technische universität
dortmund

Masterarbeit

Effiziente Berechnung von K_5-Minoren in Graphen

Julian Sauer

27. September 2019

Betreuer:
Prof. Dr. Petra Mutzel
Prof. Dr. Jens Schmidt

Fakultät für Informatik
Algorithm Engineering (LS 11)
Technische Universität Dortmund
http://ls11-www.cs.tu-dortmund.de

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Motivation und Hintergrund

Zunächst kann durch die Berechnung von K_5-Minoren entschieden werden, ob ein Graph K_5-Minor frei ist. Das ist insofern interessant, als das einige Algorithmen effizienter auf solchen Graphen ausgeführt werden können. So ist die Berechnung eines *maximalen Schnittes* (*Max-Cut Problem*) - also das Aufteilen der Knoten eines Graphen in zwei Mengen, sodass die Kanten zwischen diesen beiden Mengen in Summe ein maximales Gewicht besitzen - NP-schwer [12]. Allerdings wird etwa in [3] gezeigt, dass es für Graphen ohne K_5-Minor in Polynomialzeit gelöst werden kann. Darüber hinaus können viele kombinatorische Optimierungsprobleme wie quadratische 0-1 Probleme als Max-Cut Probleme formuliert werden [8]. In [11] transformieren Jünger et al. Ising Spin Glass Probleme zu Max-Cut Problemen, um mit Hilfe von einem Branch-and-Cut Algorithmus eine optimale Lösung zu finden. Neben diesem Ansatz können Ising Spin Glass Probleme ebenfalls auf Quantencomputern wie dem D-Wave 2000Q heuristisch, aber dafür schneller gelöst werden. Zur Bewertung ihrer Qualität können sie mit denen von Jünger et al. verglichen werden. Innerhalb von diesem exakten Verfahren stellen die K_5-Minoren die Schwierigkeit des Problems dar. Von daher ist es einerseits interessant für die Berechnung, wenn kein K_5-Minor enthalten ist, sodass das Problem effizient gelöst werden kann. Andererseits können einer oder mehrere gefundene K_5-Minoren benutzt werden, um innerhalb des linearen Programms den Suchraum zu beschränken und die Berechnung zu beschleunigen.

1.2 Aufbau der Arbeit

Kapitel 2

Definitionen

Vorab werden einige Definitionen und Notationen festgelegt, die im Verlauf der Arbeit verwendet werden. Der Algorithmus arbeitet mit einem ungerichteten Graph $G = (V, E)$, wobei E die Menge der Kanten und V die Knotenmenge sei. Eine Kante $e \in E$, die zwei Knoten u und v verbindet, wird durch $e = (u, v)$ angegeben. Ein Pfad $P(u, v)$ verbindet zwei Knoten u und v über eine Folge von Knoten, die adjazent zueinander sind. Für $P(u, v)$ werden u und v als *Endpunkte*, die übrigen Knoten des Pfades als *innere Knoten* bezeichnet. Falls nicht anders angegeben, wird für jedes Knotenpaar in einem Graphen erwartet, dass es durch maximal eine Kante verbunden wird – Mehrfachkanten sind nicht erlaubt. Genauso sind die betrachteten Graphen frei von Schleifen der Form $e = (v, v)$ mit $e \in E$ und $v \in V$.

Die Menge adjazenter Knoten eines Knotens v wird durch $N(v)$ angegeben. Bei einer *Kantenkontraktion* einer Kante $e = (u, v)$ wird ein neuer Knoten w hinzugefügt, sodass $N(w) = N(u) \cup N(v)$ gilt und anschließend e aus dem Graphen entfernt. In Abb. 2.1 ist das Vorgehen skizziert. Analog kann, wie in Abb. 2.2 gezeigt, ein Pfad kontrahiert werden, indem nacheinander je eine Kante des Pfades kontrahiert wird.

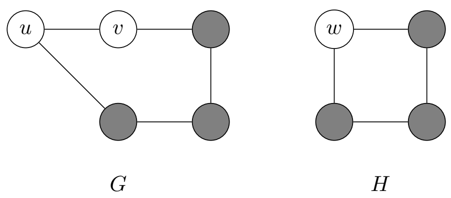


Abbildung 2.1: Die Kante, die u und v in G verbindet, wird kontrahiert, sodass sie in H durch den neuen Knoten w ersetzt wird.

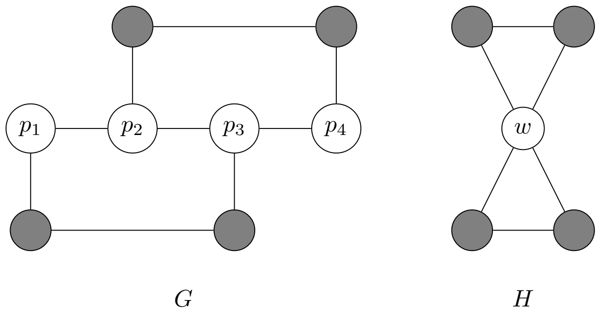


Abbildung 2.2: Der Pfad von p_1 bis p_4 wird kontrahiert. Der neue Knoten w in H enthält alle Nachbarn der Pfadknoten in G.

Ein *Minor* H eines Graphen G bezeichnet einen Graph, der isomorph zu G ist, nachdem eine beliebige Menge an Operationen von Kantenkontraktionen, Kantenentfernungen und Knotenentfernungen durchgeführt wurde. Ein Beispiel dazu findet sich in Abb. 2.3. Jeder Graph ist sein eigener Minor, genauso ist jeder Teilgraph ein gültiger Minor. Dass H ein Minor von G ist, wird dargestellt durch $H \prec_M G$. Ist V eine Menge von Knoten, die einen zusammenhängenden Teilgraphen formen, der durch Kontraktionen durch einen neuen Knoten w ersetzt wurde, dann bezeichnet das *Branch-Set* von w die Menge V. In Abb. 2.3 besteht beispielsweise das Branch-Set von g aus $\{a, b, c\}$, zu f in H_2 gehört die Knotenmenge $\{f\}$ in G.

Für die Knotenmenge $U \in V$ bezeichnet $G - U$ den Teilgraph, der entsteht, wenn alle Knoten aus U mit ihren inzidenten Kanten aus G entfernt werden. Eine *Subdivision* H eines Graphen G enthält alle Knoten aus $G = (V, E)$, sodass jedes inzidente Knotenpaar $\{u, v\} \subseteq V$ in H durch einen Pfad verbunden ist.

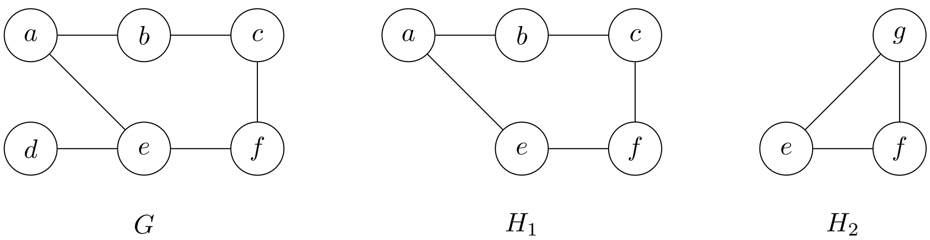


Abbildung 2.3: Ein Graph G mit seinen Minoren H_1 und H_2. Um H_1 zu erhalten, wurde in G die Kante (d, e) und anschließend der Knoten d entfernt. Für H_2 wurden außerdem der Pfad $P(a, c)$ kontrahiert.

Ein Graph wird als *planar* bezeichnet, wenn er sich so in der Ebene einbetten lässt, dass sich keine Kanten kreuzen. Ein K_5 (s. Abb. 2.4) ist ein Graph bestehend aus einer Clique von fünf Knoten. Ein $K_{3,3}$ (s. Abb. 2.5) ist ein vollständig biparti-

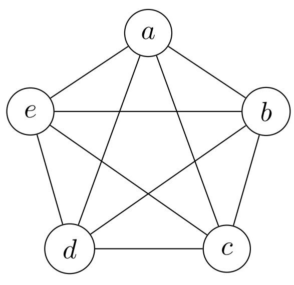


Abbildung 2.4: Der Graph K_5.

ter Graph mit sechs Knoten, sodass jede Bipartition drei Knoten enthält. Er lässt sich also in zwei Knotenmengen unterteilen (im Folgenden als rote und blaue Menge bezeichnet), sodass alle Knoten der einen Menge zu allen Knoten der anderen Menge benachbart sind. Nach dem Satz von Kuratowski ist ein Graph genau dann planar, wenn er keine K_5- oder $K_{3,3}$-Subdivision als Teilgraph beinhaltet. Eine alternative Formulierung von Wagner [14] sagt aus, dass ein Graph genau dann planar ist, wenn er keinen K_5-Minor oder $K_{3,3}$-Minor enthält [7].

Als nächstes wird ein $K_{3,3}$ genauer betrachtet. Sei dessen rote Knotenmenge $R = \{a, b, c\}$ und blaue $B = \{x, y, z\}$. Diese sechs Knoten werden in einer $K_{3,3}$-Subdivision (analog in einem $K_{3,3}$-Minor) H *Branch-End* genannt und zeichnen sich dadurch aus, dass sie als einzige Knoten in H den Grad 3 haben. Ein *Branch-Path* in H ist ein Pfad, der zwei Branch-Ends verbindet, beispielsweise $P(a, x)$. Ein *Branch-Fan* bezieht sich immer auf eines der Branch-Ends und wird z. B. für a als $F(a)$ geschrieben. Bezeichnet werden dadurch alle Pfade, die zu einem anderen Branch-End führen – für a also die Pfade $P(a, x)$, $P(a, y)$ und $P(a, z)$.

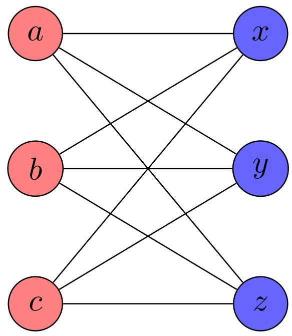


Abbildung 2.5: Der Graph $K_{3,3}$.

Der Graph W bezeichnet einen speziellen Graph, der aus acht Knoten besteht. Seine äußeren Kanten bilden einen Kreis, außerdem sind die Knoten jeweils adjazent zu den gegenüberliegenden. Eine Darstellung findet sich links in Abb. 2.6. Er enthält einen $K_{3,3}$ als Minor (in der Abbildung rechts angedeutet), jedoch keinen K_5. Als M wird ein Graph bezeichnet, der einen $K_{3,3}$ als Minor enthält, jedoch einen zusätzlichen Knoten und zwei zusätzliche Kanten hat. Er ist interessant, weil er, wie in Abb. 2.7 zu sehen, neben einem $K_{3,3}$-Minor auch einen K_5-Minor enthält.

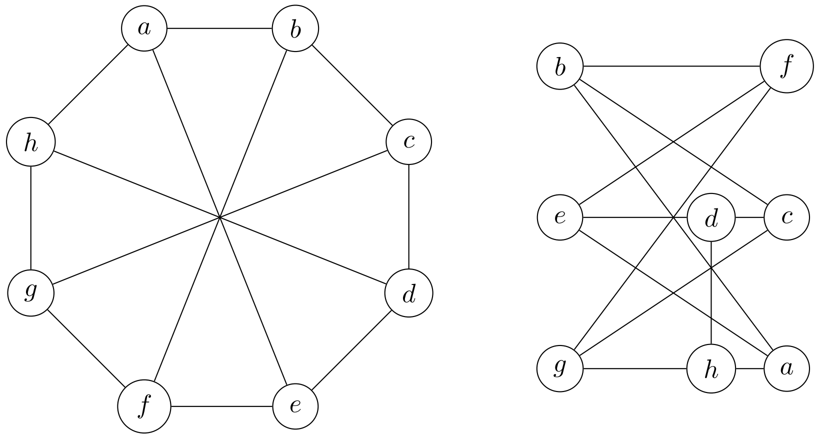
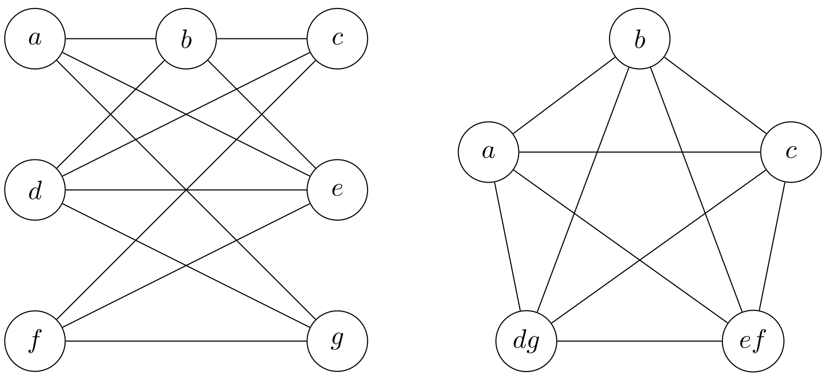


Abbildung 2.6: Der Graph W, links in seiner üblichen Darstellung, rechts mit angedeutetem $K_{3,3}$-Minor.



M K_5-Minor

Abbildung 2.7: Der Graph M sowie ein K_5-Minor aus M.

Als i-*Separator* wird eine Menge S bestehend aus i Knoten in einem zusammenhängenden Graphen G bezeichnet, sodass $G-S$ nicht mehr zusammenhängend ist. Ein Graph ist k-*zusammenhängend*, falls er keinen $(k-1)$-Separator enthält. Ein (i,j)-*Separator* ist ein i-Separator, sodass $G-S$ aus mindestens j Zusammenhangskomponenten besteht. In Abb. 2.8 wird ein $(3,3)$-Separator im oberen Graphen gezeigt, der aus den Knoten $\{c_1, c_2, c_3\}$ besteht. Sind $C \subset V$ die i Knoten, die zu einem (i,j)-Separator in G gehören, wird der Graph durch $G-C$ zunächst in j Zusammenhangskomponenten $Z_1, ..., Z_j$ zerlegt. Dann wird für jedes Z_k mit $1 \leq k \leq j$ ein neuer Graph A_k erzeugt, der aus dem Teilgraphen $Z_k \cup C$ besteht – die Knoten von C in A_k werden paarweise durch Kanten verbunden, falls sie noch nicht adjazent zueinander sind. Jeder der resultierenden Graphen wird als *augmentierte Komponente* des Eingabegraphen G bezeichnet, die durch den Separator C definiert wird.

Ist V eine Menge von Knoten oder Kanten, dann bezeichnet $|V|$ die Kardinalität von V. Sei A_1 ein Graph mit einer Clique C_1 und A_2 ein Graph mit einer Clique C_2, sodass $|C_1| = |C_2| = i$ ist. Dann bezeichnet eine Cliquen-Summe das paarweise Zusammenfügen von je einem Knoten $c_1 \in C_1$ mit einem $c_2 \in C_2$, sodass ein neuer Graph G entsteht, der sowohl einen A_1-, als auch einen A_2-Minor enthält. Innerhalb einer solchen Operation dürfen zusätzlich beliebige Kanten in G gelöscht werden, die zwei Knoten in der Clique verbinden. Dadurch ist es möglich, einen Graphen G durch einen Separator in augmentierte Komponente aufzuteilen und anschließend durch eine Cliquen-Summe wieder G zu erhalten. Dieses Vorgehen wird in Abb. 2.8 gezeigt, dabei wird der obere Graph in die unten abgebildeten augmentierten Komponenten aufgeteilt bzw. von unten nach oben werden drei Graphen durch eine Cliquen-Summe zusammengefügt.

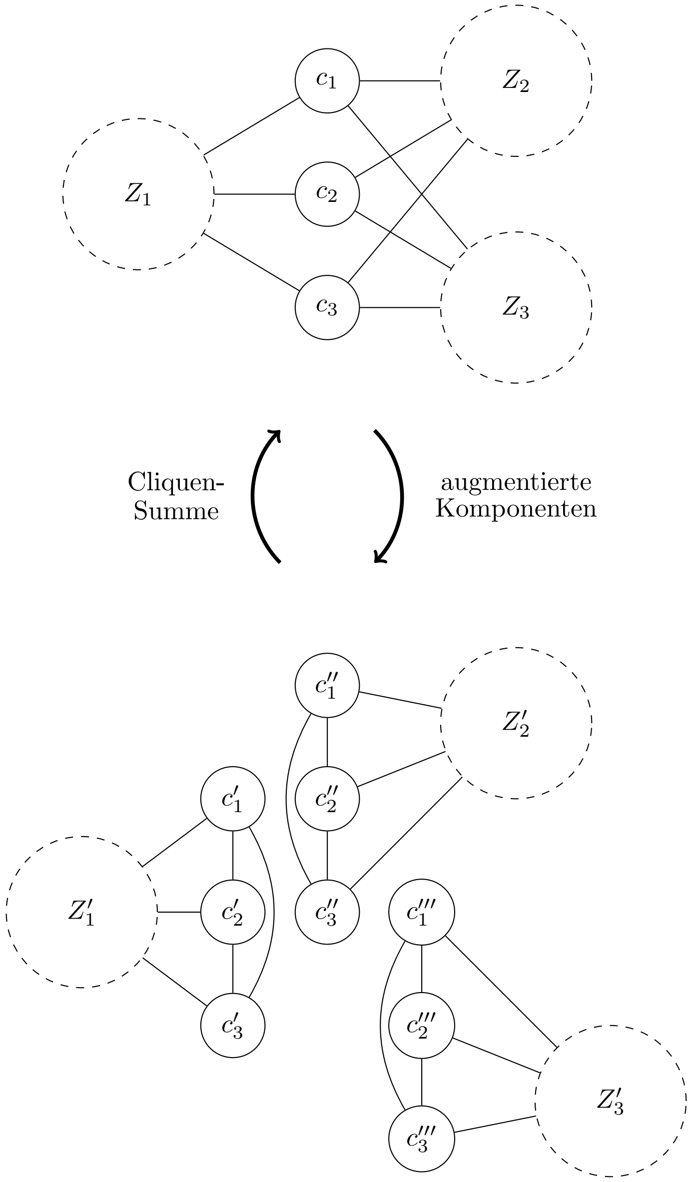


Abbildung 2.8: Der obere Graph wird in drei augmentierte Komponenten durch den $(3,3)$-Separator $C = \{c_1, c_2, c_3\}$ geteilt. Alle Z_i und Z'_i stellen Teilgraphen dar, die zur Übersicht zu einem Knoten zusammengefügt wurden. Aus den drei unteren Graphen kann durch die Cliquen-Summe der Cliquen $\{c'_1, c'_2, c'_3\}$ sowie $\{c''_1, c''_2, c''_3\}$ und $\{c'''_1, c'''_2, c'''_3\}$ der oberen Graphen erzeugt werden. Während der Cliquen-Summen Operation dürfen beliebige Kanten, die die Knoten in den Cliquen verbinden, gelöscht werden.

Kapitel 3

Algorithmus von Kezdy und McGuinness

Die Arbeit auf dem in [2] vorgestellten sequenziellen Algorithmus von Kezdy und McGuinness, der im Folgenden erklärt wird. Als Eingabe wird ein ungerichteter Graph ohne Mehrfachkanten oder Schleifen erwartet, ausgegeben wird, ob der Graph einen K_5-Minor hat. Für den Fall, dass einer gefunden wurde, kann zusätzlich ausgegeben werden, welche Knoten den Minor formen. Die Laufzeit liegt in $\mathcal{O}(n^2)$.

Planaritätstests können bereits in linearer Laufzeit entscheiden, ob ein Graph einen K_5- oder $K_{3,3}$-Minor enthält [10]. Es muss folglich der Fall behandelt werden, in dem der Test stoppt, weil er einen $K_{3,3}$-Minor gefunden hat. Denn dann kann zunächst keine Aussage darüber getroffen werden, ob zusätzlich ein K_5-Minor enthalten ist. Als Lösung bildet der Algorithmus von Kezdy und McGuinness in einem 3-zusammenhängenden Graphen solange augmentierte Komponenten mit $(3, 3)$-Separatoren, die Teil von $K_{3,3}$-Minoren sein können, bis entweder alle Komponenten planar bzw. isomorph zu W sind oder aber ein K_5-Minor gefunden wurde.

3.1 Behandlung von $K_{3,3}$-Minoren

Die folgenden Theoreme, Lemmata und meisten Beweise sind [2] entnommen.

Um das zentrale Theorem aus [2], welches den $K_{3,3}$-Minor untersucht, zu erklären, wird zunächst die Gültigkeit augmentierter Komponenten behandelt:

3.1.1 Theorem. *Für $k \geq 3$: Sei G ein k-zusammenhängender Graph und C ein k-Separator in G mit $k \geq 3$. Jede durch C definierte augmentierte Komponente ist ein Minor von G genau dann, wenn es entweder mindestens k Komponenten sind oder mindestens $k - 1$, von denen mindestens zwei jeweils aus mehr als einem Knoten bestehen. [2]*

Beweis. Seien $c_1, c_2, ..., c_k$ die Knoten von C und $Z = \{Z_1, Z_2, ..., Z_k\}$ beziehungsweise $Z = \{Z_1, Z_2, ..., Z_{k-1}\}$ die Zusammenhangskomponenten, die durch $G - C$ entstehen. Die zugehörigen augmentierten Komponenten seien $A_1, A_2, ..., A_k$ bzw. $A_1, A_2, ..., A_{k-1}$. Betrachtet wird eine beliebige dieser augmentierten Komponenten A_i. Der Definition der augmentierten Komponenten nach finden sich bereits alle Knoten von A_i in G wieder. Weiterhin enthält G mindestens alle Kanten in $A_i - C$ sowie die verbindenden Kanten zwischen A_i und C. Jedoch bilden in A_i die Knoten von C eine Clique, es existieren also ggf. Kanten zwischen den Knoten von C in A_i, die es nicht in G gibt. Es bleibt zu zeigen, dass die Kanten, die für diese Clique in A_i nötig sind, durch Kantenkontraktionen in G erzeugt werden können. Dadurch, dass G k-zusammenhängend ist, besitzt jede Zusammenhangskomponente von $G - C$ Kanten zu allen $c_1, c_2, ..., c_k$. Würde eine Kante zu einem Knoten c_j mit $1 \leq j \leq k$ fehlen, wäre ein $(k-1)$-Separator bestehend aus $C - c_j$ möglich, was im Widerspruch zu dem k-Zusammenhang stehen würde. Das Theorem unterscheided nun zwei Fälle, um die fehlenden Kanten bereitstellen zu können:

1. Es existieren k Zusammenhangskomponenten. Wird A_i betrachtet, kommen die Knoten in $Z - Z_i$ infrage, um durch Kantenkontraktionen die fehlenden Kanten für die Clique von C in A_i zu erzeugen. Um die Kanten von C in A_i in G zu erzeugen, kann zunächst der Pfad, der c_1 mit Z_1 verbindet, kontrahiert werden. Anschließend ist c_1 mit allen Knoten in C verbunden. Dies kann analog für alle Knoten in C und den entsprechenden Zusammenhangskomponenten durchgeführt werden, außer für c_i, da A_i der gesuchte Minor ist. Allerdings ist c_i aufgrund des k-Zusammenhangs mit allen anderen Zusammenhangskomponenten verbunden, sodass C nach den beschriebenen Kontraktionen eine Clique bildet.

2. Es existieren $k - 1$ Komponenten von denen mindestens zwei aus mehr als einem Knoten bestehen. Analog zum vorherigen Fall können die Pfade zwischen den Knoten von C und den Zusammenhangskomponenten A kontrahiert werden. Es fehlt jedoch ein Pfad, da eine Zusammenhangskomponente weniger vorliegt. Es gibt mindestens eine Zusammenhangskomponente aus $Z - Z_i$, die aus zwei oder mehr Knoten besteht. Da der Graph k-zusammenhängend ist, sind mindestens zwei dieser Knoten mit allen in C verbunden, sodass sie durch Kontraktionen mit zwei unterschiedlichen Knoten aus C genutzt werden können, um die gesuchte Clique zu erzeugen.

Ein Beispiel, in dem die augmentierten Komponenten keine Minoren sind, findet sich in Abb. 3.1. □

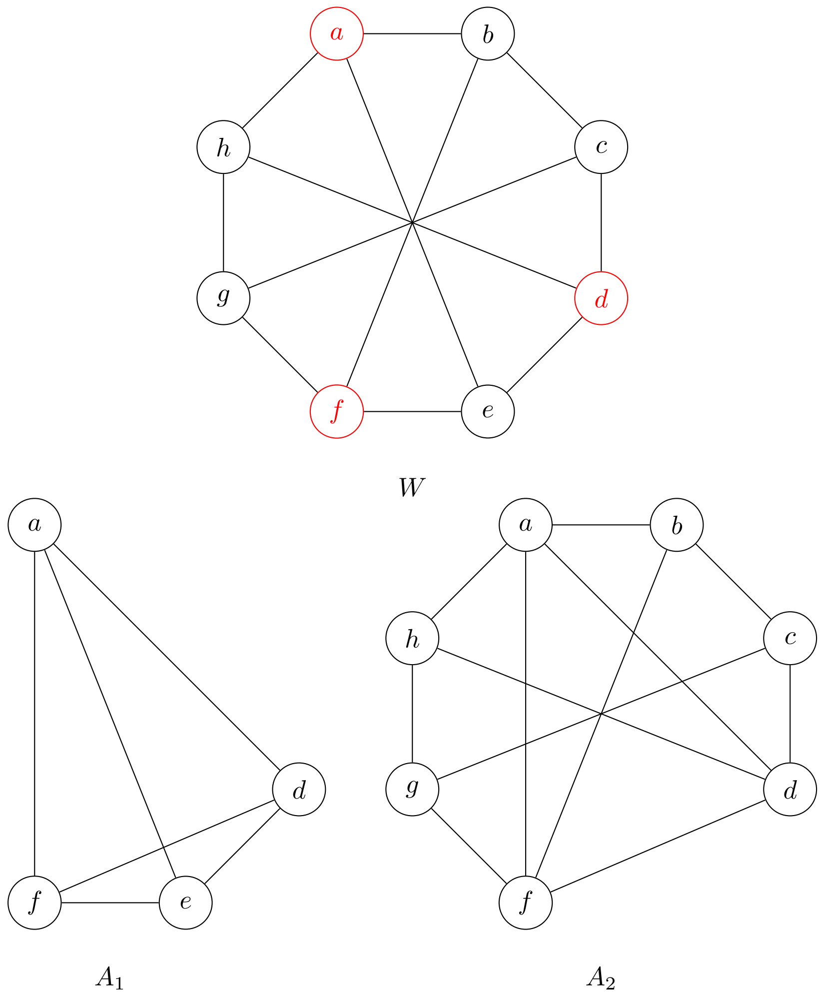


Abbildung 3.1: Gegenbeispiel zu Theorem 3.1.1 für $k = 3$ mit $C = \{a, d, f\}$ in W, wodurch die $k-1$ Komponenten A_1 und A_2 entstehen Es entstehen durch $W - C$ zwei Zusammenhangskomponenten, die die Knoten $\{e\}$ und $\{b, c, g, h\}$ enthalten. Entgegen der Voraussetzungen des Theorems gibt es daher nicht genügend Zusammenhangskomponenten bzw. keine zwei Zusammenhangskomponenten, die je mehr als zwei Knoten enthalten: Die Komponente A_1 ist zwar ein gültiger Minor, da sie etwa durch die Kontraktionen der Kanten $(a, b), (a, h), (f, g)$ und (c, d) aus W erzeugt werden kann. A_2 dagegen kann aber nicht durch Kontraktionen aus W erzeugt werden – wird beispielsweise die Kante (d, e) in W kontrahiert, fehlt die Kante (a, f) in A_2. Analog kann e mit keiner seiner inzidenten Kanten kontrahiert werden, um einen zu A_2 isomorphen Graphen zu erhalten.

Als nächstes stellen Kezdy und McGuinness fest, dass ein Graph in augmentierte Komponenten zerlegt werden kann, wenn ein $(3, 3)$-Separators in ihm gefunden wurde:

3.1.2 Theorem. *Sei G ein 3-zusammenhängender Graph mit einem $(3, 3)$-Separator C. G hat einen K_5-Minor genau dann, wenn eine der durch C definierten augmentierten Komponenten einen K_5-Minor enthält. [2]*

Beweis. Zunächst kann festgestellt werden, dass, falls eine der augmentierten Komponenten einen K_5-Minor enthält, dieser laut Theorem 3.1.1 auch ein Minor von G ist. Es bleibt zu zeigen, dass sich ein K_5-Minor nicht auf zwei augmentierte Komponenten erstreckt, sondern sich ausschließlich in einer befindet. Angenommen es gilt $K_5 \prec_M G$ und zwei der Branch-Sets, die den K_5-Minor bilden, befinden sich jeweils vollständig in unterschiedlichen Zusammenhangskomponenten. In diesem Fall wäre C ein 3-Separator in dem gefundenen Minor, was im Widerspruch zu dem 4-Zusammenhang des K_5 steht. [2] □

Das zentrale Theorem ist darauf zurückzuführen, dass jeder Graph ohne K_5-Minor durch Cliquen-Summen von Teilgraphen, die planar oder isomorph zu W sind, gebildet werden kann [14]. Eine genauere Betrachtung findet in Kapitel 4 statt. Es kann benutzt werden, um einen Graph rekursiv in augmentierte Komponenten aufzuteilen, bis der Planaritätstest entweder einen K_5-Minor findet oder alle augmentierten Komponenten planar bzw. isomorph zu W sind.

3.1.3 Theorem. *Sei G ein 3-zusammenhängender Graph, der ein $K_{3,3}$-Subdivision S enthält, dessen Branch-Ends in die rote Knotenmenge $R = \{a, b, c\}$ und blaue $B = \{x, y, z\}$ unterteilt sind. Mindestens eine der folgenden Bedingungen trifft auf G zu:*

1. *G enthält einen K_5-Minor.*
2. *G ist isomorph zu W.*
3. *$\{a, b, c\}$ bilden einen 3-Separator, sodass $\{x, y, z\}$ in separaten Komponenten liegen.*
4. *$\{x, y, z\}$ bilden einen 3-Separator, sodass $\{a, b, c\}$ in separaten Komponenten liegen.*

[2]

Durch die Theoreme 3.1.1 und 3.1.2 wurde gezeigt, dass der Graph in den Fällen 3 und 4 in augmentierte Komponenten zerlegt und darauf ein Planaritätstest ausgeführt werden kann. Anschließend stellen die Autoren einige Lemmata auf, mit denen untersucht wird, ob S einen K_5-Minor enthält – also ob Bedingung 1 zutrifft.

3.1.4 Lemma. *Sei G ein 3-zusammenhängender Graph und S ein $K_{3,3}$-Subdivision in G. Enthält G drei Pfade von einem Knoten $w \in G - S$ zu drei nicht im selben Branch-Fan liegenden Knoten in S, enthält G einen K_5-Minor. [2]*

Beweis. Seien t, u, v die drei Endpunkte der Pfade in S. Mindestens einer von ihnen ist ein innerer Knoten, da sonst alle im selben Branch-Fan liegen würden. Sei o. B. d. A. t ein solcher innerer Knoten auf dem Pfad $P(a, x)$. Folglich können u und v nicht beide in $F(a)$ oder $F(x)$ liegen, sonst lägen alle drei im gleichen Branch-Fan. Es wird unterschieden, wo genau die drei Knoten in S liegen – die Fälle sind in den Abbildungen 3.2, 3.3 und 3.4 skizziert.

1. u und v sind nicht im gleichen Branch-Fan wie t. Dann müssen u und v ebenfalls innere Knoten sein, im Beispiel auf den Pfaden $P(y, b)$ bzw. $P(z, c)$. Es kann ein M-Minor durch folgende Kontraktionen erzeugt werden: u mit einem der roten und v mit einem der blauen Knoten (analog u mit blau und v mit rot) sowie $P(w, t)$.

2. u oder v liegen auf $P(a, x)$. Sei o. B. d. A. $u \in P(a, x)$. Da t ebenfalls in diesem Pfad liegt, gilt $\{t, u\} \in F(a) \cup F(x)$, sodass v nicht in diesen beiden Branch-Fans liegen kann. Es können t und v getauscht werden, sodass eine Reduktion auf Fall 1 erreicht wird.

3. Entweder u oder v liegen im gleichen Branch-Fan wie t. Sei o. B. d. A. $u \in F(x) - P(a, x)$, im Beispiel auf dem Pfad $P(b, x)$. Es gilt $\{t, u\} \in F(x)$, weshalb v in einem anderen Branch Fan sein muss. Da alle roten Knoten in $F(x)$ liegen, gilt konkreter $v \in (F(y) \cup F(z)) - \{a, b, c\}$ Es können $P(b, u)$ kontrahiert werden sowie je nach Fall entweder $P(v, y)$ oder $P(v, z)$. Wird $P(w, t)$ ebenfalls kontrahiert, entsteht erneut ein M-Minor.

[2] □

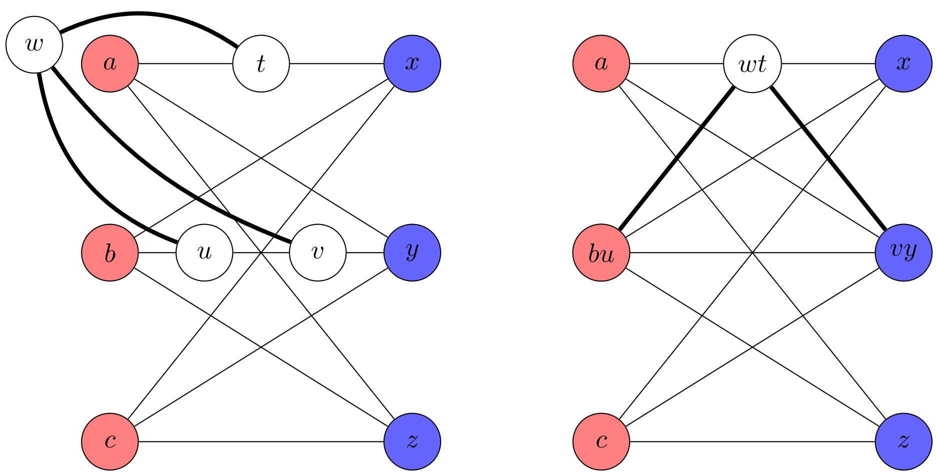


Abbildung 3.2: Beispiel zum ersten Fall von Lemma 3.1.4.

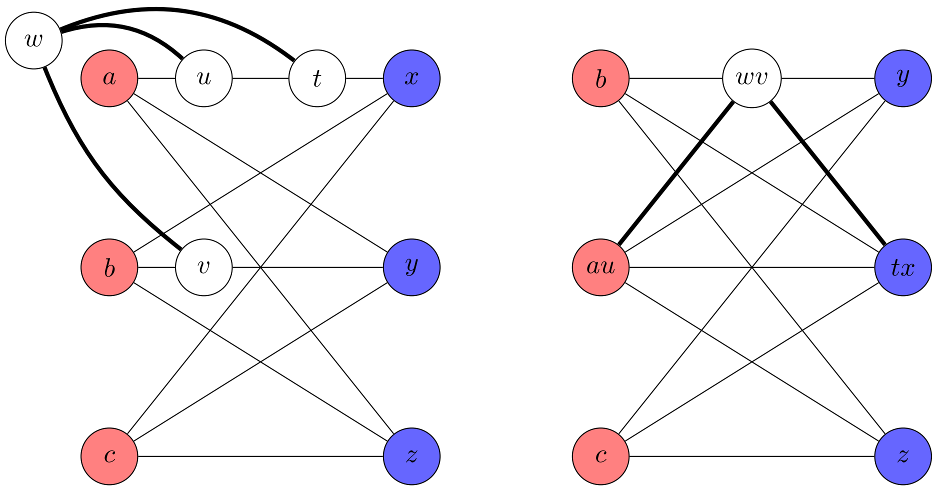


Abbildung 3.3: Beispiel zum zweiten Fall von Lemma 3.1.4.

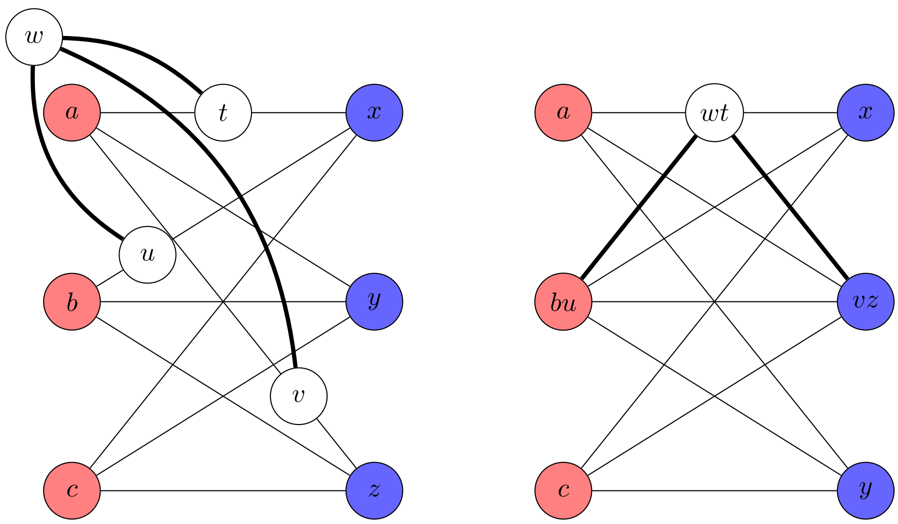


Abbildung 3.4: Beispiel zum dritten Fall von Lemma 3.1.4.

3.1.5 Lemma. *Sei G ein 3-zusammenhängender Graph und S ein $K_{3,3}$-Subdivision in G. Betrachtet wird ein Pfad außerhalb von S, der zwei Endpunkte hat, die in einem roten Branch-Fan, aber nicht im gleichen Branch-Path liegen. Analog dazu wird ein Pfad außerhalb von S gesucht, dessen Endpunkte in einem blauen Branch-Fan liegen, ohne dass diese auf dem gleichen Branch-Path in S liegen. Existieren diese beiden Pfade in G, dann enthält G einen K_5-Minor. [2]*

Beweis. Sei P_1 der Pfad, der zwei Knoten in einem roten Branch-Fan verbindet und P_2 der, der zwei in einem blauen Branch-Fan verbindet. O. B. d. A. hat P_1 Endpunkte in $F(a)$ und P_2 in $F(x)$. Da laut Bedingung die Endpunkte nicht in einem einzelnen Pfad von S

liegen, kann a kein Endpunkt von P_1 und x kein Endpunkt von P_2 sein. Es ergeben sich zwei Fälle:

1. Die beiden Pfade haben keine gemeinsamen Knoten. Da P_1 beide Endpunkte in $F(a)$ hat, liegen diese beiden Endpunkte in zwei unterschiedlichen blauen Branch-Fans. Entsprechend sind die Endpunkte von P_2 in unterschiedlichen roten Branch-Fans. Werden die Endpunkte von P_1 je mit den beiden blauen und die von P_2 mit den beiden roten Knoten von S kontrahiert, entsteht ein K_5-Minor. Abb. 3.5 skizziert den Fall beispielhaft.

2. Die beiden Pfade haben einen gemeinsamen Knoten w. Liegt dieser gemeinsame Knoten außerhalb von S, kann Lemma 3.1.4 angewendet werden, da die Endpunkte der Pfade nicht alle im gleichen Branch-Fan liegen. Liegt w innerhalb von S, ist er ein Endpunkt von P_1 und P_2 und muss auf dem Pfad $P(a, x)$ liegen, da dieser der einzige gemeinsame Pfad ist, siehe links in 3.6. Sei $P_1 = P(w, u)$ und $P_2 = P(w, v)$. Da u nicht in $F(x)$ liegt und v nicht in $F(a)$, gibt es einen Pfad von u zu einem blauen Knoten und von v zu einem roten Knoten, die sich nicht kreuzen und daher kontrahiert werden können. Durch die Kontraktion dieser beiden Pfade entsteht, wie in Abb. 3.6 rechts zu sehen, ein M-Minor.

[2] □

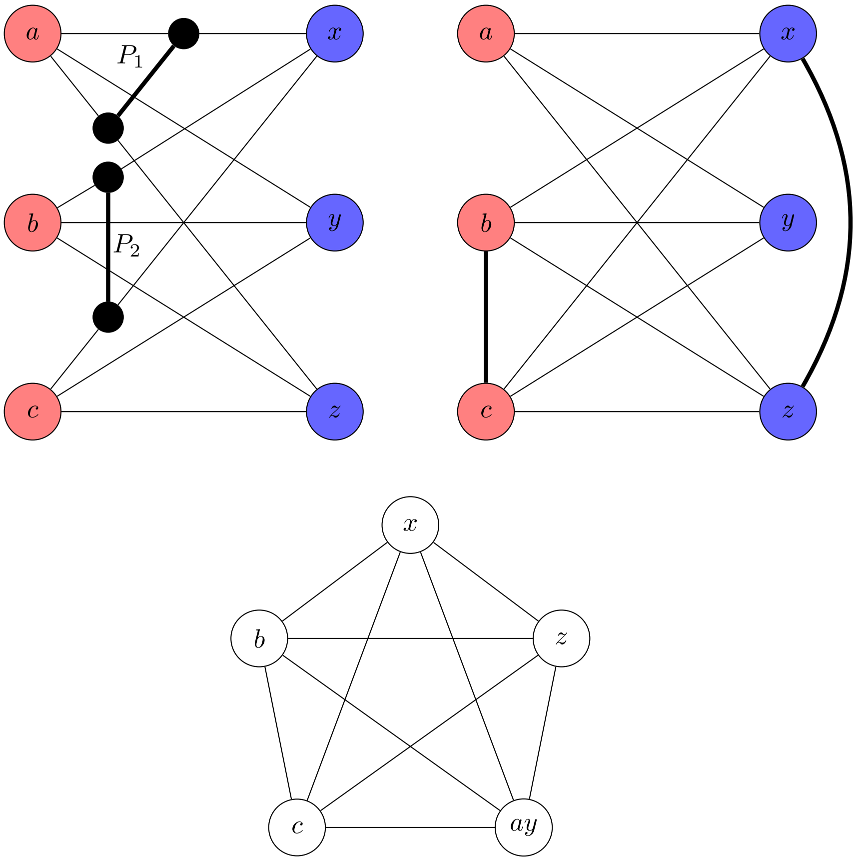


Abbildung 3.5: Beispiel zum ersten Fall von Lemma 3.1.5. Links oben ist S mit zwei zusätzlichen Pfaden aus G abgebildet. P_1 hat beide Endpunkte in $F(a)$ sowie den blauen Branch-Fans $F(x)$ und $F(y)$, mit denen die Endpunkte kontrahiert werden. P_2 hat seine Endpunkte in den roten Branch-Fans $F(b)$ und $F(c)$, mit denen sie kontrahiert werden. Rechts oben ist der dadurch entstehende Minor abgebildet und unten wird der enthaltene K_5-Minor gezeigt.

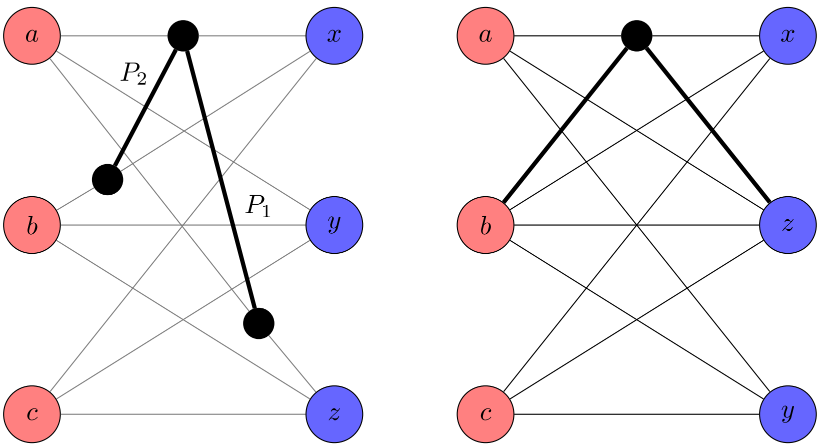


Abbildung 3.6: Beispiel zum zweiten Fall von Lemma 3.1.5. Rechts ist der enthaltene M-Minor abgebildet.

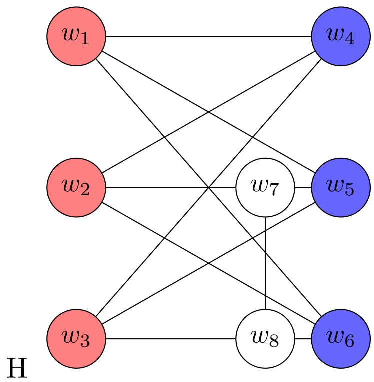


Abbildung 3.7: Darstellung von einem Graph S, der isomorph zu W ist.

3.1.6 Lemma. *Sei G ein 3-zusammenhängender Graph und S ein $K_{3,3}$-Subdivision in G. Betrachtet wird ein Pfad außerhalb von S, der zwei innere Knoten paralleler Pfade in S verbindet, sowie ein Pfad außerhalb von S, dessen Endpunkte nicht beide im gleichen Pfad von S liegen. Bestehen die Endpunkte der beiden Pfade aus mindestens drei unterschiedlichen Knoten in S, enthält G einen K_5-Minor. [2]*

3.1.7 Lemma. *Sei G ein 3-zusammenhängender Graph und S ein $K_{3,3}$-Subdivision in G mit den roten Knoten $R = \{a, b, c\}$ und den blauen Knoten $B = \{x, y, z\}$. Bilden weder R noch B einen 3-Separator, enthält G einen K_5-Minor. [2]*

Beweis. Falls R und B keinen $(3, 3)$-Separator bilden, ist sowohl der Graph $G - R$ als auch $G - B$ zusammenhängend. Sei P_1 ein Pfad, der zwei blaue Branch-Fans in $G - R$ und P_2 einer, der zwei rote Branch-Fans in $G - B$ verbindet. Beide liegen außerhalb von S. Die Endpunkte von P_1 seien u_1 und v_1, die von P_2 seien u_2 und v_2. u_1 und v_1 besitzen jeweils einen Pfad in S zu einem der roten Knoten. Folglich gibt es einen dritten roten Knoten, der keinen solchen Pfad besitzt – u_2 wird so gewählt, dass er in dem Branch-Fan dieses Knotens liegt. Demnach sind u_1, v_1 und u_2 unterschiedliche Knoten. Anschließend kann je nach vorliegendem Fall die Aussage auf eines der vorherigen Lemmata reduziert werden:

1. P_1 oder P_2 verbindet zwei parallele Pfade in S. In dem Fall kann Lemma 3.1.6 angewendet werden und G enthält einen K_5-Minor.

2. Die Endpunkte von P_1 liegen in einem einzelnen roten Branch-Fan – analog liegen die von P_2 in einem blauen. Nach Lemma 3.1.5 enthält G einen K_5-Minor.

[2] □

3.1.8 Lemma. *Sei G ein 3-zusammenhängender Graph mit einer W-Subdivision. Ist G nicht isomorph zu W, enthält G einen K_5-Minor. [2]*

Beweis. G enthält mindestens acht Knoten, die S formen. Sei $G = (V_G, E_G)$ der Graph und $S = (V_S, E_S)$ der Teilgraph mit $V_S \subseteq V_G$ und $E_S \subseteq E_G$, der die W-Subdivision formt. Da W keinen K_5-Minor enthält, besitzt jeder Graph, der isomorph zu W ist, ebenfalls keinen. Es wird deshalb angenommen, dass G nicht isomorph zu W ist. Damit G nicht isomorph zu W ist trifft mindestens einer der Punkte auf G zu:

1. Es gibt eine weitere Kante $e = (w_i, w_j) \in E_G$ mit $\{w_i, w_j\} \in V_S$, aber $e \notin E_G$. Seien die Knoten $w_1, ..., w_8 \in V_S$ o. B. d. A. so gewählt, wie in Abb. 3.7 gezeigt. Seien $R = \{w_1, w_2, w_3\}$ als rote und $B = \{w_4, w_5, w_6\}$ als blaue Knoten bezeichnet. R bildet keinen $(3, 3)$-Separator in S, da es den Pfad $P(w_5, w_6)$ in $S - R$ gibt. Analog ist B kein $(3, 3)$-Separator wegen dem Pfad $P(w_2, w_3) \in S - B$. Verbindet die zusätzliche Kante zwei Knoten in R, dann ist e eine Kante, deren Endpunkte in einem blauen Branch-Fan, aber nicht auf dem gleichen Branch-Path liegen. Außerdem gibt es durch den Pfad $P(w_5, w_6)$ eine Kante, die zwei Endpunkte in einem roten Branch-Fan verbindet, die nicht auf dem gleichen Branch-Path liegen. Das Problem kann zu Lemma 3.1.5 reduziert werden (Fall **??**). Die Reduktion kann analog für $P(w_2, w_3)$ angewendet werden, wenn e zwei blaue Knoten verbindet. Übrig bleibt der Fall, dass ein Endpunkt von e entweder w_7 oder w_8 ist und der andere Endpunkt einer der roten oder einer der blauen Knoten ist. Sei o. B. d. A. $e = (w_1, w_7)$. Dann liegen die Endpunkte von e in einem blauen Branch-Fan $(F(w_5))$, aber nicht dem gleichen Branch-Path. Außerdem verbinden die Endpunkte der Kante (w_7, w_8) zwei parallele Branch-Paths $(P(w_2, w_5)$ und $P(w_3, w_6))$. Damit kann der Fall zu Lemma 3.1.6 reduziert werden.

2. Es gibt einen weiteren Knoten $v \in V_G$ mit $v \notin V_S$. Er hat aufgrund des 3-Zusammenhangs mindestens drei Pfade zu Knoten von S. Seien diese drei Knoten $\{w_1, w_2, w_3\} \in V_S$ beliebig. Da es in W keine Cliquen der Größe 3 gibt, sind mindestens zwei der drei Knoten nicht adjazent. Sei o. B. d. A. $e = (w_1, w_2) \notin E_S$. Da $\{(w_1, v), (v, w_2)\} \in E_G$ gilt, kann eine dieser beiden Kanten kontrahiert werden, sodass durch die jeweils andere w_1 und w_2 adjazent sind. Der Minor S' mit $E'_S \cup e$ für $e = (w_1, w_2)$ kann auf den vorherigen Fall reduziert werden, da eine Kante außerhalb von S gefunden wurde. □

Als nächstes folgt der Beweis zu 3.1.3.

Beweis. Gezeigt wird, dass, falls S keinen $(3, 3)$-Separator bildet, G entweder einen K_5-Minor enthält oder isomorph zu W ist. Falls kein K_5-Minor enthalten ist, gilt nach Lemma 3.1.7, dass $G - R$ oder $G - B$ nicht zusammenhängend ist. Demnach ist B ein 3-Separator, der den Graph teilt, aber die Knoten aus R liegen nicht alle in unterschiedlichen Zusammenhangskomponenten. Deshalb muss es außerhalb von S mindestens einen Pfad P_1 geben, der zwei der roten Knoten in $G - B$ verbindet. Analog gibt es einen Pfad P_2, der zwei blaue Knoten in $G - R$ verbindet. Da P_1 zwei rote Branch-Fans verbindet, liegen seine Endpunkte in zwei verschiedenen Pfaden von S. Gleiches gilt für die Endpunkte von P_2. Liegen die Endpunkte von P_1 beide in einem einzelnen blauen Branch-Fan und die von P_2 in einem einzelnen roten, dann enthält G laut Lemma 3.1.5 einen K_5-Minor. Liegen die Endpunkte von P_1 in parallelen Pfaden von S, enthält G laut Lemma 3.1.6 einen K_5-Minor, da die Endpunkte von P_2 nicht auf einem Pfad von S liegen (analog falls P_2 auf parallelen Pfaden liegt). Übrig bleibt die Möglichkeit, dass die Endpunkte der beiden Pfade paarweise identisch sind, siehe $P(w_7, w_8)$ in Abb. 3.7. Dann ist G ein Subdivision zu W und enthält laut Lemma 3.1.8 keinen K_5-Minor bei Isomorphie zu W. [2] $\square$

3.2 Sequenzieller Algorithmus zum Finden von K_5-Minoren

Da die Theoreme größtenteils auf 3-zusammenhängenden Graphen arbeiten, muss der Eingabegraph ggf. zunächst angepasst werden, bevor der Planaritätstests angewendet werden kann. Ist der Graph 1-zusammenhängend, gibt es einen Knoten, der einen $(1, j)$-Separator für $j \geq 2$ bildet. Genauso müssen zwei Knoten existieren, die einen $(2, j)$-Separator bilden, falls der Graph 2-zusammenhängend ist. In beiden Fällen kann der Separator benutzt werden, um den Graphen in j augmentierte Komponenten zu zerlegen. Anschließend kann der 3-Zusammenhang der einzelnen Komponenten rekursiv geprüft werden. Sind die Komponenten alle 3-zusammenhängend, kann auf jede ein Planaritätstest angewendet werden. Kezdy und McGuinness verwenden den Williamson-Algorithmus [15], welcher in Linearzeit für einen Graphen einen K_5- bzw. $K_{3,3}$-Subdivision ausgibt oder feststellt, dass der Graph planar ist. In der Implementierung wird stattdessen der Planaritätstest von Boyer und Myrvold [10] verwendet, der bereits in OGDF existiert. Ergibt der Planaritätstest, dass eine Komponente planar ist, wird sie nicht weiter beachtet. Enthält sie einen K_5-Minor, kann der Algorithmus stoppen und diesen ausgeben. Wird ein $K_{3,3}$-Minor gefunden, wird geprüft, welcher der vier Fälle aus Theorem 3.1.3 zutrifft. Bei Isomorphie zu W wird die Komponente nicht weiter beachtet. Ist der $K_{3,3}$-Minor ein $(3, 3)$-Separator in der untersuchten Komponente, kann sie in weitere augmentierte Komponenten zerlegt und der Algorithmus rekursiv darauf angewendet werden. Andernfalls müssen genügend Pfade in der Komponente existieren, sodass der $K_{3,3}$-Minor auch einen K_5-Minor bildet und der Algorithmus ihn ausgeben und anhalten kann. Diese Schritte werden solange wiederholt, bis alle augmentierten Komponenten planar bzw. isomorph zu W sind oder ein K_5-Minor gefunden wurde.

Für einen Graphen G sind die einzelnen Schritte zusammengefasst:

1. Gibt es einen 1-Separator in G, dann bilde damit augmentierte Komponenten und wende den Algorithmus rekursiv auf jede an.

2. Gibt es einen 2-Separator in G, dann bilde damit augmentierte Komponenten und wende den Algorithmus rekursiv auf jede an.

3. Planaritätstest:

   (a) G ist planar: Gib zurück, dass kein K_5-Minor gefunden wurde.

   (b) G enthält ein K_5-Subdivision: Gib den K_5-Minor zurück und beende den Algorithmus.

   (c) G enthält ein $K_{3,3}$-Subdivision: Weiter zu Schritt 4

4. G ist isomorph zu W: Gib zurück, dass kein K_5-Minor gefunden wurde. G ist nicht isomorph zu W: Weiter zu Schritt 5

5. Seien $R = \{a, b, c\}$ die roten und $B = \{x, y, z\}$ die blauen Branch-Ends der $K_{3,3}$-Subdivision. Ist R einen $(3, 3)$-Separator, sodass alle Knoten aus B in unterschiedlichen Zusammenhangskomponenten sind, dann bilde damit augmentierte Komponenten und wende den Algorithmus rekursiv auf jede an. (Analog für B als Separator)

6. Ist dieser Schritt erreicht, enthält G einen K_5-Minor bestehend aus der $K_{3,3}$-Subdivision, dem Pfad, der zwei rote, sowie dem Pfad, der zwei blaue Knoten verbindet.

Kapitel 4

Wagner Struktur

In diesem Kapitel soll Theorem 3.1.3 genauer betrachtet werden. Zunächst werden einige Baumstrukturen eingeführt, die den 3-Zusammenhang herstellen können. Anschließend werden $(3, 3)$-Separatoren behandelt und eine Formulierung von dem Theorem von Wagner, das in Theorem 3.1.3 benutzt wurde, vorgestellt. Letztlich werden die gewonnen Erkenntnisse zu einer neuen Struktur zusammengefasst, die als Zertifikat für den Algorithmus von Kezdy und McGuinness dienen kann und dieser entsprechend angepasst.

4.0.1 2-Zusammenhang

4.0.1 Definition. Als *Block* eines Graphen G wird jeder seiner maximalen Teilgraphen bezeichnet, der keinen 1-Separator enthält – als 2-zusammenhängend ist. Je zwei Blöcke haben maximal einen gemeinsamen Knoten, der einen 1-Separator in G bildet. Der *Block-Cut Tree* von G ist ein Baum, der für jeden Block von G einen Knoten besitzt und für jeden Separator eine Kante [9].

Als Beispiel wird in Abb. 4.1 ein Graph mit dazugehörigem Block-Cut Tree in Abb. 4.2 gezeigt. Die Knoten des Block-Cut Tree sind als durchgezogene Linien angegeben, in jedem dieser Knoten findet sich ein 2-zusammenhängender Teilgraph aus G_1.

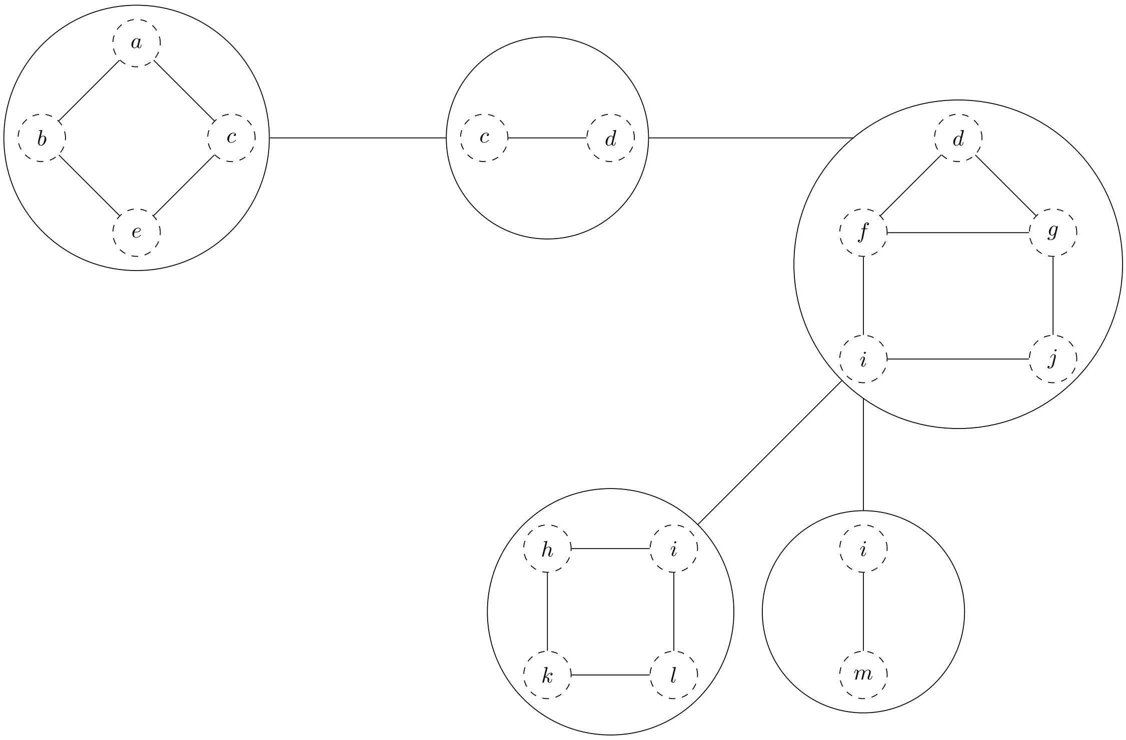


Abbildung 4.2: Der Block-Cut Tree zu G aus Abb. 4.1.

In [13] stellt Li eine Erweiterung zum Block-Cut Tree vor, die als 1-*Block-Tree* bezeichnet wird, da neben den Blöcken auch 1-Separatoren enthalten sind.

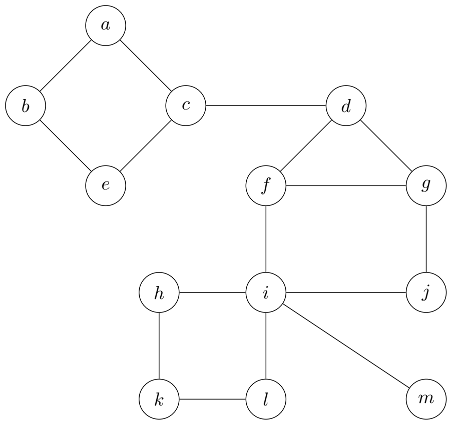


Abbildung 4.1: Ein nicht 2-zusammenhängender Graph G_1.

4.0.2 Definition. Seien für einen Graphen G die Knoten seines Block-Cut Trees $T_{bc} = (V_{T_{bc}}, E_{T_{bc}})$ als Blockknoten bezeichnet. Der 1-*Block-Tree* von G ist ein Baum $T_{(1)} = (V_{(1)}, E_{(1)})$. Dann gilt $V_{(1)} \subseteq V_{T_{bc}}$. Außerdem enthält $T_{(1)}$ einen zusätzlichen Knoten c für jede Kante $(u, v) \in E_{T_{bc}}$, sodass $\{(u, c), (c, v)\} \subseteq E_{(1)}$ und $E_{T_{bc}} \cap E_{(1)} = \emptyset$ gilt. Dabei wird c als Cliquenknoten bezeichnet und beinhaltet genau den 1-Separator, den die zu u und v gehörenden Teilgraphen in G als gemeinsamen Knoten besitzen.

Ein Beispiel findet sich in Abb. 4.3. Es kann beobachtet werden, dass die in Blockknoten enthaltenen Teilgraphen identisch zu den augmentierten Komponenten sind, die im Algorithmus von Kezdy und McGuinness in Schritt 1 durch 1-Separatoren gebildet werden.

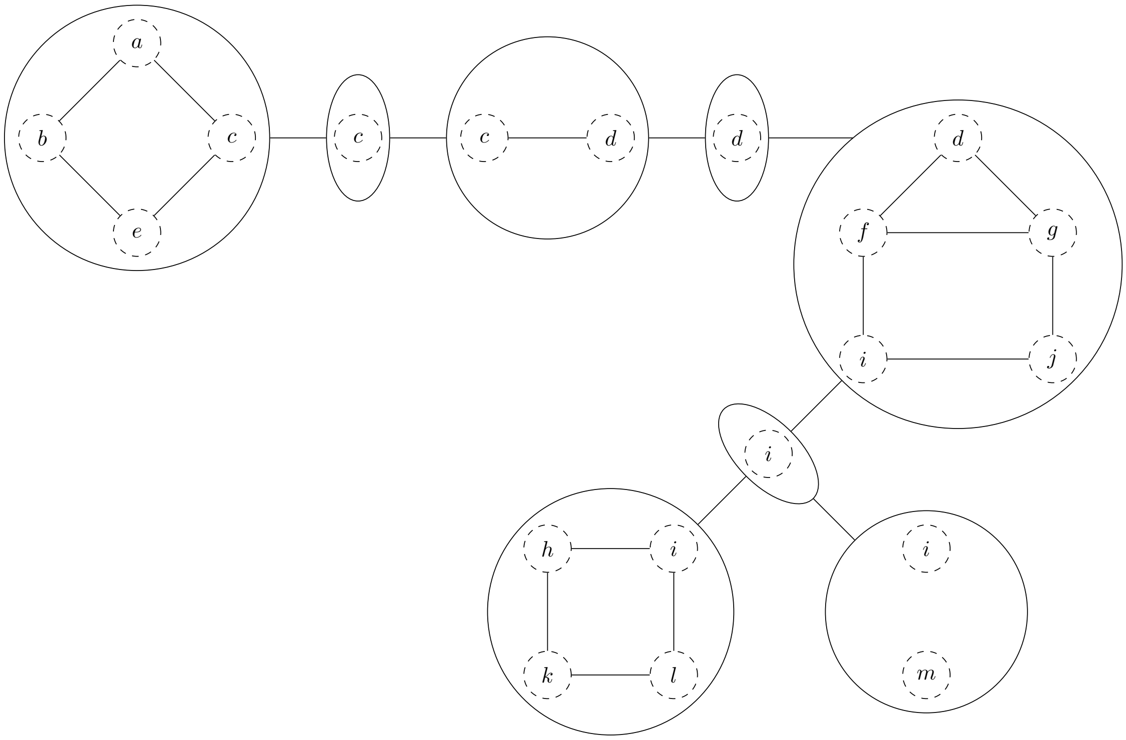


Abbildung 4.3: Der 1-Block-Tree zu G aus Abb. 4.2. Die Cliquenknoten sind ovalförmig dargestellt.

4.0.2 3-Zusammenhang

Nachdem durch Block-Cut Trees bzw. 1-Block-Trees der 2-Zusammenhang in allen Blockknoten hergestellt wurde, können die darin enthaltenen Teilgraphen jeweils als Eingabe für einen SPQR-Baum genutzt werden, um 3-zusammenhängende Komponenten zu erzeugen. Die folgende Definition dazu ist [5] entnommen.

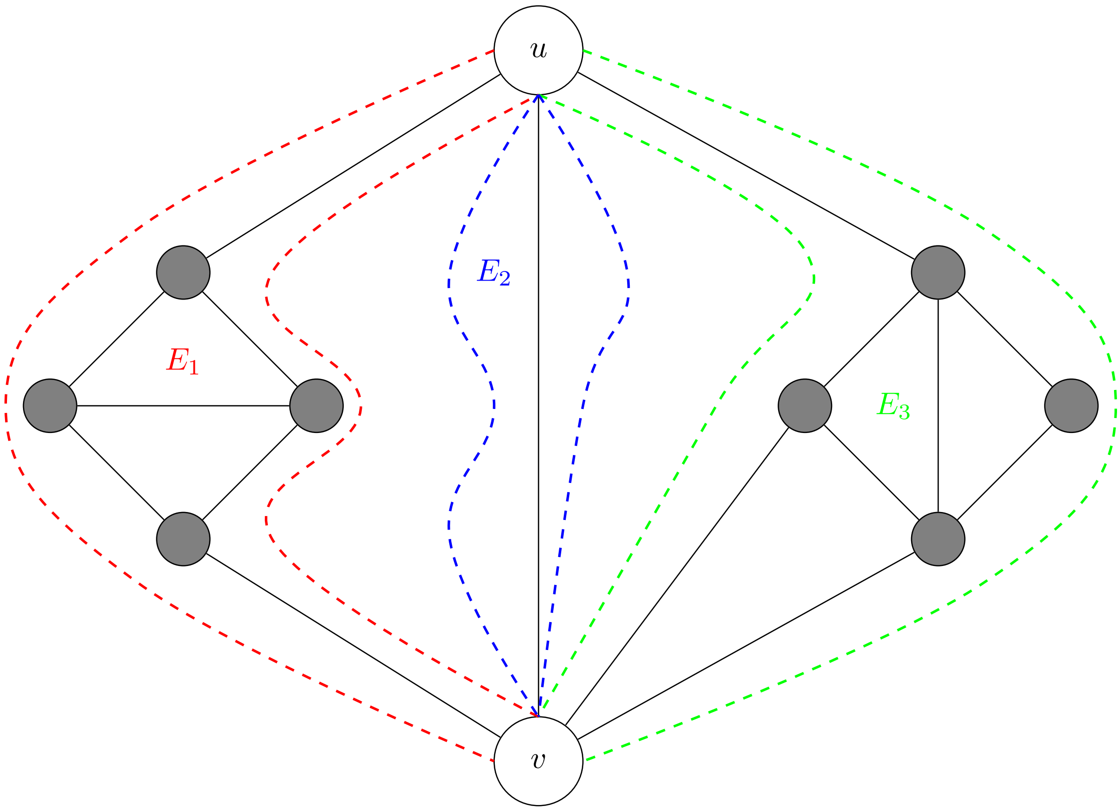


Abbildung 4.4: Unterteilung der Kanten in drei Mengen E_1, E_2 und E_3 anhand des Knotenpaares $\{u, v\}$.

4.0.3 Definition. Sei $G = (V, E)$ ein 2-zusammenhängender Graph. Als *Split Pair* $\{u, v\}$ wird entweder ein adjazentes Knotenpaar oder ein 2-Separator bezeichnet, ein *maximales Split Pair* $\{s, t\}$ für $\{u, v\}$ teilt den Graphen so auf, dass u, v, s, und t in einer 3-zusammenhängenden Komponente liegen. Eine *Split Component* für ein Split Pair ist entweder eine Kante zwischen diesem oder der maximale Teilgraph, für den es kein Split Pair mehr ist. Ein solches Knotenpaar teilt die Kanten in die Mengen $E_1, ..., E_k$, sodass eine Menge E_i alle Pfade enthält, die u oder v höchstens als Endpunkte haben. Dabei enthält E_i entweder einen Teilgraphen von G, der nur das Knotenpaar enthält, oder einen für den das Knotenpaar kein 2-Separator ist. [1] Der SPQR-Baum T_{SPQR} von G ist ein Baum, der G anhand jedem Split Pair $\{u, v\}$ rekursiv aufteilt und die entstehenden Minoren mit S, P, Q oder R markiert:

- **Q**: Tritt der Randfall auf, dass G nur eine einzige Kante (u, v) besitzt, dann enthält der SPQR-Baum einen einzelnen Q-Knoten, der auf G verweist.

[1] In Abb. 4.4 ist ein 2-zusammenhängender Graph skizziert, in dem $\{u, v\}$ eingezeichnet ist, sowie farblich hervorgehoben die dadurch entstehende Unterteilung in Kantenmengen.

- **P**: Entstehen durch das Knotenpaar die Kantenmengen $E_1, ..., E_k$ mit $k \geq 3$, dann wird ein P-Knoten hinzugefügt, der das Knotenpaar enthält sowie k viele parallele Kanten zwischen diesem.

- **S**: Andernfalls teilen u und v die Kanten in zwei Mengen, sodass es eine Kante $(u, v) \in E$ und einen weitere Pfad gibt, der das Knotenpaar verbindet, dann enthält der hinzugefügte S-Knoten die Kante und den Pfad.

- **R**: Tritt keiner der obigen Fälle ein, dann seien $\{s_1, t_1\}, ..., \{s_k, t_k\}$ für $k \geq 1$ die maximalen Split Pairs für u, v und G_i für alle $1 \leq i \leq k$ die Vereinigung aller Split Components für $\{u, v\}$ außer der, die die Kante (u, v) enthält. Der neue R-Knoten enthält G, wobei jeder Teilgraph G_i durch die Kante (s_i, t_i) ersetzt wird.

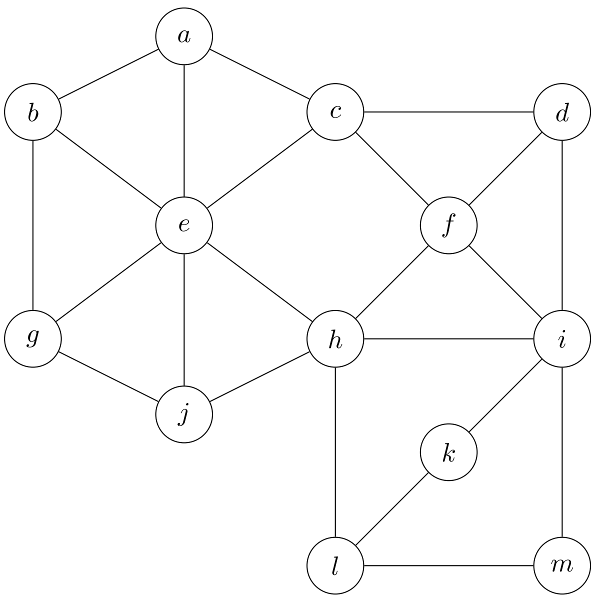


Abbildung 4.5: Ein 2-zusammenhängender Graph G_2.

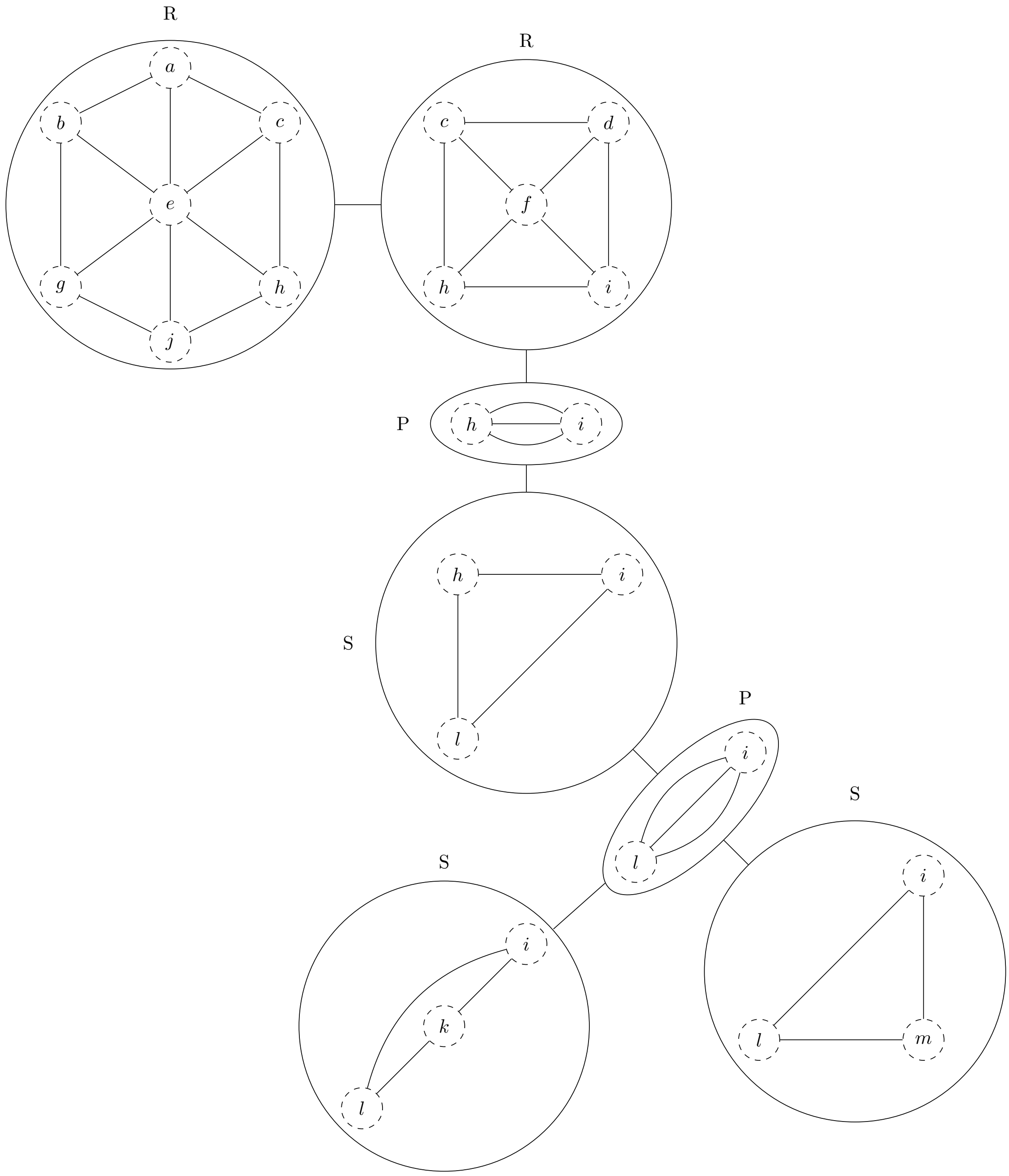


Abbildung 4.6: SPQR Baum zum Graphen G_2 aus Abb. 4.5.

Zu dem Graphen in Abb. 4.5 ist in Abb. 4.6 der zugehörige SPQR Baum skizziert. Besonders interessant sind die R-Knoten, die 3-zusammenhängende Minoren von G_2 sind. Zwar gilt das auch für die drei S-Knoten, allerdings sind Kreise generell nicht 3-zusammenhängend, aber immer planar.

In [13] stellt Li analog zum 1-Block-Tree den 2-Block-Tree vor, der aus 3-zusammenhängende Komponenten (Blockknoten) und 2-Separatoren (Cliquenknoten) besteht.

4.0.4 Definition. Sei $G = (V_G, E_G)$ ein 2-zusammenhängender, K_5-Minor freier Graph, dann ist der 2-*Block-Tree* $T_{(2)} = (V_{(2)}, E_{(2)})$ von G wie folgt definiert. Bilden die Knoten $\{u, v\} \subseteq V_G$ einen $(2, j)$-Separator in G für $j \geq 2$, dann seien $Z_1, ..., Z_j$ die Zusammenhangskomponenten von $G - \{u, v\}$. Analog zum 1-Block-Tree wird ein Cliquenknoten für den Separator angelegt, der alle Blockknoten bestehend aus $Z_i \cup \{u, v\}$ als Nachbarn besitzt. Außerdem wird die Kante (u, v) sowohl in den Block- als auch in dem Cliquenknoten hinzugefügt, falls sie nicht vorhanden ist. Ist G 3-zusammenhängend, besitzt $T_{(2)}$ einen einzelnen Blockknoten, der G vollständig enthält.

Ein 2-Block-Tree zu G_2 aus Abb. 4.5 ist in Abb. 4.7 zu sehen. Es ist zu beobachten, dass Knoten wie etwa h oder i Teil mehrerer Separatoren sein können und somit nicht nur in mehreren Graph-, sondern auch in mehreren Cliquenknoten enthalten sind. Allerdings sind keine zwei Knoten im 2-Block-Tree identisch.

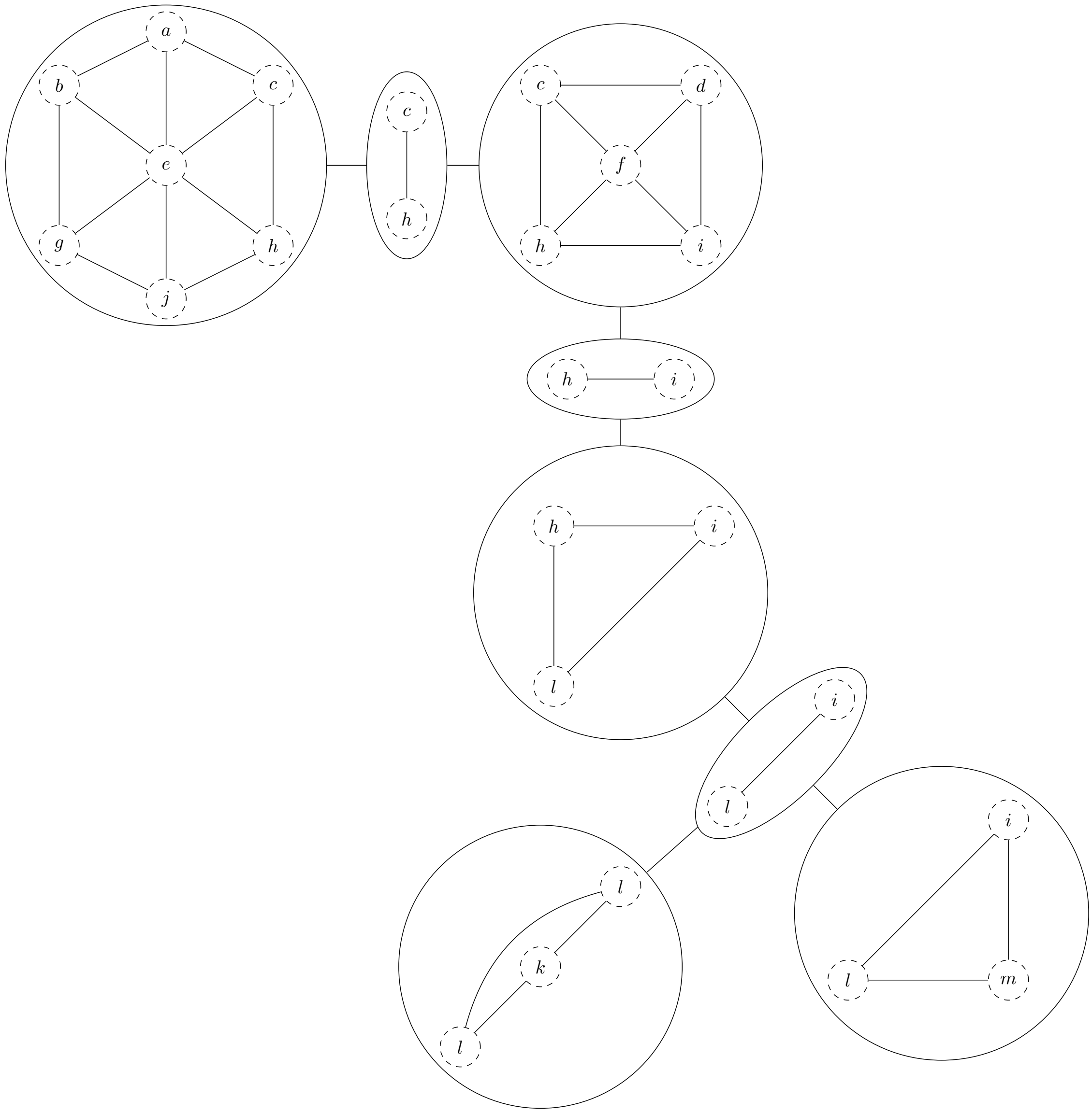


Abbildung 4.7: 2-Block-Tree zum Graphen aus Abb. 4.5. Die Cliquenknoten sind ovalförmig dargestellt und enthalten immer genau zwei Knoten, alle übrigen sind Blockknoten.

4.0.3 $(3,3)$-Separatoren

4.0.5 Definition. Sei $G = (V_G, E_G)$ ein 3-zusammenhängender, K_5-Minor freier Graph, dann ist der $(3,3)$-*Block-Tree* $T_{(3,3)} = (V_{(3,3)}, E_{(3,3)})$ von G wie folgt definiert. Enthält G keinen $(3,j)$-Separator für $j \geq 3$, so besitzt $T_{(3,3)}$ einen einzigen Blockknoten, der G komplett enthält. Sei andernfalls C ein Graph, der die drei Knoten eines solchen $(3,j)$-Separators als Clique beinhaltet. Dann werden Blockknoten in $T_{(3,3)}$ für alle $Z_i \cup C$ mit $1 \leq i \leq j$ angelegt, wobei Z_i die durch den Separator entstandenen Zusammenhangskomponenten sind. Außerdem wird ein Cliquenknoten mit C erzeugt, der im Baum Kanten zu allen zuvor erzeugten Blockknoten hat.

Ein Beispiel ist in Abb. 4.8 skizziert.

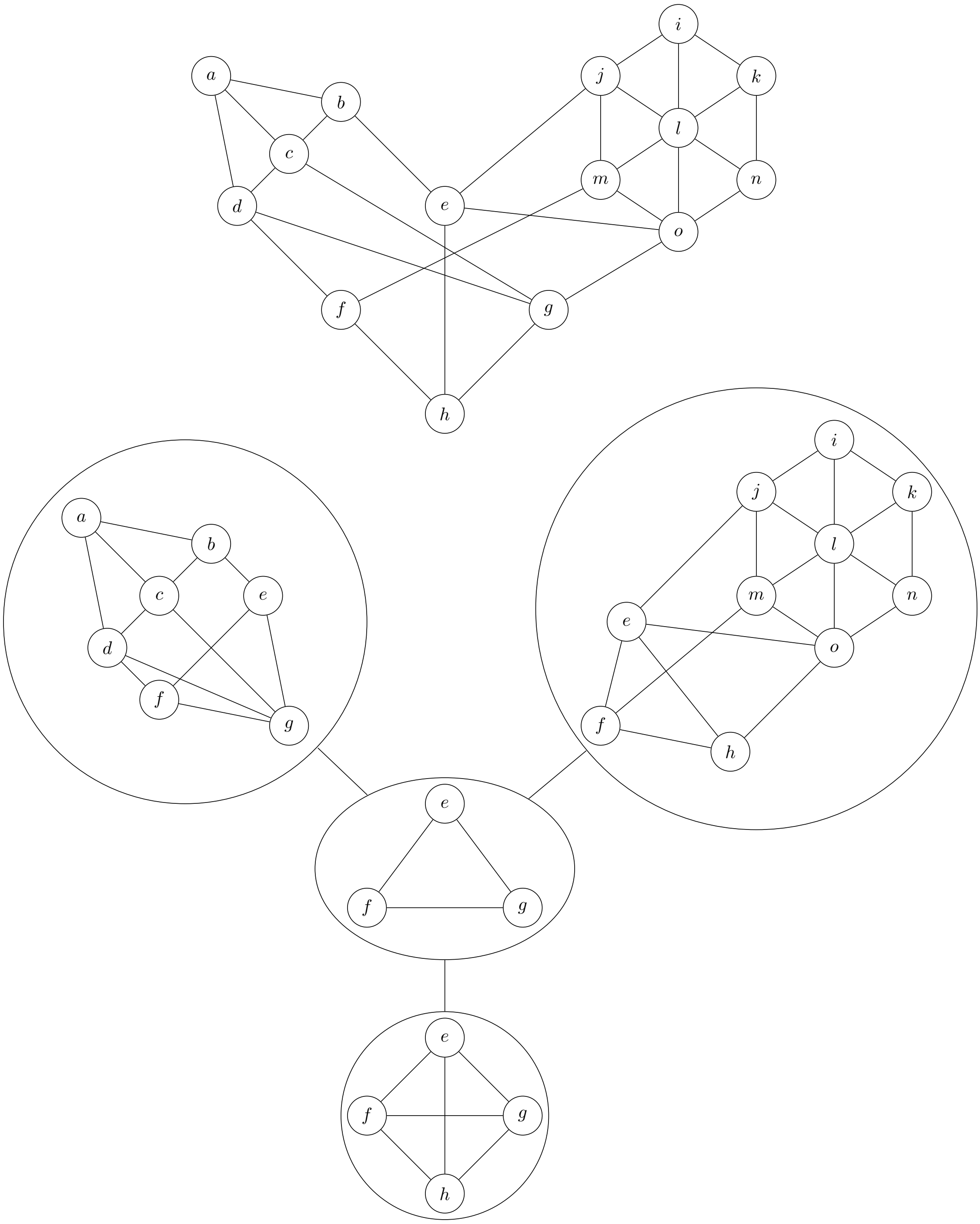


Abbildung 4.8: Ein 3-zusammenhängender Graph mit dazugehörigem $(3, 3)$-Block-Tree. Der Separator wird durch $\{e, f, g\}$ gebildet. Es ist zu sehen, dass der Eingabegraph nicht planar ist, da ein $K_{3,3}$-Minor enthalten ist. Die Blockknoten enthalten jedoch alle planare Minoren des ursprünglichen Graphs.

4.0.4 Wagner-Struktur

4.0.6 Definition. Sei G ein Graph ohne K_5-Minor. Die *Wagner-Struktur* von G ist ein Wald von Bäumen, sodass für jede Zusammenhangskomponente Z von G genau ein Baum T_Z existiert. Jeder T_Z besteht aus einem 1-Block-Tree $T_{(1)} = (V_{(1)}, E_{(1)})$ für Z, einem 2-Block-Tree $T_{(2)} = (V_{(2)}, E_{(2)})$ für alle Graphknoten $v \in V_{(2)}$ und einem $(3,3)$-Block-Tree $T_{(3,3)} = (V_{(3,3)}, E_{(3,3)})$ für alle Graphknoten $v \in V_{(2)}$ [4].

Die Wagner-Struktur von G stellt eine Dekomposition dar, sodass alle Cliquenknoten 1, 2 und $(3,3)$-Separatoren von G enthalten und die Graphknoten aller $(3,3)$-Block-Trees entweder planar oder isomorph zu W sind. Trifft das auf einen Graphknoten nicht zu, dann enthält G einen K_5-Minor und die Wagner-Struktur ist ungültig. Das Theorem von Wagner kann daraufhin wie folgt formuliert werden:

4.0.7 Theorem. *Ein Graph enthält genau dann keinen K_5-Minor, wenn für ihn eine Wagner-Struktur existiert [14].*

Als Beispiel ist dazu in Abb. 4.9 ein Graph G zu sehen, der nicht planar ist, aber keinen K_5-Minor enthält. In Abb. 4.10 sind die 3-zusammenhängende Graphen G_1, G_2, G_3, G_4, G_5 und G_6 abgebildet, von denen ebenfalls keiner einen K_5-Minor enthält. Außerdem ist G_6 nicht planar. Einige Cliquen sind in beiden Abbildungen mit je einer Farbe hervorgehoben, sodass durch das Zusammenfügen gleichfarbiger Knoten ein Graph G' erzeugt werden kann, der G als Teilgraphen enthält. In Abb. 4.11 ist die Wagner-Struktur zu G skizziert. Der 1-Block-Tree wurde nicht eingezeichnet, da er aufgrund des 2-Zusammenhangs von G aus nur einem Graphknoten besteht. Foglich existiert ein einzelner 2-Block-Tree, der aus blauen Knoten und Kanten besteht. Für jeden seiner Graphknoten existiert ein $(3,3)$-Block-Tree (rot). Es kann beobachtet werden, dass die Cliquenknoten aller Block-Trees genau die farblich markierten Cliquen aus Abb. 4.10 enthalten und in G Separatoren bilden. Außerdem ist zu sehen, dass die Graphknoten aller $(3,3)$-Block-Trees entweder planar oder isomorph zu W sind. Daraus folgt, dass G keinen K_5-Minor enthält.

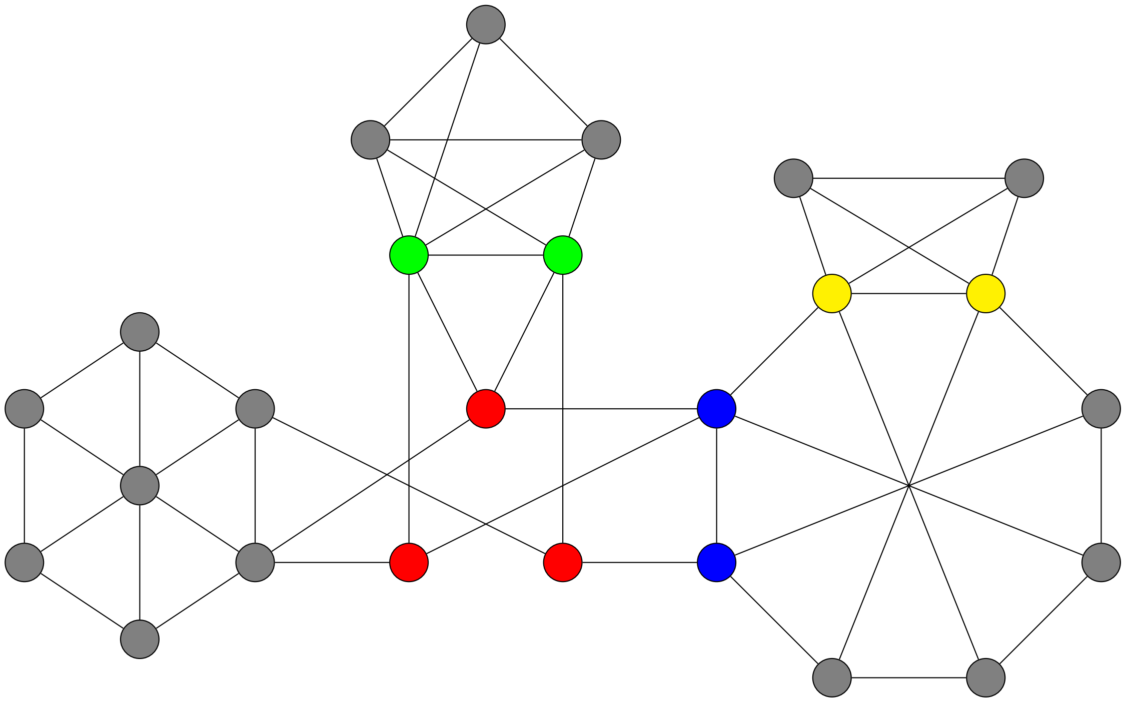


Abbildung 4.9: Ein nicht-planarer Graph G, der keinen K_5-Minor enthält.

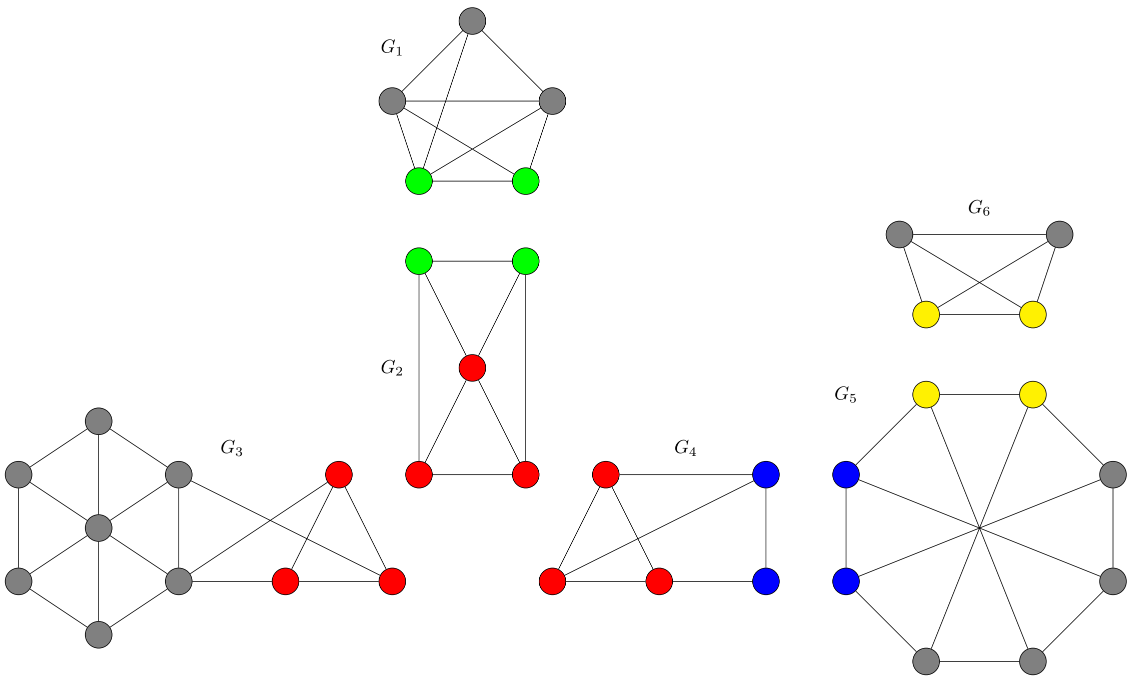


Abbildung 4.10: Mehrere Graphen, die planar oder isomorph zu W sind.

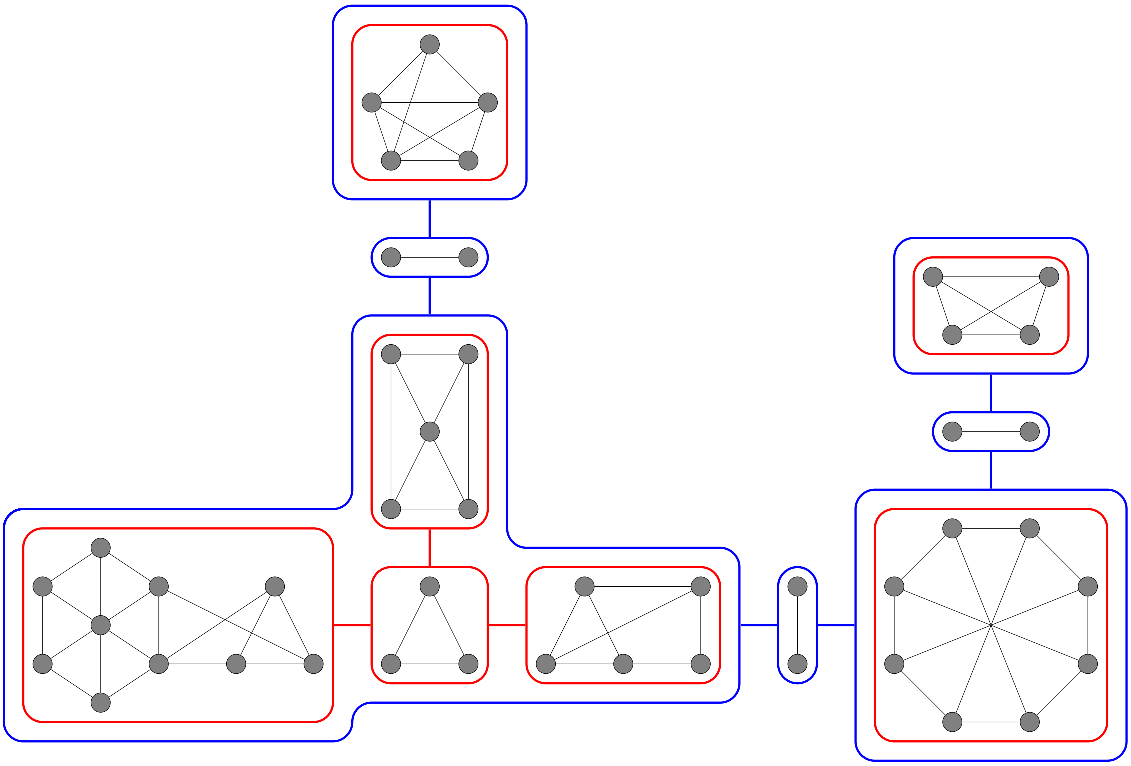


Abbildung 4.11: Wagner-Struktur von G aus Abb. 4.9. Da G bereits 2-zusammenhängend ist, enthält der 1-Block-Tree einen einzelnen Graphknoten für den kompletten Graphen und wurde nicht eingezeichnet. Die Knoten und Kanten des 2-Block-Trees sind blau, die der $(3,3)$-Block-Trees rot hervorgehoben.

4.1 Algorithmus von Kezdy und McGuinness als Wagner-Struktur

Im Folgenden wird die durch Block-Trees definierte Wagner-Struktur mit dem Algorithmus von Kezdy und McGuinness verglichen und dieser so modifiziert, dass er sie erzeugen kann. Anschließend werden augmentierte Komponenten für 1- und 2-Separatoren durch Block-Cut Trees und SPQR-Bäume erzeugt sowie ein zertifizierender Algorithmus beschrieben.

4.1.1 Block-Trees

4.1.1 Beobachtung. Wird der Graph in augmentierte Komponenten aufgeteilt, sind diese isomorph zu mindestens einem Blockknoten der verschiedenen Block-Trees, wenn bis dahin alle Separatoren gleich gewählt wurden.

Im Algorithmus von Kezdy und McGuinness wird der Graph in augmentierte Komponenten aufgeteilt, wenn ein $(1,2)$-, $(2,2)$- oder $(3,3)$-Separator gefunden wird. Sei C ein solcher (i,j)-Separator, der einen zusammenhängenden Graphen G in genau j Zusammenhangskomponenten $Z_1, ..., Z_j$ aufteilt, wenn die i Knoten von C aus G entfernt werden.

Angenommen, es gäbe eine augmentierte Komponenten G_a, zu der es im zugehörigen $(3,3)$-Block-Tree keinen Blockknoten t mit assoziiertem Graphen G_t gibt, sodass G_a und G_t isomorph sind. Dann gibt es einen Separator C_a in G, aus dem G_a entstanden ist. Da es kein t mit einem zu G_a isomorphen G_t gibt, wurden keine Knoten im Block-Tree mit C_a erzeugt, sodass es ebenfalls keinen Cliquenknoten mit C_a gibt. Dann wurde mindestens ein Separator C'_a für den 1-Block-Tree gewählt, wodurch die Knoten von C_a so in Blockknoten enthalten sind, dass sie dort keinen Separator bilden, was der gleichen Wahl von Separatoren widerspricht. Oder C_a ist vollständig in einem Blockknoten enthalten und bildet dort einen Separator, was entweder dem 2-Zusammenhang oder der Planarität bzw. Isomorphie zu W des Blockknotens widerspricht.

Der Algorithmus von Kezdy und McGuinness erzeugt also zu Beginn alle Blockknoten eines 1-Block-Trees. Zusätzlich können die Knoten der gefundenen Separatoren gespeichert werden, um ebenfalls die Cliquenknoten zu erhalten, sodass der Algorithmus einen 1-Block-Tree erzeugt. Im zweiten Schritt werden die augmentierten Komponenten bzw. Blockknoten des 1-Block-Tree in 3-zusammenhängende augmentierte Komponenten bzw. einen 2-Block-Tree überführt. Analog zum vorherigen Fall gibt es für jede augmentierte Komponente, die aus einem Blockknoten des 1-Block-Trees mit einem $(2,2)$-Separator gebildet wird, einen isomorphen Blockknoten im 2-Block-Tree, falls alle Separatoren gleich gewählt wurden. Erneut können aus den Separatorknoten Cliquen gebildet und in zusätzlichen Knoten gespeichert werden, um als Ausgabe des zweiten Schrittes einen 2-Block-Tree zu erzeugen.

Wird ein 1-Block-Tree durch einen Block-Cut Tree aufgebaut, dann muss zusätzlich jedes adjazente Knotenpaar des Block-Cut Trees nach dem gemeinsamen Knoten durchsucht werden, der den 1-Separator bildet. Analog müssen für einen 2-Block-Tree Knotenpaare des SPQR-Baumes, die aus zwei R-Knoten bestehen, nach den zwei gemeinsamen Knoten durchsucht werden, die ein (2)-Separator waren. Neben den Knotenpaaren, die adjazent zueinander sind, müssen Cliquenknoten zusätzlich für solche gebildet werden, die durch einen Pfad miteinander verbunden sind, der ausschließlich aus S-, P- oder Q-Knoten besteht. Um in der Praxis Laufzeit zu sparen, kann dieser Schritt auf den Fall verschoben werden, dass ein K_5-Minor gefunden wurde und die zugehörigen Pfade im Eingabegraph geprüft werden sollen.

Um die Blockknoten des 2-Block-Trees in je einen $(3,3)$-Block-Tree zu verwandeln, müssen nicht nur entsprechende Separatoren gefunden und augmentierte Komponente gebildet, sondern auch garantiert werden, dass die Blockknoten der $(3,3)$-Block-Trees entweder planar oder isomorph zu W sind und in keinem ein K_5-Minor enthalten ist. Hier kann das Theorem 3.1.3 in Kombination mit dem zuvor angewendeten Planaritätstest genutzt werden, um die Struktur aufzubauen. Sei dazu G_t der Graph, der zu einem beliebigen Blockknoten t im 2-Block-Tree gehört. G_t muss 3-zusammenhängend sein, da er als Graph vollständig in t vorkommt. Im Algorithmus von Kezdy und McGuinness wird G_t nun im dritten Schritt durch einen Planaritätstest überprüft. Wird ein K_5-Minor gefunden, ist die Wagner-Struktur ungültig und der Algorithmus terminiert. Andernfalls kann G_t planar sein – dann wird im $(3, 3)$-Block-Tree ein einzelner Blockknoten angelegt, der G_t enthält und als planar markiert ist. Letztlich kann der Planaritätstest ein $K_{3,3}$-Subdivision S finden, sodass die Voraussetzungen für Theorem 3.1.3 gegeben sind. Im ersten Fall von Theorem 3.1.3 enthält G_t jedoch einen K_5-Minor, da keine Knoten von S einen $(3, 3)$-Separator bilden – der Beweis dazu findet sich in Lemma 3.1.7. Damit wäre die Wagner-Struktur erneut ungültig. Im zweiten Fall ist G_t isomorph zu W. Analog zum planaren Fall wird ein neuer Blockknoten im $(3, 3)$-Block-Tree angelegt, der G_t enthält und entsprechend gekennzeichnet ist. Der dritte und vierte Fall treten ein, falls drei Knoten aus S einen $(3, 3)$-Separator bilden. Kezdy und McGuinness erzeugen hier augmentierte Komponenten, die als Blockknoten des $(3, 3)$-Block-Trees genutzt werden können. Zusätzlich müssen die drei Knoten des Separators als neuer Cliquenknoten eingefügt und mit den entstandenen Blockknoten verbunden werden. Der Algorithmus wird rekursiv auf jeden der Blockknoten angewendet, bis alle zugehörigen Teilgraphen planar oder isomorph zu W sind bzw. ein K_5-Minor gefunden wurde. Somit ist das Ergebnis des modifierten Algorithmus entweder eine Wagner-Struktur, die aus einem Wald von 1-Block-Trees mit Blockknoten, die aus 2-Block-Trees mit Blockknoten aus $(3, 3)$-Block-Trees bestehen und die die planaren Minoren und Kopien von W im Eingabegraph zeigen. Oder es wird ein K_5-Minor als Teilgraph des Eingabegraphen gefunden, sodass die Wagner-Struktur ungültig ist, die Frage, ob ein K_5-Minor enthalten ist, jedoch in beiden Fällen eindeutig beantwortet werden kann.

4.1.2 Zertifierender Algorithmus

Das Ziel ist, einen K_5-Minor zu finden oder ein Zertifikat zu liefern, dass keiner enthalten ist. Dafür sind besonders folgende Strukturen in einem Graphen relevant: Planare Teilgraphen, W-Subdivisions, K_5-Minoren und $(3, 3)$-Separtoren – andere Separatoren werden lediglich benötigt, um den 3-Zusammenhang zu garantieren. Beispielsweise sind zwei planare Teilgraphen, die durch einen 2-Separator verbunden sind, ebenfalls planar. Das Zertifikat für einen Graphen G besteht also entweder aus einem gefundenen K_5-Minor in Form von fünf disjunkten Knotenmengen, die jeweils einen der fünf Knoten des K_5 formen – dann ist zu prüfen, ob dadurch im ursprünglichen Graph ebenfalls einer geformt wird. Einerseits muss dazu jede Knotenmenge einen zusammenhängenden Teilgraphen in G formen. Andererseits sind genau diese fünf Teilgraphen paarweise durch Pfade in G verbunden, die beispielsweise durch eine Tiefensuche gefunden werden können. Oder es wird eine Wagner-Struktur erzeugt, es kann jeder Knoten der Struktur wie folgt geprüft werden:

1. **Planar**: Sei G' der Teilgraph von G, der alle Knoten des planaren Minoren enthält. Dann ist G' ebenfalls planar, was z. B. durch einen Planaritätstest geprüft werden kann.

2. W: Analog zum Test eines K_5-Minor können acht Knotenmengen die W-Subdivision darstellen. Der für den Algorithmus von Kezdy und McGuinness implementierte Isomorphietest kann wiederverwendet werden.

3. $(3, 3)$**-Separator**: Jeder gefundene Separator bildet auch im ursprünglichen Graphen einen Separator. Sei a die Anzahl der Zusammenhangskomponenten von G. Dann kann der $(3, 3)$-Separator aus G entfernt werden. Sei b die neue Anzahl der Zusammenhangskomponenten. Dann muss $b - a \geq 2$ gelten.

Kapitel 5

Implementierung

Im Folgenden werden Details zur Implmentierung im *Open Graph Drawing Framework* erklärt. Dazu wird zunächst das Framework vorgestellt, anschließend werden wichtige Klassen und Algorithmen behandelt und letztlich zusammengesetzt, um einen zertifierenden Algorithmus zum Finden von K_5-Minoren zu bilden.

Das *Open Graph Drawing Framework (OGDF)* ist ein in C++ geschriebenes Framework, das Algorithmen und Datenstrukturen für Graphen enthält, wobei ein besonderes Augenmerk auf dem Zeichnen von Graphen liegt. OGDF kann unabhängig von anderen Frameworks und Bibliotheken genutzt werden und läuft sowohl unter Linux als auch unter Windows und MacOS. Es wurde 2005 unter der GNU General Public License als Open Source Projekt veröffentlicht [1] [6].

```
int FindK5::getWagnerStructure(Graph &input, int k5Limit) {
  Graph minor;
  WagnerStructure structure;
  WagnerMinor m(input, minor, structure);

  int foundK5s = 0;
  while (m.getUntypedComponentCount() > 0) {

    BoyerMyrvold boyerMyrvold;
    SList<KuratowskiWrapper> kuratowskiSubdivisions;
    if (boyerMyrvold.planarEmbed(*m.minor, kuratowskiSubdivisions, 2)) {
      m.specifyNodes(m.minor->nodes, WagnerStructure::NodeType::Planar);
      continue;
    }

    KuratowskiWrapper kuratowskiSubdivision
                          = kuratowskiSubdivisions.popFrontRet();

    if (kuratowskiSubdivision.isK5()) {
      m.specifyNodes(kuratowskiSubdivision.edgeList,
                          WagnerStructure::NodeType::K5);
      foundK5s++;

    } else if (kuratowskiSubdivision.isK33()) {

      if (m.isIsomorphicToW(*m.minor, kuratowskiSubdivision.edgeList)) {
        m.specifyNodes(kuratowskiSubdivision.edgeList,
                          WagnerStructure::NodeType::W);

      } else if (m.split(kuratowskiSubdivision, *m.minor)) {
        continue;

      } else {
        m.specifyNodes(kuratowskiSubdivision.edgeList,
                          WagnerStructure::NodeType::K5);
        foundK5s++;
      }
    }

    if (k5Limit > 0 && foundK5s >= k5Limit)
      return foundK5s;
  }

  return foundK5s;
}
```

Listing 5.1: Implementierung des Algorithmus von Kezdy und McGuinness.

Kapitel 6

Experimentelle Analyse

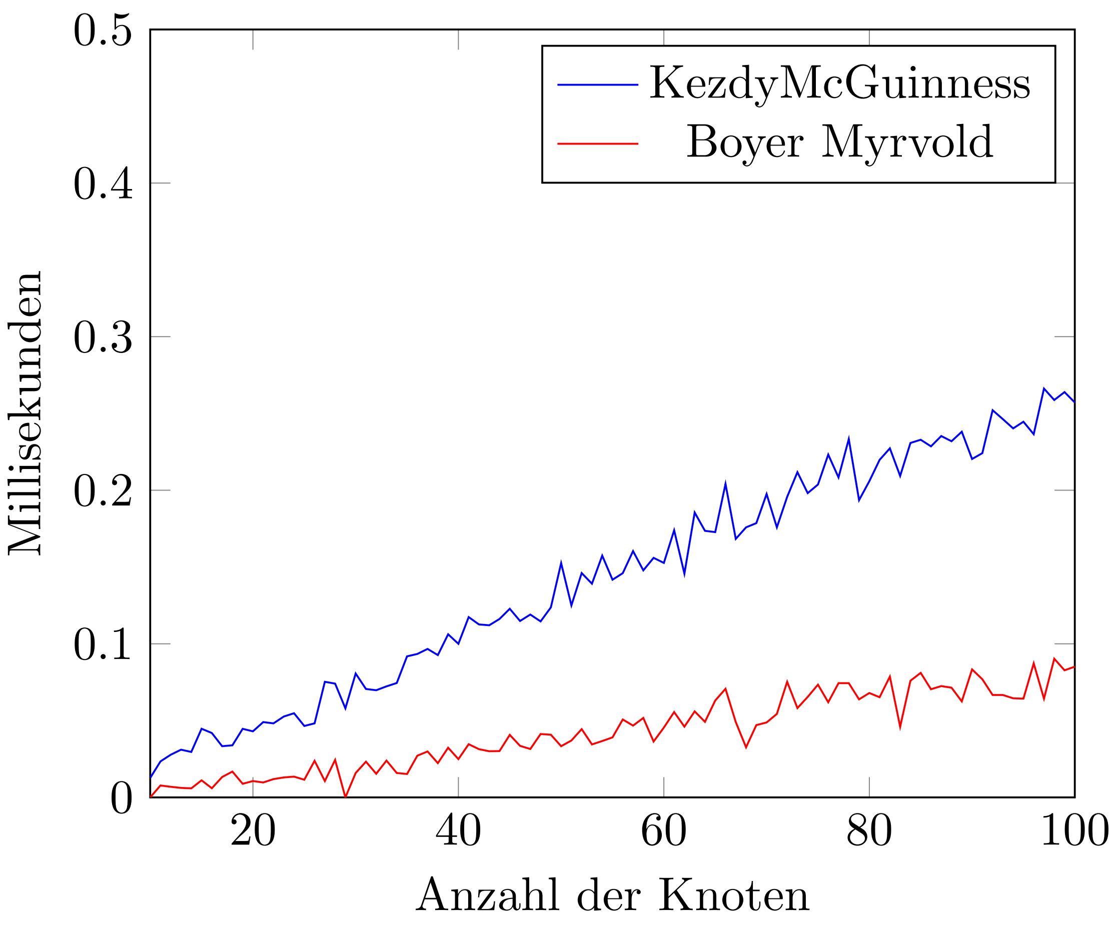


Abbildung 6.1: Benchmark der Rome Library.

Kapitel 7

Zusammenfassung und Ausblick

Anhang A

Weitere Informationen

Abbildungsverzeichnis

Algorithmenverzeichnis

Symbolverzeichnis

Literaturverzeichnis

[1] *OGDF - About.* `https://ogdf.uos.de/about/`. Stand 22. September 2019.

[2] André E. Kézdy, Patrick McGuinness: *Sequential and Parallel Algorithms to Find a K_5 Minor.* In: *Proceedings of the Third Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 27-29 January 1992, Orlando, Florida, USA.*, Seiten 345–356, 1992.

[3] Barahona, Francisco: *The Max-cut Problem on Graphs Not Contractible to K5.* Oper. Res. Lett., 2(3):107–111, 1983.

[4] Bruce A. Reed, Zhentao Li: *A Linear Time Algorithm for Testing if G has a K_5 Minor.*

[5] Carsten Gutwenger, Petra Mutzel: *A Linear Time Implementation of SPQR-Trees.* In: *Graph Drawing, 8th International Symposium, GD 2000, Colonial Williamsburg, VA, USA, September 20-23, 2000, Proceedings*, Seiten 77–90, 2000.

[6] Chimani, M., C. Gutwenger, M. Jünger, G. W. Klau, K. Klein und P. Mutzel: *The Open Graph Drawing Framework (OGDF).* In: Tamassia, R. (Herausgeber): *Handbook of Graph Drawing and Visualization*, Kapitel 17. CRC Press, 2014.

[7] Diestel, Reinhard: *Graph Theory, 4th Edition*, Band 173 der Reihe *Graduate texts in mathematics.* Springer, 2012.

[8] Francisco Barahona, Michael Jünger, Gerhard Reinelt: *Experiments in quadratic 0-1 programming.* Math. Program., 44(1-3):127–137, 1989.

[9] John A. Bondy, Uppaluri S. R. Murty: *Graph Theory.* Springer Publishing Company, Incorporated, 1st Auflage, 2008.

[10] John M. Boyer, Wendy J. Myrvold: *On the Cutting Edge: Simplified O(n) Planarity by Edge Addition.* J. Graph Algorithms Appl., 8(3):241–273, 2004.

[11] Jünger, Michael, Elisabeth Lobe, Petra Mutzel, Gerhard Reinelt, Franz Rendl, Giovanni Rinaldi und Tobias Stollenwerk: *Performance of a Quan-*

tum Annealer for Ising Ground State Computations on Chimera Graphs. CoRR, abs/1904.11965, 2019.

[12] Karp, Richard M.: *Reducibility Among Combinatorial Problems.* In: *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA*, Seiten 85–103, 1972.

[13] Li, Zhentao: *Tree decompositions and linear time algorithms.* Doktorarbeit, School of Computer Science, McGill University, Montreal, Quebec, 12 2011.

[14] Wagner, Klaus: *Über eine Eigenschaft der ebenen Komplexe.* Mathematische Annalen, 114:570–590, 1937.

[15] Williamson, S. G.: *Depth-First Search and Kuratowski Subgraphs.* J. ACM, 31(4):681–693, 1984.

Eidesstattliche Versicherung

Sauer, Julian	197859
Name, Vorname	Matr.-nr.

Ich versichere hiermit an Eides statt, dass ich die vorliegende Masterarbeit mit dem Titel

Effiziente Berechnung von K_5-Minoren in Graphen

selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Dortmund, den 27. September 2019	
Ort, Datum	Unterschrift

Belehrung:

Wer vorsätzlich gegen eine die Täuschung über Prüfungsleistungen betreffende Regelung einer Hochschulprüfungsordnung verstößt, handelt ordnungswidrig. Die Ordnungswidrigkeit kann mit einer Geldbuße von bis zu 50.000,00 € geahndet werden. Zuständige Verwaltungsbehörde für die Verfolgung und Ahndung von Ordnungswidrigkeiten ist der Kanzler/ die Kanzlerin der Technischen Universität Dortmund. Im Falle eines mehrfachen oder sonstigen schwerwiegenden Täuschungsversuches kann der Prüfling zudem exmatrikuliert werden. (§ 63 Abs. 5 Hochschulgesetz - HG -)

Die Abgabe einer falschen Versicherung an Eides statt wird mit Freiheitsstrafe bis zu 3 Jahren oder mit Geldstrafe bestraft.

Die Technische Universität Dortmund wird gfls. elektronische Vergleichswerkzeuge (wie z.B. die Software „turnitin“) zur Überprüfung von Ordnungswidrigkeiten in Prüfungsverfahren nutzen.

Die oben stehende Belehrung habe ich zur Kenntnis genommen:

Dortmund, den 27. September 2019	
Ort, Datum	Unterschrift