Oberschule an der Ronzelenstraße
Mathematik und *Informatik*

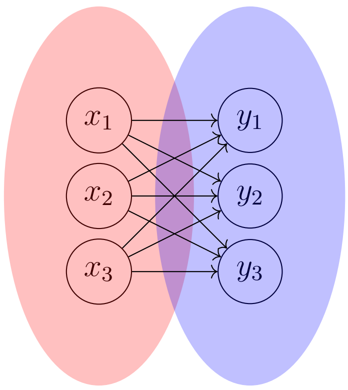
Projektarbeit im Rahmen der Qualifikationsphase

Zeitliche Optimierung der Klausurenplanung an Schulen mithilfe von Graphentheorie — eine Web-App-Entwicklung

---

2024/2025

Julius Backes | Tom Kurzke



Betreuende & Prüfende Lehrer:

Hr. V. Wolff (Mathematik LK) & Hr. O. Huras (Informatik GK)

Inhaltsverzeichnis

1 Einleitung

Die Erstellung eines ausbalancierten Klausurenplans stellt eine große Herausforderung dar, da verschiedene Anforderungen berücksichtigt werden müssen. In diesem Kapitel gehen wir auf die Problemstellung, die Motivation für die Arbeit und die allgemeinen Ziele des Projektes ein.

1.1 Problemstellung

Die Planung von Klausuren in der Oberstufe ist aufgrund verschiedener Faktoren äußerst komplex. Die große Anzahl von Fächern und Kursen erschwert die praktische Planung. Dies führt häufig zu einer unausgewogenen Verteilung der Klausuren. Ein fertiger Klausurenplan muss dabei bestimmten Regeln folgen. Unter anderem darf ein Schüler nicht mehr als eine Klausur pro Tag und maximal drei Klausuren pro Woche schreiben. Im Idealfall sollte ein Schüler sogar weniger als drei Klausuren pro Woche schreiben.

1.2 Motivation

Wir selbst haben in den letzten Jahren während unserer Zeit in der Oberstufe der Oberschule an der Ronzelenstraße erfahren, wie schwierig es ist, einen ausgewogenen Klausurenplan zu erstellen. Die Erstellung ist für die zuständigen Lehrkräfte mit einem erheblichen Zeitaufwand verbunden. So kommt es häufiger vor, dass der Klausurenplan erst eine Woche vor Beginn der Klausurenphase veröffentlicht wird. Dies kann sowohl für die Lehrkräfte als auch für die Schüler problematisch sein, da beide Parteien genügend Zeit benötigen, um sich auf eine Klausur vorzubereiten; die Lehrkräfte müssen bis zum Klausurtermin genügend Inhalte präsentiert haben und die Schüler müssen genügend Zeit zum Lernen haben. Dies ist nur mit ausreichendem Vorlauf möglich. Daher ist es wichtig, dass die Termine rechtzeitig bekannt sind.

1.3 Ziel der Arbeit

Um die o. g . Probleme in Zukunft zu vermeiden und den Prozess der Klausurplanung zu optimieren, entwickeln wir im Rahmen dieser Projektarbeit eine Webapplikation. Ziel ist die einfache Erstellung eines Klausurenplans, an den folgende Anforderungen gestellt werden.

1. Kurse, in denen zwei Klausuren pro Halbjahr vorgesehen sind, werden entsprechend umgesetzt.
2. Ein Schüler schreibt maximal eine Klausur pro Tag.
3. Ein Schüler schreibt maximal drei Klausuren pro Woche.

Zur einfachen Bedienung wird eine vom Benutzer hochgeladene Excel-Datei, die die Kurse und Schüler enthält, verarbeitet, um automatisch einen Klausurenplan zu erstellen.

2 Grundlagen der Graphentheorie

Im vorliegenden Kapitel werden grundlegende Konzepte der Graphentheorie eingeführt. Diese sind essenziell, um ein tiefes Verständnis von der Funktionsweise der finalen Anwendung zu entwickeln.

Die Graphentheorie stellt einen Teilbereich der Mathematik dar, der sich mit der Untersuchung von Graphen befasst. Die Graphentheorie stellt eine wesentliche Grundlage zur Erstellung und Analyse mathematischer Modelle dar und findet darüber hinaus Anwendung bei der Lösung von Optimierungsproblemen.

2.1 Grundbegriffe

Ein Graph aus der Graphentheorie unterscheidet sich signifikant von einem "typischen"Graphen, wie man aus der Schulmathematik und aus der Analysis kennt.

Ein graphentheoretischer Graph G besteht aus einer Menge von Knoten (engl. vertices), die durch Kanten (engl. edges) miteinander verbunden sein können. Diese Knoten können gar nicht oder mehrfach mit anderen Knoten verbunden sein. Wenn ein Knoten mit sich selbst verbunden ist, spricht man von einer Schleife.

Ein Graph G wird formal als ein Paar $G = (V(G), E(G))$ definiert, wobei $V(G)$ die Menge der Knoten darstellt, beispielsweise $\{1, 8, 7\}$ oder $\{A, B, C\}$, und E_G die Menge der Kanten ist, also Verbindungen zwischen jeweils zwei Knoten. Für die Kantenmenge $E(G)$ gilt:

$$E(G) = \{\{v_x, v_y\} : \ v_x, v_y \in V(G) \land v_x \neq v_y\}$$

Die Einschränkung $v_x \neq v_y$ ist optional. Sie gibt an, dass es keine Schleifen im Graphen geben darf. (Diestel, 2017, S. 2). In dieser Arbeit werden Schleifen nicht genutzt. Zwei Knoten a und b nennt man benachbart bzw. adjazent (engl. adjacent) wenn $\{a, b\} \in E(G)$ gilt. Um die Darstellung zu vereinfachen, definieren wir $E(G) = \{\{A, B\}, \{A, C\}\}$ als $E_G = \{AB, AC\}$ (Diestel, 2017, S. 3).

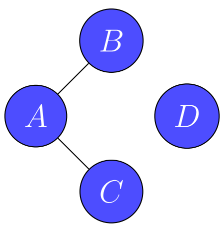


Abbildung 1: Ein einfacher ungerichteter Graph G mit $V(G) = \{A, B, C, D\}$ und $E(G) = \{AB, AC\}$

Die Anzahl der Kanten an einem Knoten v wird durch den Grad (eng. degree) des Knotens beschrieben. Ein Knoten, der mit n-vielen Knoten verbunden ist, hat den Grad n. Es gilt

$$\deg_G(v) = |E(v)|$$

Die Knoten werden in schwach verzweigte und stark verzweigte Knoten unterteilt. Ein Knoten mit z.B. acht verbundenen Knoten ist stärker verzweigt als ein Knoten mit zwei verbundenen Knoten (Diestel, 2017, S. 5).

Graphen lassen sich in zwei Arten unterscheiden: gerichtete Graphen und ungerichtete Graphen. Ein ungerichteter Graph zeichnet sich dadurch aus, dass die Kanten zwischen den Knoten keine Richtung aufweisen (vgl. Abb. 1). Im Unterschied dazu besteht die Kantemenge $E(G)$ eines gerichteten Graphen aus einer Menge von geordneten Paaren anstelle einer Menge von Mengen mit Knoten. Sie wird mit

$$E(G) = \{(v_x, v_y) \colon v_x, v_y \in V(G) \land v_x \neq v_y\}$$

definiert. Die Kanten eines gerichteten Graphen werden i. d. R. mit einem Pfeil dargestellt (Diestel, 2017, S. 30).

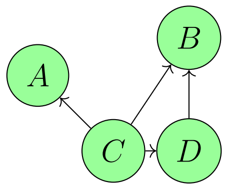


Abbildung 2: Ein einfacher gerichteter Graph
mit $V(G) = \{A, B, C, D\}$ und $E(G) = \{CA, CB, CD, DB\}$

Sei $N_G(v)$ die Menge aller adjazenten Knoten eines Knoten v in einem Graphen G, auch Nachbarn (eng. neighbours) von v genannt. (Diestel, 2017, S. 5)

$$N_G(v) \subseteq V(G)$$

$$N_G(v) = \{w \in V(G) \colon \{v, w\} \subseteq E(G)\}$$

2.2 Graphenfärbung

Die Knotenfärbung (eng. Graph Coloring) stellt eine Methode zur Färbung von Knoten in einem Graphen G dar. Es wird demnach gefordert, dass keine zwei adjazenten Knoten die gleiche Farbe aufweisen. Die Farben werden i. d. R. als Buchstaben oder Zahlen dargestellt, wobei sie Gruppen oder Zuständen zugeordnet werden. Ziel der Färbung ist es, die kleinstmögliche Anzahl von Farben zu bestimmen. Diese Anzahl wird als chromatische Zahl des Graphen bezeichnet.

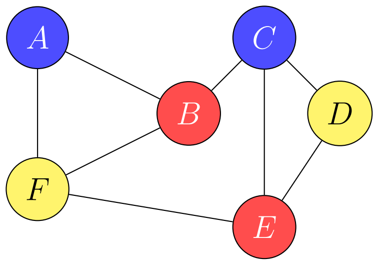
Des Weiteren findet die Färbung Anwendung in der Stundenplanerstellung sowie der Färbung von Karten, beispielsweise dem sogenannten Vier-Farben-Satz („Vier-Farben-Satz“, 2017). Ein weiteres Anwendungsgebiet stellt die Klausurenplanerstellung dar, wie sie auch in dieser Arbeit erfolgt.

Mathematisch wird das Graph Coloring als Abbildung c von der Menge $V(G)$ als Knoten auf eine Menge $C(G)$ als Farben für einen Graphen G definiert (Diestel, 2017, S. 121).

$$c(v) : V(G) \to C(G)$$

$$\forall \{v_x, v_y\} \in E(G) \colon \; c(v_x) \overset{!}{\neq} c(v_y) \tag{1}$$

Mit (1) wird gewährleistet, das zwei adjazente Knoten nicht mit derselben Farbe gefärbt werden.



$$\begin{aligned} V(G) &= \{A, B, C, D, E, F\} \\ E(G) &= \{AB, BC, CD, CE, DE, EF, FA, FB\} \\ C &= \{\bullet, \bullet, \bullet\} \end{aligned}$$

Abbildung 3: Gefärbter Graph G

Ein wichtiges Konzept in der Graphenfärbung ist der **Sättigungsgrad** (eng. degree of saturation). Für einen gegebenen Knoten v beschreibt der Sättigungsgrad $\mathrm{sat}_G(v)$ die Anzahl der verschiedenen Farben, die in der Menge der Nachbarn vorkommen(Lewis, 2021, S. 39). Formal gilt:

$$\mathrm{sat}_G(v) = |\{C_G(w) \colon w \in N_G(v) \;\wedge\; w \notin U(G)\}|$$

wobei $C_G(w)$ die Farbe des Knotens w angibt und $U(G)$ die Menge der ungefärbten Knoten. Für $U(G)$ gilt:

$$U(G) \coloneqq \{v \in V(G) \colon \; C_G(v) = \varnothing\}$$

Die Funktionen zur Bestimmung des maximalen Sättigungsgrads und des maximalen Knotengrads können wie folgt definiert werden:

$$\begin{aligned} \max \mathrm{sat}_G(v) &= \max(\{\mathrm{sat}_G(v) \colon \; v \in V(G) \setminus U(G)\}) \\ \max \deg_G(v) &= \max(\{\deg_G(v) \colon \; v \in V(G)\}) \end{aligned}$$

3 Klausurenplanung als Optimierungsproblem

3.1 Modellierung

3.2 Zielsetzungen

4 Entwicklung der Web-Applikation

4.1 Anforderungen und Spezifikationen

4.1.1 Funktionale Anforderungen

4.1.2 Nicht-funktionale Anforderungen

4.2 Systemarchitektur

4.2.1 Frontend-Design

4.2.2 Backend-Design

4.2.3 Technologie-Stack

4.3 Implementierung

4.3.1 Algorithmische Umsetzung

4.3.2 Benutzeroberfläche

5 Mathematische Analyse der Optimierung

5.1 Bewertung der Lösungsqualität

5.2 Effizienz und Komplexität

6 Evaluation und Tests

7 Zusammenfassung und Ausblick

7.1 Zusammenfassung

7.2 Ausblick

8 Literaturverzeichnis

Diestel, R. (2017, Februar). *Graphentheorie*. Springer Verlag GmbH, Heidelberg.

Lewis, R. (2021). *Guide to graph colouring*. Springer Verlag GmbH, Heidelberg.

Vier-Farben-Satz [Letzter Zugriff am 30. November 2024]. (2017, Juli). Springer Verlag GmbH. https://www.spektrum.de/lexikon/mathematik/vier-farben-satz/10765