AIML402 Assignment 1

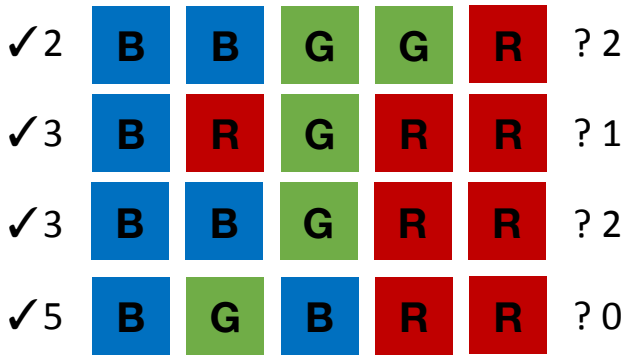
Mastermind playing agent

Weight:10% Lecturer: Lech Szymanski

For this assignment, you will create an agent that plays the game of Mastermind.

Task 1 (10 marks)

Write a Python program that plays the game of Mastermind.



In the game of Mastermind the player attempts to guess the pattern of colours in the line of squares in as few attempts as possible. After each guess the player is given information about the number of squares in the correct position and the number of squares of the correct colour, but out of position according to the intended solution. The objective for your agent will be to play the game and score the lowest average number of guesses across a number of games.

The environment

The environment for this game is configured to randomly pick a target pattern of a given length using a number of allowed colours. In a given game the player/agent has up to some maximum number of guesses to guess the target sequence. The environment is episodic – each guess of the agent is treated independently of the previous guesses, providing information about whether colours in the last guess were in the right place and/or present at all in the target solution.

The agent is scored over a series of plays with the average number of guesses it takes to find the solution word. There is an extra penalty that doubles the score for the particular run if the agent does not guess the word within the allowed maximum number of guesses. Thus, for example, in a game of maximum of 6 guesses, an agent can obtain a score of 1,2,3,4,5,6 or 12.

Game parameters

For the purpose of development and testing, the game environment in this assignment will have the following configurable parameters: length of the guess, number of colours, maximum number of guesses allowed, number of games that are played, seed of the pseudorandom number generator picking the solution words.

The agent

The agent whose behaviour you have to implement for this assignment is a Mastermind player. On initialisation the agent is provided with the length of the guess, valid alphabet of colours (represented as a list of unique alphabetic characters), and the max number of guesses per game.

The agent function

The agent function is invoked to get agent's next guess. Its single argument is a tuple of percepts, which relate the information about the last guess provided by the environment. The agent function needs to returns a list of characters that constitutes the next guess.

Percepts

The percepts of the Mastermind-playing agent is a tuple that contains four pieces of information:

- `guess_counter` – an integer value indicating which guess the agent is currently on, `guess_counter`=0 indicates the first guess;
- `last_guess` – a list of characters with information on the previous guess; at `guess_counter`=0 this list contains all 0's and it should be ignored by the agent.
- `in_place` – integer value relating the number of characters/colours in the correct place in the last guess;
- `in_colour` – integer value relating the number of characters/colours from the last guess that are also found in the target solution, but weren't in the correct position.

For an illustrative example of how the percepts work, see Figure 1.

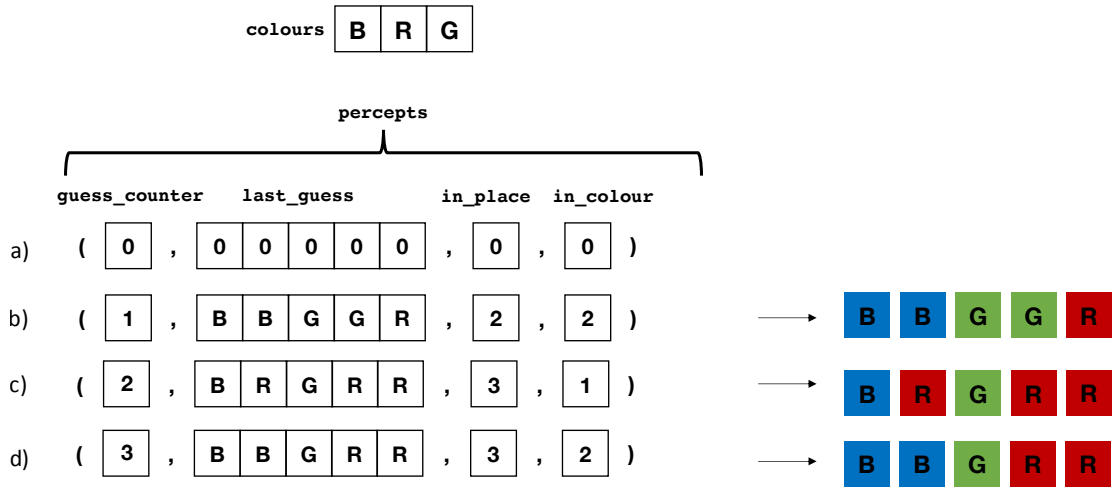


Figure 1: Percepts corresponding to a sequence of guesses in a game of code length of 5 with the alphabet of three colours [B,G,R] as shown on top of the figure; a) first guess, the values of `last_guess` should be ignored by the agent; b) second guess indicating the previous guess was [B,B,G,G,R] and that, in total, it contains four correct colours, two of them in the right position; c) third guess indicting the previous guess was [B,R,G,R,R] and that, in total, it contains four correct colours, three of them in the right position; c) foruth guess indicating the previous guess was [B,B,G,R,R] and that, in total, it contains five correct colours, three of them in the correct position.

Actions

The action of the agent is a list of the colour characters, which constitutes the next guess. The game ends if the agent guesses correctly or maximum number of guesses is exhausted. The next time agent is invoked, the `guess_counter` will be 0, which indicates a new run of the game is played.

The engine

A framework with Python code implementing the environment for this assignment is provided for you. For files and instructions on how to use it see the "How to use the Mastermind Engine for Assignment 1" in the "Assignments" section on Blackboard.

Task 2 (10 marks)

You must also write a report explaining the strategy of your agent and providing some results of its evaluation. The report should include:

- a brief introduction – you don't have to repeat the entire description of the game and environment given, but a short intro to what your report is about is required;
- explanation of how your agent works – the strategy, high level algorithm, not explanation of you code line by line;
- analysis of your agent's performance with some results – ideally conveyed via graphs and/or tables;
- brief conclusion – summary/discussion of the results/conclusions;
- citations – if needed;
- information on how to run your code – if extra libraries are required.

Screenshots of the text in your terminal and/or photos of hand-drawn diagrams are not the best way to add figures to your report. Figures and diagrams are great to have, but draw them properly (in Inkscape or PowerPoint for instance).

To give you a bit of guidance for the report structure, a LaTeXtemplate is provided (you can find it on Blackboard in the "Assignments" section under "Report template in LaTeX"). You don't have to use LaTeX to write your report – but it might be a good idea to follow the general structure provided in the template.

Finally, to pre-empt inevitable questions about the length of the report, let's say: "around 1500 words (which is roughly 3 pages)". But this is not an absolute number – a bit more is fine; more pages if you have lots of figures is not a problem; a shorter (but really good) report won't be marked down for length.

Marking scheme

This is an individual assignment and the marks will be allocated as follows:

- Task 1: **10 marks**. This task will be assessed by running your code and inspecting the performance of your agent. I am looking for evidence of "better than random" behaviour under different game settings. I don't expect implementation of the optimal solution possible. While doing some research on existing possible solutions (at algorithmic level) is definitely welcome, this is not required. Trying to come up with the an algorithm of your own is perfectly fine, which may or may not involve strategies learned in the paper so far. A low bar target is a random guess agent that takes into account information from the history of previous guesses and eliminates solutions that are not possible; for the top grade the agent, in addition to looking "backwards", must somehow look forward to make strategic guesses that improve chances of quicker arrival at the target.

- Task 2: **10 marks**. Marks will be awarded for clarity of the report, good explanation of the approach taken, and at least a bit of evaluation of your algorithm with some analysis.

The percentage of the obtained marks out of 20 will be converted to a fraction of 10% that constitutes the weighting of this assignment in AIML402.

Submission

The assignment is due at **4pm on Tuesday, 8 Aug 2023**. You should submit via Blackboard two files: my_agent.py and the pdf file containing the report. Don't zip these files into one attachment – add them separately to the submission.

Late submissions will incur a 5% penalty per day.

Academic Integrity and Academic Misconduct

Academic integrity means being honest in your studying and assessments. It is the basis for ethical decision-making and behaviour in an academic context. Academic integrity is informed by the values of honesty, trust, responsibility, fairness, respect and courage. Students are expected to be aware of, and act in accordance with, the University's Academic Integrity Policy.

Academic Misconduct, such as plagiarism or cheating, is a breach of Academic Integrity and is taken very seriously by the University. Types of misconduct include plagiarism, copying, unauthorised collaboration, taking unauthorised material into a test or exam, impersonation, and assisting someone else's misconduct. A more extensive list of the types of academic misconduct and associated processes and penalties is available in the University's Student Academic Misconduct Procedures.

Use of generative software such as ChatGPT is allowed as long as it is for the purpose of aiding, not subplanting the effort of development code and/or improving the writing. If generative software is used, students must specify (in the report's appendix) how it was used and on what aspects of the assignment.

It is your responsibility to be aware of and use acceptable academic practices when completing your assessments. To access the information in the Academic Integrity Policy and learn more, please visit the University's Academic Integrity website or ask at the Student Learning Centre or Library. If you have any questions, ask your lecturer.

- Academic Integrity Policy
- Student Academic Misconduct Procedures