CS344 Project Specification

Laveen Chandnani - 2004842

1 Title

An analysis of strategies in the re-pairing game

2 Problem

A Dyck word is a string consisting of opening and closing brackets, in such a way that at any point in the string there are at least as many opening brackets as there are closing brackets on the left side of this point, and there are an equal number of opening and closing brackets in the entire string. For example, ((())) and (()()) are valid Dyck words, but (() and ())(are not.

Dyck words have some interesting combinatorial properties, but also have their applications in theoretical computer science, more specifically in the context of formal languages. The paper by Chistikov and Vyalyi [1] is an example of such an application, and explores a one-player game called the "re-pairing game"; given a Dyck word, a move consists of taking an opening bracket and pairing it with any closing bracket to its right, then erasing them. This process is repeated until the string no longer has any characters. The width of a play (or strategy) is the maximum number of nonempty segments of symbols seen during a play, and to compute the width of a Dyck word is to work out the minimum width sufficient for re-pairing the given Dyck word.

The paper gives a simple re-pairing strategy for Dyck words, proves upper and lower bounds for the width of this strategy, and beats this strategy for a subset of Dyck words that have a one-to-one correspondence with complete ordered binary trees. This game along with the results proven is then linked back to a question in automata theory. One of the open problems at the end of this paper is computing the width of a general Dyck word. This project will conduct a survey on Dyck words and the re-pairing game, explore established results from the paper and attempt to make progress on the given open problem.

3 Objectives

The first part of this project will focus on conducting a well-structured survey of the field and the paper this project is based on. This involves:

- An introduction to the relevant definitions and content to lay out some background knowledge
- Expanding on and providing potential novel insights/perspectives on well established results

The second part of this project will focus on the open problem. This involves:

- Establishing the research landscape on the re-pairing game and relevant topics
- Creating software to demonstrate re-pairing strategies from literature and allow manual experimentation on Dyck word re-pairings
- Using the software to analyse subcases of Dyck words and exhaustively find their width in an attempt to pin down a formula for the width of a general Dyck word

4 Technical considerations

In order to tackle the open problem, I will be using an experimental approach. This will allow me to input a given Dyck word, and either run a given re-pairing algorithm on it, create our own re-pairing algorithm, pair up opening brackets with closing brackets manually, or brute force the Dyck word to work out the minimum width of the given Dyck word. I have considered the following two options:

- **A Python based GUI:** As I am most familiar with this language, this seems like a natural option to explore light Python GUI frameworks to build an application with for experimentation, and the framework that appeared to be most accessible and suitable for this was Tkinter. However, upon further research and experimentation, the library felt unwieldy to work with and the dated visual style and application package format meant that it would be difficult not only for myself to work with, but for users to experiment with the resulting software.
- **A web based application:** Although web development is an area I am less familiar with, the experience of implementing a UI using React and Materials UI is far less cumbersome than that of using Tkinter, whilst also having a more modern appearance. I would also still be able to develop the re-pairing algorithms in Python and merge them with the frontend code using Flask, which allows me to utilise my current knowledge. An advantage of using a web based application over a Python GUI is that the software can be made accessible to other users and students over the internet.

I will be following a web based application approach for its ease of access and visual style. This approach will also introduce me to web development as a skill. The website will be hosted using Github pages due to ease of use, since this can be used directly from the Github repository I will be storing my code in.

5 Timetable

Week	**Task**
Term 1 Week 2	**Submission of project specification**
Term 1 Week 2-5	Understanding the relevant literature and dissecting proofs
Term 1 Week 5-8	Writing first half of report
Term 1 Week 8	Writing up progress report
Term 1 Week 9	**Submission of progress report**
Term 1 Week 9-10	Begin thinking about web application implementation
Christmas Holidays	Write up web application, collect results and begin to analyse
Term 2 Week 1	Continue analysis of results from web application
Term 2 Week 2-6	Further work on the open problem and writing second half of report
Term 2 Week 7	Preparing for project presentation
Term 2 Week 8-10	**Project presentation** and finishing up final report
Easter Holidays	Polishing up final report
Term 3 Week 1	**Submission of final report**

Note that there will also be weekly meetings with my project supervisor to ensure the project is going as expected, and any issues can be addressed as soon as possible.

6 Risks

There are two main risks with this project. The first is a technical one i.e. loss of power whilst working, loss of data etc. These will be addressed by the following:

- Writing the code within VScode using an autosave setting. This ensures that in the event of power loss, the progress I make is not lost.
- Using Github for backup and source control. This ensures that in the event of data loss or any other reasons which can cause my work to get lost, the work is still available to access and progress is not lost.

The second risk is within the nature of the project itself; the second part of the project deals with attempting to obtain new findings on an open problem. For this, I have decided to create a tool to allow

me to tackle the problem experimentally, not just on pen and paper. This way, in the event that the goal is infeasible, I have some work and results to show for the project.

7 Legal, Social, Ethical and Professional issues

Due to the nature of the project, this will not involve any social or ethical issues.

However, with the web based application approach, a possible legal issue is with the licensing of any libraries/frameworks used. Looking further into this reveals that React and Materials UI (which I plan to use for the frontend) are under the MIT license, and Flask (which I plan to use to connect the backend python code to the frontend) is under the BSD-3-Clause license, both of which do not pose any issues for the intended use case in this project.

Another potential issue is a professional one; with a large portion of this project being a survey on a field and based on a paper, there is a risk of unintentionally plagiarising content from literature. This will be kept in mind whilst writing the final report, ensuring appropriate references are made where required.

References

[1] Chistikov, D. and Vyalyi, M. (2020) 'Re-pairing brackets', Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science, pp. 312-326.