# Evaluation of Mastermind Solutions
# Deliverable 1: Research Report

Kyle Dick
F20PA Project
supervised by Kathrin Stark

November 28, 2022

**Abstract**

Describe the problem that is to be approached.

# Contents

# 1 Introduction

Mastermind is a two-player coded-breaking game in which one player is tasked with discovering a secret combination set by the other player. The puzzle is a member of problems known as combinatorial probles where the goal is to find the correct combination of elements from a finite set to satisfy a given set of conditions. The process of investigating possible solutions to the Mastermind puzzle has been underway for decades however there exists some difficulties in defining a clear solution to this day. The implementation of a solution to the Mastermind puzzle would provide benefits to severeal problems face in many sectors of the real world such as cyber security which retains the solutions value as a goal worth seeking. Currently the challenges that are being faced in this endeavour relate to finding methods of determing the correct decisions to be made when formulating guesses at each step of the puzzle. Several attempts at definind a solution have yeilded promising results yet a common occurence is that these implementations struggle when attempting to address variations of the standard Mastermind puzzle.

The implementation that this project seeks to develop will use the foundations provided by the advancements made until the present day. An implementation based in the field of functional programming was chosen as the ideal candidate as among other factors which will be discussed further in later sections, functional programming provides mitigation against variance in outputs which would contribute to confusion when seeking a rigid solution.

## 1.1 Aims and Objectives

The aims and objectives of this project can be surmised in two specific goals. The first goal is to derive a solution to the logical puzzle Mastermind. The second goal is to evaluate solutions implemented during this project with the aim to find methods to improve later iterations. An in depth explanation of the chosen aims and objectives are as such:

- Aim 1: To derive a solution to the puzzle Mastermind.

The goal to find a solution to the problem which minimizes the number of guesses required to discover the correct combination of pegs which comprise the code. Intially the goal will be to derive a base solution to the Mastermind puzzle. The base solution being the most intuitive solution to the problem which does not aim to be the most efficient but to provide a foundation on which improvements can be made. The following objectives are associated with this aim:

- Objective 1.A: Investigate the problem space.

The Mastermind puzzle has previously been the subject of similar research regarding the ability to efficiently find the correct code combination. This stage of the project will concern itself with investigating these previous implementations to guide the project. Through exploring the methods utilised in other investigations into this problem the areas in which these solutions are lacking or could be improved can be found. The research presented in this report represents the progress of this objective.

- Objective 1.B: Implement a base solution.

The ideal base solution should acheive the basic goal of arriving at the correct code combination but should not be the most elegant solution at this point. Instead the base solution should be the foundation for which improvements are made in later iterations. The base solution will be guided through the research conducted through objective 1.A.

- Aim 2: Optimise the Mastermind Solution.

The next step following the creation of a base solution is the optimization of this implementation. The goal with optimization is to discover new methods of how the solution can be improved in regards to its efficiency. In the context of the Mastermind puzzle the concept of a search space is an important factor to optimization as it refers to the set of possible codes which satisfy the problem. An example of how the solution could be optimized is by considering ways in which this search space could be either minimized or how the navigation of the space can be improved.

- Objective 2.A: Explore Methods of Optimizing the Search Space.

The optimization of the problem's search space is directly linked to the measure of efficiency when considering Mastermind solutions. An of how this can be achieved is by implementing methods of minimizing the search space such that the quantity of possible codes which could satisfy the problem is reduced. The other method that should be investigated is improvements to how the search space is navigated by the solution. This is a method which relates heavily to the concept of heuristics and assigning values to the items within the search space which will be covered in a later section of this report. These are not the only methods that exist to optimize the search space and the exploration of differing methods is to be encouraged and sought out should it be possible.

- Objective 2.B: Explore Methods of Measuring the Efficiency of a Solution.

To achieve the goal of optimizing the solution it is important to define the elements which are being optimized for. An example of this is already defined by the previous objective in regards to search space however this is not the only area which can be optimized. The question that is to be answered by this objective is how other factors such as the speed at which a solution can reach a code which satisfies the problem should be considered.

- Objective 2.C: Improve the Solution.

This is a continual objective which encapsulates the main goal of this aim. Measurements of the optimization as defined by previous objectives should reflect the improvements made in later iterations of the solution. Documentation of the results of each optimization method should be presented as a component of this objective.

- Aim 3: Evaluate Solution.

This aim constitutes the final stages of the project. During this stage a final solution should be implementation with appropriate reasoning as to the optimizations which will be compared to already existing solutions. The comparison process should provide insight into the areas which could still be optimized.

- Objective 3.A: Design an Evaluation Process.

The evaluation process should consider the optimization methods implemented during the development of each solution iteration. It is important that the evaluation process considers factors in which the solution can be compared to similar solutions as to provide reasoning for why one solution may be an improvement over another. This objective is covered in the Evaluation Strategy chapter later in this report

- Objective 3.B: Present the Results of the Evaluation Process.

The evaluation process provides insight in how an optimization procedure may have given one solution an advantage over a similar implementation. The results of the process should be communicated clearly such that these insights can be reasoned and explained.

- Aim 4: Investigate Solutions to Mastermind Variants.

This aim and its associated objectives are intended to be goals which are desired to be included with the results of the project but are not fundamental to the investigation. These objectives should be pursued if the time allows for these areas to be explored. The topic of investigation for this aim is to explore how variations of the Mastermind puzzle can be processed by the solution implemented in this project.

Possible optimization methods in respect to these changes should be documented and communicated alongside the challenges encountered.

• Objective 4.A: Investigate Solution with Variance in Set Size for Possible Symbols.

A topic that is a recurring source of intrigue with similar research into the Mastermind puzzle is the potential for a solution to scale with changes in the way that the code is constructed. One such change relates to the possible set of symbols which the code could contain. This objective should consider methods of how the solution can adapt when the number of possible symbol increases above the standard six.

• Objective 4.B: Investigate Solution with Variance in Code Length.

An additional scaling factor which is an area of possible investigation is changes in the length of the code. The considerations regarding this objective relate to the ways in how a solution will handle the larger search space and the changes in optimization.

• Objective 4.C: Investigate Solutions to Similar Problems.

The solution implemented in this project could have applications beyond the intial Mastermind puzzle. This objective is concerned with how the solution can be applied to other similar problems, an example of such a puzzle would be the popular language game Wordle [1].

# 2 Background

The Mastermind puzzle has been subject to investigations regarding solutions to the code-breaking aspect of the game since its release. This section will provide background material which aims to give context to the aims and objectives of this project. The project was inspired by a paper exploring sudoku solutions from a series of problems known as functional pearls, the processes used to derive those solutions will be used to guide this project in this current stage. Following this brief introduction is an explanation of the game Mastermind which the solution will be derived from along with references to previous work by others. The previous work examined will focus on the specific area of search spaces in regards to finding the optimal best move at each position in the puzzle. At the conclusion of this section the goal is that the reader has an understanding of the important concepts relating to this project such that the aims and objectives are clear in their feasability and relevancy.

## 2.1 The Mastermind Puzzle

The Mastermind puzzle began as a two-player game in which one player seeks to break the code created by the other player [2].Originally a physical board game designed by an Isreali Postmaster named Mordecai Meirowitz, distributed through Invicta Toys and Games [3], the game has evolved into a rich area of investigation due to the strategies associated with solving its simple rule set.

### 2.1.1 The Rules of Mastermind

The standard variant of the game as it was introduced in 1970 used a simple plastic board with coloured pegs to represent individual elements for the code. Each game would begin with a secret code being constructed from c colours and l length, in the standard variant this would result in a code of length four constructed from a set of six coloured pegs [2].

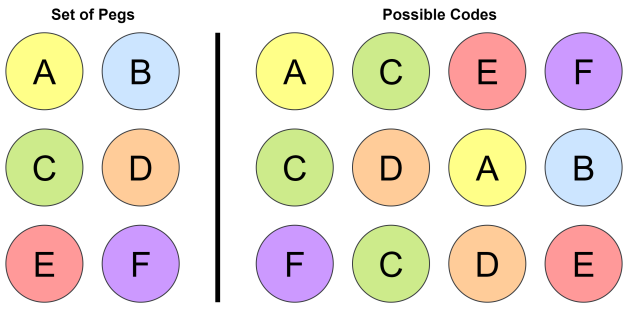


Figure 1: Example codes which would satisfy the contraints of the standard variant of Mastermind.

The codebreaker would attempt several guesses by constructing codes of a similar shape and rewarded with clues as to how closely their guess resembled the hidden code. The physical game represented these hints as smaller plastic pegs of two colours, one which would denote that a correct colour had been selected but was used in an incorrect position and one which would confirm that a correct colour and position had been selected for an element.

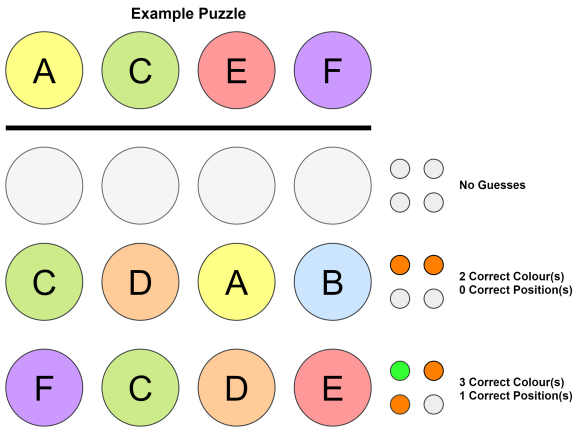


Figure 2: A simplified representation of $CB$ attempting to break the code set by $CM$

There are two end states of the game, one in which the code is broken and discovered by the guessing player and one in which the guessing player is unable to discover the code within a limit of guesses.

### 2.1.2 Variations and Similar Problems

The game can be treated as a individual puzzle in the absence of a human $CM$ through randomly generated code combinations. This project aims to implement a solution to this puzzle that can search for the correct code combination that is comparable with exisiting solutions. The act of searching for the correct code in this context is known as a combinatorial optimization problem with a wide range of applications [5] and it is because of this applicability that the search for a solution is subject to continued investigation.

## 2.2 Problem Description and Fundamental Concepts

### 2.2.1 The Goal of a Solution

At its core the Mastermind puzzle can be surmised as a combinatorial search problem over the set of all possible combinations, most often this set is referred to as the search space. In respect to this idea a solution to Mastermind would be concerned with implementing a method of selecting to correct combinations to play from this

search space. The search space in the standard variant of Mastermind encompasses a total of 1296 unique combinations, the total size of a search space can be found through a simple permutation formula below,

$$c^s$$

for c colours and a combination of length s.

### 2.2.2 Combinatorial Search Spaces

A complication encountered by previous attempts to develop a solution is the increased strain that larger search space place on the algorithms that traverse them with respect to time and memory. Consider the standard variant of Mastermind which as stated before results in a search space population of 1296 combinations. Adding a single new element to the set of possible combination members increases the population to 2041 however incrementing the code length will result in exponential growth increasing the search space instead to 7776.

### 2.2.3 Risks in Searching

### 2.2.4 Donald E. Knuth Implementation

One of the earliest proposed solutions to the Mastermind puzzle was a paper by Donald E. Knuth in which the claim is made that it is possible in all cases for the $CB$ to find the correct code within five attempts [2] [6]. The solution introduces a concept which is important to solving search problems known as the *consistent search space*, a term which would be defined further in later work [7]. The standard variant of Mastermind assumes that the code is of length four with six possible elements it can be constructed from. This definition would mean the code constructed by the $CM$ would be one of $6^4 = 1296$ possible combinations, this set of combinations is referred to as the *s*earch space of the puzzle.

The algorithm that Knuth proposes seeks to restrict this search space after each guess using the feedback provided by the $CM$. The algorithm proceeds after each guess by picking the best possible choice for the subsequent guess at each step, this greedy strategy [2] is possible as the feedback provided can eliminate combinations which are not consistent with previously guesses.

The first stage of Knuth's investigation was to define the rules of the game in a way that a computer would be able to understand. First defining the two important variables being the code $x_1x_2x_3x_4$ and the test pattern, or guess, $y_1y_2y_3y_4$ The initial rules that were derived were as follows:

- The number of "black hits (B)", i.e., the number of positions j such that $x_j = y_i$

- The number of "white hits (W)", i.e, the number of positions j such that $x_j = !y_j$ *but* $x_j = y_k$ for some k and $y_k$ which has not at this point been a hit [6].

The process of restricting the search space can be seen in the example below with the symbols represented as integers (1, 2, 3, 4, 5, 6):

| No. of Guesses | Test Pattern | Hits | Possible Combinations |
|---|---|---|---|
| 0 | no guess | N/A | 1296 |
| 1 | 1122 | WWW | 16 |
| 2 | 1213 | BWW | 4 |

At the initial point where no guess had been made the search space is still comprised of all 1296 possible code combinations. This space of possible combinations is restricted using the result of the first guess which states that three of the chosen symbols were correct however their positions were incorrect. This information allows a great amount of restriction to only 16 possible code combinations which the code could belong to. The second guess restricting this space even further to just four possible combinations. Through this method Knuth asserted that it should be possible to achieve a correct guess within five test patterns. The starting pattern discovered by Knuth to give proof to this claim was found to be 1122. A possible progression of stages from this initial test pattern would be: Another example is as follows:

| No. of Guesses | Test Pattern | Hits | Possible Combinations |
|---|---|---|---|
| 0 | no guess | N/A | 1296 |
| 1 | 1122 | B | 256 |
| 2 | 1344 | W | 44 |
| 1 | 3526 | W | 7 |

This situation would also allow for the next test pattern to distuingish the final possible combinations using the test pattern 1462. The algorithm places a high value on test patterns which coax the process towards the fourth guess being able to distinguish between the possible combinations. This method of solving the problem however is admitted within the paper to not be the most optimal solution. This can be shown by the way that the algorithm processes the following situation:

| No. of Guesses | Test Pattern | Hits | Possible Combinations |
|---|---|---|---|
| 0 | no guess | N/A | 1296 |
| 1 | 1122 | BWW | 16 |
| 2 | 1213 | BB | 4 |

which results in a possible four remaining code words (2212, 4212, 5212, 6212). The algorithm would select the test pattern 1145 as the next guess however when compared to the test pattern 4222 it can be shown that it in actuality results in less distinguishing results:

| Test Pattern | Hits | Code |
|---|---|---|
| 4222 | BWW | 2212 |
| | BBB | 4212 |
| | BB | 5212 |
| | BB | 62122 |
| 1145 | W | 2212 |
| | WW | 4212 |
| | WW | 5212 |
| | W | 62122 |

It is clear that should 4222 be used at the test pattern there are two possibilities in 2212 and 4212 where the code can be known by the third guess. This is a better result compared to the test pattern 1145 where two possibilities will always be the result.

## 2.3 Progression of Solutions

### 2.3.1 Evolutionary Computing

A trend in modern solutions to Mastermind is the exploration of evolutionary computation as a tool for developing the algorithms which evaluate the best next step at each stage. This was the basis of research for the 2005 GenMM (Genetic Mastermind) algorithm which sought to compete with the exhaustive search implementations which were currently being investigated at that time [1].

Evolutionary computing serves as an adequate route for tackling decision problems such as the evaluation of possible steps to take whilst traversing the search space. The most common way this area is implemented is through the use of Evolutionary Algorithms which simulate the Darwinian natural evolution process to incrementally change the heuristics of a system [2].

## 2.4 Functional Programming as a Tool

### 2.4.1 Advantages of Functional Programming

### 2.4.2 Functional Pearls

The inspiration for this project was a paper by Richard Bird which implemented a solution to the puzzle game Sudoku as an example of a series of problems known as functional pearls. This sections will explore the topic of functional pearls and their relevance to the current project. Functional Pearls begin as small problems which programmers wish to explore, focusing on brief but engaging examples that showcase either a guided explanation to a proof or presenation of unique data structures. The goal of a functional pearl is to teach important programming techniques and fundamental design principles [8]. Richard Bird described functional pearls in a speech as 'elegant, instructive exampels of functional programming' while showcasing his implementation of a sudoku solution[9].

The solution which this project aims to explore would enter the scope of a functional pearl due to its aim to document an example of functional programming. The project would however deviate from a traditional functional pearl as the scope would be expanded to consider similar solutions in the problem space to evaluate the different methods of solving Mastermind.

The functional pearl which inspired the project is Richard Birds paper 'Functional Pearl: A Program to Solve Sudoku' [10]. The aim of the solution was to define the function:

$$Sudoku :: Board \rightarrow [Board]$$

This function would take an input in the form of a Sudoku board, represented by a matrix of characters, and output a list of possible completed boards. Following this declaration the paper would proceed to use logic and equational reasoning to define the function using the functional language haskell to achieve the solution. Finally Richard Bird was able to arrive at the following definition for his solution:

$$Sudoku :: Board \rightarrow [Board]$$

$$Sudoku = map(maphead) \bullet search \bullet prune \bullet choices$$

where search, prune and choices representing functions declared earlier in the implementation. This sudoku solutions provide an example of how the mastermind solution this project aimst to implement may be approached. Aiming for a solution that resembles this process of declaring an initial function or set of functions then deriving a defintion for them that can be evaluated through comparison with similar solutions. The next section will examine the problem that this project will aim to solve and the work that has been explored in the area previously.

## 2.5 Applications Beyond Mastermind

### 2.5.1 Mastermind Complexity Class

Mastermind belongs to a class of problems known as NP-Complete decision problems. This means that they are solveable in exponential time. A solution to Mastermind in polynomial time would be an incredibly valuable resource as it would be proof that P = NP. This meaning that for all problems in class NP which are verifiable in polynomial time there exists a polynomial solution to the problem. It is currently not known if this hypothesis that P = NP is true however it would be a great benefit to computer science as it would allow for problems such as protein folding to be solved however it would also mean that encryption, a security method which relies on P != NP, would be rendered obselete.

# 3 Research Methodology and Requirements Analysis

The scope of a Mastermind solution should focus on the combinatorial problem of how the search space, the set of all possible combinations, should be traversed. As has been explored in the background material the ability to navigate the possible combinations and the hints derived is integral to evaluating the correct steps to take to minimize the distance to a combination which matches the secret code. The research methodologies described in this chapter will propose several questions that this project hopes to answer through its implementation of a unique solution to the Mastermind puzzle. In support of these questions the requirements of the system which will be used to explore the research methodology will also be described.

## 3.1 Research Methodology

### 3.1.1 Research Questions

The scope of this project's investigations are broad due to the many differing approaches that have previously been explored. By defining a set of research questions the process of developing a solution can be structured around the aims and objectives stated at the beginning of this report. The research questions are as follows:

**1. How do exhaustive solutions to traversing the search space of a combinatorial problem compare to evolutionary methods?**

A large body of past work has focused on the applications of evolutionary computation, specifically genetic algorithms, to evaluate the combinations within the search space. The results of these implementations have been shown to have an advantage over previous solutions, some of which utilised an exhaustive approach which analysed each member of the search space. This inquiry provides a foundation for the base case of a solution, the results of which can be used as a metric for the evaluation of possible improvements in later iterations.

**2. What are the important factors to be considered when selecting the optimal combination from a search space?**

When approaching a large set of elements it is important to understand why a specific element from that set may be selected over another. The concept of consistency was introduced in the background material of this project which leads to the core of this question which asks how should this specific implementation evaluate which combination to select at each step. There already exists some work in this area presently such as the approach of partitioning combinations based on the hints provided through each guess, a starting step investigating this question can explore implementations utilising this method.

**3. What are the advantages of using functional programming as the basis for a solution to combinatorial search problems?**

If the implementation derived from this project are able to return a positive development in finding a solution to the Mastermind puzzle it should be noted to what extent the specific paradigm of functional programming is able to provide support. Following from this it can be investigated if the unique behaviours of functional programming provide benefits which would lend itself to being the optimal tool for further research into the area.

## 3.2 Requirements Analysis

### 3.2.1 MoSCoW Prioritisation

Description of the moscow prioritisation system.

### 3.2.2 Functional Requirements

Environment Functional Requirements

| ID | Description | Priority |
|---|---|---|
| FR 1.1 | The system must be capable of storing an integer combination of a given length. | Must |
| FR 1.2 | The system must be capable of generating an integer combination from a range of given integers and of a given length. | Must |
| FR 1.3 | The system must be capable of evaluating equality between two integer combinations. | Must |
| FR 1.4 | The system must be capable of returning information on the similarity of a given input to the stored combination. | Must |

### 3.2.3 Non-Functional Requirements

# 4 Evaluation Strategy

## 4.1 Defining Metrics

The evaluation of the solutions implemented in this project relate strongly to the research questions that we have defined in the previous section. Following from this means that the metrics on which our solutions will be evaluated on are as such:

- A solution which is able to restrict the consistent combination search space is considered more valuable.
- A solution which is more accurate in selecting the correct code from the consistent search space is more valuable.
- A solution which is able to scale elegantly with both an increase in possible symbols which comprise the code and an increase in the length of the code.

# 5 Preliminary Work: The Base Solution

## 5.1 Design of a Base Solution

## 5.2 Results and Analysis of Base Solution

In here talk about the work done with the current base solution. It will be originally done based on the donald knuth implementation.

# 6 Project Management

## 6.1 Agile

Description of the agile methodology and its advantages as it pertains to this project

## 6.2 Gantt Chart

The organisation of this project is represented by the following Gantt Chart, by utilising this chart the progress of later stages can be tracked and adjusted if there is a risk to the schedule.

[NEED TO FINISH THE GANTT FIGURE

It is important that each major milestone of the project is following by a brief review period. The reasons for this is to examine the schedule and provide mitigatin measures if required.

## 6.3 Risk Analysis and Mitigations

### 6.3.1 Risk Definitions

To manage the progress of this project efficiently it was important to identify possible risks that could prevent the realisation of the goals laid out earlier in this document. To aid in the identification of the risks the following key was used to classify the associated risks:

- People (P) - Risks which are the result of issues related to those individuals involved in the project. This relates to the wellbeing, scheduling and personal issues that can be encountered.
- Technological (T) - Risks which can result from the technology being used to engineer the solutions. This relates to the technologial constraints that could be encountered or the hardware required.
- Requirement (R) - Risks which can result from changes to the requirements of the project. This relates to problems encountered with the work being implemented for the project. Logical problems and issues with the material involved would be included in these risks.

| ID | Risk | Description |
|---|---|---|
| P/T/R1 | Textual Title of Risk | Textual Description of the Risk |

### 6.3.2 Risk Identification

| ID | Risk | Description |
|---|---|---|
| P.1 | Illness and Health Complications | The situation in which an individual involved in the project is unable to contribute due to illness or health complications. This risk is slightly heightened as the world recovers from the effects of the Covid-19 pandemic however it also considers other conditions which would require time away from the project. |
| T.1 | File Loss | This is the situation in which files or documents relating to the project is lost. This is a low priority issue as precautions have been taken to mitigate this such as using version control through Github. |
| R.1 | Insufficient Base Solution | The success of this project is dependant on having a solid base solution. The success of the base solution is assessed by a set of conditions which deems it suitable for solving the problem. The base solution being unable to meet this conditions is a serious situation and as such should be of a high priority. |

### 6.3.3 Risk Mitigation Procedures

## 6.4 Considerations of Professional, Legal, Ethical, and Social Issues

# References


[1] New York Times, *"Wordle"* https://www.nytimes.com/games/wordle/index.html (Online, accessed 21-11-2022)

[2] Weisstein, Eric W. *'"Mastermind." From Mathworld – A Wolfram Web Resource'* https://mathworld.wolfram.com/Mastermind.html (Online; accessed 9-November-2022)

[3] Invicta Toys and Games ltd. *'History of Mastermind'* https://web.archive.org/web/20070812104420/http://dspace.dial.pipex.com/town/road/gbd76/toys.h (Archived 2007)

[4] Nelson, Toby. *'Investigations into the Master Mind Board Game'* https://web.archive.org/web/20150906043015/http://www.tnelson.demon.co.uk/mastermind/index.h (1999, Archived 2015)

[5] Merelo-Guervós, J.J., Castillo, P. and Rivas, V.M. *'Finding a needle in a haystack using hints and evolutionary computation: the case of evolutionary MasterMind', Applied soft computing* **6(2)** (2006) pp. 170–179. doi:10.1016/j.asoc.2004.09.003.

[6] Knuth, Donald E. *'The Computer as Master Mind', Recreational Mathematics* **9(1)** Stanford University (1976, Baywood Publishing Co., Inc.)

[7] Merelo, J.J., Mora, A.M., Cotta, C. and Runarsson, T.P. *'An experimental study of exhaustive solutions for the Mastermind puzzle'.* (2012)

[8] Bird, Richard. *How to Write a Functional Pearl* International Conference on Functional Programming, Portland (2006)

[9] Gibbons, Jeremy. *University of Oxford, Functional Pearls* http://www.cs.ox.ac.uk/people/jeremy.gibbons/pearls/ (2009)

[10] Bird, Richard. *Functional Pearl, A Program to Solve Sudoku* Cambridge University Press, Cambridge, 2006