under composition of mappings, and each such permutation is characterised by the fact that its action on the columns of the Hadamard matrix can be matched by a corresponding action on the rows of the same matrix. On the other hand, theorem 5.4 shows that an element of $GL(n,2)$ permutes the elements of the domain in such a way that it induces a permutation on the corresponding Walsh coefficients.

Since the rows of the relevant Hadamard matrix correspond to Walsh functions, it is tempting to enquire what is the relation between these two groups. Computer simulations with CAYLEY[8] show that the set of permutations generates $GL(n,2)$. The proof of this fact can be set out in a rigorous way [48]. As an example, consider again the Hadamard matrix H_4 and the corresponding permutations described in section 5.4.2. The permutation given in cycle notation as (1 2) can be represented by the matrix $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ while the permutation (1 3 2) can be represented by the matrix $\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$.

[8]A computer system for symbolic (algebraic) manipulation, with emphasis on groups and group theory [13].

5.6 From cosets to equivalence relations

In the previous sections we have examined two group-theoretical ways to establish a partition of a group. It is natural to enquire how one might obtain a specified representation of such domains given these alternatives. Consider first partitioning a given group G by the cosets of one of its subgroups, say H. H is itself a group, so one may consider that it could be partitioned in the same way provided H contained a different subgroup K. It is easy to see that K partitions G and that if $K \leq H \leq G$ and $[G : K]$ (the index of K in G) is finite then $[G : K] = [G : H][H : K]$ as is intuitively expected.

Note that in this approach we are trying to establish a series of nested subgroups of a group G each of which has greater resolution than the previous one in the series. Each such series is designed to shed some light on the structure of G. As argued by Ledermann in [47], it is a common practice in mathematics to study complex entities by resolving them into simpler components which are "irreducible" (e.g. integers are factorised into primes and polynomials are split into irreducible factors [47]). Ledermann points out that in order to be significant such a resolution must display features of uniqueness which correspond to intrinsic properties of the structure being investigated. This is the basic motivation for the study of series of subgroups, in particular the concept of **composition series**.

A composition series consists of a sequence $\{A_i\}$ of subgroups of a given group G such that $G \rhd A_1 \rhd A_2 \rhd \ldots \rhd A_r \rhd \mathbf{1}$ and where $G/A_1, A_1/A_2, \ldots, A_r/A_{r-1}, A_r$ are simple groups. Recall that a group is simple if it is of order greater than unity and has no nontrivial normal subgroup. The quotient groups $G/A_1, A_1/A_2, \ldots, A_r/A_{r-1}, A_r$ are called **composition factors** [47]. A group may possess more than one composition series. However, the following theorem asserts that the set of composition factors constitutes an intrinsic property of the group [47]:

Theorem 5.5 *(JORDAN-HÖLDER) In any two composition series of a finite group the composition factors are, apart from their sequence, isomorphic in pairs.*

A finite group is said to be **soluble** if all its composition factors are of prime order.

With these ideas in mind we are ready to examine the possibility of finding a suitable equivalence relation on the elements of $GL(n, 2)$. For lack of any further knowledge we shall base our arguments on the theoretical tools explored in this chapter. We want to stress again that this might be questionable since we are severely restricting the equivalence relations to only those provided by the group structure. For example, Radcliffe [70] has examined representations for a certain class of real-valued problems based on the concept of locality. Many of his equivalence classes had the same number of elements. At the other extreme we shall be concerned only with nested sequences of subgroups, in particular composition factors (much in line with the concept of schema). What we are trying to investigate is the possibility of using some structural property of $GL(n,2)$ provided by insights gained from group theory such that it can help in the design of a GA on the space of linear invertible transformations. In doing so, we hope that the arguments provided in the next section can highlight some of the difficulties in applying a metalevel GA on the space of representations.

5.7 Metalevel GA on the space of invertible transformations

In this section we explain the reasons that have led us to abandon the pursuit of a metalevel search on the space of linear invertible transformations. The discussion is by no means complete (and certainly it lacks formal proofs) and therefore its conclusions can only stand as conjectures. However, it is hoped that it will serve as a baseline from which further theoretical and experimental insights can be gained.

Group theory tells us that the number of elements in $GL(n,2)$, which we represent by $|GL(n,2)|$, is given by $(2^n-1)(2^n-2)\dots(2^n-2^{n-1})$ [9]. This is equivalent to $2^{1+2+\dots+(n-1)}(2^n-1)(2^{n-1}-1)\dots(2-1) = 2^{n(n-1)}m/2$, where m is a positive integer not divisible by p. In general m is a very large number, but for our purposes it is enough to concentrate on the other factor on the right-hand side of the equation.

It is immediate to see that for all $n > 3, GL(n,2)$ is exponentially larger than the domain of f (the function being optimised). This implies that the task of searching for the optimum in a given search space is transformed into an immense search over the set of linear invertible transformations! Although this task may seem unreasonable at a first glance, let us stress that we are interested in the possible architectural extensions to the simple genetic algorithm that enables it to tackle more complex problems.

Our only hope for progress therefore relies on the exploration of the structure of $GL(n,2)$. First we note that if a linear invertible transformation is not trivial, then the number of points that it moves is quite large.

Theorem 5.6 *If an element of $GL(n,2)$ moves one point of the space on which it acts, then it moves at least half of the points of this space.*

Proof: Consider any linear transformation $g \in GL(n,2)$ and let $F = \{x : xg = x\}$ be the set of all points fixed by g. It is easy to see that F is a linear subspace and that the number of points belonging to it divides the number of points contained in $\pi(n)$. From the latter observation we know that this is equal to 2^n, so the number of points in the subspace should be equal to 2^s where $s < n$. This is at most $2^{n-1} = \frac{2^n}{2}$. Therefore the number of points that are moved by g is at least 2^{n-1} (q.e.d.) .□

It is obvious that what renders a function easy for genetic search is not the position of the points themselves but the simplification of the objective function from a hyperplane sampling perspective. Hence what is important is how the partitions are changed, and not the points themselves. What we are trying to emphasise here is that, unfortunately, in principle there is not an easy way to prescribe how any given linear transformation acts on the domain. Certainly this can be achieved for particular cases (e.g. the class of Gray codes examined is one particular case) but what we are interested in understanding here is to what extent these ideas can be implemented in general. Moreover, the calculation of how the transformation acts on the partitions may in fact be even more complicated than for the individual points and hence it appears to the author that in general this is a very difficult task.

The best one can aim for is to restrict the class of transformations, the action of whose elements is well-understood. The question then would be transferred to the problem of examining what are suitable candidates for this set. Or else, one should restrict the set of problems for which the technique is applicable. But in the latter case we run the risk of not tackling problems that could be solved by means of simple modifications of the basic algorithm.

We can at least have some idea about the action of an element of $GL(n,2)$ due to the fact that elements of $GL(n,2)$ preserve linear independence (and also linear dependence!). In the words of Biggs and White [9], $GL(n,2)$ is not quite **2-transitive** on the space on which it acts, since it has no element that takes a linearly independent pair of points into a dependent pair.

Can we devise a suitable crossover scheme between good transformations? (provided a convenient measure of fitness could be devised for each implementation of the simple GA together with the corresponding linear transformations). As discussed above, two transformations (one the inverse of the other) can exist such that each transforms a fully deceptive problem into a fully easy one. So simple matrix multiplication is by itself not a good candidate for implementing a crossover scheme, since in the above case the resulting permutation is the identity which would of course render the original problem fully deceptive.

In addition, matrix multiplication should not be regarded as a natural candidate for a crossover operator, because a sensibly designed crossover acting on equal individuals cannot produce a different offspring[9].

A different alternative (inspired by crossover used in the simple GA) could be based on the exchange of parts of the parent matrices to generate the offspring. Note however that the resulting transformations should be invertible. Checking whether this holds for any generated transformation is a very time-consuming process. (The same argument applies for a random mutation on the matrices.)

The problem here can be stated as follows: what (if they exist) are the building blocks of the space of linear invertible transformations? In an attempt to answer this question one could try to examine the group structure of $GL(n,2)$. Our previous discussions suggest that the examination of series of subgroups or composition factors, in the hope of partitioning the space in ever more refined partitions, is one way to proceed. Unfortunately $GL(n,2)$ is simple, and so at least this group does not possess a nontrivial composition series.

This does not mean that we are discarding the use of linear permutations, but simply pointing to the possible difficulty of implementing a successful general meta-level GA on $GL(n,2)$. Although in principle there is some freedom as to where to map the high-fitness points, the linear independence/dependence constraint should always be respected. In fact, any such transformation is specified by the way it maps n independent vectors into n other independent vectors.

[9]On the other hand, conjugation avoids this problem, as clearly $a^{-1}aa = a$. Also, conjugation of an element by its inverse results in the same element, and in addition it does not change the cycle pattern of the permutation. However, it is not clear what sort of correlation exists between two conjugate transformations that would make it useful to guide a genetic search. Also, in principle one may lose all the information about a when one goes to $a^{-1}ba$.

It is important to stress here that the problem we are facing is due to the fact that we are trying to be overambitious by simply placing all our bets on an algorithm that learns not only how to solve a given problem to optimality but also builds an adequate representation of the problem itself on this process. This is a very difficult problem for which there is no current solution.

5.8 Experiments with modifier genes

The discussions in the last section have led to a few different possibilities. One of them is to restrict the class of transformations to a small size and run multiple versions of the basic algorithm, each with its specific (and fixed) encoding transformation. It is conceivable that multiple runs of the algorithm, each one using a different encoding, may produce a better result than multiple runs using a fixed encoding.

The above may sound interesting in principle, although it does not cast any light on extensions of the basic framework that could accommodate the idea of self-adaptation of the representation. So how could one accommodate the idea of self-adaptation in the simple GA? Consider the alternative by which a gene is introduced in the basic chromosome such that it is associated with a linear transformation that acts on the other representation elements. Borrowing some nomenclature from studies in theoretical population biology (see for instance [42]), we shall refer to each such gene as a **modifier gene**. According to such a scheme, our "complete chromosomes" would also carry not only the representation elements but also the way to interpret them. Figure 5.1 depicts one individual chromosome that incorporates a modifier gene for the representation.

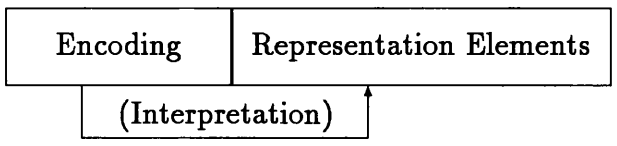


Figure 5.1: An extended chromosome with a modifier gene for the representation

Ideally one could hope that genetic search would not only find a good linear transformation of the parameter space but also the best set of parameters related to the problem itself. Unfortunately, even if we limit the size of the modifier gene this alternative may not work in general. To see that, consider the case where the function (with a particular encoding) to be optimised is fully deceptive at the origin and where the modifier gene has two values. When this gene is 0 the encoding is the original encoding that renders the function fully deceptive. When the modifier gene is 1 the encoding is the particular linear encoding (if it exists) that transforms the function into a fully easy one. Since the origin is fixed by any linear transformation, it will not be moved. If the linear transformation is such that it maps the highest-value vertices to points near the origin, the end result is that the goal of finding the optimum vertex may prove to be difficult. As an illustration of this possibility, consider the case of the Liepins and Vose fully-deceptive function described in [51].

One may be tempted to say that this undesirable feature is due to the fact that the origin has a preferential status with regard to the transformations considered here, since it is fixed by all of them. Hence, an affine transformation might be able to circumvent this problem. Note, however, that a different linear transformation may not map the highest-value vertices to points near the origin. It appears that in general predicting all the interactions provided by this approach is a very complicated matter.

In searching for a better alternative, we were drawn to the fact that experimental results from Schaffer et al [79] suggest that mutation may have a stronger role than previously admitted. Based on these results, we have introduced modifier genes that specify the mutation rates for each individual chromosome. Figure 5.2 illustrates how this was accomplished:

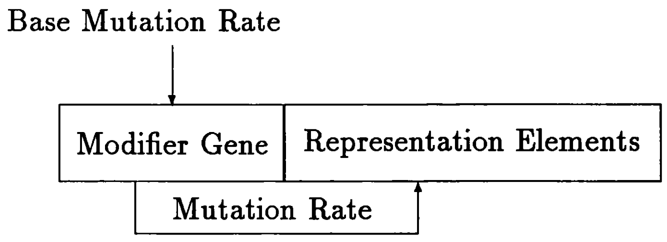


Figure 5.2: An extended chromosome with a modifier gene for mutation

In the above implementation the modifier genes are subjected to a fixed mutation rate (the **base mutation rate**). Each modifier gene comprises three bits and controls the mutation rate for the rest of its associated chromosome. Crossover was allowed to act normally on the whole structure. The experiments we report here were performed with GENESIS [82] using *sigma scaling* [21], *elitist* selection strategy [28] (whereby the best performing structure always survives intact from one generation to the next) and populations of 50 individuals. Two-point crossover was used, with a crossover rate of 0.6 per individual. The modifier's mutation rates were allowed to assume values ranging from 0 (i.e. the no-mutation case) to 200% of the base rate. The base mutation rate is given by:

$$\frac{0.5\sqrt{\frac{2.72}{ChromosomeLength}}}{PopulationSize} ,$$

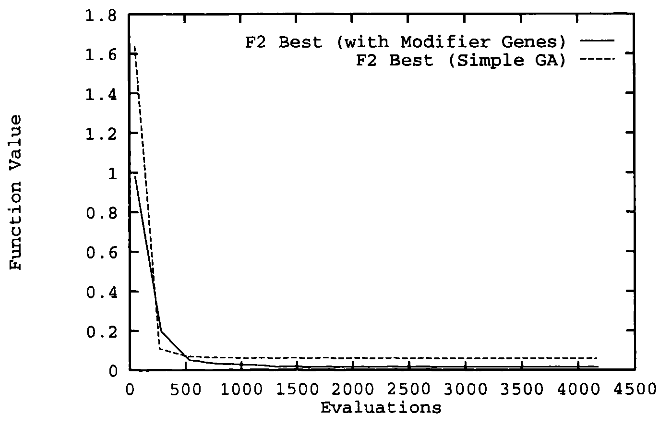
which according to [82] is based on an empirical relationship found by Schaffer et al. in [79]. (Although there is no theoretical support for the above relation, experimental evidence suggests that it provides good values for the mutation rate). Note that the mutation rate associated with a modifier gene operates on a range near to the recommended values for mutation rates. In this way it exercises a fine control on the mutation parameter of the basic algorithm. The modified algorithm was applied to the De Jong suite of tests (see [28] for more details). Figure 5.3 illustrates typical observed behaviour for the best overall fitness averaged over 10 runs.

The results of the simulations show that the use of the modifier-gene approach enables the GA to find values that are equal to or better than those obtained with the basic algorithm (also using less trials) in 4 out of the 5 test functions that constitute the suite of tests. The exception was function f_4 which is given by [28]

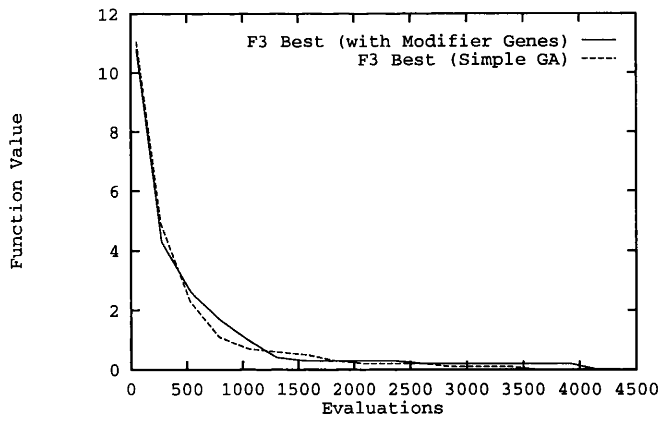
$$f_4(x) = \sum_{i=1}^{30} i x_i^4 + N(0,1) \quad , \quad | x_i | \leq 1.28$$

where $N(0,1)$ is a Gaussian noise with mean 0 and variance 1. Note that (in the experiments conducted) the modifier genes were represented by only 3 bits. It is remarkable that (simple as it may be) this approach seems to be beneficial in terms of actual performance and in this way may point to future viable extensions of the basic GA[10].

[10]At the same time that this investigation was being undertaken, Bäck [5] used similar ideas on studies of self-adaptation within GAs. The basic difference between his approach and the one reported here is that he allowed up to 20 bits for the modifier genes which could vary in the range $[0, 0.5]$. The mutation scheme reported by him is also different from the one implemented above in the following way: the mutation rate of an individual i was first changed using the associated value of the modifier and then the resulting rate was used to mutate the rest of the structure. The results reported in [5] are based on simulations of a simple bit-counting objective functions and are used in the context of finding optimal mutation rates.



(a)



(b)

Figure 5.3: Best individual (average of 10 runs)

Although we have not produced a theoretical analysis of the above approach, we believe the experiments per se indicate that this might be an interesting alternative to try in future implementations. In particular, in parallel implementations of GAs, modifier genes could be introduced to control the rate of migration from one population to another. But this remains a topic for future research.

5.9 Summary

This chapter attempts to explore the possibility of subjecting the representation itself to adaptation. The discussion was centred on the set of linear transformations, as most of the published results are related to them. To the author's knowledge the only published proof of a linear transformation that turns a fully deceptive problem into a fully easy one is given in [51]. The work behind the thesis suggests that in this class of problems the inverse of the given linear transformations may have the same effect (it certainly does that for a particular case). This implies that transformations with very different characteristics may exist that can turn a given difficult problem into a fully easy one. We have also produced a set of linear transformations that is structurally related to Walsh-Hadamard matrices. We have conjectured that attempts to devise a general scheme for submitting the representation to adaptation may prove very difficult to implement. As a way of possibly avoiding such a difficulty, we have suggested a simple extension to the basic architecture based on the ideas of modifier genes, which may eventually lead to future viable extensions of the traditional GA.

Chapter 6
Distributed Artificial Intelligence

Research in Distributed Artificial Intelligence (DAI) usually considers problems in which a group of independent problem-solvers (agents) needs to accomplish a given goal that is beyond the capability of any agent alone. This requires cooperation between the problem-solvers. A simple type of Pursuit Problem has been suggested as a useful tool for evaluating alternative approaches to the distribution of knowledge and control among cooperative problem-solvers [50].

This chapter covers briefly some of the relevant background to DAI in general and the Pursuit Problem in particular. It is intended as a reference point for the more detailed work on the Pursuit Problem that is stated in chapter 7.

6.1 Introduction

Distributed Artificial Intelligence (DAI) is the subset of Artificial Intelligence (AI) that is concerned with the interaction of knowledge-based entities which are separated in some sense (e.g. space, time or field of competence) and which are self-contained.

The oldest continuous work that has been effectively in DAI is by C. Hewitt and his group (e.g. [35]) on "actors": a form of agent. The work is presented at greatest length in a 1986 book "Actors" [2] by Hewitt's former graduate research student G. Agha. Current work on DAI was influenced by the actor approach and by developments of the basic blackboard architecture [32, 62] for cooperating expert systems, among other things.

Since about 1986 the basic literature on the subject can be found in the *Proceedings of the International Joint Conference on Artificial Intelligence* (IJCAI) or in publications such as [39, 11, 4, 24]. Areas in which DAI ideas and software have been tested include air-traffic control [85, 62], intensive-care monitoring of patients in hospitals [33], document retrieval [40], control systems for discrete manufacturing environments [66], speech understanding [49] and distributed vehicle monitoring [19].

The rest of the activity in DAI, as judged by papers in typical workshops, consists of theoretical or technical studies (e.g. the *Contract Net* [84], cooperation without explicit communication [25]) that might be absorbed into multi-feature agent designs.

A common problem seems to exist for all the above applications. It is captured in the following issues:

1. How can one design an optimal agent to solve a particular problem?
2. If one has a special architectural feature like the *Contract Net Protocol* [84], what are the criteria and the methods that should be used to embed this feature in a given system?
3. How does one treat different architectural design choices and decide on their integration? How does one select, from among them, those that have the best chance of providing the most efficient solution for a given job?

It can be said, therefore, that the "agent design problem" *is* a primary problem of DAI. Still, in the current state of development of DAI, one can find that a given design is good because the engineer who conceived it has made a particular design choice (in an unsystematic way) that has turned out to be good. It can be assumed informally, therefore, that one is in fact dealing with a design space where one particular choice corresponds to a point in that space. There is no systematic way to search the space of design choices. Hence any idea that contributes to making a search more systematic is in principle a useful contribution in the current state of DAI. This is the basic motivation for the work conducted in this thesis, where we investigate the extent to which GAs can help to advance the subject.

6.2 Issues of DAI

Research in Distributed Artificial Intelligence (DAI) typically considers the problem of how a group of independent problem-solvers (agents) can accomplish a given goal that is beyond the capability of any agent alone. This requires cooperation between the problem-solvers, since one agent cannot simply proceed to perform its action without considering what the other agents are doing [94]. As a result of that, several alternative approaches to the distribution of knowledge and control among cooperative problem-solvers have been proposed (see [22, 50] for examples). As argued in [24], in general these are still too specialised and project-specific.

The issues of interaction and communication are of central importance in DAI. The way a given problem is decomposed and distributed will determine the interdependencies among problem-solvers, and thus will require them to coordinate their activities by sharing information, plans, goals, tasks or resources [22]. Within this framework it is up to the system designer to provide a given organisational structure and the means for agents to interact within it. At the same time there is a need to develop more extensive theories of DAI and of social knowledge and action [22]. Additionally, as advocated by Avouris and Gasser [4] it is important to have standard and widely-used tools for building DAI systems as a means to free developers from many mundane tasks, and to enhance the comparability of systems and techniques.

It is often very difficult to identify what a minimal architecture for a given problem is. Moreover, many practical applications are not amenable to a detailed mathematical modelling, and hence one is usually satisfied with a solution that has an acceptable level of performance. This requires the design of suitable metrics or performance indices to gauge the quality of the solutions and the effectiveness with which they are produced. Once a particular index is found, a problem can be recast as an optimisation exercise where the space being searched is a restriction of the space of possible architectures. In mainstream DAI, typical papers on architectures pay attention to details of coordination and distribution of knowledge. We therefore examine these in our work later in the thesis, and review several existing approaches in the next section.

6.3 Theories of coordination: literature review

One of the important issues for DAI is the establishment of a general theory of coordination and cooperation. The basic question to be answered here can be restated as the problem of defining *which agent does what, when* [23]. Werner [94] argues that cooperation is made possible through communication and social organisation. The degree of communication that takes place ranges between two extremes: those involving no communication to those involving sophisticated high-level protocols [94].

In the first case an agent should be provided with some *rational* capacity to reason and infer the other agents' intentions without having to communicate with them. Examples of this school are Rosenschein and co-workers' attempts to provide agents with common knowledge that permits them to coordinate their efforts, without explicit communication (e.g. [76, 50]). From a theoretical point of view these attempts are important because: (*i*) they prescribe some lower bound on the amount and type of information needed about a conflict in order to resolve it, and (*ii*) they give some indication of when no good resolution mechanism exists [1]. On the other hand, they assume that agents are provided with enough knowledge of the other agents' beliefs but fail to clarify how this can be accomplished without some sort of communication [94]. Also, as pointed out by Werner in [94], communication is necessary to resolve uncertainty that results when there are several optimal paths to the same goal. In addition, in practical applications an architecture based on rational agents may be computationally expensive to implement.

Werner presents a detailed survey of increasingly sophisticated communication techniques including **primitive communication** (where communication is restricted to some finite set of fixed signals), **plan and information passing, message passing** and **high-level communication.** He argues that primitive communication is often used to avoid conflicts and that sophisticated cooperative action is virtually impossible to be accomplished through it. On the other hand, plan and information passing is usually computationally expensive and suffers from the fact that there is no guarantee that the resulting plan will be agreed to by the recipient agent [94]. Lastly, an explicit theory of how complex high-level communication acts are established is still missing.

Gasser et. al [24] view a coordination framework as a particular set of settled and unsettled questions about belief and action through which agents view other agents. Again, nothing is said about how the agents build their beliefs and how they engage in a particular action.

Other approaches to coordination include: (*i*) a **contract net** , which consists of a group of distributed agents that communicate (negotiate) to solve a problem by task sharing [94]; (*ii*) **partial global planning** by which agents can represent and reason about the action and interactions for groups of agents and how they affect local activities [18]; (*iii*) solving ensuing conflicts either by endowing the agents with **detection** and **conflict-resolution mechanisms** [1] or (*iv*) through **direct negotiation** using **case-based reasoning** techniques and **multiattribute utility theory** [87] and (*v*) viewing negotiation as a **constrained directed search** in a problem space [78].

6.4 The importance of suitable testbeds

While there are many examples of working DAI systems, it is difficult to compare their effectiveness directly because of differences in the applications tackled and in the systems' architectural features. For this reason, the idea of testbeds or test problems has been present in DAI for some time[1].

One such problem had considerable currency for several years, because of its simplicity and apparent generality, but then fell out of favour. This is the Pursuit Problem. It is described in the next section.

Work on the Pursuit Problem has been carried out by several researchers. Gasser *et al.* [23] explore a simple Pursuit Problem to discuss a coordination framework based on patterns of settled and unsettled problems. Stephens and Merx [86] compare three alternative approaches for solving the simple Pursuit Problem while Levy and Rosenschein [50] apply game-theoretic techniques as a methodology for solving the simple Pursuit Problem.

The simplest form of the problem ("4 $\times$ 1": see the next section) was selected by an informal consensus, and later lost favour because of a similar informal consensus process (the expression of the view that it was not complex enough to serve as an effective test for DAI architectures). This last step, we believe, also made a premature and unsupported generalisation: that, since the simplest problem was unsuitable, the general class of Pursuit Problems was therefore unsuitable. Chapter 7 examines this view systematically, and reaches different conclusions.

6.5 The Pursuit Problem

The Pursuit Problem has been suggested as a testbed problem for evaluating alternative approaches to distributed reasoning and performance studies. It was first proposed in [7] and models a configuration of two classes of agents, **red** and **blue**, which move on a rectilinear grid. Blue agents have the goal of blocking the path of any red agent. All agents take turns either moving to an empty neighbouring cell

[1]A recent example of a paper discussing the significance and desirable properties of DAI testbeds is by Hanks, Pollack and Cohn in the Winter 1993 issue of "AI Magazine". It was not available to us before the original writing of the text of the thesis was completed.

(either horizontally or vertically, but not diagonally), or remaining stationary. The red agents move randomly in any possible direction (i.e. one not blocked by a blue agent) which includes their current position. Blue agents are alternatively described as **pursuing agents** while the red agents are often described as **prey**. A Pursuit Game of N agents chasing M prey will be referred to as a $N \times M$ game.

According to [50] the Pursuit Problem as formulated by Benda considers only four blue agents, positioned at different locations on a grid, and whose task is to capture a fifth red agent by surrounding it, i.e., a 4×1 game. When a capture occurs, the red agent stops moving. If a red agent escapes across a boundary of the grid, blue agents no longer target it, and it cannot be captured[2].

Gasser *et al.* [23] explored the 4×1 game using a coordination framework based on patterns of settled and unsettled problems. Stephens and Merx [86] compared particular versions of three different control strategies (local control, distributed control, and central control) and arrived at conclusions similar to those of Benda *et al.* [7], i.e. that the central control strategy is superior to the other two although less efficient (and less robust, e.g. in terms of malfunctioning of the agents) than distributed control. Levy and Rosenschein [50] approached the same game by incorporating global goals into the local interests of all agents through the use of Game Theory techniques.

There are some points that are worth mentioning about the above techniques. First Levy and Rosenschein's technique relies on the fact that each agent, at all times, has complete knowledge of all agent locations (the same argument holds true for any central control strategy, where the controlling agent has complete and accurate information about the location of all other blue agents and prevents collisions among them). This stands counter Gasser et al's [24] aspiration of having theories that hold under a **no global viewpoint** condition, i.e., theories assuming that (in general) global knowledge is impossible or undesirable. It also puts a heavy burden on the amount of computation in any particular problem-solver, assumes that the information provided to the agents is noise-free, and may be computationally expensive to implement in games with more 5 agents. On the other hand one must ask whether the game itself justifies the complexity of the algorithm, and how the algorithm can be extended to more complex games.

[2]One of the reasons that made this domain popular is the fact that it is simple to describe and implement yet appears to require cooperation and some sort of coordination (explicit or implicit) on the part of the four blue (intelligent) to achieve their goal.

By the same token the distributed-control architecture of Stephens and Merx [86] relies on agents communicating their intended targets at each iteration of the algorithm. They argue that the distributed-control system could be implemented with much lower communication if the blue agents agreed to follow the strategy of allowing the most disadvantaged agent his choice and if a collision-avoidance convention were established. This brings into perspective the potential problems this architecture faces if the communication channel is subjected to disturbances from noise and how this architecture could be extended to games involving more than one prey.

The above discussion also raises several further important points.

- How can one judge whether a given problem can be solved by a community of problem-solvers using only local knowledge? When should global knowledge be a necessary factor for a solution to exist?
- What classes of communication and coordination structures are necessary in a particular problem?
- If one aspires to a comprehensive theory of interaction among multiple autonomous agents, how is one to define the building blocks that comprises coordination, cooperation and communication among several independent agents?
- What are the issues that lie at the foundations of the design of autonomous agents, and what can one do to design automatic tools that relieve the DAI developers from many mundane modeling tasks?

The rest of this thesis reports on our attempts to address the above questions. Our approach will be to consider first a community of very simple reactive agents that have no global knowledge and rely on a minimal amount of communication to coordinate their action and resolve conflicts when they occur. In the latter case communication is considered in an implicit way as agents follow a predefined set of rules about how to behave when conflicts arise.

As a definite and comprehensive theory is not currently available, it seems to be a daunting task to judge whether a given Pursuit Game possesses enough structure and complexity to serve its main current purpose in DAI, namely, a testbed to explore and compare alternative distributed architectures.

We shall take the experimental viewpoint that a pursuit game should be considered *easy* whenever it can be solved by a community of simple agents with a reasonable success ratio and with good efficiency of performance. This implies that no higher level of control structures is necessary. Games where reasonable performance measures cannot be achieved with this architecture will be regarded as having enough complexity that they can only be solved by the introduction of high-level control and coordination structures.

There are many avenues open to explore, but this thesis presents just our specific design decisions. We have concentrated on properties that we believe are general enough to form a good basis from which concepts can be explored even further.

6.6 Aims of this research

One basic aim of this work is to provide a framework to explore DAI problems in general. We shall focus attention on the design of suitable testbeds, a particularly important area of research in DAI. The following are the basic steps that will be considered (although not in the order that they occur in this thesis):

- Search for a minimal set of behaviours for a DAI architecture such that agents endowed with this set of behaviours can be regarded as behaving in an "agent-like" manner.
- Parametrise the architectural features in a convenient way so as to allow efficient search by a GA.
- Define (or examine) suitable performance measures for the Pursuit Problem.
- Search for an architecture that has the best performance according to these measures, using the GA as a means of carrying out the search heuristically.
- Explore sets of Pursuit Problems that have the right structure to be considered as suitable testbeds for DAI architectures.

6.7 Summary

Distributed Artificial Intelligence often considers the problem in which a group of independent agents needs to accomplish a given goal that is beyond the capability of any agent alone. As a field of both practical and theoretical importance, DAI continues to expand. As the scale of applications grows, and the requirements for embedding ever more sophisticated knowledge in the operations of such systems increases, the necessity for continued development of techniques of DAI has become more central to their success [4]. A large number of different techniques exists but it is difficult to compare their performance. Hence the need for specific test problems, such as the Pursuit Problem, where different architectural approaches can be compared. Another aim is to provide meaningful problems that may offer insights for research in distributed reasoning. This thesis offers a framework where some of the central ideas of DAI can be advanced from an experimental point of view. Using this framework we shall explore pursuit problems that were never considered before, as a means to offer researchers in DAI suitable testbeds.

Chapter 7
Using a Genetic Algorithm to Design a Minimal DAI Architecture

This chapter builds on the explorations conducted in previous chapters to propose a minimal DAI architecture that can be subjected to genetic search in an efficient way. Design principles based on the concepts previously explored are described. Implementation details that try to balance both the requirements of a class of minimal DAI architectures and its corresponding amenability to genetic search are also brought in perspective. It also presents a systematic exercise in evaluating the quality of pursuit games as potential testbeds for DAI, in which GAs are used both to optimise some simple and basic architectural features of agents and to help the search for meaningful testbeds in the (N, M) space of games. The conclusion from experiments is that $(M+4)\times M$ games have the right complexity to be good testbeds, provided that $M > 4$. Additionally, the chapter demonstrates the usefulness of GAs as tools to help DAI designers, and argues that boredom is a concept that deserves consideration as a feature of general agent architectures. The experiments also suggest that with regard to communication it pays an agent to listen selectively (or periodically) rather than all the time. Moreover, it seems that the performance of the problem-solvers is more sensitive to the fraction of time during which an agent actually listens to messages broadcast by others rather than (for example) the exact model by which messages are attenuated in the communication medium.

7.1 Objectives

One of the central features of the experimental work behind this thesis concerns how best to explore fundamental aspects of a DAI architecture that can be amenable to genetic search.

It is important at this stage to emphasise several points that distinguish the approach taken in this thesis from other applications using GAs.

1. This work is not concerned with using GAs and other natural metaphors such as Classifier Systems [28] as paradigms for the evolution of syntactically simple rules to guide an agent's performance in an arbitrary environment.

2. This work is concerned with building a minimal DAI architecture with which to explore questions that are of interest to DAI designers, and to suggest possible interesting features for general DAI systems. A second basic purpose is to examine the relative quality of different Pursuit Games within this framework, and to find the games that look best as testbeds for agent architectures completed by alternative high-level design features.
3. This work is concerned with building suitable parametrisations for an agent architecture that can be explored efficiently using GAs.
4. This work does not aim to propose an optimal architecture to solve any Pursuit Problem. Ideally we would like to, but we recognise the difficulties of such an approach. At the least, we could consider the optimisation exercise to be reported in this work as being restricted to one particular class of systems (i.e. the one we shall propose in what follows).

The first point above is based on the evidence of different examples found in the GA literature that highlight the numerous difficult theoretical and practical problems that need to be overcome here. In several of these examples classifier systems were able to evolve simple and interesting rules, but the scope of the problem was either small or the rules evolved did not offer any new insights beyond what was already known by the system designer from the outset[1]. The exercise explored in this thesis is highly computationally demanding. In the current stage, the reported evidence in the literature points to the limitations of these paradigms for delivering good solutions in a reasonable amount of time.

Additionally, we want to build our framework on top of already-existing DAI experience, and shall not try to derive (or possibly rederive) results using paradigms that at this stage of time might be too risky to rely upon. We consider that enough knowledge has been gathered during previous theoretical and experimental explorations in DAI and that those explorations provide a suitable set of concepts from which to explore further issues[2]. The recognition of these current limitations has led us to

[1]This does not mean that they should not be regarded as serious and potentially fruitful topics to address. The author also accepts that only by realising their full capabilities in domains already well-known can they be fully understood for application to more general contexts.

[2]In a sense, we regard the currently-available knowledge in the DAI area in an "evolutionary way", i.e., that through the last few years of exploration many ideas related to central DAI questions have already been sieved by their application to different problems. The ones most often cited in the literature reflect their overall success.

discard the techniques described in the first item above from further considerations in this thesis.

The second and third points above are interrelated, and in a way they distinguish the work done here from mainstream GA applications usually found in the literature. In the latter case the function to be optimised (and the corresponding parameter set) is known in advance. Hence, the application of a GA becomes an optimisation exercise where discussions of convergence to the optimum, and modifications of the basic algorithms, are central. In contrast, the problem considered here has no definite functional form to start with, nor is the parameter space specified from the outset. In fact, if it were so, much progress could have already been achieved by traditional DAI methodologies. At the same time, the success of the experimental exploration conducted for this thesis may point to a new application area of GAs for which they can be used with reasonably promising results and performance.

In this respect it is rather important to design a suitable architecture and corresponding fitness function for which a genetic search stands a chance to be successful. Ideally, we would like to avoid common problems such as premature convergence and deceptiveness that are so often reported throughout the literature. Moreover, due to limits on our computational resources, we have had to aim for a particular architecture that could generate consistent behaviour across a whole range of ever-increasing complex problems, without resorting to repeated experiments. We also wanted to use an experimental approach (for lack of a theoretical one) to convey important and relevant issues to DAI users: in particular, with respect to the Pursuit Problem. To sum up, we were faced with the following difficult hurdles to overcome:

1. Lack of a comprehensive knowledge of the problem domain.
2. Stochastic nature of the different games.
3. Lack of a firmer theoretical foundation with which to avoid the reported problems a GA may face when exploring a given domain.
4. Limits on our computational resources.
5. Necessity to specify a suitable fitness function to rank the success of alternative architectures.
6. The specification of a class of architectures that not only would be convincingly minimal from a DAI point of view but also could be explored successfully by a GA.

The first and second points above actually are the driving force and serve as the main motivation for the work reported here. In fact, as an outcome of the experiments conducted in this thesis we were able to gain not only insights in the particular domain of Pursuit Problems but also in interesting aspects related to the architectural design choices made. As a result, I believe this thesis provides enough experimental support for the use of GAs in the context of DAI.

The current best alternatives to deal with the problems referred to in the third item were already explored in previous chapters. They are discussed further in sections 7.3 and 7.7. We hasten to point out that ultimately it is the results of the experiments which will be the best assessment of the suitability of the design options selected. Finally, item 4 is discussed in detail in section 7.6, while the last item is explored in sections 7.3, 7.4 and 7.7.

7.2 Working hypothesis

We shall assume as a working principle that agents are endowed with some capability to recognise patterns (either by having this capability programmed by the designer, or by providing the agents with some learning mechanism or even by allowing a GA to evolve architectures where the agents have this capability "hardwired" from the outset.).

Whatever the set of patterns is, we shall assume that agents can recognise differences in the patterns presented to them and they can also recognise similarities among these patterns (and also with "pre-wired" patterns stored in their memory). Agents will be assumed to possess some correlation mechanism such that they can do things with two or more of those patterns (for example, they can look at differences or similarities of a given set of patterns). A design approach based on this idea should aim to find some criteria for deciding whether two given patterns are the same or not. Since an exact match is always ideal, any such criteria should not just say that the patterns are exactly the same, but rather whether they are the same under some tolerance or some given *threshold.* Outside the defined boundaries the patterns will be considered not to match. For example, a very simple agent would consider that anything closer than that threshold is equivalent to the stored pattern and would make its decisions (whatever they might be) without searching for further explanations to account for the observed difference. In a correlation-based design a simple agent should be endowed with a very simple capability to look for similarities between patterns and also with a simple counting capability[3].

There are two natural ideas that emerge from the above discussion. The first is based on the fact that even a minimally simple agent that works under the above principles can be equipped with some capability to predict the future state of the patterns or correlations that it is observing. The second one is based on the idea of allowing the agent to recognise that a given pattern has not changed enough over a period that should have been long enough for some (any) desired change to happen. That is, it may be advantageous for the agent to seek for some difference in what it sees relative to what static things it has been seeing in its previous observations. We shall address the above issues in more detail in later sections.

7.3 Design principles

Given a certain DAI problem, consider the architecture of the individual agents and the cooperative that they form in order to explore it. It is convenient to view it as partitioned into "low-level" and "high-level" components[4], where the former are present in any reasonable architecture in structures that do not vary in any essential or surprising way from one architecture to another. "High-level" is a label that we shall use to denote other components of an architectural design: those that are novel or that add special knowledge-intensive value to a specific design (and are presumably intended to make that design work better than its rivals).

There are at least two main reasons for considering the partition of a general architecture in such a way.

First, it may be argued that in many problems a high-level component could by itself solve a given problem to optimality. However, it is important to judge whether the proposed solution is amenable to implementation in systems that operate under

[3]It must be observed that in general the implementation of the above ideas is related to the details of the particular architecture.

[4]For the purpose of this thesis, we use the distinction between high and low levels informally; no formal definition is intended or implied, as the dividing line between levels is obviously subjective and not clear-cut. The existence of the informal distinction is, we think, helpful just to support the expression of some of our ideas and results.

severe constraints[5]. Because of that, a compromise should be struck between the desirability of a proposed solution and the practical limitations of its implementation. Within this context, the low-level components play an important role in relieving the high-level components from burdensome activities that could hinder the latter's performance in real-world applications.

More generally, by observing the performance of a minimal DAI architecture on an as yet poorly understood problem, a system designer may fill the existing gaps with personal knowledge about the specific problem domain. The choice of high-level components in an architecture allows DAI researchers to be creative in their designs for effective agents and collectives. In the present state of DAI, the question of what high-level components to select is wide open - which is why the issue of testbeds and test problems for comparison of different architectural approaches is currently alive.

Bearing these points in mind, we have focused our attention on exploring the low-level components of the architecture. Our objective was to design a system which could grow smoothly to accommodate several dimensions of an agent architecture by observing the performance of the architecture in problems of increasing complexity. In this way the introduction of new dimensions was only considered when a solution that proved satisfactory in games of less complexity did not provide reasonable performance with more complex games. Moreover, as we were setting out to explore problems hitherto unexplored, it was important to observe patterns of unforeseen behaviour that resulted from the interaction of only a few simple behaviours. The intention here was to observe *emergent behaviours* arising from those interactions such that they could be categorised into more abstract concepts which could then be transferred to general DAI systems, or in particular to those used in problems similar to the Pursuit Problem.

Our emphasis on simplicity was also related to the observation that (much as in nature) a given problem can be explored successfully by a variety of different classes of agents, each class with its own specific behavioural patterns. However, as argued in chapter 6, we wanted to restrict the number of allowable behaviours to a bare minimum such that they remained non-problem-specific at least in the domain of Pursuit Games. Sections 7.4 and 7.7 consider the practical implications of this idea

[5]In fact as argued in [22] "the possible limits on communication time, bandwidth, etc., so that a global viewpoint, controller, or solution is not possible" is one of the criteria that identifies when coordination among intelligent agents is a basic issue.

by motivating and explaining in greater detail the specific implementation choices taken in this work.

On the other hand, as we intended to submit a given class of architectures to genetic search it was important to prescribe some design guidelines that would make this search as effective as possible. The theoretical considerations explored in previous chapters suggest that a function should be suitable for exploration by a GA if it can be decomposed as a linear superposition of functions each of which is itself easy for genetic search. Although we have not discussed the choice of the fitness function yet, we have used this principle when specifying the choice of the architectures (i.e. the parameter space) in the following way.

Suppose that a fitness function can be found such that it ranks successfully the performance of the candidate architectures. This function may depend on the modules that comprise the given architecture in a non-linear way. However, if the interaction is additive among the modules, this might allow independent optimisation in each one of them. In other words, we aimed at an architecture that could be decomposed into loosely-coupled parts.

Each module was parametrised by a simple smooth function or a function defined only at discrete points but supporting a smooth "fit" to these values by a continuous curve. In doing so we hoped that correlations with respect to the performance should be explored efficiently by the GA. Whenever that could not be achieved, a given parameter was to be encoded with a small set of alternatives so that they could all be sampled evenly in the early stages of the optimisation. The emphasis here was on simplicity and ease of programming. However, we hasten to add that (unfortunately) even these guidelines do not guarantee the absolute success of the optimisation exercise. Despite this disclaimer, we do feel that it is better to base the design on a sensible guideline than none at all. Also, the results of the simulations to be presented seem to point to the correctness of the choices we have made by following them.

It is useful to consider the process described in this section as the search for orthogonal features of an agent architecture in a bottom-up manner. This is a process analogous in style to the Gram-Schmidt procedure known in vector algebra [10]. Abstractly speaking, we aimed for a decomposition of the architecture in such a way that the fitness function could be expressed as a linear superposition of simpler functions each of which could be amenable to independent optimisation.

7.4 Design objectives

In the previous section we have described the basic principles that guided the decisions on how to build individual modules. While these principles are important, the greater goal was to develop a convincing minimal architecture of agents that would behave in an "agent-like" way. In order to assess better how this may be accomplished, it will be instructive to make a brief review of standard ideas used in general DAI systems.

Although a definite theory of coordination has not yet emerged [24], a quick survey of the available literature shows that the current approaches to coordination and cooperation among problem-solvers divide into several lines, among which the most often cited are interactions of rational agents (e.g. [76]) or the modelling by an agent of other agents' actions or beliefs (e.g. [23, 22]), distinct theories of communication, cooperation and social structures (e.g. [94]) and different proposals for conflict resolution (such as [1, 87]).

Our intention was to build an architecture that would take most of the above issues into account in some basic but convincing way. As a consequence, agents within the desired minimal architecture should have in principle a recognised minimal capacity to plan their actions and coordinate themselves within the cooperative that they form. They should also have a convincing minimal capacity to model their own and other agents' beliefs, plans or actions. Also, the communication mechanisms and interaction protocols among problem-solvers should be restricted to be as simple as possible. Finally, they should possess a very simple resolution mechanism with which they could resolve ensuing conflicts.

7.4.1 Minimal belief capacity

The simplest way to respond to this concept for an agent's architecture is to avoid its implementation altogether by forbidding agents to reason about other agents' knowledge, beliefs, and goals. This may seem rather restrictive; hence we have adopted the following alternative which appears to be the next simplest choice. First, we have restricted our models to be applicable only to the prey.

The pursuing agents' model of the prey's beliefs is based on a black-box approach. In a black-box approach if one is talking about beliefs then it is the belief about the black-box that really matters (e.g. one may then say "I believe that the black-box will do so and so"). Agents are assumed to possess a black-box model "hardwired" in their machinery. The parameters of this model are to be tuned by some clever algorithm (a GA in the current work). Note that a belief is something that usually is not checked out at a proof level but as a result of the observations about regularities of the world or because of trust about what other agents are saying concerning the way things are happening in the world. So, if the observed regularities are such that the level of confidence of the agent on its model is above a given threshold, the agent may then plan its future actions according to what its black-box model is telling it. The model therefore acts as a predictor and the belief of the agents concerns the predictor itself.

7.4.2 Minimal capacity of coordination

Within the above framework, the best way to guarantee a minimal level of coordination is to allow agents to pursue their local goals only, regardless of what others are doing, and communicate in the minimal way possible. In a slightly more sophisticated architecture of agents we relaxed this restriction but only allowed the agents to possess a restricted set of communication acts whose semantics was as simple as possible and only conveyed a minimal amount of information about local activities of the agents themselves.

7.4.3 Minimal capacity for resolving conflicts

Due to their simplicity, blue agents should not plan to avoid potential conflicts. Conflicts, when they occur, should be solved following very simple rules with the least amount of communication. In this way, it was hoped that the number of conflicts occurring and the amount of communication (see section 7.5 for further details) required by the architecture that achieved the best performance level (within the class of minimal architectures) could convey an idea about the complexity of planning and coordination mechanisms required in a more sophisticated architecture, if it were to achieve at least the same performance levels.

Once a minimal architecture was defined and parametrised, we explored the problems of interest for this thesis using a GA to search for the optimum architecture (using a given performance criterion to be discussed later). In doing so, we regarded the performance of the optimum solution found by the GA as an indicator of the difficulty a given pursuit game would offer. We have also considered the amount of communication and the number of conflicts occurring as other indicators of the complexity of the given game.

Before going on to examine the specific design issues in detail, it is useful to introduce the experimental framework in which they were tested so that the concepts to be discussed will be seen in the proper perspective.

7.5 An experimental framework

We have built a system of programs to implement pursuit games that follow the same general line as the one in [86]. In their formulation they considered the simplest 4×1 game in which no two agents may occupy the same position and the red agent moves randomly among the possible directions and its current position. Blue agents can only sense the position of the red agent (i.e. they do not possess knowledge about the position of other blue agents unless any of them specifically broadcasts its position). When an agent blocks a position surrounding the red agent the former broadcasts to the others that that position has become captured. Thus the other agents may reassess their plans about which position should be targeted, while the agent that has blocked the position basically follows the movement of the prey by remaining stationary relative to it.

In the Stephens and Merx [86] formulation a game is considered to be finished when the prey moves outside the grid or when the blue agents manage to surround it completely inside the grid. In the latter case the prey is considered to be **captured** while in the former case the prey is considered to have **escaped** when it has at most two captured positions occupied by blue agents. If the prey has moved outside the grid while surrounded by 3 blue agents, the game is considered to be in a **stalemate** situation.

In games involving a larger number of red agents one is faced with the dilemma of deciding when the game should end since it is possible that in some moves some of the prey remain on the board while others have already moved outside the grid boundaries. Since we were interested to make an extension to more complex games that stemmed naturally from the above framework we selected the following option. Each red agent that crosses the board boundaries is considered out of the game by being "frozen" outside the board boundaries. This also applies to the blue agents that, having captured positions surrounding a given prey, follow it on its path across a boundary of the board. The configuration of prey and surrounding blue agents is obviously kept for purposes of collecting performance statistics at the end of the game. Note also that in more complex games one must allow for the fact that some of the red agents may continue to move on the grid and there may not be a sufficient number of blue agents to surround it. Hence a decision on a maximum number of iterations is necessary. As discussed in section 7.6.3, we allowed 100 iterations as a reasonable preset limit on the number of iterations of the game.

Another major difference between our implementation and the one reported in [86] is that we allow a blue agent to release a captured position that it has occupied, with the provision that it should communicate (by broadcasting) when it captures or releases a given position. Another important distinction is related to the way in which the agents react to conflicts. In the Pursuit Game a conflict occurs when multiple agents attempt to move to the same cell. As discussed in section , we allowed conflicts to occur and be solved by the agents following very simple rules, with the provision that agents must solve their conflicts in one step of discrete time. By that is meant that, whenever agents are in conflict, they are allowed to make a second choice of a neighbouring cell to which to move. If after that choice an agent still remains in conflict with another one, they both stay in their current positions.

For ease of comparison with Stephens and Merx [86], each pursuit game was run on 30 initial grid configurations which were generated by placing the red agents symmetrically with respect to both the horizontal and vertical axes. Along each axis the minimum distance between neighbouring red agents was one unit square. Blue agents were placed randomly on the board subject to the condition that they should not be on the central horizontal and vertical lines. Figure 7.1 illustrates one such initial configuration of five red agents in a 10 × 10 grid.

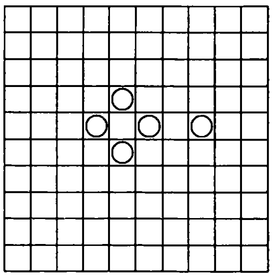


Figure 7.1: Initial configuration of five red agents

7.6 Performance metrics

It often occurs that a given problem may be explored successfully (or with reasonable success) by different approaches. The decision of which alternative should be chosen depends on its capacity to deliver the best available solution using least resources, e.g., time, complexity of a given architecture, computational cost or the amount of communication between different problem-solvers. In a large number of cases of interest these are conflicting objectives and a user may be satisfied with a given solution provided it has some degree of acceptability and does not compromise some important constraint. It often occurs that even an exact calculation of bounds on these constraints is difficult (if not impossible) to derive.

The trade-off between the acceptability of a given solution and the cost of finding it is a fundamental theme, and to assess it a distance function (**performance metric**) is usually specified. Performance metrics rank alternative methods with respect to one that would, theoretically speaking, provide an optimum solution. Within this scope, they should reflect correctly how distant from the optimum a given alternative is. It is assumed that the higher the rank of a given alternative the smaller the number of modifications that should be introduced in its design to achieve a desired level of performance.

It often happens that the design of a suitable performance metric is a very difficult problem by itself. This is a particularly sensitive issue in learning metaphors such as GAs, for which intermediate non-optimal solutions have to be rated in order to facilitate the discovery of the optimal one, if that is at all possible. A classical discussion of these problems can be found in Samuel's work [77] with a learning checkers player. Here we shall examine these issues in light of the specific problem we are concerned with in this thesis, i.e. the Pursuit Problem.

7.6.1 Performance metrics for the Pursuit Problem: discussion

We are interested to find simple metrics that rank alternative architectures as a means to sieve those architectures that explore interesting mechanisms for the distribution of knowledge and control. Simplicity is important because of considerations of computational tractability and also for capturing essential aspects that may be obscured otherwise[6]. It is also important that the GA should be able to capture (through a given metric) good architectural choices among alternative solutions to produce ever better solutions.

In a pursuit game, a good architecture should be one that generates a large number of captures in an efficient way. In principle, one should hope that this happens as a result of reasonable architectural choices that can be used in general DAI systems or at least in more complicated pursuit games. In these terms efficiency conveys the subjective idea that the architecture is exploring some important canonical feature of agents' design. For example, an optimal architecture for a pursuit game should be one that allows a maximum number of red agents to be captured and no other alternative exists that admits this in a more efficient way (i.e. using a smaller number of time steps)[7]. The question therefore is whether we can capture these two related (but not equal) concepts with a given metric.

Bearing these points in mind, we shall focus our attention on two aspects of interest: (*i*) the number of captured red agents, (ii) the efficiency with which these captures have been accomplished.

It is reasonable to accept that in a complex pursuit game the task of capturing *all* red agents may be difficult to achieve. It is therefore important to rank intermediate solutions i.e. to assign a value to a final configuration in which a red agent is "surrounded" by either three, two, one or no blue agents. For example, a 4×1 pursuit game that ends with the red agent surrounded by three blue agents seems to indicate a better coordination mechanism being used, in comparison to a game that ends with the red agent "surrounded" by only one blue agent. There are other factors that might be of interest to assess the quality of a sub-optimal solution (e.g. the average distance between captured red agents). However, as a first approximation it seems that a metric that assigns higher values for configurations of red agents that end the game surrounded by a larger number of blue agents has the desired characteristics.

[6]If two metrics are simple and have some simple relationship between them (say, a linear relationship) then if an architecture A is better than an architecture B by the standards or the first metric, then one may say automatically without testing that architecture A is better than architecture B with respect to the second metric as well.

[7]Note that it is difficult to conceive that in the average game one can optimise to find a single most efficient architecture, i.e. one that achieves a maximum attainable value. However, the problem in this work is not to find such an architecture in *a* game but to find a game that admits the possibility of finding such an architecture somewhere over the set of possible games.

The specification of the efficiency criterion is more subtle and deserves a more careful discussion.

The most immediate alternative to assess the efficiency of a given architecture is to compute the amount of time spent by the problem-solvers to achieve a solution, and to relate it to a known minimum time that it would take for an optimum architecture to achieve its solution. As argued above, in most cases it is not known what the optimum architecture is, so a suitable bound has to be calculated. This may be by itself difficult to achieve (for example, in games involving a large number of agents). An immediate alternative is to regard the distance travelled by all agents that have accomplished the surrounding of red agents as a reasonable indicator for the efficiency of the corresponding architecture. The reason is because the smaller the distance travelled by all agents that have managed to surround the prey, the more efficient the system is likely to be.

This solution begs the following questions: should we conclude from the above information that the community of agents, through their interaction, is really exploring some desirable high-level feature of the problem itself? Or should we conclude that the problem itself offers enough latitude for designers to specify architectures that are opportunistically lucky in exploring features that yield a high value of the efficiency criterion and yet produce no discernible high-level behaviour? (i.e. that a given architecture is specially tuned to take advantage of the performance criterion being optimised or even to the particular way the prey move.) In this case, could we conclude that it would perform well on games that indeed demand coordination and knowledge exchange mechanisms among problem-solvers?

Note that we are interested to examine whether the Pursuit Problem stands a chance of being regarded as a suitable testbed for different alternatives to the distribution of knowledge and control among cooperative problem-solvers. If a given problem does not exercise these important features adequately then it does not serve the prime reason for which it was originally conceived. Now, what can one do to certify that the agents indeed distribute control and knowledge among themselves? One possible approach would be to analyse (or measure) the "intentions" of the agents that form a cooperative. Unfortunately in many situations we have no access to the internal states of the agents to decide whether they are generating the intended behaviours. By the same token, it is apparent that this should be difficult to quantify. Consider for example the case where a given game is to be explored by two systems: one that obviously uses clever architectural choices for the distribution of knowledge and control and the second constituted by agents completely (or almost) devoid of these characteristics. Suppose that both systems manage to solve the problem with comparable degrees of efficiency. An observer who did not have any information on how the systems were built might conclude that both systems were equipped with comparatively powerful mechanisms to solve the problem.

This dilemma could be resolved by counting the number of messages delivered assuming they have the intended features of distribution of knowledge and control. The fragility of this argument is apparent if for example one of the designs is based on ideas of "cooperation without communication" (e.g. [50]).

The information that one of the systems lacks any coordination or similar rational mechanism, however, would be sufficient to point out the inadequacy of the game itself as a suitable testbed. It might also highlight the difficulties in devising a suitable efficiency metric.

7.6.2 Performance metrics for the Pursuit Problem

With respect to the Pursuit Problem, Stephens and Merx have considered four metrics that summarise the performance of a given architecture in a 4×1 game: (1) pursuit outcome, (2) capture ratio, (3) success ratio, and (4) success efficiency. Pursuit outcome, success ratio and success efficiency will be discussed in sections 7.6.3, 7.6.4 and 7.6.5 respectively. *Capture ratio* refers to the number of experiments that result in a capture divided by the total number of experiments run. In a general pursuit game there is a high probability of capturing at least one prey per experiment, so this metric is not particularly helpful unless a good generalisation can be suggested.

Note that the purpose of using a test problem may vary for different architectures. One particular architecture may be tailored to work in environments where there are stringent limits on communication bandwidth. Another may be designed to work in domains where conflicts occur very often and hence should be exercised in test problems with conflict-rich characteristics.

One of our purposes is to suggest games that may offer interesting challenges to DAI designers. At the same time we recognise that communication and conflict-resolution mechanisms usually result from high-level architectural choices. Since our design incorporates very primitive communication and conflict-resolution mechanisms, we are interested to propose a low-level measure that relates to them[8]. We have therefore considered the number of conflicts that occur per experiment and also the number of messages broadcast per experiment, as very simple measures of how much communication and conflict take place in each set of games. These are simple measures, but they offer hints of the intrinsic complexity of a given pursuit problem with respect to coordination of agents. The standing challenge is for a designer with particular high-level architectural ideas to see how those ideas, when implemented, can improve performance by comparison with the same test game before high-level treatment.

7.6.3 Pursuit outcomes

A game is considered finished if: (*i*) all red agents are either captured or have crossed the grid boundaries, (*ii*) a preset limit on the number of iterations is reached. The latter condition is necessary to limit the number of iterations in one game to a reasonable value. In all games that we have played, we found that 100 iterations was a suitable value since the board configurations reached after that number of steps were fairly stable and apparently did not change much afterwards.

In the same way as for the 4×1 game considered in [86], once the termination of a particular game is reached any of the red agents can be in one of the following states: in a **capture** when it is completely surrounded by blue agents; in a **stalemate** when it is surrounded on three sides; in an **escape** when it has at most two captured positions occupied by blue agents. Also, a red agent will be in a **supercapture** state when it is surrounded by less than four blue agents but the remaining captured positions are blocked by other red agents or by blue agents that are surrounding a different red agent. In this way a supercapture reflects a more sophisticated capture by the blue agents, since red agents can be captured using less than four blue agents per red agent (in contrast with the 4×1 game). In fact, if it were not for this coordination, the blue agents could never capture both red agents in even a game as simple as 6×2.

[8]In the architectures that we have examined, communication is very limited and occurs explicitly when an agent captures/releases a given capturing position. Conflicts refer to the situation when two blue agents want to occupy the same position.

7.6.4 Success ratio

The **success ratio** as defined by Stephens and Merx is the expected value of a weighting function $f(.)$ that maps the outcomes to values in the closed interval $[0,1]$. The values chosen for $f(.)$ were (following [86]) 1.0, 0.5, and 0.0 for capture, stalemate and escape, respectively. We give a supercapture the rating of 1.5. Since we consider games with more than one red agent, success ratio here is the ratio of the above quantity to the number of red agents.

In addition, we have considered a **normalised success ratio** which is the ratio of the success ratio to its maximum achievable value. For example, in a 6×2 game the latter is equal to 1.5 since red agents can always be supercaptured in principle. On the other hand, in a game such as 5×7 the maximum allowed value for the success ratio is 1.0/7 since in each experiment at most one prey can be captured. A common measure across such disparate games is therefore needed.

7.6.5 Success efficiency

Stephens and Merx define **success efficiency** as an efficiency metric that can give some basis for comparing the performance of different architectures. The calculation of this metric depends on a global assignment of agents to capture positions. In games involving more than five agents, this procedure may lead to difficult and time-consuming computations. We have therefore defined success efficiency in the following way. For each prey captured (or supercaptured), we compute the ratio of the Manhattan distance travelled by the prey to the average Manhattan distance traversed by all blue agents having *captured* or *blocked* a given position on that prey. A prey that ends in a stalemate or escape has corresponding pursuit efficiencies of 0.5 and 0.0. The success efficiency with respect to that prey is then defined as

$$\frac{\sum_{i=1}^{n_{sc}} E_i f(sc) + \sum_{i=1}^{n_c} E_i f(c) + n_s f(s) + n_e f(e)}{n_{sc} + n_c + n_s + n_e}$$

where E_i is the capture efficiency with respect to the i-th red agent that has been either captured or supercaptured; n_{sc}, n_c, n_s, and n_e are the number of red agents that become involved in a supercapture, capture, stalemate or escape. The success efficiency of the entire architecture is the average of the above measure over all prey.

7.7 Design issues

In the process of designing autonomous agents using a GA approach we were confronted with the following issues: (*i*) What questions should DAI designers ask? (*ii*) How can we apply GAs to the Pursuit Problem?

One answer to (*i*) deals with the problem of defining *which agent does what, when* [23]. With regard to (*ii*) we were faced with the problem of finding suitable dimensions of agent architectures to allow for convenient parametrisation and genetic search[9]. It is obvious that some dimensional choices may be meaningless or not interesting while others may turn out to be useful not just because they promote the applicability of GA methods but also because they suggest new interpretations or ideas to the DAI designer. In trying to identify interesting dimensions for agent architectures (i.e., *what is parametrisable?*) we found ourselves asking the same question as (*i*) above, recast in the language of reactive architectures: *what are the observable quantities that should be measured and what are the thresholds for reactions? Are the thresholds based on correlations, predictions? And if so, of what and with what? What actions should be taken when a given threshold is exceeded?*

In trying to give an answer to (*ii*) we were drawn to the fact that in a multiagent system each individual agent should have at least a minimal capacity to plan its activities and coordinate them with those of other agents. It should also be provided with some minimal capacity to deal with conflict situations whenever they occur. It is immediate to see that in many problems no agent is capable of performing a given task by itself. Hence agents should rely on coordination with others so that the community of agents will be able to accomplish the desired goal. In a complex pursuit game, it is not sufficient for an agent to block a given prey's direction of motion as there may not be enough other agents to block the prey's other possible directions for free movement (e.g. consider the case of a 5×7 pursuit game). As a consequence, if we want to endow even minimal simple agents with some capacity to cope with more complex games we have to make a choice about how they should

[9]The use of GAs as function optimisers in this context can also be motivated by the following argument. If one has some performance measure that one trusts as a good measure of performance and the value of this performance metric increases as a result of some tuning, then we can say either that this happened as a lucky coincidence or explicitly as a result of our attempts to optimise the architecture for this sort of performance.

react in these particular situations (subject to the fact that the agents have no global knowledge and very limited skills).

Bearing the above points in mind, we decided to concentrate our design on three main issues: (*i*) prediction, (*ii*) boredom and (*iii*) conflict resolution. As a subsidiary issue we have also considered basic communication between agents (as described in section 7.10) but there has not been time to do full justice to this in the thesis.

Prediction refers to the agents' capacity to identify certain patterns/situations and immediately realise a possible version of their future consequences. In many circumstances agents should be designed with an innate capacity to react to unforeseen pattern/situations as this may enhance their efficiency (and sometimes their survival when confronted with dangerous environments).

Prediction also offers a finer control in the planning capability of an agent, as it allows the possibility of recognising future conflicts, or anticipating its target position (e.g. in a pursuit game a given agent chasing a prey could "cut corners" in its pursuit, if it were able to recognise correctly some regularity in a prey's moves). On doing so it could reduce its expenditures of energy and improve the efficiency of the whole community of problem-solvers.

Boredom[10] is the recognition by an agent of certain patterns/situations that are not changing and that correlate with the agent's inability to accomplish its primary goal or to the community's inability to solve a given problem. An agent (even a very simple one) ought to perceive (or be told) that it is "not doing well" and it should have the ability (even if it is very limited) to look at the environment and reach the conclusion that something in its immediate neighbourhood should be changed to give any hope of improving its current state. Since our primary goal is to design an architecture that is very simple, we must include this issue as a low-level architectural choice if we include it at all. Fortunately a low-level version is realisable, using no more than the ideas we have just mentioned. Even when dealing with high-level design choices, however, a designer will be faced with similar issues.

[10]The term "boredom" is used here to emphasise the kind of agent actions that would occur on observation that a given situation, value of a variable or a score etc., may have persisted for too long, given that progress towards a goal is not satisfactory. It is not intended to imply that our treatment is a model for boredom in living agents. For example, it is consistent with actions that follow from preset timeout conditions in a monitoring program, and could in principle be described in terms of "timeouts" rather than "boredom".

Within the scope of the pursuit problem, conflicts occur because two agents may want to occupy the same position simultaneously. In our implementation we allowed conflicts to occur and be solved by the agents following very simple rules [11]. Agents were allowed to solve their conflicts in one step of discrete time, after which any agent remaining in conflict over an intended move was required to stay in its current position.

7.7.1 Prediction

Since we wanted to keep the architecture as simple as possible, we decided to allow agents to predict only the prey's moves (and, for example, not to allow other inferences about a prey's internal state or structure). Bearing in mind the geometrical flavour of the pursuit problem, in which straight lines belong to the set of primitive concepts, the simplest predictive choice was to allow the agents to predict the prey's moves in a linear manner. Two different *decision modules* were implemented and both architectures were subjected to genetic search (which also included the search for their parameters). Figure 7.2 depicts one of these modules.

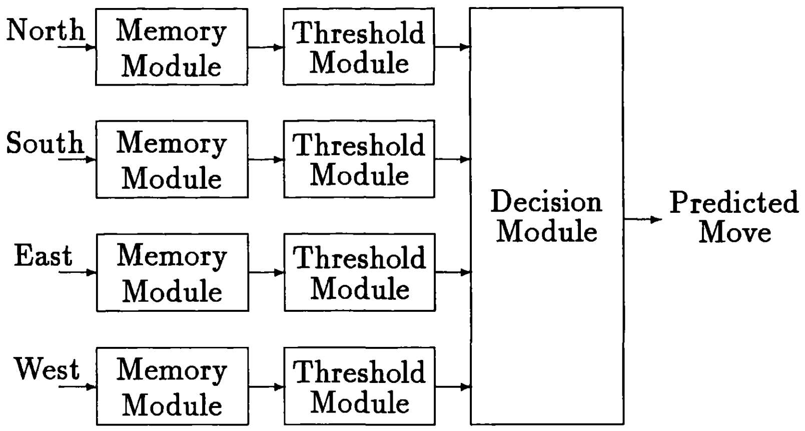


Figure 7.2: Basic predictor model

In this module the prey was considered to be moving in a given direction if the corresponding memory count was greater than or equal to the threshold. In the second option a direction was accepted as valid if the ratio of its corresponding

[11]e.g., in one implementation an agent engaged in conflict performed in a *polite way* by changing its intended move to another direction which was chosen (from a set of alternatives) in a random way or by taking its second-best alternative from the same set.

memory count to the sum over all memory counters was greater than the threshold value. The decision module was included since in both implementations two different directions could be accepted as valid.

Once a predicted direction of move was found, the agent used a simple linear rule by which it projected the prey's future position on the direction of move. The final decision on where to move in the next step was made by comparing the distance towards the target position and the corresponding prediction. If the former was bigger, the agent moved to the latter. Otherwise it moved to the target position.

In the same way as above, we considered the design of memory modules throughout the entire architecture in the simplest way possible. This is illustrated in figure 7.3.

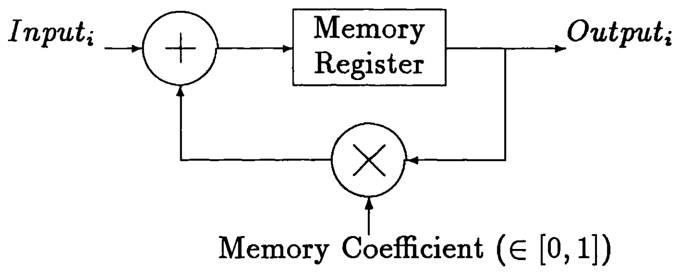


Figure 7.3: Memory model of past observations

7.7.2 Boredom

Figure **7.4** shows how we have implemented a simple architectural correlation-based scheme.

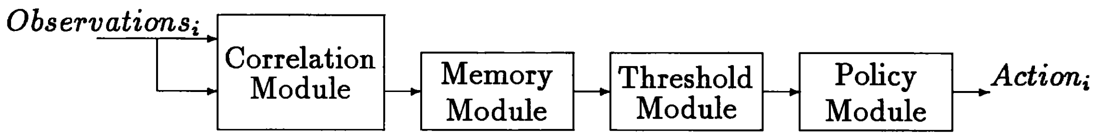


Figure 7.4: Basic correlation scheme

Whenever an agent captures a position it broadcasts a message to indicate that that position is no longer vacant. Once in a captured position an agent will tend to remain there by following the prey's movements. If, however, insufficient other agents have turned up to surround the prey after some interval, the agent may become bored (i.e. when its associated boredom level exceeds a given threshold value) and eventually may release its position. The information that the prey has not been surrounded is correlated with information on the prey's movements, such as change of direction, detection of movement and the number of agents surrounding the prey. Once an agent releases a position it may for a number of iterations (*i*) stay in the same position, (*ii*) choose any random movement, (*iii*) backtrack, i.e., move in the direction opposed to its preferred choice of movement or (*iv*) when bored, choose the best alternative movement apart from following the prey and then target the nearest capture position that is available.

The last option has several interesting consequences. If the second-best choice is a distant target, then the agent may refocus its attention (after some time steps when it goes back to its normal state) on exactly the same position that it decided to release when it became bored. To a human observer this action would be akin to one where the bored agent is broadcasting a very primitive and "deceitful" message ("*I intend to release a captured position*"). This is because some agent in the vicinity of the bored one may thus opt to target the available position. Because the bored agent may also try to target the released position in a future move (possibly the next one), this will result in conflict. In the ensuing resolution of the conflict the agents could find themselves in a better configuration. Viewed in this way the "deceitful" message is a call for cooperation.

In addition, in situations where an agent is blocking a capture position adjacent to another prey, the release of its current captured position may be beneficial as it allows for some reconfiguration that may result in the capture of two or more prey at the same time.

Once an agent has become bored, it will remain bored for some time until its boredom level falls below a second threshold. If this latter value is reached, the agent returns to its normal state.

For this module, we used the GA to search for the best combination of correlation strategies, memory parameters and also the best policy to be implemented once an agent reaches a given boredom level.

We believe the concept of boredom is a primitive but interesting architectural idea that is worth considering for inclusion in the design of many types of agents and DAI systems. In the pursuit problem (as in many other conceivable DAI applications) "a little boredom goes a long way". This is illustrated by Table 7.1, which depicts the best pursuit outcomes found by the GA on a 5×7 pursuit game for architectures including or excluding boredom (see Section 7.8 below for further details on the optimisation issues).

Boredom			No Boredom		
Escapes	Captures	Stalemates	Escapes	Captures	Stalemates
185	10	15	200	2	8

Table 7.1: 5×7 Pursuit outcome

The introduction of boredom allowed captures to occur in one-third of the games while stalemates occurred in half of them[12]. This is reflected by the normalised success ratio, which was equal to 58% in the architecture with boredom as opposed to only 22% in the simple architecture.

7.7.3 Conflict resolution

To examine conflict resolution, we implemented several simple policies and used the capability of the GA to search for the most effective one. In a pursuit game involving more than 4 pursuing agents the simplest form of conflict is related to the fact that two or more agents may want to occupy the same location simultaneously. In accordance with the design restriction of having no global knowledge, agents resolved their conflicts by changing their choices of where to move next following very simple rules. If an agent persisted in conflict after trying to follow the rules, it was forced to stay in the same position. Conflicts also occurred due to the prey's movements, since an agent that had captured a position and was following the prey's move could be dragged to the same position as another agent. Whenever this occurred, in our approach, the docked agent was forced to undock.

7.8 Optimisation issues

We have so far described the model parameters that we have subjected to genetic search and have discussed some performance metrics that can be used for comparing

[12]In effect one out of the two captures in the no-boredom architecture occurred because a red agent was surrounded by three blue agents and another red agent that had previously moved to an effective "capture" position by being stationary just out of the grid boundaries (so it should be considered as a supercapture, although it was a very opportunistic one).

the efficiency of different architectures. Recall that one of our aims is to suggest games that look best as testbeds, i.e. games in which the low-level components of an agent's architecture do not lead to a performance that has either a very low or a very high rating. As such, a game will be considered very complex if a community of simple problem-solvers cannot be found such that it attains a minimum level of performance with respect to the former. On the other hand, if we can find an architecture of simple agents that achieves a high performance in a given set of games, we shall consider the set of games as not being of interest for the purposes of a DAI testbed. Stated in other words, given a set of games we shall use a GA to evolve an architecture of simple agents that, among all those generated in the genetic search, achieves the highest performance. A set of games will be considered as a good testbed if the performance of this architecture does not rate either too low or too highly by the given measure of success.

In this way, the search for good games will comprise three phases: (1) finding sets of games that look interesting in principle; (2) submitting each set of games as a problem to be solved by an architecture of simple agents that will be evolved using a GA; and (3) identifying among the set of games those for which the performance of the architectures evolved by the GA falls within a given range. In this section we shall consider how item (2) can be accomplished.

Each candidate solution produced by the GA will comprise the parameters of a given agent's architecture, the performance of which will be calculated by requiring it to play a given set of games. A GA evolves sets of candidate solutions towards *populations* of *fitter* solutions. Within our interpretation, a solution (or an agent's architecture) will be considered satisfactory (or fit) if a corresponding community of agents (with the given parameters) achieves a reasonable performance when playing a given set of games. Ideally we would like to regard the GA as being successful in its search for the architecture's parameters if it can produce solutions near the global optimum, i.e., if no other set of parameters exists such that these result in a better performance. This brings into perspective the following interrelated points:

- What is the criterion to be optimised by the GA?
- What will be considered as a good "fit" solution?
- How does a given architecture that was found by optimisation against a fixed set of games perform with respect to different initial configurations?

The natural candidates for the optimisation criterion were initially the success ratio and the success efficiency. We first examined these criteria with respect to the 4×1 game and the performance metrics suggested in [86]. Once a particular architecture was found by the GA, we inspected the solutions via an animated visual display. Based on this, we observed that solutions produced by optimisation of the success-efficiency metric were of worse quality than those produced by optimising the success-ratio criterion. The former led to communities of agents that hovered in the grid without approaching the prey in a consistent way, and yet achieved high values of the efficiency metric.

The last observation can be understood by the fact that, if a blue agent does not move but still manages to block one direction of the red agent, it is behaving in a very efficient way. The GA was able to find architectures where agents seemed to be exploiting this approach passively rather than actually exploring the possibilities that were open to them. We have observed that minimisation of the sum of the success-ratio and success-efficiency metrics resulted in systems with the most effective game-playing behaviour. Other metrics could be generated by different weighted combination of the performance metrics. However, we have used the success ratio instead since it produced good-quality solutions and it was simpler and faster to calculate.

Since theoretical information about the global optimum was in general unavailable, we decided to use the GA's results as natural candidates. In the 4×1 game this offered no problem as solutions achieving the maximum allowed value were always produced. In more complex games, however, we took the viewpoint that (lacking any other significant information) the solutions found by the GA could be used as a baseline for comparison. The GA started with a population of 50 individuals chosen at random and we let it run for 3500 trials. The individual solution with the highest success ratio was chosen as the fittest one. In general, due to limits on our computational resources, we ran the GA once for each set of games. It is a known fact in the GA literature [28] that a GA can become trapped on local optima and hence running the algorithm several times is usually recommended when one can afford to do so. As we used it in very complex games that demanded a great number of computations, running the algorithm several times would have been prohibitive (although playing the set of games in parallel on several machines would make the task much easier).

Tests were performed to assess the variance of the solutions generated by the GA using 7×3 as an example. We ran the GA 10 times and collected statistics concerning the results. The standard deviation of the best success ratio obtained in each run was more than an order of magnitude smaller than the corresponding mean. We observed that in all experiments conducted, the GA managed to find good-quality solutions and did not seem to get trapped prematurely in any particular area of the search space. This is particularly pleasing once it is recognised that the GA was attempting to find good combinations of very different parameters such as thresholds and policies to be adopted when a given threshold was achieved. The solutions provided by the GA continued to perform well when we considered different sets of initial configurations.

To assess the performance of the GA against random search we ran both algorithms on a 8×4 game. The solution found by the GA was 72% better than the one found by random search (which incidentally was found in the first few iterations).

7.9 Experimental results

We have conducted experiments to provide suggestions of answers to the following questions:

- Is the 4×1 game sufficiently complex to be useful as a testbed for different architectures?
- How does a particular architecture that was optimised for a 4×1 game behave in more complex games?
- In case an answer to the first question happens to be negative, what games have enough structure and complexity to be exploited as useful testbeds?

The last question is particularly important, and our response to it constitutes a significant novel result for this work.

The answer to the first question proved to be particularly easy: a 4×1 game could be played successfully by a community of very simple agents with a performance equal to that of the best architectures reported elsewhere. Table 7.2 illustrates the performance of an architecture in which agents relied only on their rudimentary predictive capabilities and conflict-resolution strategies obtained through optimisation via the GA (in this table, Conflicts refers to the number of conflicts per experiment while Messages refers to the number of capture/release messages per experiment). Since the number of agents in a given conflict was almost always two, the number of actual messages broadcast was less than 15 per experiment. This compares rather well with the results reported in [86].

Escapes	Captures	Stalemates	Success Ratio	Conflicts	Messages
0	29	1	98.33 %	3.067	7.533

Table 7.2: Performance metrics of predictive architecture in a 4 × 1 game

The inclusion of boredom in the basic architecture resulted in 100% success ratio with a very much reduced number of conflicts per experiment (1.5) and a slightly higher number of capture/release messages per experiment (8.1). Hence, 100% success ratio can be achieved in a 4 × 1 game with a minimal amount of coordination and communication.

Figure 7.5 illustrates the success-ratio performance of a simple architecture of agents optimised by the GA when used to play more complex games. It is consistent with the observation [86] that a distributed-control system degrades gracefully and that enough simple agents can be added to achieve a desired success ratio.

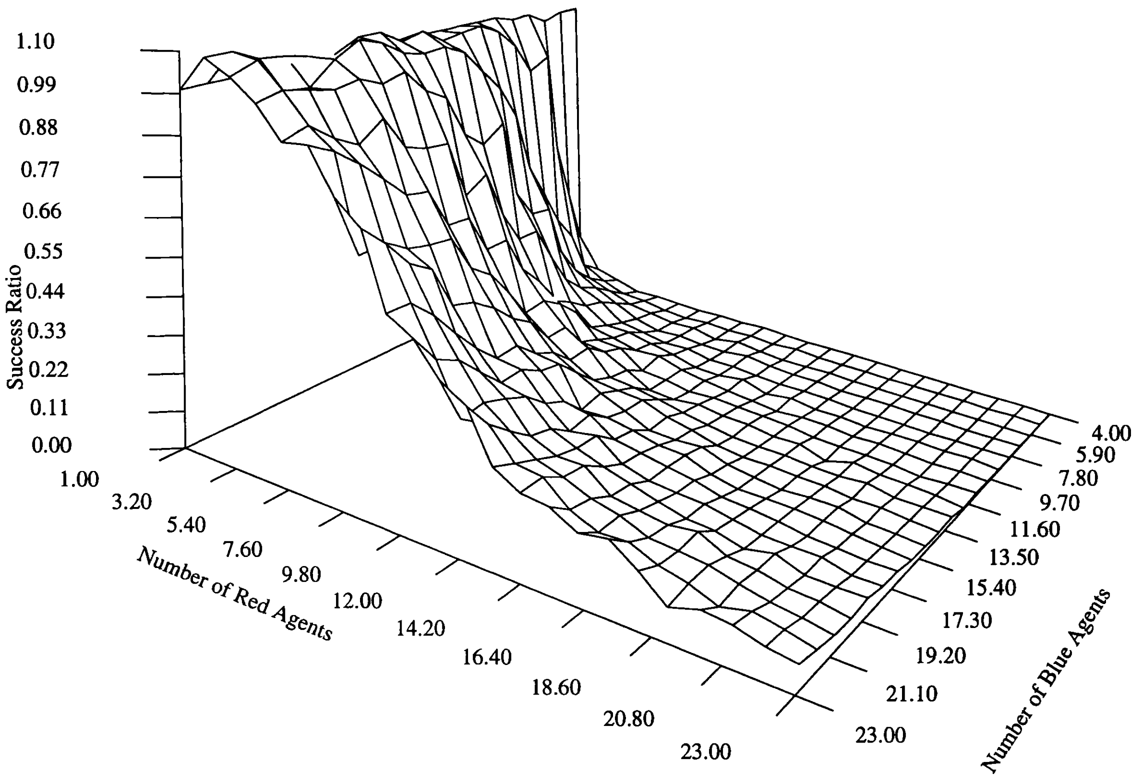


Figure 7.5: Success-ratio performance of an optimised "4 × 1" architecture in complex games

Note that some of the games have an associated value for the success ratio greater than 1.0 which is due to the fact that (as explained in section 7.6.4) we give a super-capture the rating of 1.5. Figure 7.5 suggests degradation of the performance metric in a Gaussian way where the standard deviation is proportional to the number of blue agents. This behaviour is illustrated in Figure 7.6 where we plot success ratios against the ratio of red to blue agents (and where the family of curves shows the different success ratios that occur in figure 7.5 as one moves along the lines parallel to the number-of-red-agents-axis in the red agent-blue agent plane). Figure 7.5 also serves to suggest areas of the game space where games of greater complexity (i.e. in which the performance of the agents' architecture achieves very low values) can be found. It must be emphasised, however, that it reflects the performance of an architecture specially optimised for a 4×1 game when confronted with more complex games. (A specific architectural optimisation for each particular game would in principle produce better results).

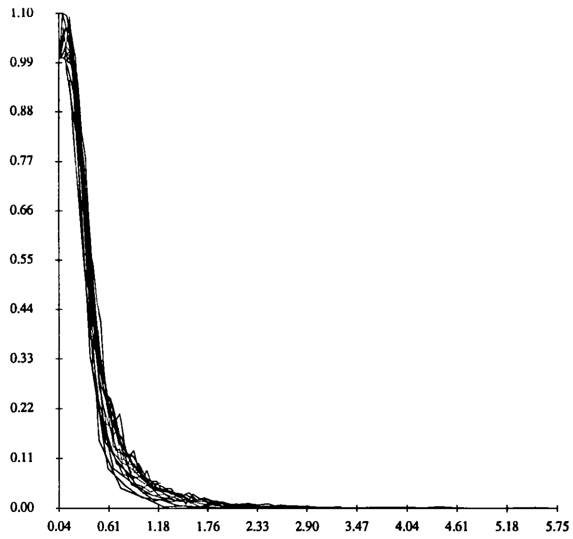


Figure 7.6: Success-ratio performance against ratio of red to blue agents

Using the information illustrated in figure 7.6 as a subjective guideline, we reasoned that games within the region $[0.3, 1.2]$ formed an initial set of good testbed problems. Yet this still left us with a huge number of games to be evaluated. The next step was to narrow down the search to a class of games for which all the red agents could be captured (at least in principle) using all the available blue agents (albeit probably demanding a lot of coordination among them). Our interpretation of the data is that **the best class is the class of $(M+4) \times M$ games**. (For example, $N = M + 4$ holds on a line that is a good subjective separator of the high-gradient region and the plain-like region on the surface in figure 7.5). Incidentally, the 6×2

game is the first element of this class of games, for which the above ratio is exactly 1/3.

Figure 7.7 shows the optimum success ratio found by the GA when used to search for the best value of the performance metric in this set of complex games. It illustrates the utility of implementing boredom in an agent architecture over one that does not take it into account. In this class of games the degradation in performance is exponential at first but falls off very slowly for games involving more than 7 red agents with the curves oscillating slightly but showing a steady and slow decline as the number of red agents increases. It appears to indicate that the increase in the number of agents above a suitable non-small value does not introduce much extra complexity into the games played. Values of M below or equal to 4 seem to lead to games that can be tackled with reasonable success by communities of very simple problem-solvers and as such should be playable with very high efficiency by any agent architecture with good high-level design features. Therefore it is possible to conclude that the 6×2, 7×3 and 8×4 games represent qualifying exercises for any such candidate architecture, while the serious tests for the qualifiers start with 9×5.

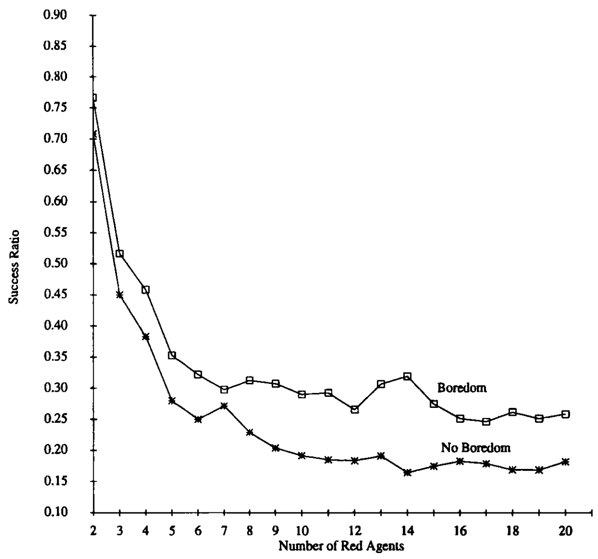


Figure 7.7: Unnormalised success-ratio

Figure 7.8 shows the number of conflicts occurring per experiment while figure 7.9 shows the number of capture/release messages broadcast. In all experiments conducted, almost all conflicts involved only two agents. Obviously an increase in communication should be taken into account in those architectures where a conflict-resolution policy relies on agents exchanging some form of communication. But figure 7.9 in particular indicates that even a simple change in the features of agent design can produce a surprisingly large increase in the demand for communication bandwidth.

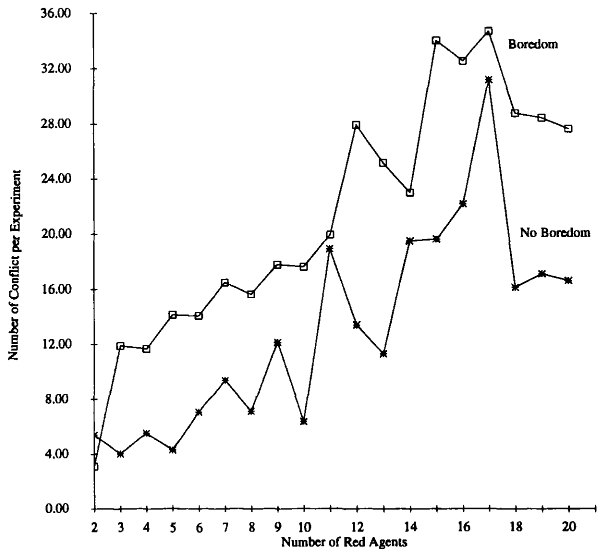


Figure 7.8: Conflicts per experiment

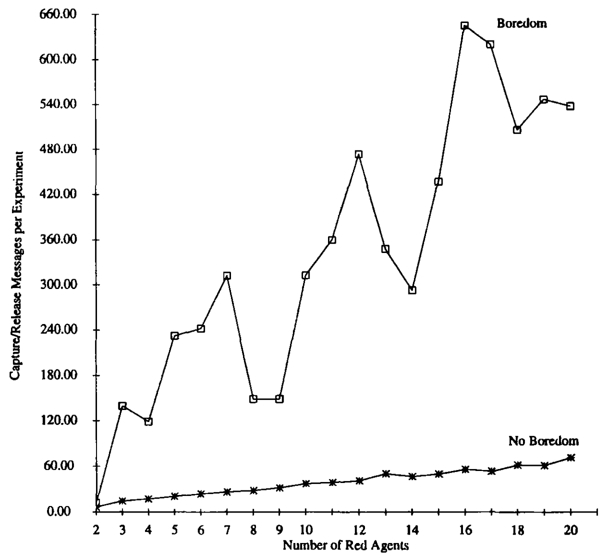


Figure 7.9: Broadcast messages per experiment

7.10 Communication

As already argued in chapter 6, cooperation is often made possible through communication and social organisation. We shall follow Wilson in [96] by regarding communication as an act on the part of one agent ("the sender" or "the transmitter") that alters the probability pattern of behaviour for another agent ("the receiver") in an adaptive fashion. From this perspective the architecture so far proposed already has embedded in it a minimal capacity for communication which becomes apparent when an agent captures/releases a position, or when conflicts are resolved. In the former case communication is explicit as we enforce the condition that an agent should always broadcast a message in such situations. In the latter case communication is implicit, as an agent will probably change its immediate goal as a result of the conflict resolution policy.

In complex DAI systems, however, communication should in principle be more sophisticated than the form discussed above. It is therefore natural to enquire to what extent a minimal communication capacity (beyond the one considered so far) can be designed into the agents' architecture such that it is beneficial to the community of problem-solvers, if this capacity can be designed at all. We want to stress once again that we are not interested here in studying paradigms for the evolution of communication. Rather, we shall assume that communication is one extra dimension that a DAI designer should take into account when confronted with a given problem, and we would like to convey some information about the increase in efficiency, if any, that is provided by simple communication mechanisms. We consider next the particular design decisions that we took with regard to the pursuit games.

7.10.1 Issues of communication

We are interested to investigate here the possibility of embedding a minimal communication protocol in the agents' architecture. The simplest way to consider communication from the point of view of the sender is to allow it either to emit a signal or not[13]. We have previously considered agents' architecture for which no explicit

[13]A more sophisticated agent, for example, could be provided with a high-level mechanism to decide in what circumstances it should communicate and what message should be broadcast in each one.

message was broadcast apart from the occasions when an agent captures/releases a position. It is now reasonable enough to focus attention on agents that would under certain circumstances try to communicate at all times.

When should a simple agent wish to communicate? In a pursuit game it is fairly obvious to assume that once an agent captures a given position it would wish that the remaining positions surrounding the prey should be captured as well. Therefore, once in a captured position an agent should always broadcast a simple message: "*I need help to capture this prey*". We have implemented this capability by allowing an agent to broadcast only its coordinate positions. The semantics of the message is to be understood as a call for help (or cooperation) by an agent at the transmitted position. Any agent receiving the message will therefore assume that an agent at that position is requiring some help to surround a prey.

Communication can of course be omnidirectional or directional. Since we wanted to keep the agents' architecture as simple as possible, we only considered communication to be omnidirectional.

So far for the transmitter of the message. An agent that receives a message should be confronted with the following issues:

1. Shall I listen to all broadcast messages or shall I have a selective response to them?
2. What are the thresholds for reaction?
3. What shall I do about any signal that I hear above the threshold?
4. What sort of information do I hold? What sort of correlations do I make?

There are various alternatives in tackling these issues. For example, an agent that receives a given message (with the coordinates of the sender) could compute its distance to the sender and disregard that message if it is below a given threshold ("*I can't do much to help in this situation*"). We have adopted instead a different approach that is grounded in the physics of radio communication and which views communication as being subject to attenuation. Within this interpretation a message can be regarded as having a certain intensity and as it propagates it is attenuated by the medium. What the receiver "listens to" is the intensity of the messages reaching it[14].

Given the above framework an agent would still be confronted with the task of integrating all the information that it received. It would also be faced with the decision of how to react to the incoming flux of information. We discuss in the following sections the design alternatives that we have taken. We hasten to point out that we believe they are general enough to be of practical use in other applications besides the Pursuit Problem.

7.10.2 Agents' communication model

As far as communication is concerned an agent views its world as a circle with an infinite radius in which it is placed at the centre. The circle is divided into a given number of sectors (the exact number is to be evolved by a GA search). In this way an agent effectively clusters all the information on a per-sector basis. The size of a sector corresponds to the maximum resolution that an agent has such that it is able to distinguish between different sources of information if and only if they are located in different sectors.

The amount of communication in a sector is equal to the sum of the intensities of all signals emitted in that sector. If the amount of communication in a sector is below a given threshold ("squelch value"), the agent regards the information coming from that sector as **noise** and simply does not take it into account. Obviously, the GA is used to search also for the squelch value.

7.10.3 Attenuation law

Different alternatives exist to model the attenuation law corresponding to the communication medium. The most obvious of them is given by the following expression:

$$Intensity(r) = \frac{I_0}{r^p}$$

[14]The intensities are actually calculated by providing each agent with a rule to compute the attenuation law in that communication medium. The GA in this case tries to find an "optimum medium" to propagate the messages!

where r corresponds to the distance to the transmitter of the message, p corresponds to a "power law" for the communication medium[15] and I_0 is taken to be equal to 1 without loss of generality. In principle there is no particular reason why one should only consider an exponential power law and not others (e.g. Gaussian). We have adopted the following attenuation profile which is inspired by Cauchy's distribution in statistics [65] and which will therefore be referred to as a Cauchy profile:

$$Intensity(r) = \frac{\alpha^2}{\alpha^2 + r^2}$$

where α is a parameter that corresponds to a power law. A value of α equal to 0 corresponds to the situation where an agent cannot hear anything. A value of α equal to 80 corresponds to a situation for which the intensity of a signal transmitted 30 units away from a receiver is attenuated by less than 13% when it reaches the latter. We have thus considered this case to be a reasonable approximation to the situation where an agent can hear anything broadcast in the grid that we have been using. Hence $[0, 80]$ seemed to be a reasonable choice for the interval of α.

The decision to opt for the Cauchy profile was based on the following reasons. First, the shape of the profiles changed only marginally for values of α that differed by a small amount. This is consistent with the idea of supporting a smooth "fit" to these values by a continuous curve discussed in previous sections.

Secondly, we wanted to avoid the situation where the attenuation varied by a large margin for agents whose Manhattan distances differed only by small values. For example, consider an exponential attenuation law with $p = 2$. An agent that is one unit away from the transmitter will receive a signal with intensity equal to the transmitted one. On the other hand an agent that is placed one horizontal unit and one vertical unit away from the transmitter will receive the same signal attenuated by a factor of 2. As a result, no attenuation factor occurs in the interval $(1, 2)$. The Cauchy distribution appears to offer more control than the traditional exponential law in avoiding this potentially undesirable situation. We remark in passing that we are considering a two-dimensional space and not the space of 3 or more dimensions where one normally finds power-law potentials. It is not unknown in physics that in two-dimensional problems where potentials exist, the potentials are not straight power laws.

[15] $p \in [0, 3.5]$ seems to be a good range of allowable values.

7.10.4 Reacting to communication

Once the calculations of the signal intensities corresponding to all sectors is completed, an agent will only consider for its interest the highest-intensity sector among those that are above the squelch value. It will then try to move in the direction of the bisector of the corresponding angle. This was judged to be a fairly simple and unbiased choice of direction. If no sector existed such that its corresponding intensity was greater than the squelch value, the agent should choose to move to the nearest prey.

A latency time for listening to communication was also introduced to allow an agent to choose a time interval[16][17] in which it did not hear any signal regardless of its intensity. This proved to be particularly useful in connection with an artifact in the design that we have called the "Keystone Kops" effect[18] and whose removal exemplifies the benefit of using a GA technique as a tool to assist the design of the architectures.

7.10.5 The "Keystone Kops" effect

From the previous discussions, it follows that once a sector was chosen an agent would try to move in the direction of the bisector of the corresponding angle. If, however, it was one unit away from a transmitter, it might decide to go to the transmitter's position instead of capturing a possible vacant position around the prey. In this way it would generate a detrimental conflict. Without modifying this design choice, the GA consistently generated solutions where agents opted not to

[16]A latency time equal to 0 implies that an agent should listen to messages broadcast in the communication medium at each time step. On the other hand a latency time equal to n implies that in only 1 out of $(n+1)$ consecutives moves does an agent check the incoming messages. Note also that an agent uses the squelch value to assess whether the information coming from a given sector should be treated as relevant or just noise. As a result, even in the case when an agent decides to listen to any messages broadcast, it may not react to any one of them as it may decide that they are just noisy information.

[17]In effect, this behaviour shows an agent tuning its listening rate to the bandwidth of incoming information that it can handle best, given its other characteristics. This is an echo of the cybernetic "law of requisite variety" idea, produced some time ago to refer to similar situations [3]. I am grateful to an examiner for pointing that out.

[18]A classical comedy feature in some of Hollywood's silent movies, in which policemen in a chase end up colliding with and grabbing each other or innocent bystanders instead of the actual object of their pursuit.

hear anything, thanks to a suitable choice of a latency time. Once this artifact was corrected, the solutions generated by the GA were consistently better than the ones previously obtained.

We note in passing that the Keystone Kops effect is not restricted to the particular application explored here. It may well appear disguised in others such as path-planning via potential fields. Here for example, a robot whose task is to capture an egg might end up with its hand-grip in the correct place although the result of its approach might be catastrophic to the egg's structure if care were not taken to ensure that it should reach the egg's surface with a velocity very close to zero!

7.10.6 Experimental results

The GA was used to search for architectures that included communication in the design. With regard to communication the search included sector size, latency time, the parameter α for the Cauchy profile and the squelch value. Since the introduction of these parameters increased the length of the chromosome, a population of 70 individuals was used instead of the previous 50.

Figure 7.10 compares the optimum success ratio found by the GA with the previous results. It shows that communication is beneficial in most games, especially those involving a large number of agents. The results are even more remarkable if one considers the fact that they were obtained using the same number of trials as in the previous experiments, i.e. 3500, although the chromosome length had increased by more than 29%. When the number of trials was allowed to be 5000 the results improved consistently throughout the games considered (although in most of them only by a small fraction). In either case, in some games performance increased by 40% over the case when communication was not implemented. The performance curve has similar characteristics to the ones considered before, i.e. with degradation being exponential at first but falling off very slowly for games involving more than 7 red agents. The conclusions with respect to the most suitable games to be used as testbeds are consistent with the remarks made previously, and hence reinforce them (since the architectural considerations of section 7.10 and previous sections of chapter 7 are basically independent).

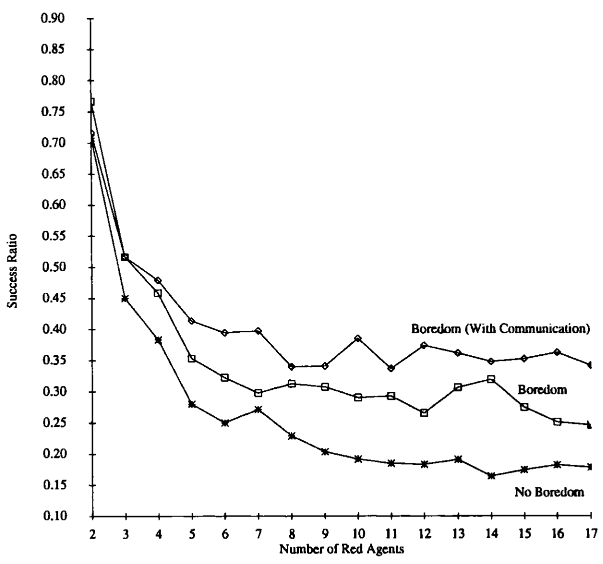


Figure 7.10: Unnormalised success-ratio

It is instructive at this point to make some speculations suggested by the experiments conducted. It must be observed, though, that to make more robust and finer conclusions would require a bigger volume of experiments than we can currently provide. Hence the following conclusions should be read with care, although in my opinion they suggest interesting lines of research to be pursued in future work.

In order to understand how I reached them, let me first describe the conditions on which they are based. For each game explored, we saved 17 architectures that achieved the highest scores according to the performance criterion. In general, we rarely observed differences of performance of more than 1.2% among the best architectures found by the GA in each game explored[19].

Table 7.3 shows the percentage of games where each communication parameter was fixed in all the 17 best architectures.

Sector Size	Latency Time	Attenuation Law (α)
70.6%	82%	23.5%

Table 7.3: Communication parameters and percentage of games where they were fixed in all the 17 best architectures.

We see from table 7.3 that one has much more freedom in choosing different values of α rather than the other two parameters without compromising the optimum

[19]In fact, in most cases the best architectures achieved the same value of the performance criterion and whenever they differed, the corresponding values usually varied by at most 0.5%.

performance level. Usually the range of variation of α was very large. Note that the rate of change of the attenuation law with respect to α is given by,

$$\frac{d}{d\alpha}Intensity(r) = \frac{2\alpha r^2}{(\alpha^2+r^2)^2} \approx \begin{cases} \frac{2\alpha}{r^2} & \text{if } \alpha \ll r \\ \frac{1}{2\alpha} & \text{if } \alpha \approx r \\ \frac{2r^2}{\alpha^3} & \text{if } \alpha \gg r \end{cases}$$

Thus in general we can expect quite different values of α to produce similar profiles, particularly when α is large. The results of the experiments indicate that the performance of the collective of problem-solvers is not particularly sensitive to changes in the profiles.

Table 7.4 lists the range/value that occurred most often for the sector-size and latency-time parameters in all the best architectures (in all games explored). We should point out that with regard to the sector size we have observed values of 60 and 90 degrees occurring in 16% of the total cases while we did not observe any correlation between this parameter and the number of agents involved in the games. That was not the case for the latency-time parameter, as in all games involving at least 12 red agents all the best architectures found by the GA had latency times equal to 0.

Sector Size		Latency Time	
Range (degrees)	Percentage of Total	Value	Percentage of Total
[10, 30]	71%	0	47%

Table 7.4: Percentage of occurrence for all the best architectures (range/value).

We may conclude from the above observation that in games with a larger density of agents, i.e. those with at least 12 agents, it pays to listen to the communication medium at all times. This may be linked to the dynamics of the games, since a larger number of red agents in the grid may act disruptively in configurations of pursuing agents that were almost achieving a capture. The movement of the prey may induce conflicts among the pursuing agents and as a result, "near captures" may result in escapes.

It appears that, once we introduce communication in the architecture, things get much more complicated than before. One reason for this is related to the fact that the communication scheme implemented here incorporates different design choices and there are many different ways in which they can interact without compromising the overall performance. The experiments also appear to indicate that one may get much more mileage out of exploiting the idea of latency times rather than focusing too much attention on the attenuation profiles.(This conclusion would actually be rather counter-intuitive from the point of view of (say) a radio communications engineer).

7.11 Conclusions

In this chapter we have examined the design of autonomous agents using a GA approach to explore the space of pursuit games and to suggest games in this space that may serve as good testbeds for alternative DAI architectures. Based on the experimental evidence collected, we have concluded that the 4×1 game does not serve the purpose for which it was conceived, since a minimal agent's architecture using little coordination and communication can solve the problem so well that there is nothing left for more sophisticated architectures to improve upon. To gain further insights about the game space we have explored it using an architecture of simple agents that was specially optimised for the 4×1 game. The interpretation of the results obtained suggests that the class of $(M + 4) \times M$ games forms an initial class of good testbed problems.

The experiments also suggest quite strongly that in the class of $(M + 4) \times M$ games the simplest ones that have enough complexity to serve as useful supports for tests and evaluations of DAI architectures are those with $M \in [5, 7]$. For $M > 7$, the games certainly require more than low-level architectures if the pursuing agents are to achieve good performance, but the overall computational cost of playing them may be undesirably high when one considers that the outcomes may give little extra enlightenment to DAI designers.

The results reported here have a value beyond the domain of Pursuit Problems. We have demonstrated that DAI can profit from the application of GAs as experimental tools to assist the *design* of different architectures. The assistance is of two kinds. First, it allows searches of suitable spaces (here, both "architectural spaces" and the (N, M) game space) for good design choices, when no other approach can guarantee to make such choices. Second, it is an aid to thinking about the dimensions of the spaces, as one has to consider the question of what can be parametrised when one is trying to set up a problem for GA analysis. One may not know in advance which kind of assistance will be of greater value for a given problem. In the research reported here, we have been able to take advantage of both kinds of assistance. Most of the chapter is devoted to discussion of optimal choices of low-level architectures and pursuit games. However, we have also identified some basic dimensions in agents' architecture that are potentially valid for many different DAI applications. Within this activity, we believe that the issues introduced through the concept of *boredom* are the most novel. In addition, we have also found that the concepts of "listening rate" (i.e. the latency time for listening to communication described in section 7.10.4), thresholds and attenuation of communication are significant ideas to be taken into consideration by designers who aim at an effective increase in the performance of the problem-solving agents. With regard to communication, the results reported in section 7.10.6 appear to indicate that the performance of the collective of problem-solvers is less sensitive to changes in the α parameter that controls attenuation profiles than the value of the latency-time variable. They suggest that it may be more useful to investigate details of the idea of latency times rather than focusing too much attention on the attenuation profiles. This is a surprising result which was not expected when we set out to explore these dimensions (and, as we have said, it may be counter-intuitive from at least one point of view - though possibly quite intuitively sensible from another point of view, e.g. that of a sociologist). Hence it may be a fruitful topic for further research.

Chapter 8
A non-standard approach for some standard optimisations

The two previous chapters have considered a non-standard application of GAs in which they are used as tools to assist the design of DAI architectures. However, no theoretical evidence exists to suggest that the results obtained are the optimal solutions for the games explored, because of the absence of a "theory" for the games and for DAI in general. This may leave the impression that the approach taken so far cannot be beneficial in more traditional applications where the formulation of the problem considers the optimisation of a function given in an explicit form. The present chapter attempts to correct this impression by focusing on a nontrivial application that shows the benefits one can obtain through the flexibility provided by the GA in exploring dimensions not immediately suitable for exploration by any classical treatment. The application considers a nontrivial nonlinear constraint optimisation problem with a large search space where some dimensions are discrete and other continuous. As an added bonus it also suggests a simple methodology to handle the constraints, which may be an interesting alternative whenever a problem should have a similar underlying structure.

8.1 Motivation

In chapters 6 and 7 we have considered a non-standard application of GAs in which they are used as experimental design tools. The approach taken is particularly useful in problems where a designer wishes to explore suitable spaces for good design choices when no other approach can guarantee to make such choices. We can see the general quality of results for the DAI application, where the fact that one could not guarantee the optimality of the proposed solutions for the problems being explored was clearly not detrimental to the aims of the application.

On the other hand the relevant chapters may have left the impression that the standpoint taken there may not be beneficial in other circumstances for which the optimisation of a function given in an explicit form is required, especially when some classical treatment already exists. The purpose of this chapter is to correct this possible misconception. It is obvious by now that unless appropriate knowledge about the domain exists, not every function can be searched efficiently by a GA approach[1]. One of the goals of this chapter is therefore to indicate some features characterising application domains where the flexibility provided by the GA in exploring non-traditional dimensions proves to be very attractive. Specifically, we shall do this by considering the application of GAs to splines for curve fitting.

Although the conclusions presented in this chapter are based on experimental results, we consider them worth mentioning for the following reasons. First and foremost, they demonstrate that optimisation over parameters that are not obviously suitable for "classical" treatments of optimisation can be conducted simultaneously with exploitation of the obvious or classical variables. Second, the application is a real-world problem whose solution is of practical industrial importance to a large number of users. In addition, we discuss some technical issues that may be of interest in other GA applications that have a similar underlying structure.

Since the chapter does not fall directly within the main course of the thesis, only the most relevant points will be discussed here. The interested reader is referred to [56] and [89] for further details.

The organisation of the chapter is as follows. Section 8.2 describes the application and the problem area where the technique was applied. Section 8.3 briefly reviews the mathematical ideas that are important for what follows. Section 8.4 describes alternative solutions that have been proposed in the past while section 8.5 details the GA solution and exemplifies the benefits of using the technique and the flexibility of exploring dimensions not amenable to classical treatments.

8.2 Introductory remarks on the application

We shall centre our discussion on an application area, namely fermentation processes for pharmaceutical production[2], where the use of smoothing splines, especially cubic splines, is widespread (e.g. in the analysis of process and laboratory data). Fermentation processes are used in the manufacture of many pharmaceuticals and

[1]Which is why the issue of the identification of suitable domains for genetic optimisation is currently alive.

[2]This by no means restricts the results discussed here to this area only, as we shall point out later.

numerous food products. A major challenge in controlling these processes is their optimisation.

There are several issues here. One is related to the fact that the fermentation companies store historical measurements (or historical data) from their processes and they would like to make use of those. But such use is not necessarily simple, as one often wants to interpolate between discrete measurements and one may also want to differentiate curves[3]. Usually the elapsed time of the process is very long and the data are discrete and sparse. One may, for example, only have access to 21 measurements for a period of 200 hours.

In terms of interpolation, we would like to fill out the gaps between the data points. In terms of differentiation we would like to have a smooth curve (i.e. without noisy spikes) so that future users could be able to differentiate it in a reliable way. It is therefore important to remove noise from the data by smoothing the data properly.

8.3 Theory and methods

Data-smoothing using spline functions is an example of a "black box" approach to modelling. The spline is a parametric model which is fitted to measured data, but the model coefficients have no particular biological or physical meaning. A spline function has no dynamic or time-series component.

Spline functions are well-known to give excellent approximations for interpolation and/or smoothing of data [72]. A smoothing spline, unlike an interpolating spline, does not go through all the measured points. Instead, the smoothness of the function (measured for example, in the case of a cubic spline, by the sizes of the discontinuities in the third derivatives) can be traded against the accuracy of the fit. An interpolating spline should be used only if the data are known to be completely free of noise, otherwise some degree of smoothing is called for.

A spline function is a piecewise continuous function consisting of polynomial segments which join at points referred to as **knots**, where the values and all but their highest derivatives are continuous.

[3]For example, it is very important for fermentation engineers to know the rate of growth of cells in a culture.

A spline function $s(x)$ is defined over a set of basis functions $b_{j,k}(x)$. We shall follow [17] in using normalised B-spline functions. Given the set of data points $(x_i, y_i), i = 1, 2, \ldots, m$ in the range $[a, b]$, a spline function of degree k and with knots t_j, $j = k+1, k+2, \ldots, n-k$ may be represented as a linear combination of a finite number of kth degree B-splines:

$$s^k(x) = \sum_{i=1}^{n-k+1} c_i b_{i,k}(x, t)$$

where c_i are the representation coefficients and the B-splines, $b_{i,k}(x, t)$ are determined uniquely by the degree of spline and the knots. We shall refer to $\underline{sx}$ as the discrete samples from the spline function $s^k(x)$ at $x_1, x_2, \ldots, x_m$, and to $\underline{c}$ as the vector of coefficients.

The discontinuity jump of $s^k(x)$ at t_i is defined as:

$$r_{j,i}^k = \left[\frac{d^k b_{j,i}(t_i)}{dx^k}\right]^+ - \left[\frac{d^k b_{j,i}(t_i)}{dx^k}\right]^-$$

The superscripts + and - indicate the value of the kth derivative on either side of the knot t_i. Following [17], the spline's roughness is measured by the vector of discontinuities (evaluated at the sampled points) in its highest derivative, which we shall refer to as $\underline{rc} = R\underline{c}$. Only discontinuities at the interior knots are important and hence, for example, for a cubic spline the jth column of R is:

$$\begin{aligned} \underline{r}_j^T &= (r_{5,j}, r_{6,j}, \ldots, r_{(m-4),j}) \ , \ \text{where} \ , \\ r_{j,i} &= \left[\tfrac{d^3 b_{j,k}(t_i)}{dx^3}\right]^+ - \left[\tfrac{d^3 b_{j,k}(t_i)}{dx^3}\right]^- \end{aligned}$$

Cubic splines are able to give good fits to data that show abrupt changes. They can also mimic a wide range of analytic functions. This explains their widespread use in interpolation and/or data smoothing, although in principle other degrees of polynomials could be used.

Throughout this chapter we assume that measurements $\underline{y}$ are generated by a smooth function $g(x)$ corrupted by additive random noise. The amount of smoothing of the spline approximation should be controlled to avoid the spline following the random errors in the measurements, a phenomenon often referred to as **overfitting**.

To account for the roughness of the spline and any departure from the measurements, a cost function is introduced (describing the quality of the spline fit):

$$J(n,k,p,\underline{c}) = p(\underline{f}^T\underline{f}) + (\underline{rc}^T\underline{rc}) = p\|\underline{f}\|^2 + \|\underline{rc}\|^2$$

where f denotes the accuracy of the spline fit, $\underline{c}$ corresponds to the vector of the coefficients of the expansion of a given function in terms of the spline basis, and the elements of the vector $\underline{rc}$ are the weighted sums of the discontinuities in the highest derivatives at each interior knot. The parameter p (which must be chosen) controls the trade-off between the fidelity of the spline to the measurements and the roughness indicated by the higher derivatives.

Note that J is a quadratic function of the coefficient vector $\underline{c}$. Hence once the number of knots, their positions and a value for p have been chosen, the minimisation of J with respect to $\underline{c}$ is straightforward (using calculus techniques). The choice of p and n is a non-linear problem. It is important to note that, since the smooth function is corrupted by random noise, the standard deviations of the measurements affect the fidelity demanded of the spline. The difficult problem is to find a good estimate of the smoothing parameter.

8.4 Previous solutions

In Dierckx's solution [17] the user specifies a fidelity constraint S, and then both the number of knots and the smoothness factor p are adjusted in the search for a spline with the desired value of the constraint. The basic problem with his approach is that, if one wants to do it properly, knowledge of the errors in the data is needed. Moreover, it is also important to know the data-measurement variances precisely, because there is a statistical hypothesis test involved. Another problem with his technique is that the relationship between the specified fidelity and the quality of the visual fit is far from intuitive.

Recall that the problem is to obtain a good estimate of p that does not require knowledge of the standard deviation of the noise. Craven and Wahba [14] showed that an estimate of p is that one which gives a spline with a minimum Generalised Cross Validation (GCV) function and it provides an optimum amount of smoothing from the data. A convenient interpretation of GCV is that it optimises the ability of a set of splines fitted to all except one data point to predict the omitted value. In their implementation they considered the case where the number of basis functions was the same as the number of data points. Unlike them, we would wish to treat the number of polynomial segments as a variable. In fact, our simulations show that one does better by reducing the number of knots.

To sum up all the above, we are faced with three problems here. The first is related to knowledge of the errors of the data. This knowledge is assumed in Dierckx's solution but cannot be taken for granted in many practical applications. The second problem deals with the decision on how many knots should be used. It is partially explored by Dierckx but assumed to be known in Craven and Wahba's methodology. The last problem is concerned with the choice of the degree of the splines. This choice may be domain-dependent and may require extra knowledge or expertise about the process being analysed. A standard way to circumvent this issue is to assume that cubic splines provide good fits in general and use them accordingly.

The following questions are therefore appropriate. Are the above approaches generally sensible? Can we provide more insights about a given problem and hence improve on existing techniques by relaxing the above assumptions? Can we find an automatic way to choose the degree of the splines and the number of knots that produces spline fits with higher quality than the above methodologies?

8.5 A hybrid algorithm for fitting experimental noisy data with splines

The GA solution tries to combine the best characteristics of the two previous approaches. First we have incorporated the degree of the spline as another dimension in the search. The candidate set of parameters is chosen to be n_a, p_a and k_a where n is the number of knots, p is the smoothing parameter and k is the degree of the spline. A straightforward minimisation of $J(n_a, p_a, k_a, \underline{c})$ with respect to $\underline{c}$ gives the set of coefficients for an arbitrary spline $\underline{sx}_a$. $\underline{sx}_a$ and $\underline{y}$ are used to compute the GCV function for this particular combination of parameters. The GA searches for the combination that minimises the GCV function.

Note that the problem is nonlinear and p ranges over a large interval ($[0, \infty]$) and is continuous[4]. Also note that if we allow the degree of the knots to vary, then the "optimum" value of p is different for different spline degrees; so it would not be a good idea to allow p to vary over a small range of values. The problem also suffers from numerical instabilities for some combinations of parameters due to the existence of ill-conditioned matrices. And it also belongs to the class of constrained optimisation problems (albeit a simple constraint) because of theoretical restrictions on how the degree and the number of knots are related.

Since the constraints examined in the application defined simple regions in the search space, we have used a simple geometrical transformation so that no solution was generated outside the feasible region. By doing so, we were able to apply a simple genetic algorithm to search for the minimum of the GCV function without unnecessary modifications to the initialisation and reproduction phases.

8.6 Application to the penicillin process

The penicillin fermentations involved the aerobic growth of a mould, *P. chrysogeum P2*, on a semidefined medium containing corn steep liquor. After a period of rapid and almost exponential growth the initial charge of nutrients became exhausted and the growth was subsequently limited by a glucose feed, giving almost linear growth for the remainder of the fermentation.

Figures 8.1 and 8.2 show the results from laboratory assays of biomass concentration. The input information poses problems for a smoothing procedure because the data records are sparse and have diurnal oscillations defined by only a few data points.

Figure 8.1 (a) is the spline with the best overall GCV value found by the GA; it is a quadratic spline with 16 knots. Figure 8.1(b) is the best cubic spline found by the GA, with 24 knots, while figure 8.1(c) shows a 12-knot cubic spline from a local

[4]This is an important point that needs to be emphasised: our search space is much bigger than for example Craven and Wahba's because they do not allow n to be a variable. In doing so they have collapsed a multidimensional problem immediately into a single-dimensional one. All they had to find, therefore, was the value of p that minimised that GCV function. For example, the GCV surfaces have local minima, and the global minimum is not necessarily the one that has the number of knots equal to the number of sampled data points (which stands contrary to the Craven and Wahba methodology).

GCV minimum. Figure 8.1(d) corresponds to a 20-knot cubic spline from another local GCV minimum. The oscillations in these data are real effects, but the splines away from the global GCV minimum do not follow the oscillations reliably.

Figure 8.2(a) is the spline with the minimum GCV value found by the GA; it is a cubic spline with 19 knots. Figures 8.2(b) to 8.2(d) show cubic splines from local GCV minima. Figure 8.2(b) corresponds to a 22-knot spline with a smoothing factor of about the same value as the smoothing factor of the spline in figure 8.2(a). Figure 8.2(c) is from a local GCV minimum with 16 knots, and Figure 8.2(d) is a very smooth spline, but which nevertheless has a local minimum in the GCV value.

Figure 8.3 illustrates a comparison of cubic and quadratic splines for penicillin-concentration data for a second fermentation. It illustrates a major benefit that arose from using the GA to select simultaneously the model's structure and its parameters. Figure 8.3(b) shows a spline with minimum GCV value, found by the GA when splines were restricted to degree 3. Figure 8.3(a) is the quadratic spline found by the GA when allowed to search for splines of different degrees. The data record has a discontinuity at about 35 hours when penicillin production begins; the quadratic spline with its discontinuous second derivative shows higher fidelity to the data in the region of the "corner" than the cubic spline. Moreover, the quadratic spline, unlike the cubic spline, avoided giving impossible values for the penicillin concentration (which cannot, of course, become negative). It is important to mention that it is commonplace in the engineering literature to use cubic splines, but we have shown the advantage of being more general (i.e. considering the spline degree as an optimisation parameter) here.

8.7 Summary

The results reported in this chapter have some value beyond the description of "yet another GA application". First, they demonstrate that optimisation over parameters that are not obviously suitable for "classical" treatments of optimisation can be conducted simultaneously with exploitation of the obvious or classical variables. (Here, for example, the degree of the splines is a non-classical variable, and in fact the number of knots can be regarded as non-classical as well). Second, the application is a real-world problem whose solution is of practical industrial importance to a large number of users (we have also applied the technique to data obtained in the production of genetically-engineered organisms - where the nature of the data and our treatment are broadly the same as for the experiment that is the subject of this chapter - with very promising results as well.). The problem examined here is an instance of a larger class of problems in which one is required to produce approximations of multivariate functions from sparse data. Success of a technique of solution in one instance may strongly imply success in the others. As a result, a large body of important problems may be open to more in-depth analysis.

The technique described here achieved a considerable advance in the quality of fits of fermentation data. By that is meant the ability to handle the discontinuities that are so characteristic of these fed-batch processes and the ability to adjust the spline degree - which, as shown in figure 8.3, avoids the generation of unrealistic negative values. No previous curve-fitting approach to that particular problem has been satisfactory.

The application illustrates the flexibility provided by a GA approach whenever one is dealing with spaces where the constraints can be parametrised by means of a convenient choice of variables. It is precisely this flexibility that may be a particularly interesting advantage of the GA over other more traditional algorithms when dealing with problems of a similar structure.

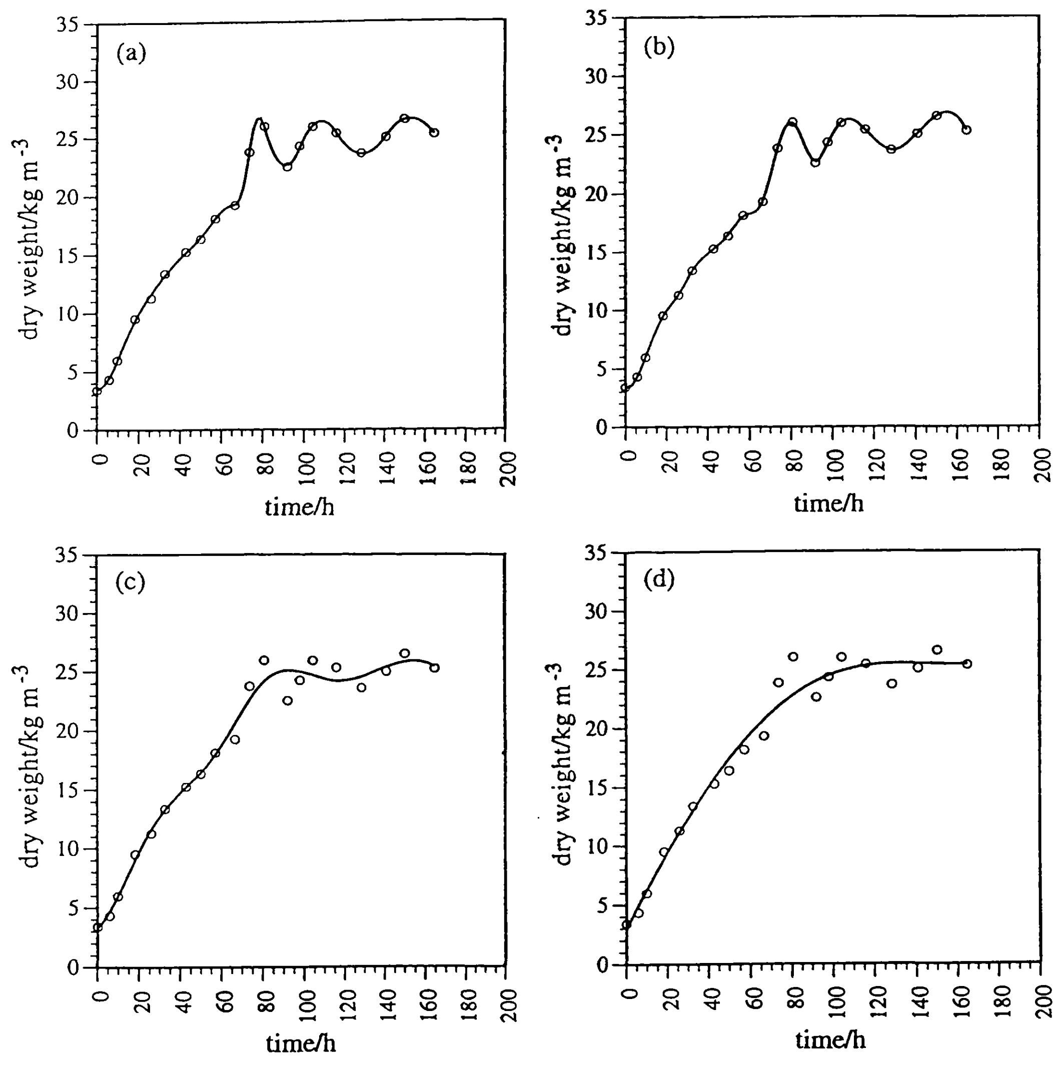


Figure 8.1: Spline fits to dry weight

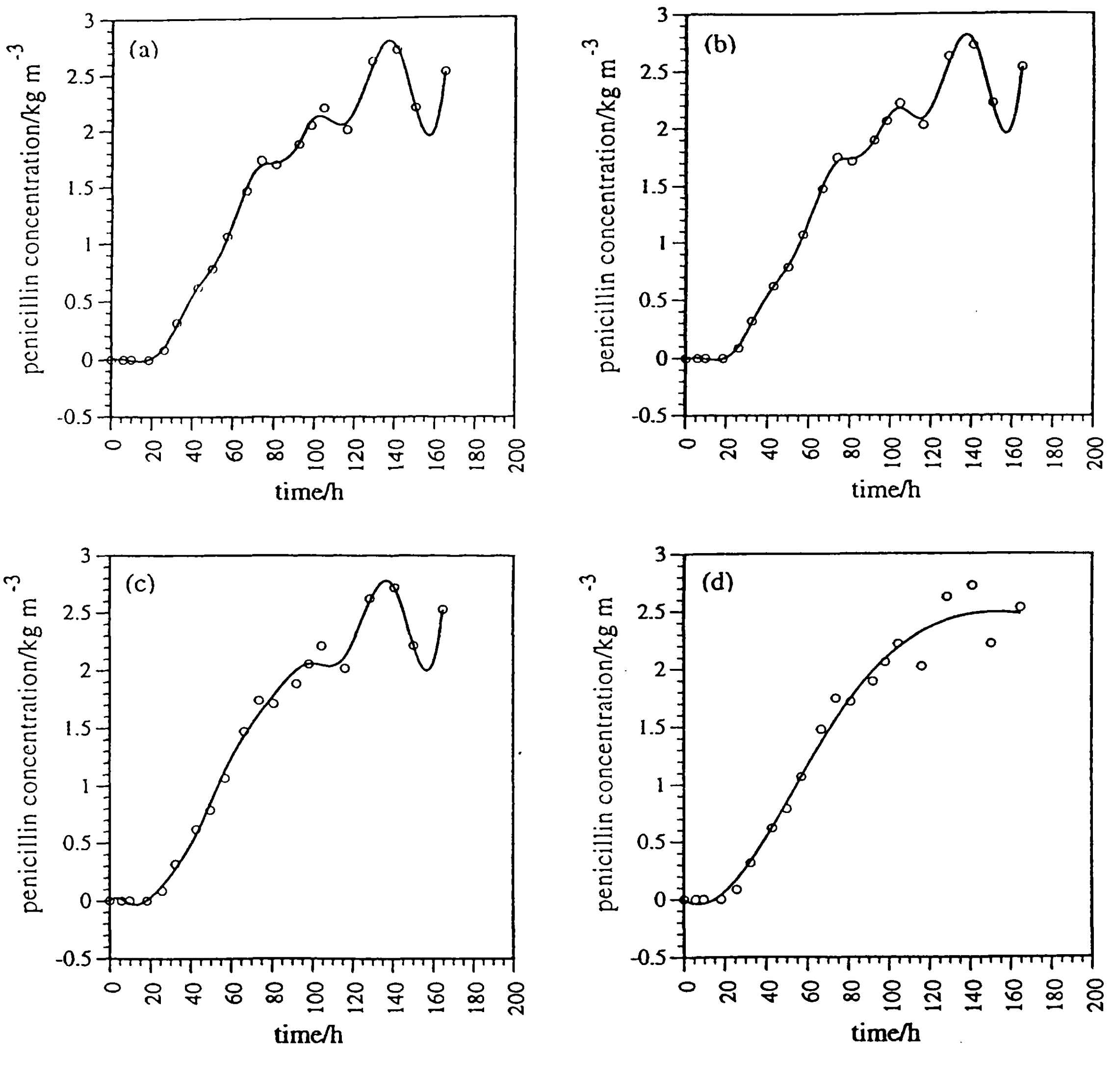
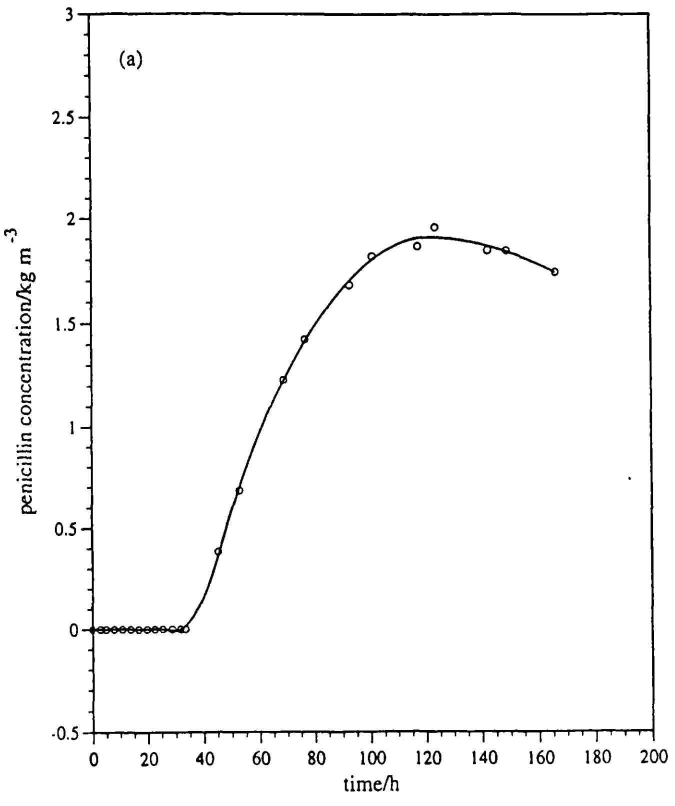


Figure 8.2: Spline fits to penicillin concentration



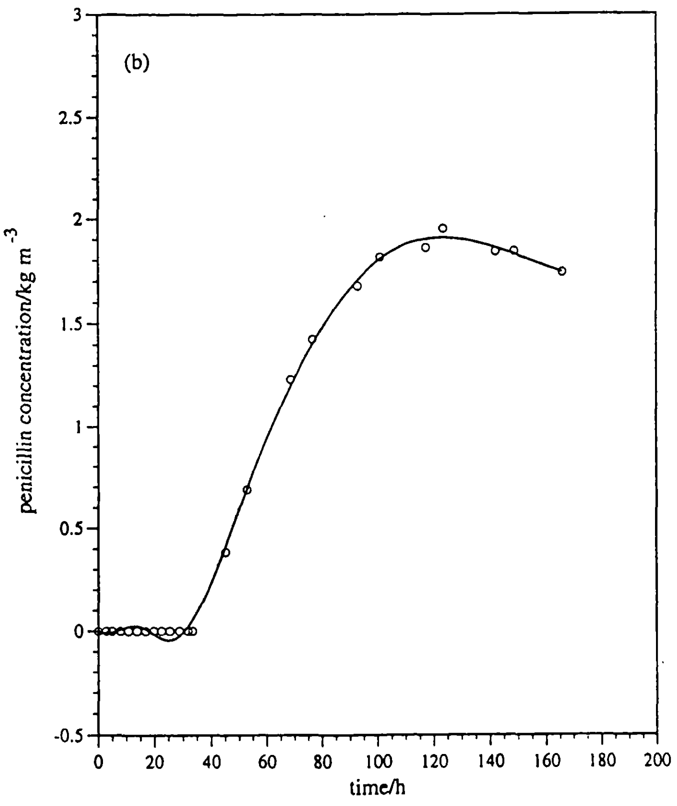


Figure 8.3: Comparison of quadratic and cubic splines fits

Chapter 9
Conclusions and Directions for Further Research

9.1 Conclusions

The research in this thesis has contributed towards advances in both theoretical and practical aspects of GAs. On the theoretical front, we have reviewed the underlying theory for many application domains of Genetic Algorithms from the perspective provided by Harmonic Analysis. One of the aims was to get further insights into the domains of applicability of GAs and to extend the analysis often applied to binary alphabets to other domains. This was accomplished in the work reported in appendix A, where we have shown that the results most often cited with regard to binary strings carry over naturally to non-binary alphabets.

A second motivation was to put the results in the GA literature into the perspective of other mathematical results already known elsewhere. This can be viewed as not only contributing useful results unknown to the existing GA literature, but also as an investigation of a more fundamental and general formulation of some of the basic concepts of the subject.

Chapter 4 has examined the idea of a suitable index to gauge the difficulty a given problem might possess for genetic optimisation. The scope of the investigation is wider than a first glance through it might indicate. One could imagine that an explicit estimation of certain Walsh coefficients would be beneficial in providing more confidence in the relevant results or serving as a suitable warning signal about the characteristics of the function being explored. Our results indicate that unless one restricts the class of functions being explored, this might not be accomplished even in the extreme case where the evaluation of the Walsh coefficients could be performed with enough statistical reliability. We have also described explicitly a whole class of functions that are of interest in some signal-processing applications, which can exhibit high degrees of nonlinearity and yet belong to the class of fully easy functions. It is interesting to note that this gives some theoretical support to recent experimental results about the ability of GAs to solve related problems which in principle (without a proper theoretical treatment) might be regarded as difficult and highly epistatic ones.

The thesis has contributed also towards the identification of two areas for which the application of GAs provides new conceptual insights and very promising results. Given the current theoretical limitations, I believe the identification of suitable domains for genetic optimisation is a desirable issue from at least two different points of view. First, it may contribute to more in-depth theoretical analysis of the domains of applicability of GAs. Secondly, from a more practical point of view one would like to advance the knowledge in areas where no other approach can currently guarantee to make such advances.

The applications in themselves share no obvious similarities. However when looked at in a "GA perspective" some commonality between them becomes more apparent. Most notable is the discipline of identifying non-standard dimensions, or even forcing the introduction of parameters in a problem description when their nature is not obvious, and subsequently subjecting them to genetic search. This contrasts with standard applications when one has only to consider the dimensions already given in the problem description. It is hoped therefore that this thesis serves also as an educational study for the cases when one has to consider a problem without clear-cut description and has to model it in such a way that a GA search can be useful and in particular, more useful than typical alternative approaches to solution of the problem.

With regard to the application involving splines, it is important to emphasise that it is an instance of a larger class of problems in which one is required to produce approximations of multivariate functions from sparse data. For example, Craven and Wahba [14] have noted that the method of generalised cross-validation is also applicable to choosing the **regularisation parameter** in the **method of regularisation** for solving Fredholm integral equations of the first kind. Poggio and Girosi [68] have noted that many problems in several different areas including system estimation, computer vision and time-series analysis can be formulated as problems of approximating multivariate functions from sparse data. When the problem involves continuous output values, regularisation techniques can be used. Since the solutions provided within the scope of **regularisation theory** are equivalent to generalised splines [68] I believe the techniques explored here can be extended to address these challenging and important domains.

One particularly pleasing outcome of the minimisation of the generalised cross-validation function using a GA approach is that the resulting technique makes very good use of very limited information. It also works rather well with sudden transitions, as indicated, for example, by the results towards the end of chapter 8 on the adaptation of the method to the penicillin data with a sharp corner discontinuity. This is a significant result for biochemical engineers, because no previous curve-fitting approach to that particular problem has been satisfactory.

The second application deals with the design of autonomous agents for distributed knowledge-based systems. One of the goals was the design of suitable testbeds for evaluating alternative architectures of distributed intelligent systems. The work focused on the Pursuit Problem, a canonical problem suggested as a useful tool for evaluating alternative approaches to the distribution of knowledge and control among intelligent, cooperative problem-solvers. Pursuit Problems are particularly interesting from an application point of view as they serve as suitable models for several real-world applications, such as finance and control. As such, they should be considered as potentially interesting testbeds for evaluating proposed software solutions for these domains as well. The contribution of the thesis is to indicate in detail (and justify) choices of good testbed games, where previous work in this direction has been more subjective than systematic.

Most of the application-oriented research conducted here has been directed towards providing good guidelines on how to use GAs as experimental tools to assist the design of autonomous agents for distributed knowledge-based systems. The effective assessment of the quality of alternative distributed knowledge-based systems is becoming increasingly important in both the theoretical and practical spectrum of distributed artificial intelligence. In the application area, the magnitude of the costs incurred with a wrong appraisal of the quality of proposed solutions can be very high if care is not taken in the design of suitable experiments before a proposed architecture is embedded in real-world applications. From a theoretical point of view it is very important to have problems that exercise the dimensions that can contribute towards the development of more extensive theories of DAI and of social knowledge and action. In a broader sense, we would like to understand how a collection of autonomous agents achieves coordination and how higher-level organisational structures evolve as a result of the interaction of the individual agents. Chapter 7 indicates that the problems proposed in this thesis have enough complexity to indeed require cooperation among problem-solvers. As such, they stand as good testbeds for exploring ideas that are important for DAI.

The research reported in chapters 7 and 8 has contributed to questioning the validity of previous standard practices in both application domains. (It has been well received at international conferences or workshops in the subject areas treated in those chapters). For example, the use of cubic splines in fermentation-process data and the use of the 4×1 game as a suitable testbed for DAI are both questioned and found to be somewhat lacking. But the thesis also offers alternatives to overcome the above limitations.

In the application involving splines, the research has provided an automatic way to determine the degree of the splines and the number of knots. In the DAI application, it has suggested not only an interesting and novel way to model and test ideas that are of interest to DAI researchers but also as a result of the explorations conducted we have been able to suggest a class of games that has a good chance of exercising precisely the dimensions that are of interest to DAI.

This is in my opinion an important point to make. The 4×1 pursuit game has remained as the standard canonical test problem in DAI for quite a long time probably because of its compelling simplicity and the ease by which results can be compared. This elegance and simplicity may have obscured the actual goals for which it was conceived in the first place, namely, to exercise interesting high-level ideas of coordination, cooperation and communication that may be used elsewhere. For example, recently Korf [45] explored a class of pursuit problems in which only one prey was to be surrounded. This included the 4×1 game although the grid size he considered was larger than ours (100×100).

He arrived at conclusions similar to ours, namely that a 4×1 game can be solved without any explicit cooperation. But probably as a result of his experiments he then views coordination in general as an emergent property of the interaction of simple greedy agents subject to environmental constraints. This may be a premature and misleading message and exemplifies the importance of supporting one's claim with careful experiments. In fact, our research has revealed that to achieve reasonable performance levels in more complicated games the agents' architecture should include more than low-level components. It is certainly true that many problems in the real world look more complex and require more high-level decisions from the problem-solvers than the 4×1 game does. It was with this idea in mind that we set out to explore the game space and to suggest games that should work well for the purposes for which they were designed in the first place. It is also true that ultimately it will be the community of users who will decide on the appropriateness of a particular class of problems to be used as good testbeds. But it is safe to believe that one should not rule out pursuit problems other than 4×1 from consideration as good testbeds for DAI - provided the correct games are considered - without trying to be systematic in investigating them.

The experiments conducted in this work suggest quite strongly that in the class of $(M+4) \times M$ games the simplest ones that have enough complexity to serve as useful supports for tests and evaluations of DAI architectures are those with $M \in [5,7]$. By focusing attention on this class of games, we have also identified some basic dimensions in agents' architecture that are potentially valid for many different DAI applications. Within this activity, we believe that the issues introduced through the concept of boredom are the most novel. In addition, we have also found that the concepts of "listening rate", thresholds and attenuation of communication are important ideas to be taken into consideration by DAI designers. With regard to communication, our experiments appear to indicate that the performance of the collective of problem-solvers is less sensitive to changes in the α parameter that controls the attenuation profiles than the value of the latency-time variable. We do not yet have a language or the tools to make finer distinction, but it does seem that one may get much more mileage by experimenting with the idea of latency times rather than focusing too much attention on the rather more obvious (at least to a physicist or a communications engineer) issue of the attenuation profiles. This is certainly a topic worth further research.

9.2 Directions of future research

In this section we shall comment briefly on possible future directions related to the two main applications, i.e. first of all the spline and then the DAI, and finally we shall make some more general remarks.

In the application involving splines we have not tried to find the minimum number of knots, or their optimal position. Instead, we have followed Dierckx [17] in generating an arbitrary distribution of the knots which concentrates them on areas with relatively high densities of data points. In principle, there is no reason why this particular distribution will be the optimum for different data sets, and a more challenging problem is to evolve the distribution of knots using a GA. This might require the use of a non-fixed representation for the chromosomes, and accordingly, the introduction of suitable crossover operators. The placement of knots is an active area among researchers who use splines, and I believe this is an area where GAs can play an important role.

With regard to the Pursuit Problem several interesting possibilities suggest themselves. For example, we have found that the concept of "superagents" (assemblies of blue or pursuing agents that have reached positions relative to each other, e.g. while docked against red agents, that it is in their best interests to maintain), is worth further research. (Superagents are not mentioned in the DAI chapters because we have not had enough time to investigate them seriously). It is desirable to work towards concepts that have implications similar to those that arise from the ideas of colonies of cells or animals in sociobiology.

We would also like to explore the extent to which the idea of establishing ranks (and the ensuing dominance relations) among agents can be effective in further improving the performance of their collectives. Another interesting area to explore is how the problem-solvers can recognise that they have achieved positions that are good enough that (from their point of view) the game ought to stop.

Still further dimensions or concepts deserve examination, e.g. non-dimensional quantities associated with the pursuit games explored here. One might, for example, be interested in exploring games where the density of agents in a grid is held constant so that different methods could be compared independently of mesh size and prey number but purely by means of ratios. Since the problems considered here are computationally very demanding, it would be desirable to assess the population of proposed solutions in a parallel way (as opposed to the current implementation, which works sequentially). Once that is accomplished, more in-depth knowledge of the structure of the problem (and related solutions) could be gained due to the faster way of assessing different performance parameters or design choices.

One final approach that deserves a mention is to consider the parametrisation of the prey's actions and use the GA to search for the combination of such parameter(s) with others discussed above that results in a performance that leaves enough room for improvement (from the perspective of the "good DAI testbed" reasoning in chapter 6) through the addition of suitable high-level components. Finally, it is reasonable to hope that the framework provided here can be used or abstracted to other domains for which the construction of suitable experiments or designs is an important issue.

REFERENCES

[1] M. R. Adler, A. B. Davis, R. Weihmayer, and R. W. Worrest. Conflict-resolution Strategies for Nonhierarchical Distributed Agents. In L. Gasser and M. N. Huhns, editors, *Distributed Artificial Intelligence*, volume 2, pages 139–161. Pitman, London, 1989.

[2] G. Agha. *Actors.* The MIT Press, Cambridge, Massachusetts, 1986.

[3] W. Ross Ashby. *An Introduction to Cybernetics.* Chapman & Hall, London, 1957.

[4] N. M. Avouris and L. Gasser. Introduction. In N. M. Avouris and L. Gasser, editors, *Distributed Artificial Intelligence: Theory and Praxis*, pages 1–7. Kluwer Academic Publishers, Dordrecht, 1992.

[5] T. Bäck. The interaction of mutation rate, selection, and self-adaptation within a Genetic Algorithm. In R. Männer and B. Manderick, editors, *Proceedings of the Second Conference on Parallel Problem Solving from Nature*, pages 85–94, Brussels, Belgium, September 1992.

[6] D. L. Battle and M. D. Vose. Isomorphisms of Genetic Algorithms. In G. J. E. Rawlins, editor, *Foundations of Genetic Algorithms*, San Mateo, CA, 1991. Morgan Kaufmann.

[7] M. Benda, V. Jagannathan, and R. Dodhiawala. On optimal cooperation of knowledge sources. Technical Report BCS-G2010-28, Boeing AI Center, Boeing Computer Services, Bellevue, Washington, August 1986. Cited in L. Gasser, N. F. Rouquette, R. W. Hill, and J. Lieb. Representing and Using Organizational Knowledge in Distributed AI Systems. In L. Gasser and M. N. Huhns, editors, *Distributed Artificial Intelligence*, volume 2, pages 55-78, Pitman, London, 1989.

[8] A. D. Bethke. *Genetic Algorithms as function optimizers.* PhD thesis, University of Michigan, 1981. Dissertation Abstracts International, 41 (9) ,3503B (University Microfilms No. 8106101).

[9] N. L. Biggs and A. T. White. *Permutation Groups and Combinatorial Structures.* Cambridge University Press, Cambridge, 1979.

[10] G. Birkhoff and Saunders MacLane. *A Survey of Modern Algebra.* The MacMillan Company, New York, 1953.

[11] A. H. Bond and L. Gasser, editors. *Readings in Distributed Artificial Intelligence.* Morgan Kaufmann, San Mateo, CA, 1988.

[12] S. H. Brooks. A discussion of random methods for seeking maxima. *Operations Research*, 6(2):244–251, March 1958.

[13] J.J. Cannon. An Introduction to the Group Theory Language Cayley. In M.D. Atkinson, editor, *Computational Group Theory*, pages 145–183. Academic Press, London, 1984.

[14] P. Craven and G. Wahba. Smoothing noisy data with spline functions. *Numerische Mathematik*, 31:377–403, 1979.

[15] J. F. Crow. *Basic Concepts In Population, Quantitative, and Evolutionary Genetics.* W. H. Freeman and Company, New York, 1986.

[16] Y. Davidor. *Genetic Algorithms and Robotics: A Heuristic Strategy for Optimization.* World Scientific Publishing Co. (Pte.) Ltd., Singapore, 1991.

[17] P. Dierckx. An algorithm for smoothing, differentiation and integration of experimental data using spline functions. *Journal of Computational and Applied Mathematics*, 1:165–184, 1975.

[18] E. H. Durfee and V. R. Lesser. Negotiating task decomposition and allocation using partial global planning. In L. Gasser and M. N. Huhns, editors, *Distributed Artificial Intelligence*, volume 2, pages 229–243. Pitman, London, 1989.

[19] E. H. Durfee, V. R. Lesser, and D. D. Corkill. Coherent Cooperation among communicating Problem Solvers. In A. H. Bond and L. Gasser, editors, *Readings in Distributed Artificial Intelligence*, pages 268–284, San Mateo, CA, 1988. Morgan Kaufmann.

[20] C. R. Edwards. The application of the Rademacher-Walsh transform to boolean function classification and threshold synthesis. *IEEE Transactions on Computers*, C-24(1), January 1975.

[21] S. Forrest and M. Mitchell. What makes a problem hard for a Genetic Algorithm? Some anomalous results and their explanation. *Machine Learning*, 1993. To appear.

[22] L. Gasser. An Overview of DAI. In N. M. Avouris and L. Gasser, editors, *Distributed Artificial Intelligence: Theory and Praxis*, pages 9–30. Kluwer Academic Publishers, Dordrecht, 1992.

[23] L. Gasser and M. N. Huhns. Representing and using Organizational Knowledge in Distributed AI Systems. In L. Gasser and M. N. Huhns, editors, *Distributed Artificial Intelligence*, volume 2, pages 55–78. Pitman, London, 1989.

[24] L. Gasser and M. N. Huhns. Themes in Distributed Artificial Intelligence Research. In L. Gasser and M. N. Huhns, editors, *Distributed Artificial Intelligence*, volume 2, pages vii–xv. Pitman, London, 1989.

[25] Michael R. Genesereth, Matthew L. Ginsberg, and Jeffrey S. Rosenschein. Cooperation without communication. In *Proceedings of The National Conference on Artificial Intelligence*, pages 51–57, Philadelphia, Pennsylvania, August 1986.

[26] D. E. Goldberg. Genetic Algorithms and Walsh functions: Part i, a gentle introduction. *Complex Systems*, 3:129–152, 1989.

[27] D. E. Goldberg. Genetic Algorithms and Walsh functions: Part ii, deception and its analysis. *Complex Systems*, 3:153–171, 1989.

[28] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison Wesley, Reading, Massachussets, 1989.

[29] D. E. Goldberg. Construction of high-order deceptive functions from low-order Walsh coefficients. Technical Report IlliGAL Report No. 90002, University of Illinois at Urbana-Champaign, 1990.

[30] J. J. Grefenstette. Deception considered harmful. In D. Whitley, editor, *Foundations of Genetic Algorithms 2*, Vail, Colorado, July 1992.

[31] H. F. Harmuth. *Transmission of Information by Orthogonal Functions.* Springer-Verlag, Berlin, 2 edition, 1972.

[32] B. Hayes-Roth. A Blackboard Architecture for Control. *Artificial Intelligence Journal*, 26:251–321, 1985.

[33] B. Hayes-Roth, M. Hewett, R. Washington, R. Hewett, and A. Seiver. Distributing Intelligence within an Individual. In L. Gasser and M. N. Huhns, editors, *Distributed Artificial Intelligence*, volume 2, pages 385–412. Pitman, London, 1989.

[34] I. N. Herstein. *Topics in Algebra.* John Wiley & Sons, New York, 2 edition, 1975.

[35] C. Hewitt and P. de Jong. Analyzing the roles of descriptions and actions in open systems. In *Proceedings of the AAAI83*, pages 162–167, Los Altos, CA, 1983. Morgan Kaufmann.

[36] J. Hofbauer and K. Sigmund. *The Theory of Evolution and Dynamical Systems: Mathematical Aspects of Selection.* Cambridge University Press, Cambridge, 1988.

[37] J. H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, Michigan, 1975.

[38] R. Hooke and T. A. Jeeves. Comments on Brook's discussion of random methods. *Operations Research*, 6(6):881–882, November 1958.

[39] M. N. Huhns, editor. *Distributed Artificial Intelligence.* Pitman, London, 1987.

[40] M. N. Huhns, U. Mukhopadhyay, L. M. Stephens, and R. D. Bonnell. DAI for document retrieval: The MINDS Project. In M. N. Huhns, editor, *Distributed Artificial Intelligence*, pages 249–283. Pitman, London, 1987.

[41] K. A. De Jong. Are Genetic Algorithms function optimizers? In R. Männer and B. Manderick, editors, *Proceedings of the Second Conference on Parallel Problem Solving from Nature*, pages 3–13, Brussels, Belgium, September 1992.

[42] S. Karlin and J. McGregor. Towards a theory of the evolution of Modifier Genes. *Theoretical Population Biology*, t:59–103, 1974.

[43] M. G. Karpovsky. *Finite orthogonal series in the design of digital devices.* Halsted Press, John Wiley & Sons, New York, 1976.

[44] O. Kempthorne. *An Introduction to Genetic Statistics.* Iowa State University Press, Ames, 1969.

[45] R. E. Korf. A simple solution to pursuit games. In *Proceedings of the 11th International Workshop on Distributed Artificial Intelligence*, pages 183–194, Glen Arbor, Michigan, February 1992.

[46] R. J. Lechner. Harmonic analysis of switching functions. In A. Mukhopadhyay, editor, *Recent Developments in Switching Theory*, pages 121–228. Academic Press, New York, 1971.

[47] W. Ledermann. *Introduction to Group Theory.* Longman Group Limited, London, 1976.

[48] Charles Leedham-Green. Personal communication, 1991.

[49] V. R. Lesser and L. D. Erman. Distributed Interpretation: A model and experiment. In A. H. Bond and L. Gasser, editors, *Readings in Distributed Artificial Intelligence*, pages 120–139, San Mateo, CA, 1988. Morgan Kaufmann.

[50] R. Levy and J. S. Rosenschein. A game theoretic approach to Distributed Artificial Intelligence and the Pursuit Problem. In Y. Demazeau and E. Werner, editors, *Decentralized Artificial Intelligence III*, pages 129–146. Elsevier Science Publishers B.V./North-Holland, Amsterdam, 1992.

[51] G. E. Liepins and M. D. Vose. Representational issues in genetic optimization. *Journal of Experimental and Theoretical Artificial Intelligence*, 2(2):4–30, 1990.

[52] G. E. Liepins and M. D. Vose. Polynomials, Basis Sets, and Deceptiveness in Genetic Algorithms. *Complex Systems*, 5:45–61, 1991.

[53] G. E. Liepins and M. D. Vose. Characterizing crossover in Genetic Algorithms. *Annals of Mathematics and Artificial Intelligence*, 5:27–34, 1992.

[54] M. Manela. Automorphisms of Walsh-Hadamard matrices. Technical Report RN/91/67, Department of Computer Science, University College London, 1991.

[55] M. Manela and J. A. Campbell. Harmonic analysis, epistasis and genetic algorithms. In R. Männer and B. Manderick, editors, *Proceedings of the Second Conference on Parallel Problem Solving from Nature*, pages 57–64, Brussels, Belgium, September 1992.

[56] M. Manela, N. Thornhill, and J. A. Campbell. Fitting spline functions to noisy data using a Genetic Algorithm. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 549–553, San Mateo, CA, July 1993. Morgan Kaufmann.

[57] M. Maqusi. Sampling representation of sequency-band-limited nonstationary random processes. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-28(2):249–251, April 1980.

[58] A. J. Mason. Partition coefficients, static deception and deceptive problems for non-binary alphabets. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 210–214, San Mateo, CA, July 1991. Morgan Kaufmann.

[59] K. Mathias and D. Whitley. Remapping hyperspace during genetic search: Canonical Delta Folding. In D. Whitley, editor, *Foundations of Genetic Algorithms 2*, Vail, Colorado, July 1992.

[60] D. M. Miller and J. C. Muzio. A spectral characterization of the self-dualized classification of boolean functions. *International Journal of Electronics*, 61(1):65–72, 1986.

[61] G.A. Miller, H.F. Blichfeldt, and L.E. Dickson. *Theory and applications of finite groups.* G. E. Stechert & Co., New York, 1938.

[62] H. Penny Nii, Nelleke Aiello, and James Rice. Experiments on Cage and Poligon: Measuring the Performance of Parallel Blackboard Systems. In L. Gasser and M. N. Huhns, editors, *Distributed Artificial Intelligence*, volume 2, pages 319–383. Pitman, London, 1989.

[63] A. Nix and M. D. Vose. Modeling Genetic Algorithms with Markov Chains. *Annals of Mathematics and Artificial Intelligence*, 5:74–88, 1992.

[64] A. V. Oppenheim and R. W. Schafer. *Digital Signal Processing.* Prentice-Hall, Englewood Cliffs, N.J., 1975.

[65] A. Papoulis. *Probability, random variables and stochastic processes.* McGraw-Hill, New York, 1965.

[66] H. V. D. Parunak. Manufacturing experience with the Contract Net. In M. N. Huhns, editor, *Distributed Artificial Intelligence*, pages 285–310. Pitman, London, 1987.

[67] J. Pearl. Application of Walsh Transform to Statistical Analysis. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-1(2):111–119, April 1971.

[68] T. Poggio and F. Girosi. Networks for Approximation and Learning. *Proceedings of the IEEE*, 78(9), September 1990.

[69] V. Popov. Invariant Description of Linear Time-Invariant Controlable Systems. *Siam J. Control*, 10:252–264, 1972.

[70] N. J. Radcliffe. *Genetic Neural Networks on MIMD Computers.* PhD thesis, University of Edinburgh, 1990. Compressed edition.

[71] N. J. Radcliffe. Genetic Set Recombination. In D. Whitley, editor, *Foundations of Genetic Algorithms 2*, Vail, Colorado, July 1992.

[72] C. H. Reinsch. Smoothing by spline functions. *Numer. Math*, 10:177–183, 1967.

[73] R. Rieger, A. Michaelis, and M. M. Green. *Glossary of genetics : classical and molecular.* Springer-Verlag, Berlin, 1986.

[74] M. L. Roberts. Personal communication, 1993.

[75] D. Rogers. A hybrid of Friedman's Multivariate Adaptive Regression Splines (MARS) algorithm with Holland's Genetic Algorithm. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 384–391, San Mateo, CA, July 1991. Morgan Kaufmann.

[76] J. S. Rosenschein and J. S. Breese. Communication-free interactions among rational agents: A probabilistic approach. In L. Gasser and M. N. Huhns, editors, *Distributed Artificial Intelligence*, volume 2, pages 99–118. Pitman, London, 1989.

[77] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):210–229, 1959.

[78] A. Sathi and M. S. Fox. Constraint-directed negotiation of resource reallocations. In L. Gasser and M. N. Huhns, editors, *Distributed Artificial Intelligence*, volume 2, pages 163–193. Pitman, London, 1989.

[79] J. David Schaffer, R. A. Caruana, L. J. Eshelman, and R. Das. A study of control parameters affecting online performance of Genetic Algorithms for function optimization. In J. D. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 51–60, San Mateo, CA, 1989. Morgan Kauffmann.

[80] J. David Schaffer and L. J. Eshelman. Designing multiplierless filters using Genetic Algorithms. In Stephanie Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 439–444, San Mateo, CA, July 1993. Morgan Kaufmann.

[81] N. N. Schraudolph and R. K. Belew. Dynamic Parameter Encoding for Genetic Algorithms. *Machine Learning*, 9:9–21, 1992.

[82] N. N. Schraudolph and J. J. Grefenstette. A User's Guide to GAucsd 1.4. Technical Report CS92-249, University of California, San Diego,CA, 1992.

[83] C. G. Shaefer. The ARGOT Strategy: Adaptive Representation Genetic Optimizer Technique. In *Proceedings of the Second International Conference on Genetic Algorithms*, pages 50–58, Hillsdale, 1987. Lawrence Erlbaum Associates.

[84] R. G. Smith. The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. In A. H. Bond and L. Gasser, editors, *Readings in Distributed Artificial Intelligence*, pages 357–366, San Mateo, CA, 1988. Morgan Kaufmann.

[85] R. Steeb, S. Cammarata, F. Hayes-Roth, P. Thorndyke, and R. Wesson. Distributed Intelligence for Air Fleet Control. In A. H. Bond and L. Gasser, editors, *Readings in Distributed Artificial Intelligence*, pages 90–101, San Mateo, CA, 1988. Morgan Kaufmann.

[86] L. M. Stephens and M. B. Merx. The effect of agent control strategy on the performance of a DAI Pursuit Problem. In *Proceedings of the 10th. International Workshop on Distributed Artificial Intelligence*, Bandera, Texas, October 1990.

[87] K. Sycara. Multiagent compromise via Negotiation. In L. Gasser and M. N. Huhns, editors, *Distributed Artificial Intelligence*, volume 2, pages 119–137. Pitman, London, 1989.

[88] A. Terras. *Harmonic Analysis on Symmetric Spaces and Applications*, volume 1. Springer-Verlag, New York, 1988.

[89] N. Thornhill, M. Manela, K.M. Stone, and J.A. Campbell. Two methods of selecting smoothing splines applied to fermentation process data. *Journal of the American Institute of Chemical Engineers*, 1993. To appear.

[90] H. L. V. Trees. *Detection, Estimation, and Modulation Theory: Part I Detection, Estimation, and Linear Modulation Theory.* John Wiley and Sons, New York, 1968.

[91] M. D. Vose. Modeling Simple Genetic Algorithms. In D. Whitley, editor, *Foundations of Genetic Algorithms 2*, Vail, Colorado, July 1992.

[92] M. D. Vose and G. E. Liepins. Punctuated Equilibria in genetic search. *Complex Systems*, 5:31–44, 1991.

[93] M. D. Vose and G. E. Liepins. Schema disruption. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 237–242, San Mateo, CA, July 1991. Morgan Kaufmann.

[94] E. Werner. Cooperative agents: A unified theory of communication and social structure. In L. Gasser and M. N. Huhns, editors, *Distributed Artificial Intelligence*, volume 2, pages 3–36. Pitman, London, 1989.

[95] D.J. Wilde. *Optimum seeking methods.* Prentice-Hall, Englewood Cliffs, N.J., 1964.

[96] E. O. Wilson. *Sociobiology.* Belknap Press of Harvard University Press, Cambridge, Mass., 1980. Abridged ed.

Appendix A
Fourier Series Analysis

This appendix extends the results discussed in chapter 3 using the unified framework provided by abstract harmonic analysis, so that questions of interest in the GA literature can be discussed in a more general way. We also extend results from deceptive analysis for the binary case [51] to non-binary alphabets to show that there exist functions that are fully deceptive in all such cases.

A.1 Introduction

Most studies of the behaviour of genetic algorithms (GAs) refer to schemata that are expressed in a binary alphabet. In some instances where the efficiency of GAs is unsatisfactory, there is a view that the problem should be overcome or reduced if an alphabet with more characters is used. The debate on which alphabet is most appropriate, i.e. a binary one or an alphabet of higher cardinality, is still ongoing (see e.g [58, 75]) and any clarification with respect to this issue should therefore be appropriate.

In the binary case Bethke [8] introduced the idea of exploring which conditions should make a function easy or difficult for genetic search by means of Walsh-function analysis. His approach was extended by others, most notably by Goldberg (see e.g. [26, 27, 29]). Central to this analysis are the important notions of *schema* and the *Schema Theorem* [37, 28]. A schema [28] is a similarity template describing a subset of strings with similarities at certain positions and can be viewed as defining hyperplanes in the search space [28]. A GA is thought to work by directing the search towards schemas containing highly fit regions of the search space [21]. In his thesis, Bethke [8] developed the *Walsh-Schema* transform, in which schema average fitness values were expressed in terms of Walsh coefficients and then used this transform to characterise functions as easy or hard for the GA to optimise.

The idea that a similar analysis could be extended to the non-binary case may have occurred to several authors (for example, through the obvious similarity of Walsh and Fourier functions [28, 21]). However, to our knowledge the only attempt, so far, to pursue an analysis when non-binary alphabets are considered was made by Mason [58] along completely different lines.

Most results and definitions used in this appendix are extensions of concepts already established in the GA literature for the binary case (and much earlier in the non-GA literature), and we claim no originality for them. However, we hope that the methods examined here should contribute to decisions about which alphabet to use in much the same way they are used in other signal-processing applications.

In particular, we make use of a general Fourier transform as an extension to the Walsh functions employed by Bethke [8] in the analysis of performance of schemata for binary strings, and focus on the examination of deceptive behaviour, which Mason [58] has considered recently for the non-binary case. We show explicitly how Bethke's results can be embedded in a more general framework, namely of Harmonic Analysis, where the concepts discussed above can be understood by reference to certain subspaces and their respective null spaces. We also extend results from deceptive analysis for the binary case [51] to non-binary alphabets to show that there exist functions that are fully deceptive in all such cases.

In what follows we shall consider functions from a commutative group to the field of real numbers. We shall represent a given function as an orthogonal-series expansion of a complete orthogonal base of exponential functions. This is analogous to the use of the discrete Laplace transform, a fundamental tool of the theory of sampled-data control systems. In effect, the underlying reason why that approach should pay off is that our chosen basis forms a multiplicative group isomorphic to the additive group of integers (see [43] for further details). We regard the material in this appendix as a framework for extending and improving the present somewhat fragmented understanding of the application domains referred to above.

A.2 Notation and representational issues

Let p be an integer such that $p > 2$. Let Z_p be the additive group of integers modulo p. Let ${Z_p}^n$ be the n-fold cartesian product of Z_p, alternatively the ring of residue classes mod p (since p is not necessarily a prime) or a module over Z_p. This corresponds to the domain of our search space. The range of the function is usually considered to be the set of real positive numbers. An element $x \in {Z_p}^n$ will be represented by the row vector $(x_1, x_2, \ldots, x_n)$ and it is indexed in the natural ordering defined by the correspondence between a number and its radix-p expansion using base-p n-tuples. x^T will denote the transpose of x, while $< x, y >$ will denote the inner product in Z_p of x and y given by

$$< x, y >= xy^T = \sum_{i=1}^{n} x_i y_i.$$

Usually, addition and multiplications are done in arithmetic modulo p below. However, if any such operations refer to real arithmetic, this will be clear from the context.

Let f be a real-valued function. The abstract Fourier transform of f provides a unique representation for f. Lechner [46] presents a typical treatment of this issue. Given a positive integer p, not necessarily a prime, we define $Q_\omega(x)$ and $\bar{Q}_\omega(x)$ by

$$\begin{aligned} Q_\omega(x) &= \exp(2\pi i < \omega, x > /p) \\ \bar{Q}_\omega(x) &= \exp(-2\pi i < \omega, x > /p) \end{aligned} \tag{1}$$

The abstract Fourier transform pair is given by

$$\begin{aligned} f^*(\omega) &= p^{-n} \textstyle\sum_x f(x) Q_\omega(x) \\ f(x) &= \textstyle\sum_\omega f^*(\omega) \bar{Q}_\omega(x) \end{aligned} \tag{2}$$

The direct analogy between this transform pair and the Walsh-transform pair is apparent if one uses $p = 2$ for the characteristic of the finite field over which the domain and the transform space are defined. As shown in [46], in this case the exponential functions becomes

$$\exp(\pi i < \omega, x >) = (-1)^{<\omega, x>}$$

i.e., the Fourier transform collapses to the Walsh transform.

To make the notation easier to read, we shall use the symbol X to denote $Z_p{}^n$. Note that X is an additive group with componentwise addition modulo p, denoted by $\oplus$. Any subspace of X that is closed under addition is therefore a subgroup. If V is one such subspace then a *translation* (or coset) of V is denoted by $\{V + c\}$, $c \in X$. $V + c$ represents the set $\{x \oplus c | x \in V\}$ (where $x \oplus c$ denotes the componentwise addition modulo p between x and c).

Let $H = Z_p \cup \{*\}$, where the symbol $*$ stands for the usual interpretation of "don't care"; i.e. it can be matched by any $x \in Z_p$. Let H^n be the n-fold cartesian product of H. We shall be interested in sets of the form $V = \{x \in X : x_i \in Z_p$ if $h_i = *$, otherwise $x_i = 0\ \}$. It is easy to see that V is a subgroup. We can identify each such set with the binary vector β whose i-th entry is 1 when the corresponding entry in the related set is $*$, and 0 otherwise. Hence we shall refer to V as V_β.

A *schema* is a translation of V_β. A schema is conveniently represented as a n-tuple $h = (h_1, \dots, h_n)$ where $h_i \in H$. If $h_i = *$ it is called a *free variable*, otherwise it is called *fixed* (or constrained). The *dimension* $dim(h)$ of a schema h is the number of free variables it contains. The *order* $o(h)$ of a schema h is defined as the number of fixed positions in it. Therefore, $o(h) = n - dim(h)$ while the cardinality of a schema is given by $p^{dim(h)}$. In an analogous way we define the order of x, denoted by $o(x)$, as the number of elements in x that are not equal to 0. The *defining length* of a schema h, denoted by $d_l(h)$, is the distance between the first and last free positions in it. Accordingly, the defining length of x, denoted by $d_l(x)$, is the distance between the first and last elements in x that are not equal to 0. The *weight* of x denoted by $weight(x)$ is equal to the sum (over the reals) of all elements of X, i.e., $weight(x) = \sum_{i=1}^{n} x_i$. Note that there are many vectors c which translate V into the same coset $V + c$. However, only one of these vectors has zero values for each of the free variables in V. From now on we shall use the symbol c to denote this translation vector, which is the unique vector of minimum weight in the coset $(V + c)$[1]. Finally, the set of all vectors in X that are orthogonal to all elements of V is the *nullspace* of V and is denoted by V'. Note that since the subspace V_β can be generated by unit vectors its corresponding nullspace is given by $V_{\bar{\beta}}$, where $\bar{\beta}$ is the bitwise logical complement of the binary vector β.

A.3 Poisson Summation Theorem and schema average fitness

In this section we prove the Poisson Summation Theorem for cosets of submodules of $(Z/pZ)^n$.

[1]c can also be described as the unique element of the coset that lies in $V_{\bar{\beta}}$.

Let (Z/pZ) be the set of integers modulo p, where p is not necessarily a prime. Let $(Z/pZ)^n$ be the n-fold cartesian product of Z/pZ, i.e., a module over Z/pZ. Let V be a submodule denoted by $V \leq (Z/pZ)^n$ and V' its nullspace. Define $\delta(\omega)$ to be the characteristic function of V' i.e., it is 1 if $\omega \in V'$ and 0 otherwise. We shall also use $\delta_{\omega c}$ to represent a function having the value 0 except for $\omega = c$ when it is 1.

Lemma A.1 *Let $\omega \in (Z/pZ)^n$ and $g^*(\omega) = \sum_{x \in V} \exp(2\pi i < \omega, x > /p)$. Then $g^*(\omega) = |V|\delta(\omega)$, i.e., it is 0 except on V', where it is equal to the cardinality of V.*

Proof of Lemma A.1 *Consider*

$$\begin{aligned} \phi_\omega : \ & V \rightarrow Z/pZ \\ & z \rightarrow \phi_\omega(z) = < \omega, z > \end{aligned}$$

It is easy to see that ϕ_ω is a Z/pZ homomorphism with kernel $\mathcal{K}\phi_\omega$. The image of ϕ_ω denoted by $\mathcal{I}\phi_\omega$ is a submodule of Z/pZ [34]. If $\mathcal{I}\phi_\omega = 0$ then $\omega \in V'$ and $g^*(\omega) = \sum_{s \in V} 1 = |V|$. If $0 \neq \mathcal{I}\phi_\omega \leq Z/pZ$ then $\mathcal{I}\phi_\omega$ is an *ideal* [34]; $\mathcal{I}\phi_\omega = aZ/pZ$ (for some $a|p, p = ax$). It follows that

$$\begin{aligned} g^*(\omega) &= \textstyle\sum_{u \in \mathcal{I}\phi_\omega} \sum_{z \in V, \phi_\omega(x) = u} \exp(2\pi i u/p) = |\mathcal{K}\phi_\omega| \sum_{u \in \mathcal{I}\phi_\omega} \exp(2\pi i u/p) \\ &= |\mathcal{K}\phi_\omega| \textstyle\sum_{t=0}^{x-1} \exp(2\pi i a t/p) = |\mathcal{K}\phi_\omega| \sum_{t=0}^{x-1} \exp(2\pi i t/x) \\ &= |\mathcal{K}\phi_\omega| \textstyle\sum_{t=0}^{x-1} (\exp(2\pi i x))^t = 0. \end{aligned}$$

The last result follows from the fact that if ω is an x-th root of unity then $\sum_{t=0}^{x-1} \omega^t = 0$ since $(1 - \omega) \sum_{t=0}^{x-1} \omega^t = 1 - \omega^x = 0$ (q.e.d.) . □

Lemma A.2 *Let $V + c$ be any coset of any submodule V of $(Z/pZ)^n$ whose cardinality is p^k and with $c \in V'$. Let $v_c(x)$ be the characteristic function of $V + c$. Then $v_c^*(\omega) = p^{k-n}\delta(\omega) \exp(2\pi i < \omega_1, c > /p)$.*

Proof of Lemma A.2 *Let $\omega = \omega_1 \oplus \omega_2, x = y \oplus z$ with $\omega_1, y \in V'$ and $\omega_2, z \in V$. Then $v_c(x) = \delta_{yc}$ i.e., it is 0 except when $y = c$ when it is 1. It follows that*

$$v^*(\omega) = p^{-n} \sum_z \sum_y f(z \oplus y) \exp(2\pi i < \omega_1 \oplus \omega_2, y \oplus z > /p).$$

But

$$< \omega_1 \oplus \omega_2, y \oplus z > = < \omega_1, y > \oplus < \omega_2, z > .$$

Thus $v^*(\omega) = p^{-n} \sum_z \exp(2\pi i < \omega_2, z > /p) \sum_y \exp(2\pi i < \omega_1, y > /p)\delta_{yc}$.

By lemma A.1 the first term is $p^k \delta_{\omega_2 0}$ while the second one is equal to $\exp(2\pi i < \omega_1, c > /p)$. Thus $v_c^*(\omega) = p^{k-n}\delta_{\omega_2 0} \exp(2\pi i < \omega_1, c > /p)$ (q.e.d.) .□

Theorem A.1 (Convolution Theorem). *Let $(f * g)(c) = \sum_x f(x \oplus c)g(x)$ denote the* convolution sum *between f and g, where $\oplus$ denotes addition modulo p. Then $(f * g)^*(\omega) = p^n(f^*g^*)(\omega)$, i.e, the Fourier transform of the convolution sum is equal (with proper normalization) to the product of the Fourier transforms.*

Proof: By definition the Fourier transform of f is given by
$f^*(\omega) = p^{-n} \sum_x f(x) \exp(2\pi i < \omega, x > /p)$. Therefore,

$$\begin{aligned}(f^*g^*)(\omega) &= p^{-2n} \textstyle\sum_x f(x) \exp(2\pi i < \omega, x > /p) \sum_y g(y) \exp(2\pi i < \omega, y > /p) \\ &= p^{-2n} \textstyle\sum_{x,y} \exp(2\pi i < \omega, x \oplus y > /p) f(x) g(y)\end{aligned}$$

Let $y = x \oplus c$. Summing first over x (with c fixed) and then over c the above expression becomes

$$p^{-2n} \sum_c \exp(2\pi i < \omega, c > /p) \sum_x f(x)g(x \oplus c) = p^{-n}(f * g)^*(\omega).\Box$$

The Poisson Summation Theorem states that the average of a function's value over a subspace of X may be computed by summing its Fourier transform over a suitable quotient group.

Theorem A.2 (Poisson Summation Theorem (generalisation)). *Let V be a subspace of X with cardinality p^k and nullspace V'. Let $V+c$ be a translation of V with $c \in V'$. If f is a real- or complex-valued function on X, then*

$$\sum_{x \in V} f(x \oplus c) = p^k \sum_{\omega \in V'} f^*(\omega) \bar{Q}_\omega(c) \tag{3}$$

Proof:

$$\sum_{x \in V+c} f(x) = \sum_{x \in V} f(x \oplus c) = \sum_{x \in (Z/pZ)^n} f(x \oplus c) v_c(x)$$

Note that the last term in the above equation is the convolution of f and v_c evaluated at c. The result follows by direct substitution of lemma A.2 and the Convolution Theorem (q.e.d.) . □

A.4 Deceptive problems

Liepins and Vose [51] have shown how to construct fully deceptive functions for binary alphabets on strings of arbitrary length. Mason [58] has given a construction of fully deceptive problems[2] of any given order for nonbinary codings which relies on partition coefficients. Although one can use the tools provided in the previous sections to address Mason's construction we prefer to extend the Liepins and Vose result in the following lemma.

Lemma A.3 *Let $p > 2$, $n > 1$, $x^* = (p-1, p-1, \ldots, p-1)$ and f be defined on X as follows:*

$$f(x) = \begin{cases} 1 - \frac{1+weight(x)}{pn} & \text{for } x \neq x^*, \\ 1 & \text{for } x = x^*. \end{cases}$$

Then f is fully deceptive.

Proof of Lemma A.3 *First note that $f(x) < 1$ for all $x \neq x^*$. Let h_1 and h_2 be competing schemata such that $0 \in h_1$. We divide the proof into two parts by considering first any schema h_2 that does not contain x^* and then any schema that contains x^*.*

If h_2 is a competing schema of h_1 not containing x^*, then we compare each member of h_1, say x, to its corresponding translation y in h_2 by vector c, i.e., $y = x \oplus c$. Since c has entries equal to 0 whenever the entries in the corresponding schemata are equal to $*$ it follows that $weight(y) = weight(x) + weight(c)$, and since by hypothesis $y \neq x^*$, $f(y)$ is given by

$$\begin{aligned} f(y) &= 1 - (1 + weight(y)/pn) = 1 - (\frac{1 + weight(x) + weight(c)}{pn}) \\ &= f(x) - \frac{weight(c)}{pn}. \end{aligned}$$

Therefore $f(y) < f(x)$ for each element $x \in h_1$ and its corresponding translation $y \in h_2$. Hence $f(h_1) > f(h_2)$.

[2]The extension of the definition of deceptiveness from the binary to the n-ary case is straightforward and we refer the reader to [58].

Next we consider the case when all fixed positions of h_2 have entries equal to $(p-1)$, in which case we know that $x^* \in h_2$. Since $f(x^*) = 1$, this will be the only case where a member of h_2 is greater than the corresponding member of h_1 and hence we have to examine whether the difference between them is enough to offset all other contributions, which we know by the above reasoning to be greater than zero. We prove in what follows, by an argument with general validity, that this cannot occur.

Consider for instance h_2 having $(n-l)$ fixed positions equal to $(p-1)$ and the other l positions being equal to $*$. There are p^l elements in h_1 and also in h_2. Thus there are $(p^l - 1)$ pairs of values of $x \in h_1$ and $y \in h_2$ where $f(x) > f(y)$ and only one pair (precisely when $y = x^*$) where $f(y) > f(x)$. For the first $(p^l - 1)$ pairs we have that

$$f(x) - f(y) = (1 - \frac{1 + weight(x)}{pn}) - (1 - \frac{1 + weight(y)}{pn}) = \frac{(p-1)(n-l)}{pn}$$

where in the last equation we have made use of the fact that x and y differ only on $(n-l)$ entries and in each of them the difference is $(p-1)$. When $y = x^*$ we have that

$$f(x) - f(y) = (1 - \frac{1 + weight(x)}{pn}) - 1 = -\frac{1 + l(p-1)}{pn} .$$

It follows that

$$f(h_1) - f(h_2) = \frac{[(n-l)(p-1)(p^l - 1)] - [1 + l(p-1)]}{p^{l+1}n} .$$

Rearranging the terms in the above expression we get

$$f(h_1) - f(h_2) = \frac{(p-1)[(n-l)(p^{(l-1)} - l)] - 1}{p^{l+1}n} .$$

Since by hypothesis $p > 2$ and $(n-l) >= 1$ the above expression is always greater than zero. This completes the proof. □

Following basically the same lines as above, one can prove that the function given in lemma A.3 has a further interesting property: if a schema dimension is not too small compared to n (of relative order $\log n$) then *any* schema not containing x^* has a higher fitness than a competing schema containing x^* [74].

In some circumstances it is of interest to represent the search space in which each coordinate axis has associated with it a different alphabet i.e., each coordinate axis has an associated alphabet of different cardinality. Even in these cases we can still extend lemma A.3 as follows. Let $i \in [1, n]$ where n is the length of the string ($n > 1$), p_i be any positive integer greater than 2, and Z_{p_i} be the additive group of integers modulo p_i. Let $\prod_{i=1}^{n} Z_{p_i}$ be the n-fold cartesian product of the Z_{p_i}.

Lemma A.4 *Let $n > 2$, $i \in [1, n]$, $p_i > 2$, $x^* = (p_1 - 1, p_2 - 1, \ldots, p_n - 1)$ and f be defined on X as follows:*

$$f(x) = \begin{cases} 1 - \frac{1 + weight(x)}{\sum_{i=1}^{n} p_i} & \textit{for } x \neq x^*, \\ 1 & \textit{for } x = x^*. \end{cases}$$

Then f is fully deceptive.

The proof follows the same lines as lemma A.3 and will therefore be omitted.

In the sections to follow, we shall restrict our analysis to the case where all coordinates are represented by the same alphabet whose cardinality is a prime number. In this situation X can be regarded as a vector space over the field $GF(p)$.

A.5 Domain encodings

As pointed out by Liepins and Vose [51], any function can be encoded in a way that makes genetic search easy by simply choosing a suitable permutation from the space of all possible permutations. Unfortunately, as they themselves remarked, this space is so large that the search for the good ones is intractable. Inspection of equation (3) offers one particular alternative, namely, to look for a class of permutations that induce permutations on the Fourier coefficients. The following theorem shows that linear invertible transformations provide such an alternative.

Theorem A.3 (Domain Encodings). *Let $g(x)$ be a known function from X into the reals. Let f be given by $f(x) = g(xA)$, where A is a linear invertible transformation. Then the Fourier transforms of f and g are related by $f^*(\omega) = g^*(\omega A^{-T})$, where A^{-T} is the transposed inverse of A.*

Proof: Let y be given by $y = xA$. Then $x = yA^{-1}$ and

$$\begin{aligned} f^*(\omega) &= (1/p^n)\textstyle\sum_x f(x)\exp(\frac{2\pi i<\omega,x>}{p}) = (1/p^n)\sum_x f(yA^{-1})\exp(2\pi i\omega x^T/p) \\ &= (1/p^n)\textstyle\sum_y g(y)\exp(2\pi i\omega A^{-T}y^T/p) = g^*(\omega A^{-T}).\square \end{aligned}$$

The above result shows that when A permutes X its transposed inverse permutes the Fourier coefficients. Consider next the case when A is a symmetric matrix i.e., $A = A^T$. If we consider the mapping $f(x) = g(xA^{-1})$, then it is immediate to see that $f^*(\omega) = g^*(\omega A)$. As a result, when A^{-1} permutes the original data, A permutes the Fourier coefficients. In other words, the action of the inverse transformation can be explained in the transform space in the same way as for the original transformation in the original space.

A.6 Possible extensions

In the previous sections we have described the conventional genetic algorithm as containing a combinatorial operator (crossover) acting on a set which has a group structure. The subgroups of this group are such that they possess an hierarchical structure and are invariant under the crossover operation. This explains why the above analysis (and in particular the Walsh Schema Analysis introduced by Bethke [8]) may be of theoretical interest in characterising classes of functions that are easy and difficult for genetic search. Whenever the underlying characteristic of the search space is of the same type as above, and the crossover operator preserves the subgroups involved, the theoretical techniques discussed above may be applied to try to characterise other application domains of GAs. The ideas expressed in this paragraph, although developed in an independent way, are consistent with those explored in [92, 53, 71].

As a trivial example, we mention the case of a finite Abelian group G. Any Abelian group is isomorphic to a direct product of cyclic groups, and any cyclic group is isomorphic to the additive group of integers modulo n where n is the order of the group. It follows from group theory (see [47] for instance) that any element $x \in G$ can be expressed uniquely as a product of elements from these subgroups. This provides a suitable definition for the crossover operator as a composition of projections of x

in the corresponding subgroups. In this way, the techniques studied above can be applied in a straightforward way. For further details the reader is referred to [47, 43].

A.7 Discussion

The Building Block Hypothesis [37] is a central issue for the performance of genetic algorithms. It is expressed via the Schema Theorem, and suggests that a function will be easy for genetic optimisation if short, low-order, and highly fit schemata are sampled, recombined, and resampled to form strings of potentially higher fitness [28]. It is therefore important to use this idea to help in characterising functions that are easy for genetic optimisation. One possible characterization, which can be obtained in terms of Fourier coefficients and by reference to equation (3), was first stated by Bethke [8] for the case $p = 2$. This is extended to $p > 2$ in the following lemmas.

Lemma A.5 *Let $j \in X$ and h be any schema. If the order of j exceeds the order of h, then $f^*(j)$ does not affect $f(h)$.*

Proof of Lemma A.5 *Recall that since h is a schema, it is a translation of a subspace given by V_β. Note that $o(h) = n - o(\beta) = o(\bar{\beta})$. Therefore, the hypothesis forces $o(j) > o(\bar{\beta})$, which implies that $j \notin V_{\bar{\beta}}$ since it cannot be spanned by the unit vectors of $V_{\bar{\beta}}$. Consequently $f^*(j)$ does not contribute to the sum in equation (3) (q.e.d.).*$\square$

Lemma A.6 *Let $j \in X$ and h be any schema. If the definition length of j exceeds the definition length of h, then $f(h)$ does not depend on $f^*(j)$.*

Proof of Lemma A.6 *Let $d_{\bar{l}}(j)$ be the distance between the first and last 0s in the string j. Note that when j is binary, $d_l(\bar{j}) = d_{\bar{l}}(j)$. Also, $d_l(h) = d_{\bar{l}}(\beta)$. It follows by the hypothesis that $d_l(j) > d_l(h) = d_{\bar{l}}(\beta) = d_l(\bar{\beta})$. In other words, j cannot be spanned by the unit vectors of $V_{\bar{\beta}}$, that is, $j \notin V_{\bar{\beta}}$. Therefore $f^*(j)$ does not contribute to $f(h)$ (q.e.d.).* $\square$

The above lemmas show that if the absolute values of the Fourier coefficients tend to 0 for increasing order and definition length then the Building Block Hypothesis is being respected and we expect the corresponding function to be easy for genetic optimisation.

A.8 Summary of result from the Appendix

We have extended the results reported in chapter 3 to non-binary alphabets. By showing explicitly how to calculate schema averages in non-binary alphabets we hope to present means for others to shed some light on the question of which alphabet should be considered for optimising some given particular class of functions. We have also extended results from binary alphabets to show that there exist fully deceptive problems for any such alphabet on strings of any length.

Appendix B
Agent's architecture

In this appendix we give a more detailed description about the agent's architecture whose main modules are presented in chapter 7.

B.1 Agent's description

In the implementation considered in this thesis the agent's architecture is constituted by the following basic features.

State	the agent's internal state
(X,Y)	agent's current position in the rectangular grid
(Xchoice, Ychoice)	agent's choice for its future position in the grid
Boredom Level	agent's level of boredom (which if above or below a given threshold forces it to change its internal state)
Communication Latency	associated with the time when an agent should listen for communication from other agents
Noise in Sectors[$SECTOR\_NUMBER$]	an array in which each entry contains the amount of communication in a given sector at each iteration
Prey Direction[$PREY\_NUMBER$][4]	a bidimensional array containing information on the most recent movements of each prey

Figure B.1: Basic description of blue agents

The agent may also maintain a table of the state of the positions surrounding the prey to discard from its future targets those that are already occupied by other blue agents. Alternatively, it may sense the state of those positions if it has the required sensing capability. In the thesis, only the first option was implemented.

The above parameters are used in conjunction with the following parameters[1] which are themselves selected by the GA.

Projection factor	a multiplicative factor by which a prey's current distance to an agent is multiplied and subsequently projected in the direction that the agent believes that the prey is heading
Direction threshold ($Th_{direction}$)	a value above which a given direction is considered valid for use with the predictive mode
α	auto-regressive parameter for memory model of prey's recent moves
Boredom threshold ($Th_{boredom}$)	a threshold value above which an agent changes its state to being "bored"
Boredom hysteresis	a percentage value that is used to establish the interval during which, once an agent has become bored, it will stay in that state before returning to its normal state
β	auto-regressive parameter for memory of boredom
Sector size	corresponds to the maximum resolution that an agent has such that it is able to distinguish between different sources of information
Squelch value	a value below which an agent regards the information coming from a sector as being noise
Attenuation profiles' power law	model by which messages are attenuated in the communication medium

Figure B.2: Parameters subjected to genetic search

The following sections describe how an agent's control structures were implemented and how it uses the above information. Before doing so, it is convenient to emphasise some restrictions that are imposed on the agent's capabilities.

[1]the interested reader is referred to chapter 7 and the sections that follow for more details on the specific implementation used here.

B.2 Restrictions on the agent's design

It is assumed that an agent can only sense the position of the prey. Whenever a neighbouring cell surrounding a prey is occupied or abandoned by an agent, the agent broadcasts an associated message which is received by all other agents in the grid. The agents maintain a table which describes the state of each prey's surrounding positions and which is updated whenever one of the above messages is broadcast in the communication medium.

It is also important to note that no agent has any knowledge about the game boundaries and the position of any other blue agents. It is also assumed that an agent has a capability to recognise that it is in a conflict situation with other agents either by having some sensing (or alarm) mechanism or by having this information provided by an outside source[2].

B.3 Game cycle

A typical game iteration is given in figure B.3.

```
do until Game is over
{
  Prey move
  Agents choose their future positions
  Agents resolve ensuing conflicts
  Agents in the grid move to their final choices of future positions
  If any position was captured/released, message should be broadcast
  Update agents' states and collect game statistics
}
```

Figure B.3: Game

[2]For example, in a computer network this facility could be implemented by having access to the status of a network server or a process queue.

B.4 Control structure

The basic control structure of an agent is described in figure B.4.

```
At each iteration
{
   Sense prey's direction of motion
   If in a square in direct contact with a prey then follow its move
   Otherwise establish a direction in which to move
}
```

Figure B.4: Basic control structure

The implementation of the agent's sensing capabilities is very straightforward. As already mentioned, if a prey moves from its current position it may move either *NORTH*, *SOUTH*, *EAST* or *WEST*. An agent maintains a register associated to each such direction of every prey. Let $R_t(p_i,\ dir_j)$ correspond to the register associated with the direction dir_j of prey p_i at time t. Then $R_t(p_i,\ dir_j)$ is updated at each iteration of the game by the following equation

$$R_{t+1}(p_i,\ dir_j) = \alpha \times R_t(p_i,\ dir_j) + \delta_{p_i,\ dir_j}\ ,$$

where $\delta_{p_i,\ dir_j}$ is 1 if prey p_i has moved in direction dir_j and 0 otherwise. Note that if a prey does not move at a particular iteration $\delta_{p_i,\ dir_j} = 0$ for all dir_j.

B.4.1 Threshold module

If $R_t(p_i,\ dir_j) \geq Th_{direction}$ then dir_j is accepted as a valid predictor of the prey's future moving direction. Note that in principle two or more directions may be considered as valid predictors of the prey's future moving direction. The agent is thus faced with the task of deciding to which one it should move. A decision module illustrated in figure 7.2 of chapter 7 is therefore necessary.

B.4.2 Decision module

The decision module associates a unit vector to each direction in the following way:

$NORTH$	$\Leftrightarrow (0,-1)$
$SOUTH$	$\Leftrightarrow (0,1)$
$WEST$	$\Leftrightarrow (-1,0)$
$EAST$	$\Leftrightarrow (1,0)$

Let $pred(p_i)$ be the unit vector that points to the prey's predicted moving direction. Then the prey's predicted moving direction (and future position) is found by the algorithm described in figure B.5[3].

```
Initialise $pred(p_i) = (0,0)$
For every valid direction $dir_j$ of prey $p_i$
{
  If $dir_j$ equals $NORTH$
    $pred(p_i) = pred(p_i) + (0,-1)$
  If $dir_j$ equals $SOUTH$
    $pred(p_i) = pred(p_i) + (0,1)$
  If $dir_j$ equals $WEST$
    $pred(p_i) = pred(p_i) + (-1,0)$
  If $dir_j$ equals $EAST$
    $pred(p_i) = pred(p_i) + (1,0)$
}
If $pred(p_i) = (0,0)$ prediction is prey's current position
Otherwise
{
  Normalise $pred(p_i)$ to obtain a unit vector
  Prediction is given by:
    Projection factor $\times$ agent's distance to $p_i$ $\times pred(p_i)$
        $+$ prey's current position
}
```

Figure B.5: Predicting a prey's future position

[3]The predicted future position of a position surrounding the prey is easily calculated by a suitable adjustment on the prey's future position.

B.4.3 Establishing the next move

The choice of the next move depends on the agent's state, which can be either *normal* or *bored*. The next figure illustrates how this is implemented.

```
If in normal state
{
  if listening to communication
   {
     move to sector with highest level of communication
     if no such sector exists, proceed as if no communication is implemented
   }
  else
   {
    choose nearest position to target and predict its future position
    move to the nearest of the two
   }
}
Otherwise implement boredom policy
```

Figure B.6: Agent's policies for motion

B.4.4 Boredom module

We have implemented a set of simple actions to be taken when an agent is in a bored state. These are obviously subjected to genetic search and include

- STAY_WHEN_BORED: agent remains in its current position;
- RANDOM_ACTION_WHEN_BORED: agent chooses randomly among *every* direction of move (including remaining at its current position);
- BACK_TRACK_WHEN_BORED: agent makes its best choice of move and then moves in the opposite direction;
- SECOND_BEST_CHOICE_WHEN_BOREDOM_REACHED: agent chooses to move toward the second nearest choice.

In all the above options, if the agent choice of move is onto a cell occupied by a prey, then the agent remains at its current position. As indicated in figure B.3, the agents may change their states in the updating stage of the game. Let $boredom_t$ denote the agent's boredom level at iteration t.

Then if $boredom_t \geq Th_{boredom}$ the agent's state is changed into "bored" and remains in that state until its boredom level falls below a certain value, when it then returns to its "normal" state.

The updating of $boredom_t$ is as follows

$$boredom_{t+1} = \beta \times boredom_t + boredom\_condition_t$$

where $boredom\_condition_t$ is 1 when certain conditions are fulfilled. Basically this happens when the agent is blocking a prey's direction but the latter is not completely surrounded. Note that in the latter case a prey may be surrounded by 1, 2 or 3 agents. It may have also moved or it may have remained in its position during the *current* iteration. The correlation scheme implemented is based on the simple logical conjunction of these observations. For example, $boredom\_condition_t$ may be set to 1 (and consequently an agent may increase its boredom level) if it is blocking a prey's direction which is surrounded by 2 other agents and which has moved in the current iteration.

B.4.5 Conflict Resolution

The basic conflict resolution mechanism implemented in the thesis is illustrated in figure B.7.

```
For each agent in conflict
   agent makes new choice for $(Xchoice, Ychoice)$
For each agent remaining in conflict
   agent's choice for $(Xchoice, Ychoice)$ is $(X, Y)$
```

Figure B.7: Conflict resolution

We have implemented a set of simple actions to be taken when an agent is in conflict. These are obviously subjected to genetic search and include

- COMPLETE_RANDOM_ACTION: agent chooses to move to any direction (which does not overlap with one occupied by a prey) or remain in its own position.
- BIASED_RANDOM_ACTION: similar to COMPLETE_RANDOM_ACTION except that it does not choose a direction that is opposite to its intended move.

- BACK_TRACK: if agent is surrounding a prey it chooses the second best alternative available in that prey. If it will overlap with prey, it chooses to move in the opposite direction. If agent is not surrounding a prey, it moves to the opposite direction of its previous selected choice.
- RANDOM_BACK_TRACK: similar to BACK_TRACK except that agent chooses a random direction when its best choice overlaps with prey. If not surrounding a prey makes a random choice.