Sudoku Stuff

Chris Mader[1] und Niklas Widmann[1]

[1] *Duale Hochschule Baden-Württemberg*

7. Juni 2021

Inhaltsverzeichnis

Abstract

1 Einführung

1.1 Motivation

2 Grundlagen

2.1 Technologien

2.1.1 React

3 Implementation

3.1 Generation

Ein Ziel des Projektes war das schreiben eines Algorithmus, welcher in der Lage ist, lösbare Sudokus zu generieren. Diese sollten eine eindeutige Lösung haben und abhängig von bestimmten Inputvariablen erstellt werden. Zu diesen Inputvariablen zählen sowohl die Schwierigkeit, als auch bestimmte Lösungsstrategien, welche potentiell beim darauffolgenden Lösen durch den Nutzer eine Anwendung finden sollen.

Die Generation wurde durch drei Unterschritte realisiert. Zuerst wird versucht, ein leeres Sudoku vollständig zu füllen. Dann werden wieder Felder gelöscht, wobei nach jedem Feld ein Prüfalgorithmus testet, ob das Sudoku noch eine eindeutige Lösung hat. Wenn eine gewissen Anzahl von gelöschten Feldern, welche von der Schwierigkeit abhängt, erreicht wird, ist der Generationsvorgang abgeschlossen.

3.1.1 Erzeugen eines gefüllten Sudokus

Zuerst muss ein vollständig gefülltes Sudoku generiert werden, welches das gelöste Sudoku darstellt. Dazu wird für alle Ziffern von 1-9 versucht eine Stelle in jedem Neuntel des Sudokus

zu finden, in welche diese Ziffern platziert werden kann. Um zu verhindern dass der Algorithmus an einer Stelle stecken bleibt, wird Backtracking verwendet. Sollte die Ausführung an einem Punkt kommen, an dem sie kein mögliches Feld mehr findet, wird das vorläufige Feld bis zu einem vorherig gespeichertem Meilenstein zurückgesetzt. Je öfters der Algorithmus feststeckt, ohne dass der nächste Meilenstein erreicht wird, wird das Sudoku um ein immer größeres Stück zurückgesetzt.

Beim Füllen des Sudokus werden dabei nach jeder, in jedem Neuntel erfolgreich eingetragener Ziffer ein Meilenstein gesetzt.

3.1.2 Leeren des Sudokus

Sobald das Sudoku vollständig gefüllt ist, werden Stück für Stück wieder Ziffern herausgestrichen. Dabei muss nach jedem Streichen überprüft werden, ob das Sudoku nach eine eindeutige Lösung hat. Wenn das nicht der Fall ist, wird, ähnlich wie beim Füllen, das Sudoku um einen Schritt zurückgesetzt.

3.1.3 Lösungsalgorithmus

Um zu überprüfen ob das Sudoku eine eindeutige Lösung hat, muss jeder mögliche Lösungpfad verfolgt werden. Dabei wird ein rekursiver Algorithmus verwendet, welcher bei jedem Schritt all momentan potentiell eintragebaren Ziffern einträgt und den nächsten Rekursionsschritt aufruft. Die Abbruchkondition wird dabei durch die vollständige Lösung oder ein Sudoku ohne weitere möglichen Schritte dargestellt. Nachdem alle möglichen Schritte durchgeführt werden, wird überprüft ob alle zurückgegebenen Lösungen identisch sind. Sollte das nicht der Fall sein, besitzt das Sudoku keine eindeutige Lösung mehr.

3.2 Lösungsstrategien