

Logic-Based Solutions to Puzzles and Riddles

Deliverable 1: Research Report

Kyle Dick
F20PA Project
supervised by Kathrin Stark

November 9, 2022

Abstract

Describe the problem that is to be approached.

Contents

1	Introduction	4
1.1	Aims and Objectives	4
2	Background	6
2.1	Functional Pearls	6
2.2	Mastermind	7
2.3	Donald Knuth's Solution	8
2.4	Logic and Equational Reasoning	10
3	Research Methodology	11
4	Evaluation Strategy	12
5	Preliminary Work	13
6	Project Management	14
6.1	File and Resource Management	14
6.2	Timeline and Deadlines	14
6.3	Risk Analysis and Mitigations	14

1 Introduction

The main focus of this project is to investigate solutions to the puzzle game known as Mastermind. The approach will be to develop an initial base solution in a functional programming language which can be iteratively improved with the goal of creating a solution which is logically sound and efficient. Mastermind is a simple code-breaking game in which one player will create a secret code which consists of four coloured pegs from a choice of six. This is the standard version of the game however there are multiple variations of the game, some of which will be discussed and examined within the scope of this project.

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. Initially 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.
 - Objective 1.A: 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.
 - Objective 1.B: The ideal base solution should achieve 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.
 - Objective 1.C: This objective is a continuous process which will be referred to throughout the life cycle of the project. Each iteration of the solution, starting with the base solution, should be evaluated against

a set of criteria defined later in the document. After this review possible improvements can be documented and worked towards for the next iteration.

- Aim 2: The goal of this project is to arrive at a solution which can be evaluated in terms of its soundness and efficiency. This means that the solution should be one that is easily understood with the appropriate explanation. The steps in which the solution is arrived at should follow a logical progression that is reflected in its documentation. The efficiency should be measured through how quickly the correct code combination can be found by the solution.
 - Objective 2.A: Designing an evaluation process for this project should be foremost concerned with how well the solution is able to solve the problem. The Mastermind puzzle is most commonly restricted by the amount of guesses allowed to the player. From this the metric in which a solution should be measured against is how quickly the correct code combination can be found and its minimum amount of guesses to do so. A solution which does not reach the correct code combination within a set amount of allowed guesses will be considered a failed solution.
 - Objective 2.B: Analyse an iteration using the chosen strategy.
 - Objective 2.C: Report the results of the evaluation strategy and iterate if improvements are identified.
 - Objective 2.D: Document results taken to arrive at a desired solution.

2 Background

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 will relate to the field of functional programming and the specific area of logic and equational reasoning. 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 feasibility and relevancy.

2.1 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 presentation of unique data structures. The goal of a functional pearl is to teach important programming techniques and fundamental design principles [1]. Richard Bird described functional pearls in a speech as 'elegant, instructive examples of functional programming' while showcasing his implementation of a sudoku solution[2].

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' [3]. 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 aims 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 definition 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.

2.2 Mastermind

The aim of this project is to deliver a solution to the code-breaking board game Mastermind designed by Mordecai Meirowitz and originally manufactured by Invicta Toys and Games [4]. The original board game consisted of two players with different roles. Player A would be designated the code-breaker while Player B would attempt to solve the code.

The game was designed around player A selecting a code unknown to player B and player B attempting to break the code. Player A would first construct the code from a set of six coloured pegs with the code being exactly four pegs in length. There are variants where the size of the set of choices and code length are variable however this describes the standard variant.

A variant of the original mastermind puzzle was explored

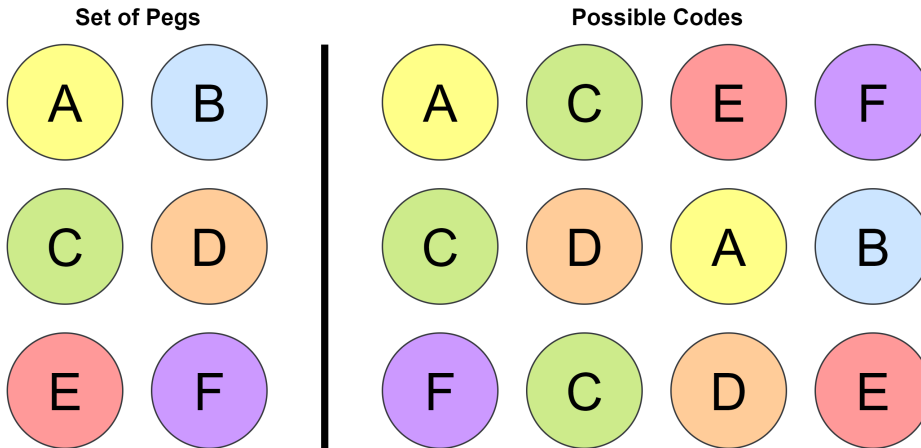


Figure 1: Example of code enumerations Player A could construct

The challenge for Player B is to correctly guess the code created by considering feedback given from Player A relating to how correct each guess was. A guess is evaluated with Player B given a number of pegs with two possible colours, one which represents a correct position and another which represents a correct colour.

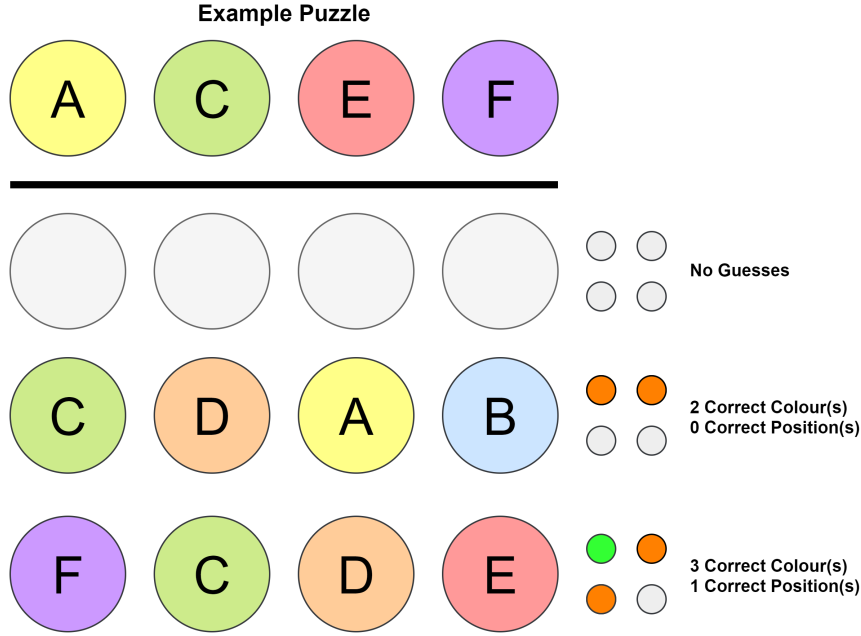


Figure 2: Simplified example of Player B guesses

The solution that this project aims to implement would assume the role of Player B attempting to correctly identify the generated code in the minimum number of guesses. To provide a solution which is not bound by the number of elements within the set of possible choices or the length of the code, the code will follow the following rules.

$$code = [c_i, \dots, c_n]$$

$$pegs = \{c_i, \dots, c_m\}$$

$$n, m = \mathbb{N} > 1$$

A solution which is agnostic of the code length and number of possible elements would support basic variants of Mastermind where the only difference are these restrictions. The base solution will solely focus on solving for a code of length four with six possible elements.

2.3 Donald Knuth's Solution

The computer scientist Donald E. Knuth proposed a solution to Master Mind in 1996 which focused on the strategy of minimizing the amount of possible enumerations for the code after each guess. While discussing this he used integers one through six to represent the six possible symbols for the code created by the "codemaker" noting that the total possible number of enumerations being equal to $6^2 = 1296$.

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", i.e., the number of positions j such that $x_j = y_j$
- The number of "white hits", i.e., the number of positions j such that $x_j = y_k$ for some k and y_k which has not at this point been a hit. [5]

The next step involved defining variables n_i as the number of times symbol i occurs within the codeword and n'_i as the number of times it occurs within the guess for $1 \leq i \leq 6$. With these variables Knuth was able to define two rules which the computer could understand:

$$1 : \min(n_1, n'_1) + \min(n_2, n'_2) + \dots + \min(n_6, n'_6)$$

for the total number of correct hits, both white and black. From this it can be followed that the number of misses is

$$2 : \min(n_1 - n'_1, 0) + \min(n_2 - n'_2, 0) + \dots + \min(n_6 - n'_6, 0)$$

The algorithm that Knuth used would track the remaining possible codes after each test pattern was checked. This would allow Knuth to use the general rule of restricting the number of possible to arrive at the correct code while minimizing the amount of guesses required. 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:

<i>test pattern</i>	<i>hits</i>	<i>possible combinations</i>
1122	WWW	16
1213	BWW	4

This idealistic situation would result in the next test pattern being able to distinguish from the final four possible combinations using the test pattern 1415. Another example is as follows:

<i>test pattern</i>	<i>hits</i>	<i>possible combinations</i>
1122	B	256
1344	W	44
3526	BWW	7

This situation would also allow for the next test pattern to distinguish the final possible combinations using the test pattern 1462. The algorithm used to decide the progression from one test pattern to next decides based on the test pattern which is able to minimize the number of possible combinations. The algorithm selects the first test pattern such that there is no situation where the fourth test pattern is unable to distinguish the final remaining code combinations giving credit to Knuth's claim that the correct code can be ascertained within five guesses.

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:

<i>test pattern</i>	<i>hits</i>
1122	<i>BWW</i>
1213	<i>BB</i>

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:

$$4222 : \quad BBW = 2212 \quad BBB = 4212 \quad BB = 5212 \quad BB = 6212$$

$$1145 : \quad W = 2212 \quad WW = 4212 \quad WW = 5212 \quad W = 6212$$

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.

2.4 Logic and Equational Reasoning

The goal with the solution this project aims for is to implement a solution which is logically sound. This means for the project to succeed

3 Research Methodology

The question that this project aims to answer is 'How can the combinatorial problem of Mastermind be solved'. The scope of this question is quite large so instead it will be broken down into a set of smaller questions that will help to define the research methodology. Those questions are as such:

- How can the search space of consistent combinations be minimized
- Which scoring algorithms provide the best possible chance of selecting the correct code from a set of consistent solutions.

4 Evaluation Strategy

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

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 File and Resource Management

6.2 Timeline and Deadlines

6.3 Risk Analysis and Mitigations

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 technological 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

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.

References

- [1] Richard Bird, *How to Write a Functional Pearl* International Conference on Functional Programming, Portland (2006)
- [2] Jeremy Gibbons, *University of Oxford, Functional Pearls*
<http://www.cs.ox.ac.uk/people/jeremy.gibbons/pearls/> (2009)
- [3] Richard Bird *Functional Pearl, A Program to Solve Sudoku* Cambridge University Press, Cambridge, 2006
- [4] Invicta Toys and Games ltd, *History of Mastermind*
<https://web.archive.org/web/20070812104420/http://dspace.dial.pipex.com/town/road/gbd76/toys.l>
(Archived 2007)
- [5] Donald E. Knuth, The Computer as Master Mind *Recreational Mathematics*
9(1) (1976)