Karlsruhe Institute of Technology

Induzierte Bäume in planaren Graphen

Bachelorarbeit
von

Simon Bischof

An der Fakultät für Informatik
Institut für Theoretische Informatik

Erstgutachter:	Prof. Dr. Dorothea Wagner
Zweitgutachter:	Prof. Dr. Peter Sanders
Betreuende Mitarbeiter:	Dipl.-Inform. Thomas Bläsius
	Dr. Ignaz Rutter
	Dr. Tamara Mchedlidze

Bearbeitungszeit: 15.07.2013 – 11.11.2013

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich diese Arbeit selbstständig verfasst habe und dass alle wörtlich oder sinngemäß übernommenen Stellen in der Arbeit als solche gekennzeichnet sind.

Karlsruhe, den 11.11.2013

Zusammenfassung

In dieser Arbeit wird das Problem betrachtet, in einem planaren Graphen einen möglichst großen induzierten Baum zu finden. Es wird bewiesen, dass dieses Problem $\mathcal{NP}$-vollständig ist. Zusätzlich werden obere und untere Schranken für die Größe induzierter Bäume in bestimmten Graphklassen gezeigt. Dabei wird bewiesen, dass in einem vollständigen 3-Baum ein induzierter Baum existiert, der mindestens die Hälfte der Knoten enthält. Für unvollständige 3-Bäume wird ein effizienter Algorithmus zum Berechnen eines maximalen induzierten Baums angegeben. Als Anwendung wird eine Möglichkeit vorgestellt, bestehende Graphvisualisierungs-Algorithmen für planare Graphen so zu erweitern, dass sie auch bei kleiner äußerer Facette kompakte Zeichnungen liefern. Dabei sollen mithilfe eines induzierten Baums im Dualgraphen Kanten gefunden werden, die temporär gelöscht werden. Daraufhin zeichnet man den entstehenden Graphen mit dem Zeichenalgorithmus. Am Ende werden die gelöschten Kanten kreuzungsfrei in die Zeichnung wieder eingefügt.

Inhaltsverzeichnis

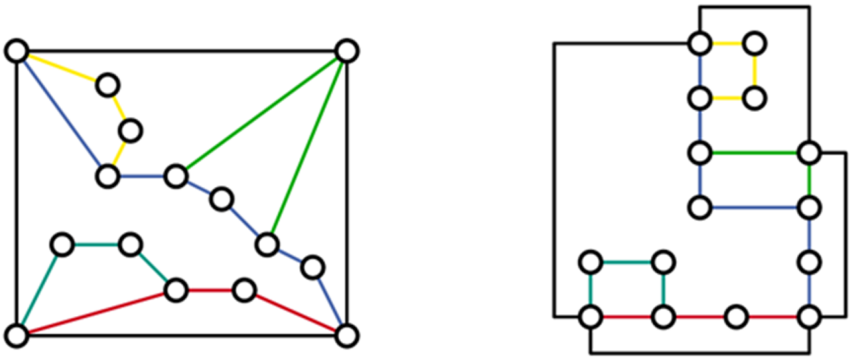
1. Einleitung

In Beruf und Alltag werden Menschen oft mit großen Mengen an Daten konfrontiert. Diese können wir in ihrer Rohform, oft Tabellen voller Zahlen, nur schwer erfassen. Deshalb ist es sinnvoll, diese Daten durch Grafiken zu visualisieren. Viele der Daten lassen sich dabei auf natürliche Weise als Graphen darstellen, wie zum Beispiel das Straßennetz oder die Beziehungen von Menschen in sozialen Netzwerken. Diese Struktur soll sich auch in der Zeichnung widerspiegeln. Aus diesem Grund entwickelt man Algorithmen, die Zeichnungen von Graphen erstellen. Häufig schränkt man sich dabei auf *planare Graphen* ein, also Graphen, die man ohne Kreuzungen in die Ebene zeichnen kann. Allerdings sind die betrachteten Graphen oftmals nicht planar. Da Kreuzungen für Menschen störend sind, versucht man, diese bei Zeichnungen zu minimieren. Eine Technik hierzu wurde von Gutwenger et al. [GMW00] untersucht. Anstelle der Kreuzungen werden dann Dummyknoten eingefügt, wodurch der Graph planar wird und mit den entsprechenden Algorithmen gezeichnet werden kann. Diese Technik wird als *Planarisierung* bezeichnet.

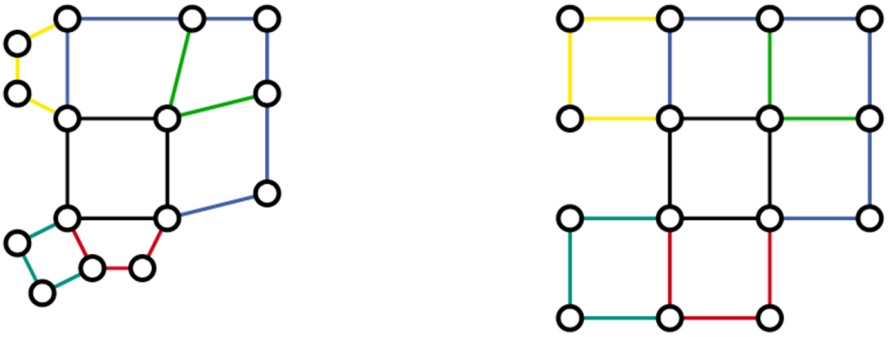
Aufgrund der Fülle der existierenden Visualisierungsalgorithmen sollen hier nur einige Beispiele vorgestellt werden. Der Algorithmus von Tutte [Tut63] ist ein klassischer Algorithmus für das Erstellen von geradlinigen Layouts, welcher für dreifach knotenzusammenhängende Graphen funktioniert. Allerdings kann der benötigte Platz exponentiell in der Knotengröße wachsen. Schnyder [Sch90] gibt einen verbesserten Algorithmus dafür an, welcher in Linearzeit ein geradliniges Layout auf einem Gitter mit linearer Höhe und Breite berechnet. Der Algorithmus von Tamassia [Tam87] ist dagegen ein Standardbeispiel für die Berechnung eines Orthogonallayouts, bei dem alle Knoten und Kanten auf ein rechtwinkliges Gitter gezeichnet werden, wobei die Kanten auch Knicke haben können. Bei fester Einbettung kann ein solches Layout mit minimaler Knickzahl in $O(n^2 \log n)$ berechnet werden.

Viele dieser Algorithmen können allerdings schlechtere Zeichnungen liefern, falls die äußere Facette sehr klein ist. Als Illustration sollen hier zwei Bilder[1] gezeigt werden. In Abb. 3.1 ist das orthogonale Layout eines Graphen zu sehen, wobei die äußere Facette mit vier Knoten sehr klein ist. Der Graph in Abb. 1.1b ist der gleiche Graph wie der andere, allerdings hier mit einer anderen Einbettung. Die äußere Facette ist hier mit 14 Knoten sehr groß, was eine wesentlich bessere Zeichnung erlaubt. Dieses Problem trifft auf praktisch alle Zeichenalgorithmen zu. Hier soll diese Arbeit ansetzen. Vorgestellt wird hier, wie man bestehende Visualisierungsalgorithmen erweitern kann, so dass in diesen Fällen

[1] Bilder von: http://i11www.iti.uni-karlsruhe.de/_media/teaching/winter2011/graphdrawing/ueb02_slides.pdf, S. 51



(a) Die äußere Facette hat nur vier Knoten.



(b) Die äußere Facette hat 14 Knoten.

Abbildung 1.1: Bei demselben Graphen entsteht durch die Wahl einer Einbettung mit großer äußerer Facette eine bessere Zeichnung.

die Zeichnungen kompakter werden. Da eine kleine äußere Facette das Problem ist, versucht man diese durch das Entfernen von Kanten temporär möglichst groß zu machen. Die Menge der Kanten, die entfernt werden dürfen, wird eingeschränkt, denn dabei sollen nicht zu viele strukturelle Eigenschaften des Graphen verloren gehen. Dafür benötigt man eine Kantenauswahlstrategie. Diese Kanten werden dann entfernt und der Restgraph mit einem bestehenden Visualisierungsalgorithmus gezeichnet. Daraufhin braucht man noch einen Algorithmus, der die gelöschten Kanten wieder in die bestehende Zeichnung einfügt.

In dieser Arbeit werden die zu entfernenden Kanten über einen induzierten Baum im Dualgraphen berechnet. Damit bleibt der Graph zusammenhängend und es entstehen keine neuen Brücken. Dabei ist der *Dualgraph* eines Graphen bezüglich einer festen Einbettung der Graph, welcher entsteht, wenn man jede Facette des Graphen durch einen Knoten darstellt und zwei Knoten genau dann verbindet, wenn die dazugehörigen Facetten im Ursprungsgraphen benachbart sind. Ein Baum T als Subgraph eines Graphen G heißt *induziert*, wenn T der durch seine Knotenmenge induzierte Subgraph von G ist. Induzierte Bäume sind bereits seit mehreren Jahrzehnten Gegenstand der Forschung. Dies ist ein interessantes Thema, zu dem es bereits viele Ergebnisse gibt. Allerdings lag der Fokus bisher auf allgemeinen Graphen. Dabei wird folgendes Problem betrachtet:

Definition 1.1 (Maximum Induced Tree (MIT))**.** *Gegeben sei ein einfacher, ungerichteter Graph $G = (V, E)$. Gesucht ist eine Knotenmenge $\tilde{V} \subseteq V$ maximaler Kardinalität, so dass der von $\tilde{V}$ induzierte Subgraph ein Baum ist.*

Ebenso wie das Problem MIT lässt sich das Problem Planar Maximum Induced Tree (PMIT) definieren. Hierbei wird zusätzlich noch gefordert, dass G planar ist. Hierüber gibt

es bisher allerdings noch nicht so viele Erkenntnisse. Deshalb werden in dieser Arbeit zuerst einige theoretische Eigenschaften des Problems PMIT untersucht. Zuerst wird dessen $\mathcal{NP}$-Vollständigkeit gezeigt. Daraufhin werden Schranken für die Größe von induzierten Bäumen in speziellen Graphklassen bewiesen.

Als Verallgemeinerung von MIT und PMIT lassen sich Probleme betrachten, bei denen in einem Graphen maximale induzierte Subgraphen mit bestimmten Eigenschaften gesucht werden.

1.1 Verwandte Arbeiten

Für allgemeine Graphen wurde von Erdős et al. [ESS86] die $\mathcal{NP}$-Vollständigkeit von MIT bewiesen sowie verschiedene Schranken formuliert. Beispielsweise gibt es für $c > 1$ zusammenhängende Graphen mit n Knoten und cn Kanten, so dass ein maximaler induzierter Baum nur $2 \log \log n + O(\log \log \log n)$ Kanten enthält. Für dünne zufällige Graphen hat Vega [dlV96] bewiesen: Der zufällige Graph mit n Knoten und Kantenwahrscheinlichkeit $\frac{c}{n}$, $c > 1$, enthält einen induzierten Baum der Größe $\alpha_c n$, wobei $\alpha_c > 0$ nur von c abhängt, mit einer Wahrscheinlichkeit, die für $n \to \infty$ gegen 1 konvergiert. Matoušek und Šámal [MŠ07] haben gezeigt, dass ein dreiecksfreier Graph immer einen induzierten Baum der Größe $e^{c\sqrt{n}}$ enthält. Diese untere Schranke wurde von Fox et al. [FLS09] auf $\sqrt{n}$ verbessert. Außerdem wurde gezeigt, dass in Graphen, welche keine Clique der Größe $r \geq 4$ enthalten, ein induzierter Baum der Größe $\frac{\log n}{4 \log r}$ existiert. Da planare Graphen keine Clique der Größe 5 enthalten, folgt daraus automatisch, dass sich in einem planarer Graphen immer ein induzierter Baum logarithmischer Größe finden lässt. Für dreiecksfreie Graphen G wurde von Pfender [Pfe10] gezeigt, dass auch, wenn für einen Knoten $v \in V(G)$ gefordert wird, dass dieser im induzierten Baum ist, sich ein solcher mit Größe $\lceil \frac{1}{2}(1 + \sqrt{8n-7}) \rceil$ finden lässt. Scott [Sco97] gibt eine Verbindung von induzierten Bäumen zur chromatischen Zahl eines Graphens an: für jeden Baum T und jede natürliche Zahl k existiert $g(k, T)$, so dass jeder Graph mit chromatischer Zahl $\chi(G) > g(k, T)$ eine Clique aus k Knoten oder eine induzierte Kopie einer Unterteilung von T enthält.

Sehr ähnlich zu MIT ist das Problem, einen maximalen induzierten Wald zu finden. Alon et al. [AMT01] haben bewiesen, dass für einen Graph G mit n Knoten, e Kanten und Knotengrad $\Delta \leq 3$ ein induzierter Wald der Größe $n - \frac{e}{4} - \frac{1}{4}$ existiert. Für $\Delta \geq 3$ wurde ein Zusammenhang mit $\alpha(G)$, der Größe des maximalen Independent Set von G, gezeigt. Dann lässt sich ein induzierter Wald mit Größe $\alpha(G) + \frac{n - \alpha(G)}{(\Delta-1)^2}$ finden. Von Pelsmajer [Pel04] wurde bewiesen, dass jeder einfache außenplanare Graph mit n Knoten einen induzierten linearen Wald mit mindestens $\lceil \frac{4n+2}{7} \rceil$ Knoten besitzt und dass diese Grenze scharf ist. Für planare dreiecksfreie Graphen haben Kowalik et al. [KLŠ10] gezeigt, dass ein induzierter Wald mit einer Größe von mindestens $\frac{71n+72}{128}$ existiert.

Neben induzierten Wäldern kann man auch induzierte Pfade betrachten. Arocha und Valencia [AV00] haben dabei gezeigt, dass $p_n \to \infty$ für $n \to \infty$, wobei p_n das Minimum der maximalen Länge eines induzierten Pfades über alle dreifach knotenzusammenhängende planare Graphen mit n Knoten ist. Di Giacomo et al. [DGLM] haben gezeigt, dass es mit vollständigen 3-Bäumen planare Graphen gibt, in denen der maximale induzierte Pfad nur logarithmische Größe hat.

Cameron [Cam89] hat gezeigt, dass das Problem, ein maximales induziertes Matching zu finden, $\mathcal{NP}$-Vollständig ist. Duckworth et al. [DMZ05] konnten dies noch verschärfen, indem sie zeigten, dass auch die Approximation für dieses Problem $\mathcal{NP}$-schwer ist: Es existiert eine Konstante c, so dass eine Approximation mit Faktor c für $3s$-reguläre Graphen, $s \geq 1$, $\mathcal{NP}$-schwer ist. Im Gegensatz dazu konnte Zito [Zit00] beweisen, dass auf Bäumen für dieses Problem ein Linearzeitalgorithmus existiert. Von Cameron und Walker [CW05]

kommt eine vollständige Charakterisierung aller Graphen, in denen ein maximale Größe von Matchings gleich der von induzierten Matchings ist.

Des Weiteren haben Angelini et al. [AEFG13] bewiesen, dass jeder planare Graph mit n Knoten einen induzierten außenplanaren Graphen mit mindestens $\frac{n}{2}$ Knoten enthält.

1.2 Ergebnisse und Struktur dieser Arbeit

Der Fokus dieser Arbeit liegt in der theoretischen Untersuchung des Problems PMIT, zusätzlich wird das Ergebnis einer ersten Implementierung für den Visualisierungsalgorithmus vorgestellt. Zuerst wird in Kapitel 2 der Beweis der $\mathcal{NP}$-Vollständigkeit von allgemeinen auf planare Graphen erweitert. Danach sollen in Kapitel 3 spezielle Graphklassen betrachtet werden. In Abschnitt 3.1 wird ein Graph mit einem kleinen induzierten Baum konstruiert. In Abschnitt 3.2 wird für vollständige 3-Bäume bewiesen, dass für sie ein induzierter Baum existiert, der die Hälfte der Knoten enthält. Für unvollständige 3-Bäume wird in Abschnitt 3.3 ein Algorithmus vorgestellt, welcher in Polynomialzeit einen optimalen induzierten Baum berechnet. Zusätzlich wird in Abschnitt 3.4 gezeigt, dass unter bestimmten Voraussetzungen auch in teilweise vollständigen Teilbäumen ein induzierter Baum existiert, der die Hälfte der Knoten enthält. In Kapitel 4 werden zuerst zwei Heuristiken, welche einen großen induzierten Baum berechnen sollen, vorgestellt und ausgewertet. Dann wird als Anwendung ein darauf aufbauender Graphvisualisierungs-Algorithmus gezeigt, der die Zeichnungen von Graphen kompakter machen und damit das Problem lösen soll, das diese Arbeit motiviert hat.

2. Komplexität

In diesem Kapitel soll die Komplexität des Problems, einen maximalen induzierten Baum zu finden, betrachtet werden. Dazu wird zuerst der schon bekannte Beweis vorgestellt, dass dies in allgemeinen Graphen $\mathcal{NP}$-vollständig ist. Dieser Beweis wird daraufhin auf planare Graphen übertragen, womit gezeigt wird, dass die Bestimmung eines maximalen induzierten Baums auch in planaren Graphen $\mathcal{NP}$-vollständig ist.

2.1 Maximum Induced Tree

Zuerst definieren wir das zu MIT gehörende Entscheidungsproblem.

Definition 2.1 (MIT (Entscheidungsproblem))**.** *Gegeben sei ein einfacher, ungerichteter Graph $G = (V, E)$ und ein Parameter k. Gibt es eine Knotenmenge $W \subseteq V$ mit $|W| \geq k$, sodass der von W induzierte Subgraph ein Baum ist?*

Lemma 2.2. *$MIT \in \mathcal{NP}$.*

Beweis. Gegeben eine Knotenmenge $W \subseteq V$ lässt sich in Polynomialzeit überprüfen, ob $|W| \geq k$ und ob W in G einen Baum induziert, wie z.B. mit Tiefensuche. □

Zum Beweis der $\mathcal{NP}$-Schwere werden wir ein verwandtes Problem verwenden:

Definition 2.3 (Constrained Maximum Induced Tree (CMIT))**.** *Gegeben sei ein einfacher, ungerichteter Graph $G = (V, E)$, eine Knotenmenge $C \subseteq V$ und ein Parameter k. Gefragt ist, ob es eine Knotenmenge W mit $C \subseteq W \subseteq V$ und $|W| \geq k$ gibt, sodass der von W induzierte Subgraph ein Baum ist.*

Ebenso bekommt man das Problem PCMIT, wenn man von G noch Planarität fordert. Die Knoten aus C nennen wir *Pflichtknoten*.

Satz 2.4. *Es gilt $CMIT \propto MIT$ und $PCMIT \propto PMIT$.*

Beweis. Gegeben sei eine Instanz $(G = (V, E), k, C)$ von CMIT bzw. PCMIT. Hänge an jeden Knoten von C einen eigenen Pfad P_v aus $n := |V|$ Knoten an. Dies liefert einen Graphen G'. Offensichtlich ist G' planar, falls G planar war. Sei nun $j := |C|$. Falls $k < j$, setze $k := j$, denn jeder Baum muss alle Knoten aus C, also mindestens j Knoten erhalten.

Wähle nun $k' = k + j \cdot n$. Die neue Instanz von MIT bzw. PMIT sei dann (G', k'). Dies geht in Polynomialzeit.

Nun wird gezeigt, dass die beiden Instanzen tatsächlich äquivalent sind. Sei dazu ein Baum $T = (V_T, E_T)$ der Größe k in G gegeben, der alle Knoten in C enthält. Dann kann man aber die bei der Transformation hinzugefügten Pfade für einen Baum in G' zu T hinzufügen. Die angehängten Pfade können keinen neuen Kreis induzieren, da jeder innere Knoten des Pfads nur zu den zwei Nachbarn inzident ist, der eine Randknoten nur mit dem Nachbarn und dem einen der Pflichtknoten aus C, und der andere Randknoten nur mit seinem einzigen Nachbarn. Die Knotenmenge bleibt auch zusammenhängend, da die Pfade von einem Knoten $v \in C \subseteq V_T$ erreichbar sind. Außerdem enthalten die Pfade zusammen $j \cdot n$ Knoten, womit der neue Baum die gewünschte Anzahl an Knoten erreicht.

Sei umgekehrt $T' = (V_T, E_T)$ ein Baum der Größe $k' \geq k + j \cdot n$ in G'. Falls $C = \emptyset$, ist $G' = G$ und $k' = k$. Damit erfüllt T' schon die gewünschten Bedingungen. Sei also nun $C \neq \emptyset$. Dann ist $k' = k + j \cdot n > j \cdot n$. Wären nur Knoten aus den Pfaden P_v in V_T, so wäre $|V_T| \leq j \cdot n < k'$, ein Widerspruch. Also existiert ein Knoten $v \in V \cap V_T$. Angenommen, aus einem der Pfade P_v wäre kein Knoten in V_T. Dann ist aber $|V_T| \leq n + (j-1)n = j \cdot n < k'$, was erneut ein Widerspruch ist. Da somit ein Knoten aus G in V_T ist, für jedes $v \in C$ ein Knoten in V_T ist und ein Pfad P_v mit dem Graphen G nur über v verbunden ist, muss aufgrund des Zusammenhangs von T' auch jeder Knoten $v \in C$ in V_T enthalten sein. Es sei T der durch $V_T \cap V$ induzierte Subgraph. Dann ist T zusammenhängend, da T' zusammenhängend ist und man T aus T' durch wiederholtes Löschen von Grad-1-Knoten erhalten kann, und zwar vom Ende der Pfade her. Außerdem ist T kreisfrei, da T' kreisfrei ist und durch das Entfernen von Knoten und Kanten kein neuer Kreis entstehen kann. Also ist T ein Baum. Außerdem gibt es genau $j \cdot n$ hinzugefügte Pfadknoten. Da $|V_T| \geq k + j \cdot n$, ist die Anzahl der Knoten in $V_T \cap V$ tatsächlich mindestens $k + j \cdot n - j \cdot n = k$. □

Von Erdős et al. [ESS86] wurde bewiesen, dass MIT $\mathcal{NP}$-vollständig ist. Da dieser Beweis später auf den Beweis der $\mathcal{NP}$-Vollständigkeit von PMIT übertragen wird, soll er hier noch einmal beschrieben werden. Im ursprünglichen Beweis wurde das Problem CMIT nicht explizit verwendet, nur die Idee dahinter. Da es jedoch bei PMIT eine große Hilfe sein wird, wird es hier in den Beweis eingebaut.

Satz 2.5. *CMIT ist $\mathcal{NP}$-schwer.*

Beweis. Wir reduzieren das $\mathcal{NP}$-vollständige Problem Independent Set auf CMIT. Sei dazu eine Instanz $(G = (V, E), k)$ von Independent Set gegeben. Sei $G' := (V', E')$ mit $V' := V \dot\cup \{v'\}$ mit einem neuen $v' \notin V$ sowie $E' := E \cup \{\{u, v'\} \mid u \in V\}$ und $k' = k+1$. Als einzigen Pflichtknoten wählen wir v'. Die Instanz von CMIT sei $(G', k', \{v'\})$. Offensichtlich ist die Transformation in Polynomialzeit möglich.

Es muss nun die Äquivalenz der beiden Instanzen gezeigt werden. Sei dazu $T = (W, \tilde{E})$ ein Baum in G' mit mindestens k' Knoten. Nach Konstruktion ist $v' \in W$. Betrachte nun eine Kante $e = \{u, v\}$ im ursprünglichen Graphen. Wären $u, v \in W$, so wären $\{u, v'\}, \{v, v'\}, \{u, v\} \in \tilde{E}$. Da dies ein Dreieck ist, wäre T kein Baum. Also ist höchstens ein Endknoten von e in W. Damit liefert $W \cap V$ ein Independent Set in G. Der Baum T hat nach Voraussetzung in G' mindestens $k' = k + 1$ Knoten, und damit in G mindestens k, weil $|G' \setminus G| = |\{v'\}| = 1$. Damit ist unser Independent Set groß genug. Sei umgekehrt ein Independent Set W mit $|W| \geq k$ in G gegeben. Dann induziert aber $W \cup \{v'\}$ einen Baum T, denn zwischen den Knoten aus W verlaufen keine Kanten, und zwischen jedem Knoten aus W und v' verläuft eine Kante. Damit ist T zusammenhängend, und da jeder Knoten aus W in T Grad 1 hat, muss T kreisfrei sein, weil für einen Kreis mindestens 3 Knoten einen Grad ≥ 2 haben müssten. Außerdem hat T mindestens $k + 1 = k'$ Knoten.

Somit ist CMIT $\mathcal{NP}$-schwer. □

Zusammen mit Lemma 2.2 und Satz 2.4 folgt das folgende Korollar.

Korollar 2.6. *MIT ist $\mathcal{NP}$-vollständig.*

Aus dem Beweis folgt, dass das Problem auch $\mathcal{NP}$-vollständig ist, wenn man einen maximalen induzierten Stern sucht. Denn der konstruierte Baum ist dort sogar immer ein Stern.

2.2 Maximum Induced Tree in planaren Graphen

Die im vorigen Beweis angegebene Reduktion funktioniert für das Problem PMIT nicht direkt, da die Planarität bei der Konstruktion von G' verloren gehen kann. Mit einigen technischen Schritten lässt sich der Beweis jedoch auf den planaren Fall übertragen. Damit gewinnt man den folgenden Satz:

Satz 2.7. *PMIT ist $\mathcal{NP}$-vollständig.*

PMIT ist ein Spezialfall von MIT $\in \mathcal{NP}$ und damit ebenfalls in $\mathcal{NP}$. Nun reduzieren wir Planar Independent Set auf PCMIT, wobei Planar Independent Set $\mathcal{NP}$-vollständig ist. Im ersten Schritt konstruieren wir dabei die Reduktion.

Gegeben sei eine Instanz $(G = (V, E), k)$ von Planar Independent Set. Zuerst bestimmen wir eine beliebige planare Einbettung von G und halten diese fest. Im folgenden betrachten wir G immer mit dieser Einbettung. Nun erstellen wir einen neuen Knoten $v' \notin V$ und verbinden ihn mit allen anderen Knoten $v \in V$ durch eine Kante $e_v = \{v', v\}$, sodass sich diese neuen Kanten nicht kreuzen und dass jede neue Kante jede Kante $e \in E$ höchstens einmal schneidet; siehe Abb. 2.1. Dies ist offensichtlich möglich. Wenn man sich eine dieser Kanten in der Richtung von v' nach v anschaut, kann man die linke und rechte Seite davon unterscheiden. Dadurch geht eventuell die Planarität verloren. Deshalb ersetzen wir jede Kreuzung durch einen Dummy-Knoten. Dann ist der entstehende Graph planar.

Wenn auf einer Kante $e = \{u, v\} \in E$ ein Kreuzungsknoten ist und dieser nicht zu W hinzugefügt wird, könnten u und v beide in W gewählt werden, was bei einem Independent Set aber nicht sein darf. Daher wird jeder Kreuzungsknoten w in einen linken und rechten Knoten gesplittet. Der linke Knoten wird dabei mit dem auf e nächsten Knoten auf der linken Seite verbunden, der rechte mit dem auf e nächsten auf der rechten Seite. Die durch Kreuzungsknoten unterteilten Kanten von v' zu den anderen Knoten aus G werden verdoppelt, sodass der eine Pfad durch die linken, der andere durch die rechten Knoten geht. Die durch das Splitten entstehenden zwei Knoten nennen wir ein *Unterteilungspaar*, die Anzahl an Unterteilungspaaren werde mit Δ bezeichnet. Die Knoten aus einem Unterteilungspaar werden durch eine Kante verbunden; siehe Abb. 2.2. Jede Kante $e\{u, v\} \in E$ ist somit durch einen Pfad P_e von u nach v ersetzt worden, der u, die Knoten der Unterteilungspaare und v enthält. Um zu garantieren, dass von jeder Kante höchstens ein Endknoten gewählt werden kann, muss aus jedem Unterteilungspaar genau ein Knoten in den Baum genommen werden. Daher fügen wir einen Pfad der Länge zwei zwischen den Knoten eines Unterteilungspaar ein. Da eine Kante zwischen diesen besteht, bleibt die Planarität erhalten. Den mittleren Knoten des Pfads bezeichnen wir als *s-Knoten*; siehe Abb. 2.3.

Nun existieren zwischen zwei auf einem Pfad von v' zu einem $v \in V$ aufeinanderfolgenden Unterteilungspaaren zwei nebeneinander verlaufende Kanten. Daher beeinflusst die Wahl der Knoten aus einem Unterteilungspaar einer Kante die Wahl auf der nächsten Kante, wodurch auch die Wahl der Endknoten dieser Kanten voneinander abhängig wird. Dadurch

könnte es sein, dass eine Instanz von Planar Independent Set keine gültige Lösung von PCMIT induziert. Wir betrachten deshalb für jedes Auftreten der vorher genannten Situation die vier Knoten der beiden Unterteilungspaare. Nach der bisherigen Konstruktion begrenzen diese Knoten eine Facette. In diese Facette wird ein neuer Knoten, *t-Knoten* genannt, eingefügt. Die beiden Kanten auf dem Pfad $v' \rightsquigarrow v$ werden entfernt, stattdessen wird der t-Knoten mit den vier Knoten verbunden; siehe Abb. 2.4.

Dies liefert einen Graphen G'. In C seien genau v' und die in der Konstruktion eingefügten s- und t-Knoten. Setze k' auf $k + |C| + \Delta$. Wähle (G', k', C) als Instanz von PCMIT. Diese Reduktion läuft in Polynomialzeit. Im folgenden wird ihre Korrektheit bewiesen.

Lemma 2.8. *Falls $W \subseteq V$ ein* Independent Set *der Größe k in G ist, dann existiert ein induzierter Baum V' der Größe k' in G' mit $C \subseteq V'$.*

Beweis. Wir fügen alle Knoten aus $C \cup W$ zu V' hinzu. Sei nun $e = \{u, w\} \in E$. Dann ist höchstens einer der beiden Knoten u, w in W, da dies ein Independent Set ist. Nun wählen wir aus jedem Unterteilungspaar auf P_e genau einen Knoten aus und fügen ihn zu V' hinzu, und zwar so, dass keine zwei benachbarte Knoten auf P_e in V' sind. Falls einer der beiden Knoten u, v in W ist, wird dabei von jedem Unterteilungspaar x, y der Knoten zu V' hinzugefügt, der auf der Kante weiter von u entfernt liegt; siehe Abb. 2.5. Offensichtlich sind alle Knoten aus C und W gewählt, genauso wie die Hälfte der Unterteilungsknoten. Damit ist $|V'| \geq |C| + |I| + \Delta$.

Es bleibt zu zeigen, dass der von V' induzierte Subgraph T tatsächlich ein Baum ist. Dazu zeigen wir nacheinander, dass T zusammenhängend und kreisfrei ist.

Für den Zusammenhang reicht wegen $v' \in C \subseteq V'$ zu zeigen, dass jeder Knoten $v \in W'$ von v' erreichbar ist. Hierbei unterscheiden wir verschiedene Fälle, je nach Art des Knotens v. Sei zuerst v ein t-Knoten oder ein gewählter Knoten eines Unterteilungspaars. Jeder t-Knoten ist in V', genauso wie einer der Knoten aus jedem Unterteilungspaar. Aufgrund der Konstruktion erhält man für jeden Konten $w \in V$ einen Pfad auf der Expansion der temporär zwischen v' und w eingefügten Kante, welcher auf der abwechselnd aus t-Knoten und Unterteilungspaarknoten besteht. Dann existiert ein Pfad von v' zu v. Falls v ein s-Knoten ist, ist dieser zu den zwei Knoten eines bestimmten Unterteilungspaars adjazent. Da davon einer gewählt wurde und eben gezeigt wurde, dass dieser von v' erreichbar ist, gilt dies auch für v. Ansonsten ist $v \in V$. Falls v direkt zu v' inzident ist, gilt die Behauptung trivialerweise. Ansonsten ist v zu den zwei Knoten desjenigen Unterteilungspaars inzident, welches durch Splitten des letzten Kreuzungsknotens auf der ursprünglichen Kante von v' nach v entstanden ist. Wie im vorigen Absatz folgt dann, dass v' mit v verbunden ist. Damit ist T zusammenhängend.

Nun wird gezeigt, dass T kreisfrei ist. Die s-Knoten sind nur zu den Knoten $u \in V'$ und $w \notin V'$ eines Unterteilungspaares adjazent, haben damit Grad 1 in T und können daher nicht Teil eines Kreises sein. Daher kann man sie im Folgenden ignorieren. Für eine Kante $e \in E$ sind nach Konstruktion von V' keine zwei benachbarte Knoten auf P_e in V'. Damit ist auch keine Kante auf P_e in dem durch V' induzierten Baum. Zusätzlich bilden sich auf den ehemaligen Kanten $\{v', v\}$ für $v \in V$ von v' ausgehende Pfade, die abwechselnd Unterteilungsknoten und t-Knoten enthalten; und eventuell noch v. Insgesamt erhält man also eine Unterteilung des Sterns $K_{1,|V|}$ mit Zentrum v'. Da dieser kreisfrei ist, muss auch T kreisfrei sein.

Also ist T ein Baum der richtigen Größe. □

Nun bleibt noch die andere Richtung der Korrektheit zu zeigen.

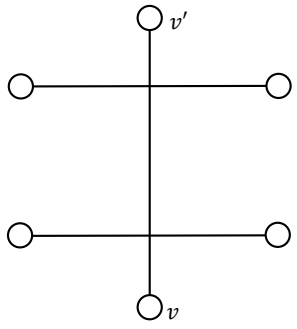


Abbildung 2.1: Nach dem Einfügen von v' entstehen Kreuzungen.

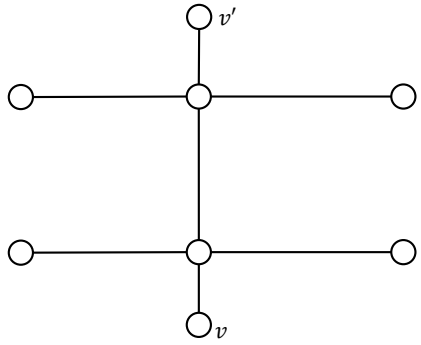


Abbildung 2.2: Die Kreuzungsknoten werden eingefügt.

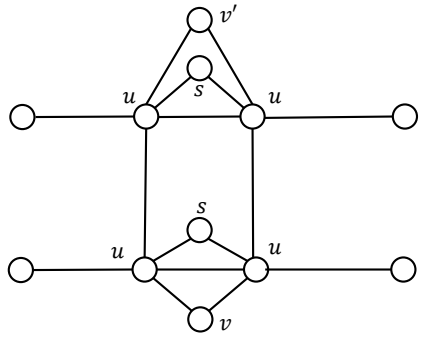


Abbildung 2.3: Die Kreuzungsknoten werden expandiert: u=Knoten eines Unterteilungspaars, s=s-Knoten

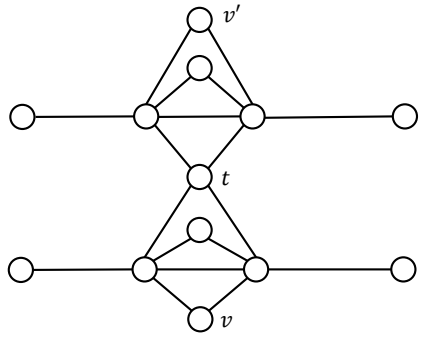


Abbildung 2.4: Ein t-Knotens t wird eingefügt.

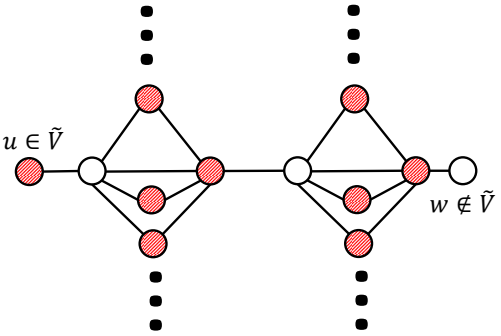


Abbildung 2.5: Die rot markierten Knoten gehören zu V'.

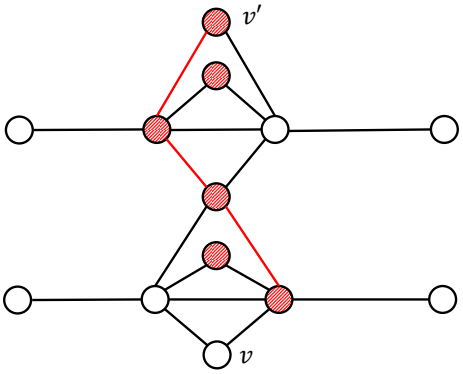


Abbildung 2.6: Die roten Kanten bilden den Pfad.

Lemma 2.9. *Sei V' eine Knotenmenge mit $C \subseteq V'$ mit mindestens k' Knoten, die einen Baum T in G' induziert. Dann ist $W := V \cap V'$ ein* Independent Set *in G mit $|W| \geq k$.*

Beweis. Wir zeigen hierfür folgende Behauptungen: Für jedes Unterteilungspaar ist genau einer der Knoten in V' enthalten, W ist ein Independent Set in G sowie $|W| \geq k$.

Sei ein Unterteilungspaar gegeben. Der s-Knoten s des Paars ist genau wie v' in $C \subseteq V'$. Da s nur zu den beiden Knoten des Paares adjazent ist sowie T zusammenhängend ist, muss auch einer der beiden Paarknoten in V' sein, denn sonst wären v' und s in T nicht verbunden. Wären beide Knoten des Unterteilungspaares in V', dann induzieren die beiden Knoten zusammen mit dem s-Knoten nach Konstruktion von G' ein Dreieck. Damit wäre aber T kein Baum. Somit ist aus jedem Unterteilungspaar genau ein Knoten in V'. Da außerdem jeder t-Knoten in V' ist, erhält man aufgrund der Konstruktion für jeden Konten $w \in V$ einen Pfad in T, welcher auf der bei der Konstruktion von G' unterteilten Kante zwischen v' und w verläuft. Dieser besteht abwechselnd aus t-Knoten und Unterteilungspaarknoten; siehe dazu Abb. 2.6.

Jetzt zeigen wir, dass W ein Independent Set ist. Wir führen einen Beweis durch Widerspruch und nehmen deshalb an, W wäre kein Independent Set. Dann existieren $u, v \in W$, sodass $e := \{u, v\} \in E$. Betrachte P_e. Die Anzahl der hierauf hinzugefügten Unterteilungspaare sei ℓ. Angenommen, zwei auf P_e adjazenten Knoten x und y wären beide in V'. Dann ist x ein Knoten aus V oder wegen der Konstruktion ein Knoten eines Unterteilungspaares. Für $x \in V$ ist x direkt zu v' inzident oder zu einem Knoten aus dem benachbarten Unterteilungspaar. Im zweiten Fall existiert aber ein Pfad $v' \rightsquigarrow x$. Falls x ein

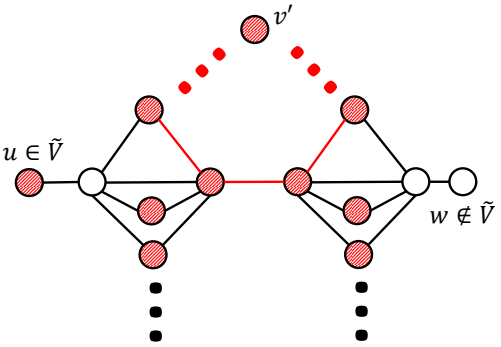


Abbildung 2.7: Ein induzierter Kreis, dargestellt durch rote Kanten, entsteht.

Knoten eines Unterteilungspaars ist, existiert ebenfalls ein Pfad $v' \rightsquigarrow x$. Damit existiert ein solcher Pfad in jedem Fall. Analog existiert ein Pfad $v' \rightsquigarrow y$. Da nun aber x und y auf dem P_e benachbart sind, wobei $\{x, y\}$ nicht in den vorher beschriebenen Pfaden vorkommt, induzieren die Pfade zusammen mit $\{x, y\}$ einen Kreis in T; siehe Abb. 2.7. Damit sind benachbarte Knoten auf P_e nicht beide in V'. Da P_e nach Konstruktion $2\ell + 2$ Knoten enthält, können höchstens $\ell + 1$ davon in V' sein. Aber für jedes der ℓ Paare ist ein Knoten in V'. Zusammen mit u und v sind dies mindestens $\ell + 2$ Knoten. Dies liefert uns den gewünschten Widerspruch.

Nun muss nur noch $|W| \geq k$ gezeigt werden. Von jedem Unterteilungspaar ist höchstens ein Knoten in V' enthalten. Von den bei der Konstruktion von $G' = (V_{G'}, E_{G'})$ hinzugefügten Knoten $V_{G'} \setminus V$ können somit nur v', die s- und t-Knoten sowie von jedem Unterteilungspaar ein Knoten in V' sein. Damit ergibt sich $|V' \cap (V_{G'} \setminus V)| \leq |C| + \Delta = k' - k$. Damit ist aber $|W| = |V' \cap V| = |V'| - |V' \cap (V_{G'} \setminus V)| \geq k' - (k' - k) = k$.

Insgesamt folgt daraus das Lemma. □

Wegen Lemma 2.8 und Lemma 2.9 gilt die Korrektheit der Reduktion. Somit ist PCMIT $\mathcal{NP}$-vollständig und damit nach Satz 2.4 auch PMIT.

3. Obere und untere Schranken in speziellen Graphklassen

Es gibt bereits diverse Untersuchungen zu oberen und unteren Schranken in allgemeinen Graphen. Nun soll versucht werden, derartige Ergebnisse auch für planare Graphen zu erhalten. Insbesondere soll versucht werden, Eigenschaften zu identifizieren, aus denen die Existenz eines Baums linearer Größe folgt. Abschnitt 3.1 wird Eigenschaften vorstellen, die dafür alleine nicht reichen. Im Abschnitt 3.2 wird diese Frage für vollständige 3-Bäume positiv beantwortet. Danach werden unvollständige 3-Bäume untersucht.

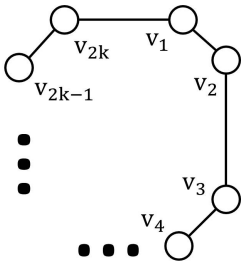
3.1 Ein Graph mit einem kleinen Baum

Erdős et al. [ESS86] haben gezeigt, dass es außenplanare Graphen mit Maximalgrad 4 gibt, die nur induzierte Bäume mit höchstens logarithmischer Größe enthalten. In diesem Abschnitt wird mit einer ähnlichen Konstruktion gezeigt, dass sogar Außenplanarität und Maximalgrad 3 zusammen nicht garantieren können, dass ein Baum linearer Größe entsteht.

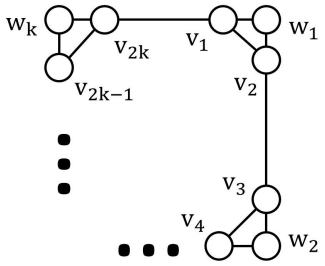
Lemma 3.1. *Für jedes $k \geq 3$ existiert ein außenplanarer Graph mit Maximalgrad 3 und $|V| = k^2$ Knoten, dessen maximaler induzierter Baum höchstens $4k-6 \in \Theta(\sqrt{|V|})$ Knoten enthält.*

Beweis. Für den Beweis wird zuerst ein derartiger Graph angegeben. Danach wird bewiesen, dass der maximale induzierte Baum die gewünschte Größe hat. Konstruiere dafür zuerst einen Kreis $v_1, v_2, \ldots, v_{2k}$ aus $2k$ Knoten; siehe Abb. 3.1a. Füge nun für jedes $i \in \{1, 2, \ldots, k\}$ einen Knoten w_i hinzu und verbinde ihn mit v_{2i-1} und v_{2i}. Nun bilden diese drei Knoten ein Dreieck; siehe Abb. 3.1b. Hänge zusätzlich an jeden Knoten w_i einen Pfad π_i der Länge $k-3$ an; siehe Abb. 3.1c. Offensichtlich hat jedes Dreieck zusammen mit dem dazugehörigen Pfad genau k Knoten, und da es k Exemplare davon gibt, hat der Graph k^2 Knoten. Außerdem ist der Graph außenplanar und jeder Knoten hat höchstens Grad 3.

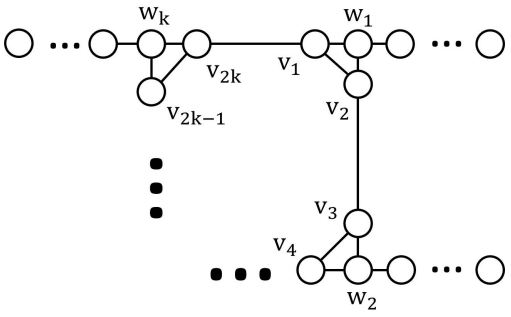
Nun wird die Größe eines induzierten Baums abgeschätzt. Von jedem Dreieck können höchstens zwei Knoten in einem induzierten Baum sein, da sonst ein Kreis entstehen würde. Wenn ein Knoten auf einem Pfad π_i sowie mindestens ein Knoten außerhalb des Pfades gewählt wurde, muss w_i auch im Baum vorhanden sein, da dieser zusammenhängend ist. Dann kann aber von den beiden anderen Knoten des Dreiecks nur einer gewählt worden sein.



(a) Es wird ein Kreis der Länge $2k$ erstellt.



(b) Die Dreiecke werden gebildet.



(c) Die Pfade π_i werden angefügt.

Abbildung 3.1: Der Graph wird mit den oben abgebildeten Schritten konstruiert.

Angenommen, es seien Knoten aus mindestens 3 Pfaden π_i, π_j und π_ℓ gewählt. Betrachten wir dazu erst π_i. Damit der Baum zusammenhängend bleibt, müssen wir ihn nun mit π_j und π_ℓ verbinden. Dies geht aufgrund der Struktur des Graphen nur über die Kreisknoten v_t. Gemäß obiger Überlegung kann diese Verbindung nur in eine der beiden Richtungen im Kreis fortgesetzt werden. Dies tut man, bis man den nächsten Pfad, o.B.d.A. sei dies π_j, erreicht. Dann aber sind bei π_i und π_j in den zugehörigen Dreiecken schon zwei Knoten gewählt. Um π_ℓ mit π_i und π_j zu verbinden, müsste aber ein weiterer Knoten aus diesen Dreiecken gewählt werden. Da dies nicht möglich ist, können höchstens aus zwei Pfaden Knoten gewählt werden. Von den Knoten der Dreiecke können höchstens $2k$ Knoten im Baum sein. Da höchstens zwei Pfade hinzukommen können und jeder Pfad $k-3$ Knoten hat, sind im Baum höchstens $2k + 2(k-3) = 4k-6$ Knoten enthalten. □

3.2 Vollständige 3-Bäume

Definition 3.2 (Vollständiger 3-Baum)**.** *Wir definieren den* vollständigen 3-Baum G_n *der Tiefe* $n \in \mathbb{N}_0$ *rekursiv. Es sei* $G_0 := K_3$*, also ein Dreieck. Die Knoten von* G_0 *nennen wir* Knoten von Level 0*. Für* $n \in \mathbb{N}$ *entstehe* G_n *aus* G_{n-1}*, indem man in jede Facette außer der äußeren einen Knoten, welchen wir* Mittelknoten des Dreiecks *nennen, setzt und diesen mit den drei an die Facette angrenzenden Knoten verbindet. Diese Knoten nennen wir dann* Knoten von Level n*.*

Satz 3.3. *In dem regelmäßigen 3-Baum* $G_n = (V_n, E_n)$ *gibt es einen induzierten Baum mit mindestens* $\frac{|V_n|}{2}$ *Knoten.*

Beweis. Zum Beweis betrachten wir folgende Konstruktion: Wir wählen als Startkonfiguration zwei beliebige der drei Knoten von Level $k = 0$, d.h. aus dem äußeren Dreieck, und

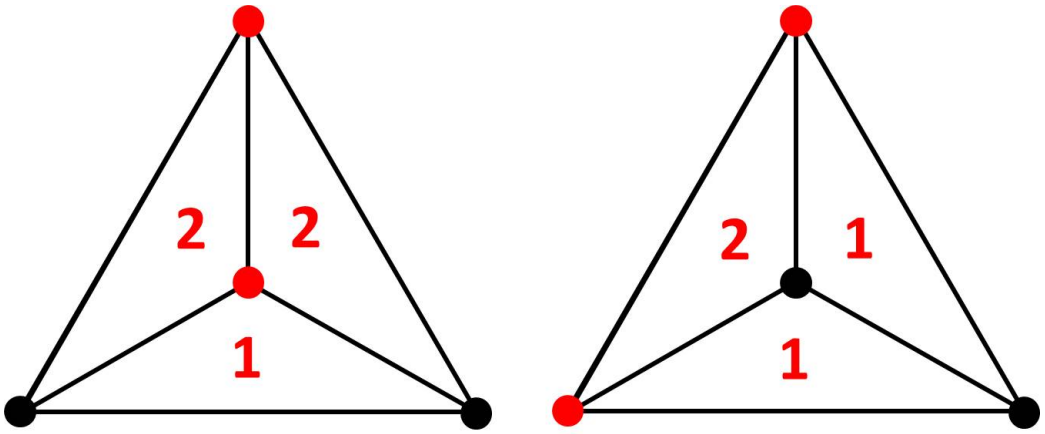


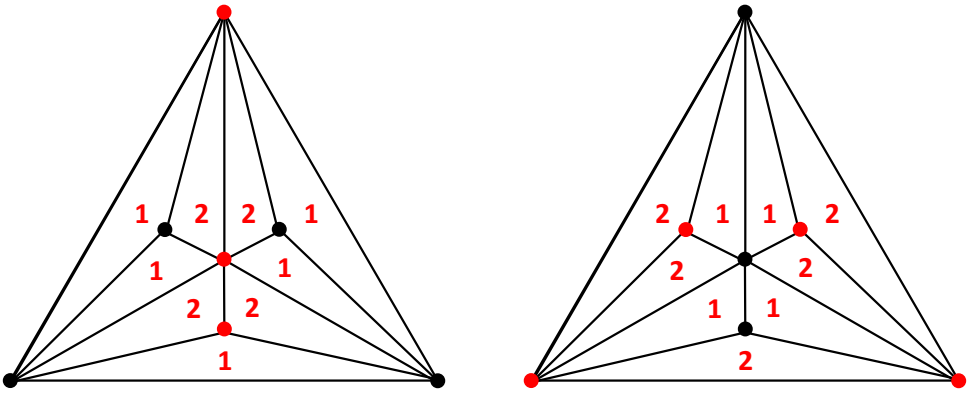
Abbildung 3.2: Gewählte Knoten sind rot markiert und die entstehenden Teildreiecke mit der Anzahl der gewählten Knoten bezeichnet.

fügen sie dem Baum hinzu. Nun fügen wir Knoten rekursiv, für jedes zuvor betrachtete Dreieck separat, hinzu. Falls in einem Dreieck keine inneren Knoten vorhanden sind, womit $k = n$ ist, bricht die Rekursion ab. Ansonsten hat der Graph noch Knoten vom Level $k+1$. Der Mittelknoten eines Dreiecks wird dabei genau dann gewählt, wenn genau einer der Außenknoten des Dreiecks gewählt ist; siehe Abb. 3.2.

Der von den gewählten Knoten induzierte Graph bleibt zusammenhängend, da ein hinzugefügter Mittelknoten eines Dreiecks t immer mit mindestens einem der Knoten aus den Eckknoten von t verbunden ist. Es können aufgrund der Konstruktion auch keine neuen Kreise induziert werden, denn wenn ein Mittelknoten von t hinzugefügt wird, wurde nur ein Eckknoten von t gewählt. Damit hat der Mittelknoten danach aber Grad 1. Somit kann man die Mittelknoten in umgekehrter Reihenfolge wieder löschen. Dann bleiben nur noch die zwei gewählten Knoten des äußeren Dreiecks, welche keinen Kreis mehr induzieren können. Außerdem haben alle drei Teildreiecke einen oder zwei Knoten ausgewählt, weshalb man rekursiv mit den Teildreiecken verfahren kann. k wird dafür um 1 erhöht.

Nun wird gezeigt, dass der konstruierte Baum tatsächlich die gewünschte Größe hat. Wir betrachten zuerst den Fall, dass n gerade ist. Nun können wir mit Rückwärtsinduktion zeigen, dass für ein Dreieck, das im i-ten Rekursionsschritt, wobei i gerade, unterteilt wurde, die Hälfte der inneren Knoten gewählt wurde. In Dreiecken der Tiefe n gibt es keine inneren Knoten, die Behauptung gilt dort also. Ansonsten sei t ein Dreieck, das im i-ten Rekursionsschritt, $i < n$ gerade, unterteilt wurde. Betrachte nun zwei Rekursionsschritte gleichzeitig. Die Abbildungen 3.3a und 3.3b zeigen die von der Zahl der gewählten Dreiecksknoten abhängige Konfiguration. Von den in den beiden Rekursionsschritten eingefügten Knoten werden immer zwei von vier, also genau die Hälfte gewählt. Für die Knoten in den inneren Dreiecken wurde aufgrund der Induktion ebenfalls die Hälfte der Knoten gewählt. Damit wurde jeder Knoten mit positivem Level genau einmal betrachtet. Bei den noch nicht betrachteten Knoten vom Level 0 wurde ein halber Knoten mehr als die Hälfte gewählt. Falls n gerade ist, ist damit mehr als die Hälfte der Knoten im endgültigen Baum und in dem Fall gilt Satz 3.3.

Für ungerades n betrachten wir die Situation nach dem ersten Rekursionsschritt. Für die drei Teildreiecke folgt nach dem vorigen Argument, dass genau die Hälfte der Knoten ausgewählt sind, weil die Tiefe ungerade und damit relativ zu den Teildreiecken gerade ist. Von den vier verbleibenden Knoten auf Level 0 und 1 ist aber ebenfalls die Hälfte ausgewählt, nämlich zwei der drei Knoten auf Level 0. Somit ist auch für ungerades n insgesamt die Hälfte der Knoten ausgewählt. □



(a) (b)

Abbildung 3.3: Die gewählten innere Knoten bei einem und bei zwei gewählten Außenknoten sind rot markiert.

3.3 Ein Algorithmus für unvollständige 3-Bäume

Zuerst betrachten wir eine alternative Sichtweise auf 3-Bäume: Stelle einen 3-Baum als Baum T dar, wobei ein Blatt μ einem Dreieck entspricht: $\Delta(\mu) = \{u, v, w\}$ sind die drei Knoten des Dreiecks, der dazugehörige Graph werde mit $G(\mu)$ bezeichnet. Für einen inneren Knoten μ mit Kindern μ_1, μ_2, μ_3 sei $\Delta(\mu) = \{u, v, w\}$, wobei $\Delta(\mu_1) = \{u, v, c\}$, $\Delta(\mu_2) = \{u, w, c\}$ und $\Delta(\mu_3) = \{v, w, c\}$. c ist dabei der Mittelknoten von μ und wird mit $M(\mu)$ bezeichnet. Der zu μ gehörende Graph $G(\mu)$ ist dann die Vereinigung von $G(\mu_1)$, $G(\mu_2)$ und $G(\mu_3)$, wobei gemeinsame Knoten und Kanten identifiziert werden. Der Baum T mit Wurzel μ heißt dann der zu $G(\mu)$ gehörende Baum.

Für den vollständigen 3-Baum G_k ist dann der dazugehörige Baum T ein Ternärbaum, in dem jeder innere Knoten genau drei Kinder hat und in dem alle Blätter die gleiche Tiefe k haben.

Definition 3.4 (Unvollständiger 3-Baum)**.** *Sei T ein Baum wie oben und μ die Wurzel von T. Falls in T jeder innere Knoten genau drei Kinder hat, aber nicht alle Blätter die gleiche Tiefe, so nennt man $G(\mu)$ einen* unvollständiger 3-Baum.

Auf unvollständigen 3-Bäumen kann das im vorigen Abschnitt beschriebene Verfahren schlechte Lösungen liefern. Um eine optimale Lösung in Polynomialzeit zu berechnen, bietet sich dynamische Programmierung an.

Gegeben sei dazu ein 3-Baum G der Tiefe k und T der zu G gehörende Baum. Für jeden Knoten μ aus T speichert man für jede Teilmenge $\Delta' \subseteq \Delta(\mu)$ die maximale Knotenmenge $V' \subseteq G(\mu)$, wobei V' einen Baum in $G(\mu)$ induziert und $V' \cap \Delta(\mu) = \Delta'$. Für die Wurzel aus T liefert das eine optimale Lösung.

Falls μ ein Blatt in T ist, so sind wir fertig, denn es können für keine Kombination innere Knoten gewählt werden.

Falls μ ein innerer Knoten ist, hat dieser Kinder μ_1, μ_2, μ_3. Wir berechnen für diese nun rekursiv die optimalen Lösungen. Nun geht man alle Teilmengen $\Delta' \subseteq \Delta(\mu)$ durch, wobei die drei Knoten nicht alle gewählt werden können, da diese sonst einen Kreis induzieren würden.

Für $|\Delta'| = 2$ kann der innere Knoten nicht gewählt werden. Es entstehen zwei Teildreiecke mit einem und eines mit zwei gewählten Knoten; siehe Abb. 3.4. Die entsprechenden Einträge für die Kinder geben an, welche Knoten darin jeweils gewählt werden sollen. Wähle als V' die Vereinigung dieser Knotenmengen mit Δ'.

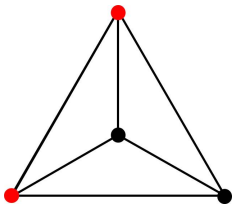
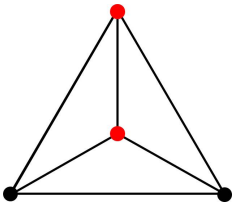
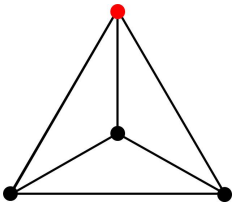


Abbildung 3.4: Die obige Konfigurationen entsteht bei einem gewählter Außenknoten.



(a) Der Mittelknoten wurde gewählt.



(b) Der Mittelknoten wurde nicht gewählt.

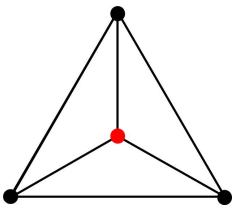
Abbildung 3.5: Die obigen Konfigurationen entstehen bei einem gewählter Außenknoten.

Für $|\Delta'| = 1$ müssen zwei Möglichkeiten überprüft werden: Wenn $M(\mu)$ gewählt wird, entstehen zwei Teildreiecke mit zwei und eines mit einem gewählten Knoten; siehe Abb. 3.5a. Wird $M(\mu)$ nicht gewählt, entstehen zwei Teildreiecke mit einem und eines mit keinem gewählten Knoten; siehe Abb. 3.5b. Aus letzterem werden keine weiteren Knoten gewählt. Vergleiche die beiden Möglichkeiten und wähle diejenige, die die höhere Zahl an inneren Knoten liefert, und setze V' entsprechend.

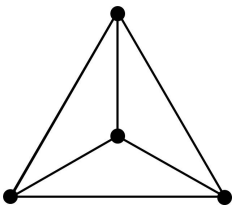
Auch für $|\Delta'| = 0$ müssen zwei Möglichkeiten überprüft werden: Wenn $M(\mu)$ gewählt wird, entstehen drei Teildreiecke mit einem gewählten Knoten; siehe Abb. 3.6a. Wird der mittlere Knoten nicht gewählt, entstehen drei Teildreiecke mit einem gewählten Knoten; siehe Abb. 3.6b. Hier muss man die entsprechenden Werte der Teildreiecke bei der Kombination, bei der nur $M(\mu)$ gewählt wurde, vergleichen und das Kind mit dem höheren Wert nehmen. Aus den anderen werden keine weiteren Knoten gewählt. Auch hier vergleicht man die beiden Möglichkeiten und wählt diejenige, die die höhere Zahl an inneren Knoten liefert. Zusätzlich wird V' entsprechend gesetzt.

Nach Abschluss des Algorithmus nimmt man sich den Eintrag der Wurzel von T und sucht nach der Kombination mit der höchsten Zahl gewählter Knoten.

Für den Beweis, dass der Algorithmus korrekt ist, wird zuerst gezeigt, dass die Konstruktion tatsächlich einen Baum liefert. Im Basisschritt werden keine neuen Knoten eingefügt, also kann die Baumeigenschaft dadurch nicht verloren gehen. In den sonstigen Schritten kann durch die Konstruktion offensichtlich kein neuer Kreis entstehen, da der Mittelknoten, falls er eingefügt wird, höchstens zu einem Außenknoten verbunden ist. Somit muss nur noch der Zusammenhang überprüft werden. Wird der mittlere Knoten nicht gewählt, so wird kein neuer Knoten hinzugefügt, diese Fälle sind daher unproblematisch. Wenn bei einem



(a) Der Mittelknoten wurde gewählt.



(b) Der Mittelknoten wurde nicht gewählt.

Abbildung 3.6: Die obigen Konfigurationen entstehen, wenn kein Außenknoten gewählt ist.

gewählten Außenknoten der Mittelknoten hinzugefügt wird, ist dieser mit diesem äußeren Knoten verbunden. Nur in dem Fall, dass kein Außenknoten gewählt wurde, ist der gewählte Mittelknoten mit keinem weiteren auf niedrigerem Level gewählten verbunden. Es muss nun gezeigt werden, dass außerhalb dieses Dreiecks t keine weiteren Knoten gewählt wurden. Die drei äußeren Knoten sind in diesem Fall nicht gewählt. Diese Konstellation tritt aber auch nur auf, falls im dem Dreieck mit einem um eins kleineren Level, in dem t eines der Teildreiecke ist, keine Außenknoten gewählt wurden. Denn im anderen Fall, wo ebenfalls ein solches Dreieck entsteht, werden aus diesem keine Knoten gewählt. Dann werden aber nach dem Vorgehen im Algorithmus in diesem Fall nur aus diesem Dreieck, und nicht den anderen beiden, Knoten gewählt. Außerdem sind hier wieder die äußeren Knoten nicht gewählt. Rekursiv folgt dann die Behauptung.

Zum Beweis der Optimalität wird zuerst gezeigt, dass für eine feste Kombination $C \neq \emptyset$ von gewählten Außenknoten eines Dreiecks t und eine optimale Gesamtlösung die Teillösung des Problems "Finde einen maximalen Baum innerhalb von t, wobei von den Außenknoten genau die Knoten aus C verwendet werden müssen" ebenfalls optimal ist. Sei M eine optimale Lösung, M_t seien die gewählten Knoten innerhalb t. Die inneren Knoten eines Dreiecks t haben keine Verbindung zu Knoten außerhalb von t, welche nicht über die Außenknoten von t gehen würden. Die gewählten Außenknoten von t sind hier aber fest gewählt. Deshalb hat die Wahl der Knoten innerhalb von t keine Auswirkungen auf die Wahl der Knoten außerhalb. Damit liefert jeder gültige Baum M_t' des Teilproblems durch $M' := (M \setminus M_t) \cup M_t'$ einen gültigen Baum des Gesamtproblems. Falls nun $|M_t'| > |M_t|$ gälte, so wäre auch $|M'| > |M|$, im Widerspruch zur Optimalität von M. Damit ist aber M_t eine optimale Lösung des Teilproblems.

Nun zeigen wir mit struktureller Induktion: Für eine feste Kombination aus gewählten Außenknoten eines Dreiecks t liefert der Algorithmus die optimale Lösung. Für Dreiecke mit mindestens einem gewählten Außenknoten gilt nach obigem Absatz die optimale Teilproblem-Eigenschaft. Als Induktionsanfang dienen uns nicht weiter unterteilte Dreiecke. Dort kann gar kein innerer Knoten aufgenommen werden, damit ist das Vorgehen optimal. Für den Induktionsschritt werden die verschiedenen Fälle einzeln durchgegangen. Falls der Mittelknoten gewählt wurde, liefert eine optimale Teillösung in den Teildreiecken gemäß der optimalen Teilproblem-Eigenschaft eine optimale Lösung für t. Falls genau ein Außenknoten, aber nicht der Mittelknoten gewählt wurde, entsteht ein Teildreieck mit 0 gewählten Außenknoten. Da aber dann schon ein Knoten gewählt wurde, der nicht mehr mit den Knoten innerhalb des Dreiecks verbunden werden kann, darf aufgrund des geforderten Zusammenhangs kein Knoten darin gewählt werden. Für die anderen Dreiecke gilt wie vorhin die optimale Teilproblem-Eigenschaft. Wird für den Fall, dass kein Außenknoten gewählt ist, der Mittelknoten auch nicht gewählt, entstehen drei Teildreiecke mit 0 gewählten Außenknoten. Sobald in einem davon ein Knoten gewählt wurde, darf in keinem anderen mehr ein Knoten gewählt werden, da diese nicht verbunden werden können. Eine Lösung in einem Teildreieck liefert uns eine Lösung für t mit den gleichen gewählten Knoten. Da nur Knoten eines Teildreiecks verwendet werden können, liefert uns das Teildreieck mit der größten Lösung auch die größte Lösung für t. Falls man entscheiden kann, ob man den Mittelknoten mit aufnehmen darf, wird jeweils die maximale entstehende Knotenmenge verwendet. Damit liefern auch diese Fälle optimale Lösungen.

Damit liefert jeder Fall eine optimale Lösung bezüglich den gewählten Außenknoten. Da alle möglichen Fälle ausprobiert wurden, und am Ende für das äußere Dreieck der Fall mit den meisten gewählten Knoten ausgesucht wird, folgt daraus die Optimalität des zurückgelieferten Baums. Weil außerdem in jedem Schritt nur Operationen mit linearer Laufzeit verwendet werden, gilt der folgende Satz:

Satz 3.5. *Der vorgestellte Algorithmus berechnet einen maximalen induzierten Baum in unvollständigen 3-Bäumen in quadratischer Laufzeit.*

Bemerkung: Die lineare Laufzeit in jedem Knoten kommt von der Vereinigung der gewählten Knotenmengen. Werden diese nur implizit über Verweise in die Kindknoten gespeichert und gemerkt, ob jeweils der Mittelknoten hinzugenommen wurde, lässt sich der Algorithmus auch in Linearzeit implementieren. Hier müssen die gewählten Knoten am Ende noch ausgegeben werden.

3.4 Größe des maximalen Baums in teilweise vollständigen Bäumen

Nun versuchen wir, das Ergebnis aus Abschnitt 3.2 auch für unvollständige 3-Bäume zu verallgemeinern. Dazu betrachten wir einen vollständigen Baum, welcher dann zusätzlich noch unterteilt wird. Dabei sollen die weiteren Unterteilungen untereinander ähnlich sein, insbesondere die gleiche Anzahl an Knoten enthalten und sich in der Anzahl ihrer induzierten Bäume ähnlich verhalten. Für solche 3-Bäume gilt folgender Satz:

Satz 3.6. *Gegeben sei ein zu einem 3-Baum G mit n Knoten gehörender Baum T. G sei bis zu einem Level k vollständig aufgebaut, d.h. alle Blätter von T haben eine Tiefe von mindestens k. Es gelte, dass die zu den Knoten μ der Tiefe k gehörenden Graphen $G(\mu)$ alle gleich viele Knoten haben. Es existiere zusätzlich eine Konstante $x \in [-\frac{1}{2}, \frac{1}{2}]$, so dass für alle Knoten μ der Tiefe k gilt: Wenn zwei Knoten aus $\Delta(\mu)$ vorgegeben werden, lässt sich ein induzierter Baum finden, der diese Knoten enthält und dass der Anteil der gewählten Knoten aus dem Inneren von $G(\mu)$ mindestens $\frac{1}{2} + x$ beträgt; sowie mindestens $\frac{1}{2} - x$, falls einer vorgegeben wird. Dann existiert ein induzierter Baum in G mit mindestens $\lfloor \frac{n}{2} \rfloor$ Knoten.*

Beweis. Zur Vereinfachung nennen wir die Knoten bis zu Level k *R-Knoten*, die Knoten mit höherem Level *U-Knoten*. Wir zählen die R- und U-Knoten jeweils separat. Die R-Knoten bilden einen vollständigen 3-Baum der Tiefe k. Die R-Knoten wählen wir gemäß der Konstruktion im Beweis von Satz 3.3. Die U-Knoten wählen wir so, dass der in der Voraussetzung genannte Anteil erreicht wird. Die Anzahl der U-Knoten werde mit u, die Anzahl der R-Knoten mit r bezeichnet.

Behauptung: Für jedes Level i mit $0 \leq i \leq k$ existiert eine Konstante $y_i \in [-\frac{1}{2}, \frac{1}{2}]$, so dass alle Dreiecke in G mit Level i der Anteil der gewählten U-Knoten innerhalb des Dreiecks mindestens $\frac{1}{2} + y_i$ beträgt, falls genau ein Außenknoten gewählt wurde, und mindestens $\frac{1}{2} - y_i$, falls zwei davon gewählt wurden.

Wir beweisen die Behauptung durch Induktion rückwärts von k bis 0. Offensichtlich ist der Induktionsanfang aufgrund der Voraussetzungen an G durch $y_k := x$ erfüllt. Für den Induktionsschritt betrachten wir nun ein beliebiges i mit $0 < i \leq k$. Für Level i gelte die Behauptung. Nun wird gezeigt, dass die Behauptung dann auch für Level $i - 1$ gilt. Die Teildreiecke eines Dreiecks t von Level $i-1$ haben Level i, für diese gilt also die Behauptung. Da der Mittelknoten von t kein U-Knoten ist und da die Dreiecke auf Level i offensichtlich gleich viele innere U-Knoten haben, kann man den Anteil an gewählten U-Knoten in t durch den Mittelwert der Anteile in den Teildreiecken berechnen.

Für einen gewählten Außenknoten entstehen zwei Teildreiecke mit zwei sowie eines mit einem gewählten Außenknoten. Damit ist der Anteil gewählter U-Knoten in t mindestens

$$\frac{1}{3}(2(\frac{1}{2} - y_i) + (\frac{1}{2} + y_i)) = \frac{1}{3}(\frac{3}{2} - y_i) = \frac{1}{2} - \frac{y_i}{3}.$$

Damit kann man $y_{i-1} = -\frac{y_i}{3}$ setzen, unabhängig vom Dreieck. Für zwei gewählte Außenknoten entstehen zwei Teildreiecke mit einem sowie zwei mit einem gewählten Außenknoten. Damit ist der Anteil gewählter U-Knoten in t mindestens

$$\frac{1}{3}(2(\frac{1}{2} + y_i) + (\frac{1}{2} - y_i)) = \frac{1}{3}(\frac{3}{2} + y_i) = \frac{1}{2} + \frac{y_i}{3},$$

was wieder, unabhängig vom Dreieck, auf $y_{i-1} = -\frac{y_i}{3}$ führt. Da offensichtlich aus $y_i \in [-\frac{1}{2}, \frac{1}{2}]$ auch $y_{i-1} \in [-\frac{1}{2}, \frac{1}{2}]$ folgt, ist die Behauptung erwiesen.

Da mindestens eine der Zahlen $\frac{1}{2} + y_0$ und $\frac{1}{2} - y_0$ größer oder gleich $\frac{1}{2}$ ist, kann man mindestens die $\frac{u}{2}$ U-Knoten auswählen.

Da die R-Knoten einen vollständigen 3-Baum bilden und die Knoten wie im Beweis zu Satz 3.3 gewählt werden, wobei man im Unterschied dazu hier auch mit nur einem Außenknoten anfangen kann, kann man diese auf die gleiche Weise zählen. Falls zwei der drei Level-0-Knoten gewählt wurden, kann man mindestens $\frac{r}{2}$ R-Knoten wählen. Bei nur einem gewählten Level-0-Knoten und ungeradem k gilt dies ebenfalls. In diesen Fällen sind somit mindestens die Hälfte der Knoten aus G gewählt. Falls k gerade ist, ist r ungerade und es werden $\lfloor \frac{r}{2} \rfloor$ R-Knoten gewählt. Für gerades u folgt wegen $\frac{u}{2} \in \mathbb{Z}$ und wegen $\lfloor \frac{n}{2} \rfloor = \lfloor \frac{r}{2} \rfloor + \frac{u}{2}$ ebenfalls die Korrektheit des Satzes. Ist u ungerade, dann sind zumindest $\lceil \frac{u}{2} \rceil$ U-Knoten gewählt. Insgesamt sind in diesem Fall $\frac{n}{2}$ Knoten aus G gewählt. □

4. Heuristiken und Anwendung

In diesem Kapitel wird eine praktische Anwendung von induzierten Bäumen vorgestellt. Zuerst werden dabei zwei Heuristiken zu deren Berechnung vorgestellt und ausgewertet. Danach werden induzierte Bäume als Teil eines Graphvisualisierungs-Algorithmus verwendet, der kompakte Zeichnungen erstellt.

4.1 Heuristische Berechnung großer induzierter Bäumen

Da das Problem, einen maximalen induzierten Baum zu finden, nach Kapitel 1 insbesondere $\mathcal{NP}$-vollständig ist, muss eine Heuristik verwendet werden. In dieser Arbeit werden eine modifizierte Breitensuche (BFS) bzw. eine Tiefensuche (DFS) von einem gewählten Startknoten aus verwendet. Dabei wird gemäß dem Greedy-Ansatz jeder durch die Suche erreichte Knoten hinzugenommen und weiter betrachtet, der zusammen mit den bisher gefundenen Knoten keinen Kreis induziert. Dies ist genau dann der Fall, wenn der Knoten keine Schleife hat und höchstens eine Kante zu einem schon gefundenen Knoten führt. Damit Knoten nicht unnötig mehrfach betrachtet werden, werden Knoten, die nicht mehr gewählt werden können, entsprechend markiert. Die Reihenfolge, in der die zu einem Knoten inzidenten Kanten betrachtet werden, erfolgt zufällig, und zwar jeweils gleichverteilt. Der Algorithmus kann deshalb mehrmals angewendet werden und die größte Kantenmenge gewählt werden.

Um zu testen, wie gut die Heuristik ist, wurden die sogenannten Rom-Graphen [WBG+97] verwendet. Diese sind eine Menge von diversen Testgraphen. Gemäß dem Ansatz dieser Arbeit wurden nur solche dieser Graphen betrachtet, die sowohl planar als auch zusammenhängend sind. Dies erfüllen 3279 Graphen; diese haben 10 bis 89 Knoten. In den folgenden Diagrammen sind die Anzahl der Kanten im durch die Heuristik induzierten Baum gegen die Anzahl der Knoten der Graphen aufgetragen. Dabei wurde jeweils das beste Ergebnis aus drei Durchgängen gewählt. Für die Breitensuche liefert eine lineare Regression für die Baumgröße y in einem Graphen der Größe x die Abhängigkeit $y = 0,7825x - 0,5545$ mit $R^2 = 0,8279$, es besteht also eine sehr starke lineare Abhängigkeit. Da in einem Graphen mit n Knoten ein induzierter Baum höchstens $n - 1$ Kanten haben kann, liefert das Verfahren oft eine lineare Approximation mit einem durchschnittlichen Faktor von mindestens $\frac{1}{0,7825} \approx 1,277$ der maximalen Lösung hat. Die entsprechenden Werte für die Tiefensuche lauten $y = 0,8435x - 1,2312$ und $R^2 = 0,9349$. Im Durchschnitt über die Graphen liefert die Tiefensuche eine um etwa $0,90$ Kanten bessere Lösung, wobei in vielen Fällen auch die Breitensuche besser ist. Allerdings gibt es bei beiden Heuristiken immer wieder

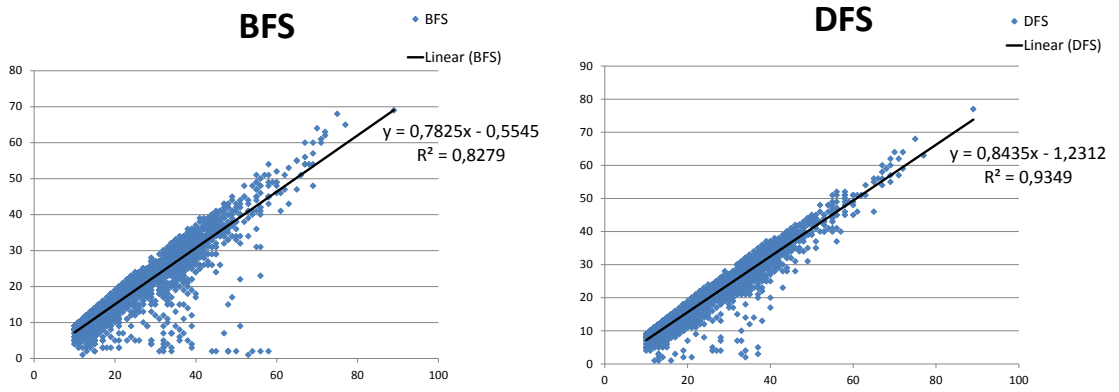


Abbildung 4.1: Dies ist das Ergebnis der Auswertung der Heuristiken: Links Breiten-, rechts Tiefensuche.

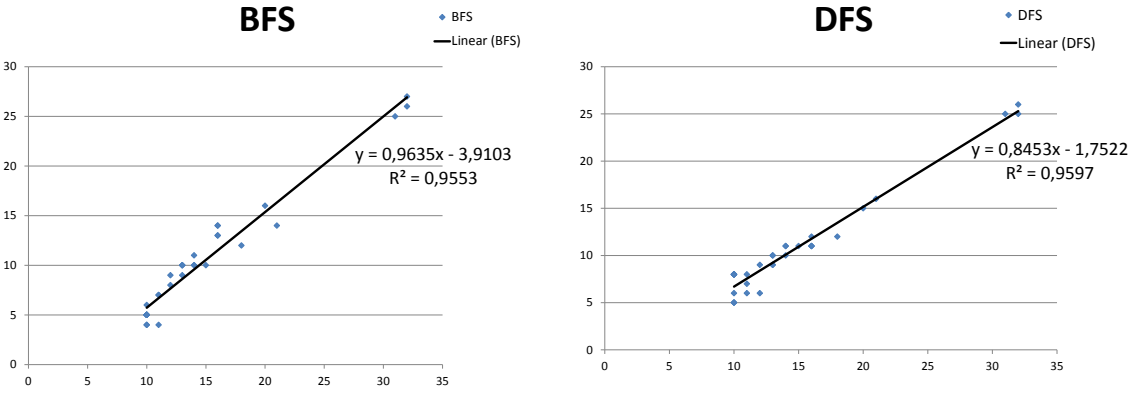


Abbildung 4.2: Dies ist das Ergebnis der Auswertung der Heuristiken für zweifach zusammenhängende Graphen: Links Breiten-, rechts Tiefensuche.

Ausreißer nach unten, wo beispielsweise in einem Graph mit 30 bis 40 Knoten nur ein, zwei oder drei Kanten gewählt wurden. Bei zweifach knotenzusammenhängenden Graphen liefert die Heuristik leicht bessere Ergebnisse: $y = 0,9635x - 3,9103$ und $R^2 = 0,9553$ für die Breitensuche sowie $y = 0,8453x - 1,7522$ und $R^2 = 0,9597$ für die Tiefensuche. Hier ist also die Breitensuche besser. Allerdings waren unter den Rom-Graphen nur 34 solcher Testgraphen, was die Aussagekraft dieses Vergleichs einschränkt.

Nun sollen induzierte Bäume in den Dualgraphen der Rom-Graphen betrachtet werden. Da die Rom-Graphen ohne feste Einbettung vorlagen, wurde hier eine zufällige gewählt. Hier wurden beide Heuristiken jeweils zehnmal angewendet und das beste Ergebnis genommen. Für beide ergibt das Verhältnis von der Anzahl gewählter Kanten im induzierten Baum zur Anzahl von Knoten im Dualgraphen im Durchschnitt nur etwa $0,108$, ein Baum ohne Kanten tritt für beide Heuristiken bei 2565 der 3279 Graphen auf. Die Heuristik liefert also deutlich schlechtere Ergebnisse. Da die Dualgraphen nicht notwendigerweise einfach sind, können Schleifen und Mehrfachkanten auftreten. Von mehrfachen Kanten kann dann keine in einem induzierten Baum vorhanden sein, da sonst die anderen Kanten davon auch darin sein müssten, wodurch ein Kreis induziert wird. Eine Schleife ist schon selbst ein Kreis, deshalb kann ein Knoten mit Schleife nicht im induzierten Baum sein. Trifft dies für den zur äußeren Facette im Ursprungsgraphen dualen Knoten zu, was gleichbedeutend damit ist, dass es eine Brücke an der äußeren Facette gibt, kann der Startknoten nicht gewählt werden und der induzierte Baum muss leer bleiben. Bei zweifach zusammenhängenden Graphen kann dieser Fall nicht auftreten, weshalb es dort bessere Ergebnisse gibt. Der

Durchschnitt der Verhältnisse von gewählten Kanten zu Knoten ist hier $0,330$ für die Breitensuche sowie $0,339$ für die Tiefensuche. Trotzdem ist das noch relativ weit von den Ergebnissen in den Primalgraphen entfernt. Dies liegt zum Teil auch daran, dass viele der Rom-Graphen eher wenige Kanten und damit auch wenig Facetten haben. Dann treten aber gehäuft Mehrfachkanten auf.

4.2 Anwendung: Erstellung kompakter Graphzeichnungen

Bei diversen Graph-Zeichnungs-Algorithmen kann durch das Hinzufügen weniger Kanten die für die Zeichnung des entstehenden Graphen nötige Fläche sich deutlich vergrößern. Oft entsteht das Problem durch eine zu kleine äußere Facette. Deshalb versucht man, diese durch das Entfernen von Kanten temporär möglichst groß zu machen. Dabei betrachten wir wie in den vorigen Kapiteln ausschließlich zusammenhängende Graphen.

Gegeben sei hierfür ein Graph-Visualisierungs-Algorithmus A für planare Graphen, der die kombinatorische Einbettung des zu zeichnenden Graphen inklusive der äußeren Facette beibehält, und ein zusammenhängender planarer Graph G mit kombinatorischer Einbettung. Der vorgestellte Algorithmus entfernt zuerst eine Menge von Kanten, zeichnet den Restgraphen mit dem Algorithmus A und fügt die entfernten Kanten planar in die erstellte Zeichnung ein, wobei die weiteren Restriktionen von A, wie beispielsweise die Forderung nach orthogonalem Kantenverlauf bei orthogonalen Layouts, für diese Kanten nicht gelten sollen. Dabei bleibt die kombinatorische Einbettung des Ursprungsgraphen inklusive der äußeren Facette erhalten.

Der Algorithmus erweitert damit bestehende Graph-Visualisierungs-Algorithmen für planare Zeichnungen zu einem Hybridlayout mit möglicherweise besseren visuellen Eigenschaften. In dieser Arbeit soll ein Orthogonal-Layout als festem Basisalgorithmus verwendet werden. Dies ist eine Variante von Tamassias Algorithmus [Tam87], der für einen Knotengrad größer als 4 generalisiert wurde.

Beim Entfernen der Kanten muss darauf geachtet werden, dass dadurch nicht zu viel strukturelle Information verloren geht. Dies soll hier folgendes bedeuten: Beim Entfernen von Kanten wird die äußere Facette jeweils mit einer anderen Facette vereinigt, ein zusammenhängender Graph soll zusammenhängend bleiben und es sollen keine neuen Brücken entstehen. Nun sucht man im Dualgraphen einen induzierten Baum. Es gilt nämlich das folgende Lemma:

Lemma 4.1. *Sei G ein zusammenhängender und einfacher planarer Graph und T^* ein induzierter Baum im Dualgraph G^*, der als Wurzel den zu der äußeren Facette von G korrespondierenden Knoten aus G^* hat. Seien T die Dualkanten der Kanten aus T^*.Dann erfüllt T die geforderten Bedingungen.*

Beweis. Da der Baum zusammenhängend ist und die Wurzel der äußeren Facette entspricht, kann man die Kanten nacheinander von der Wurzel aus gemäß der ersten Bedingung entfernen. Wenn durch das Entfernen einer Kante der Graph in zwei Zusammenhangskomponenten zerfällt, muss diese Kante eine Brücke gewesen sein. Ihre Dualkante ist dann aber eine Schleife in G^* und kann deshalb nicht in einem induzierten Baum sein. Also bleibt G nach dem Entfernen der Kanten zusammenhängend. Außerdem entsteht dadurch keine neue Brücke. Denn wenn durch das Entfernen der Kanten eine Brücke e entstehen würde, so müssten die angrenzenden Facetten beides die äußere Facette des Restgraphen sein. Damit wurden die ursprünglich an e angrenzenden Facetten mit der äußeren Facette vereinigt. Damit sind die entsprechenden Knoten im Dualgraphen aber benachbart und zusätzlich in T^*. Da aber dann offensichtlich die Dualkante von e im induzierten Baum liegt, wäre e entfernt worden. Dies ist ein Widerspruch. Damit sind alle geforderten Bedingungen erfüllt. □

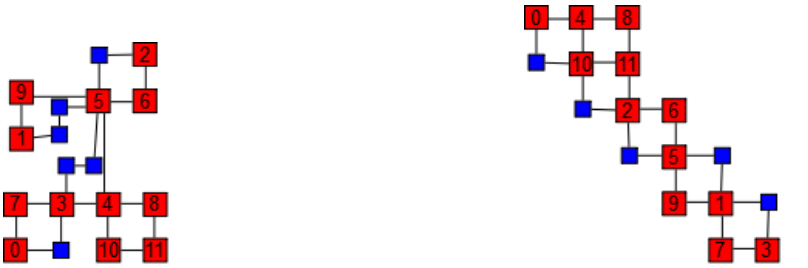


Abbildung 4.3: Diese Zeichnung entsteht als Ergebnis von A: links BFS, rechts DFS.

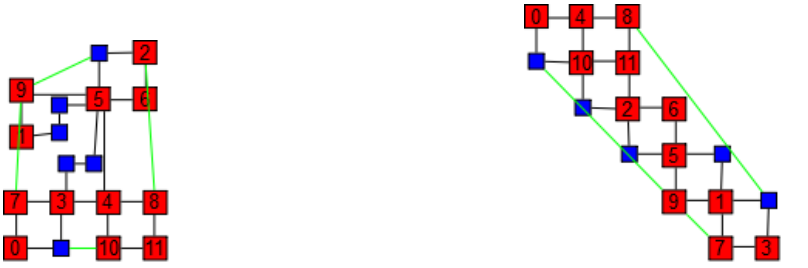
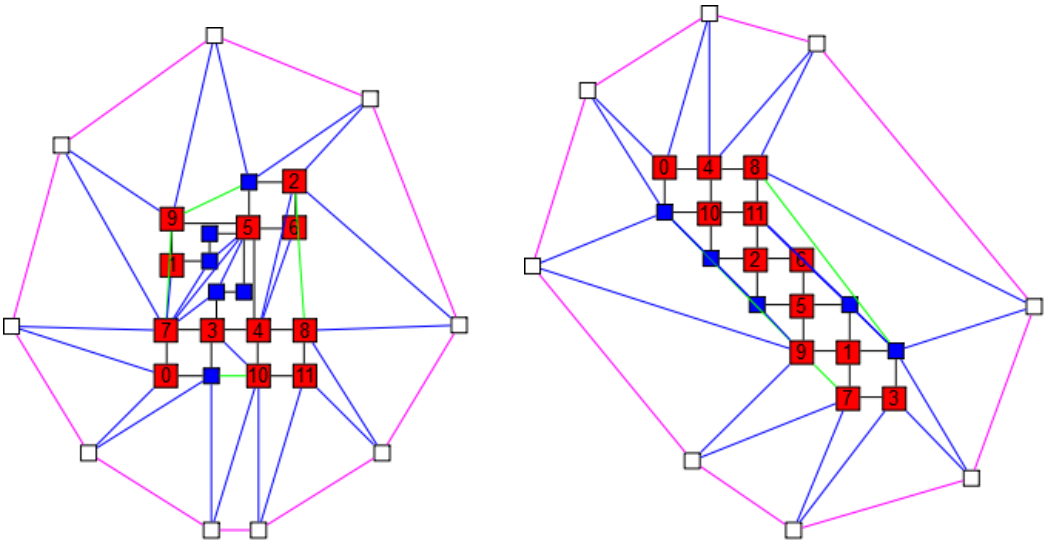


Abbildung 4.4: Die konvexe Hülle wird gebildet: links BFS, rechts DFS.

Zuerst wird mithilfe einer Heuristik, wobei in dieser Arbeit sowohl BFS als auch DFS verwendet wurden, ein großer induzierter Baum im Dualgraphen gesucht und dessen Dualkanten entfernt. Der Restgraph wird mit dem Algorithmus A gezeichnet. Hierbei wird darauf geachtet, dass die Einbettung des Graphens erhalten bleibt. Danach werden die gezeichneten Kanten an ihren Knicken im orthogonalen Layout unterteilt, so dass nun jede Kante aus einer geraden Linie besteht. Das Ergebnis ist in den Abb. 4.3a bzw. 4.3b zu finden, wobei die Knoten des Ursprungsgraphen in rot, die Unterteilungsknoten in blau gezeichnet sind.

Für das Wiedereinfügen der Kanten müssen noch einige Einzelschritte gemacht werden. Zuerst wird die konvexe Hülle des Graphen berechnet. Zur Vereinfachung der weiteren Schritte kommt ein leicht modifizierter Graham Scan zum Einsatz. Dieser gibt im Gegensatz zum herkömmlichen Algorithmus auch die Punkte zurück, die auf einer geraden Linie zwischen den Eckpunkten der konvexen Hülle liegen. Dies tritt häufiger bei orthogonalen Layouts auf. Die Kanten der konvexen Hülle werden, falls sie nicht schon vorhanden sind, in den Graphen eingefügt und grün markiert. Das Ergebnis ist in den Abb. 4.4a bzw. 4.4b zu sehen. Danach werden alle Facetten, die durch die im vorigen Schritt entstanden sind – alle Facetten, die durch eine grüne Kante begrenzt werden, mit Ausnahme der äußeren Facette – trianguliert, dass die Kanten gerade gezeichnet werden können, ohne bestehende Kanten zu kreuzen. Diese neuen Kanten werden blau markiert. Seien die Knoten der konvexen Hülle im Gegenuhrzeigersinn mit $v_1, \ldots, v_k, v_{k+1} = v_1$ bezeichnet. In die äußere Facette wird ein Kreisgraph mit k neuen Knoten $w_1, \ldots, w_k$ eingefügt, so dass jeder Knoten w_i gegenüber v_i liegt. Die Kanten zwischen diesen Knoten werden lila markiert. Danach wird jeder dieser Knoten w_i jeweils mit v_i und v_{i+1} durch blaue Kanten verbunden. Dadurch ergeben sich die Abb. 4.5a bzw. 4.5b.

Danach werden die entfernten Kanten wieder in den Graphen eingefügt, so dass die Kanten aus G wieder die Einbettung von G bilden. Der dabei entstehende Graph ist möglicherweise nicht mehr planar. Daher wird nun ein existierender Einbettungsalgorithmus verwendet. Dieser macht den Graphen planar, indem er die wieder eingefügten Kanten durch die



(a) (b)

Abbildung 4.5: Die Graphen werden trianguliert: links BFS, rechts DFS.

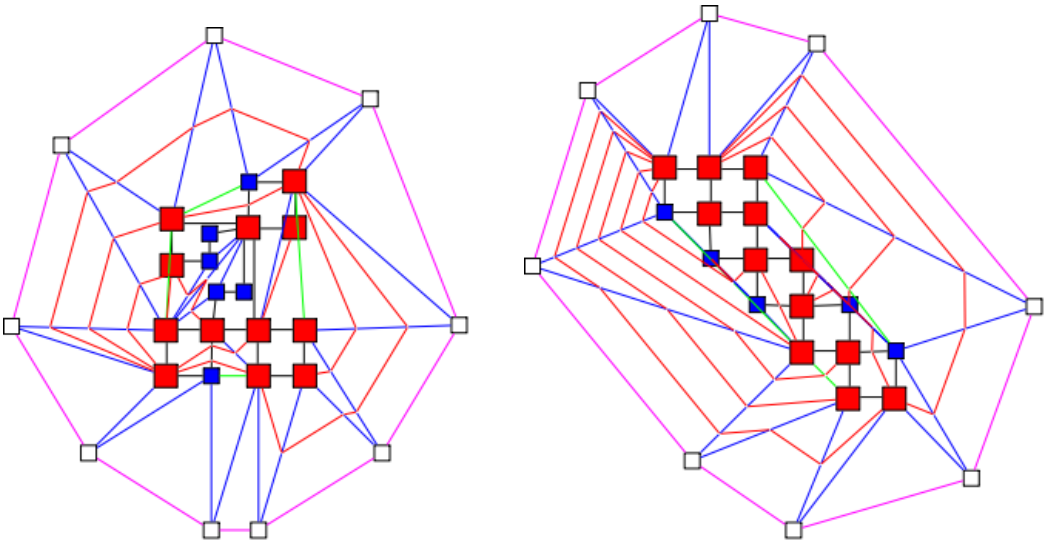
restlichen Kanten legt und Kreuzungen durch Dummy-Knoten ersetzt. Dabei werden nur Kreuzungen bei blauen und grünen Kanten erlaubt. Diese gehören nicht zum Ursprungsgraphen und damit bleibt die Planarität nach Entfernen dieser erhalten. Danach werden für jede blaue und grüne Kante e die Schnittpunkte auf e in gleichem Abstand über diese Kanten verteilt, damit die einzufügenden Kanten nicht zu nah aneinander bzw. an bestehenden Kanten verlaufen. Dies wird in den Abb. 4.6a und 4.6b illustriert, die eingefügten Kanten sind hier rot markiert. Durch die vorher erfolgte Triangulierung ist dies auch planar möglich.

Nachdem die in den vorigen Schritten hinzugefügten Kanten wieder entfernt wurden, ist der Algorithmus fertig. Das Ergebnis ist aus den Abb. 4.7a und 4.7b ersichtlich. Zum Vergleich zeigt Abb. 4.8 das Endergebnis einer orthogonalen Zeichnung des Graphen ohne das vorige Entfernen von Kanten.

4.3 Auswertung der Ergebnisse

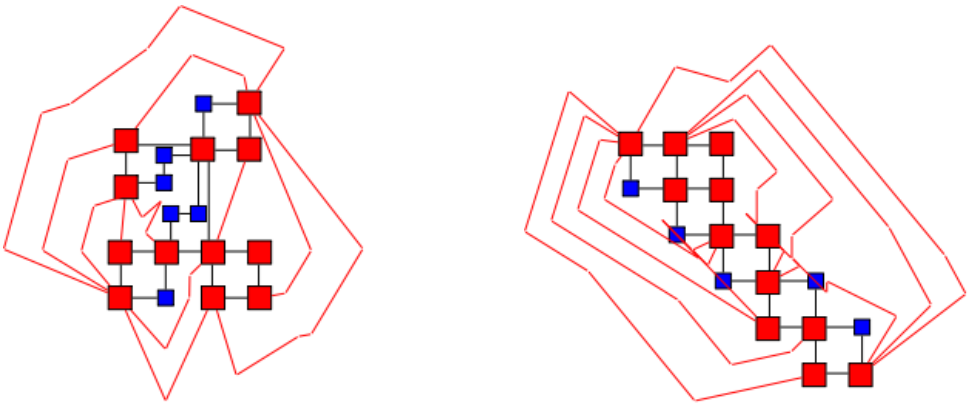
In der vorliegenden Implementierung gibt es noch einige Randfälle, in denen sie keine zufriedenstellenden Resultate liefert. Die Implementierung erlaubt es, Knoten als Rechtecke mit einer bestimmten Größe zu zeichnen. Der Orthogonal-Layout-Algorithmus lässt die Kanten aber erst an den Rändern der Rechtecke beginnen. Dadurch entstehen an den Knoten Lücken in den Kanten. Der Teil der Implementierung, der die Triangulierung berechnet, prüft bei jeder blauen Kante, ob diese eine schon bestehende Kante schneiden würde. Es wird somit allerdings nicht verhindert, dass Kanten durch die Knoten gehen. Dies ist in Abb. 4.5a zu sehen, wo die Kante $\{2, 4\}$ durch den Knoten 6 geht. Außerdem wird nicht verhindert, dass blaue Kanten bei der Triangulierung teilweise aufeinander liegen. dies ist bei der Triangulierung bei Abb. 4.5b der Fall. Die Auswirkungen sind in Abb. 4.7b ersichtlich, wo es einen Bereich gibt, in dem rote Kanten aufeinander liegen.

Des Weiteren kann der Algorithmus noch in diversen Punkten verbessert werden. Wie schon begründet, bleibt ein zweifach kantenzusammenhängender Graph auch nach dem Entfernen von Kanten noch zweifach kantenzusammenhängend. Es ist allerdings möglich, dass aus einem zweifach knotenzusammenhängenden Graph ein Graph entsteht, der nur noch einen Knotenzusammenhang von 1 hat. Um noch weniger Struktur zu verlieren, will man dies eventuell ebenfalls ausschließen. Inwieweit dies möglich ist, soll hier allerdings



(a) (b)

Abbildung 4.6: Die Kanten werden wieder eingefügt: links BFS, rechts DFS.



(a) (b)

Abbildung 4.7: Dies ist das Endergebnis des Algorithmus: links BFS, rechts DFS.

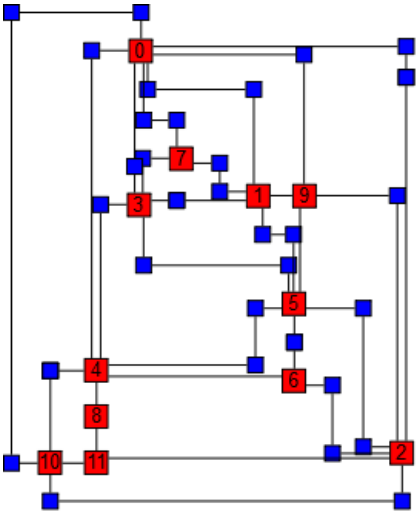


Abbildung 4.8: Dies ist das Endergebnis bei einem rein orthogonalen Layout.

nicht betrachtet werden. Außerdem wäre es möglich, eine bessere Heuristik für das Finden von großen induzierten Bäumen zu verwenden. Allerdings liefern schon die BFS- bzw. DFS-basierten Greedy-Ansätze gute Ergebnisse.

In Abb. 4.7a ist die Kante $\{1, 3\}$ mit Knicken gezeichnet, obwohl eine gerade Kante keine Probleme erzeugen würde. Ebenso gibt es in Abb. 4.7b Kanten, bei denen einzelne Abschnitte ohne Knicke gezeichnet werden könnten. Hierzu würde sich nach dem Setzen der Knickpunkte ein Korrekturschritt anbieten, in dem versucht wird, unnötige Knicke oder Umwege zu minimieren. Dies könnte vielleicht durch einen sogenannten Spring-Embedder geschehen, wobei als Ausgangskonfiguration die durch den Algorithmus generierten Knicke der eingefügten Kanten dienen würde. Dadurch würde die Zeichnung vermutlich übersichtlicher und schöner werden. Aufgrund der Komplexität dieser Aufgabe unterbleibt hier die Implementierung.

Zusätzlich könnte man noch die Triangulierungen nicht durch ein Greedy-Verfahren berechnen, sondern so optimieren, dass die entstehenden Kanten eine schönere Zeichnung erlauben. Insbesondere können bei der Triangulierung Kanten aufeinander liegen. Dies führt zu Artefakten wie in Abb. 4.7b, wo eine rote Kante einfach aufzuhören scheint. In Wirklichkeit führt die Kante bis zu diesem Punkt, macht dann eine 180°-Drehung und verläuft dann teilweise auf dem selben Weg wieder zurück. In den beiden Beispielzeichnungen scheinen auch lange blaue Kanten ein Problem darzustellen. Ebenso kann man auch die Einteilung der äußeren Facette in die Dreiecke optimieren. Insbesondere an den 90°-Ecken der konvexen Hülle knicken die Kanten eher abrupt ab. Es könnte untersucht werden, ob man durch Anpassung der Koordinaten der Dreiecke oder einfach durch mehr Dreiecke eine bessere Zeichnung erreicht.

5. Ausblick

Für induzierte Bäume in allgemeinen Graphen gibt es bereits viele Untersuchungen zu oberen und unteren Schranken. Für planare Graphen sind allerdings bisher wenige solcher Ergebnisse bekannt. Insbesondere können bestimmte Einschränkungen an die Graphen betrachtet werden und wie sie sich auf die Größe des maximalen induzierten Baums auswirken. Dabei wären vor allem Eigenschaften von Interesse, aus denen man die Existenz eines großen oder sogar linearen Baums finden kann. Ein Kandidat hierfür wären die 3-Bäume. Für vollständige 3-Bäume konnte in dieser Arbeit gezeigt werden, dass man einen induzierten Baum finden kann, der mindestens die Hälfte der Knoten enthält. Die Frage ist, ob ein derartiges Resultat auch für unvollständige 3-Bäume gilt. Es erscheint durchaus möglich, dass man in diesen immer einen induzierten Baum linearer Größe finden kann.

Für die in dieser Arbeit vorgestellte Anwendung werden die induzierten Bäume im Dualgraphen gesucht und nicht im Graphen selbst. Dieser ist im Allgemeinen nicht einfach. Hier wäre zu untersuchen, inwieweit sich dies auf die Größe eines induzierten Baums auswirken kann. Außerdem bleibt die Frage offen, ob man aus Eigenschaften eines Graphen Eigenschaften der induzierten Bäume in dessen Dualgraphen ableiten kann.

Die vorgestellten Heuristiken zur Berechnung eines großen induzierten Baums wurden hier an Beispielgraphen getestet. Eine theoretische Untersuchung ihrer Eigenschaften steht noch aus, speziell in der Frage, ob sich Approximationsgarantien für die erwartete Anzahl an gewählten Kanten ergeben. Wenn dies nicht der Fall ist, kann man stattdessen untersuchen, ob man unter zusätzlichen Einschränkungen an den Graphen bessere Garantien geben kann. Eine weitere Möglichkeit ist die Abschätzung der Wahrscheinlichkeit, mit welcher die Heuristiken einen großen Baum liefern. Da die Heuristiken eher simpel sind, können sie wohl auch noch verbessert werden. Da sie in der Praxis relativ gute Ergebnisse liefern, wird dies vor allem aus theoretischer Sicht interessant sein.

Literaturverzeichnis

[AEFG13] Patrizio Angelini, William S. Evans, Fabrizio Frati und Joachim Gudmundsson: SEFE with No Mapping via Large Induced Outerplane Graphs in Plane Graphs. CoRR, abs/1309.4713, 2013.

[AMT01] Noga Alon, Dhruv Mubayi und Robin Thomas: Large induced forests in sparse graphs. Journal of Graph Theory, 38(3):113–123, 2001.

[AV00] Jorge L. Arocha und Pilar Valencia: Long induced paths in 3-connected planar graphs. Discussiones Mathematicae Graph Theory, 20(1):105–107, 2000.

[Cam89] Kathie Cameron: Induced matchings. Discrete Applied Mathematics, 24(1–3):97–102, 1989.

[CW05] Kathie Cameron und Tracy Walker: The graphs with maximum induced matching and maximum matching the same size. Discrete Mathematics, 299(1–3):49–55, 2005.

[DGLM] Emilio Di Giacomo, Giuseppe Liotta und Tamara Mchedlidze: *Lower and Upper Bounds for Long Induced Paths in 3-connected Planar Graphs.* In: *Proc. of 39th International Workshop on Graph-Theoretic Concepts in Computer Science (WG'13)*, Lecture Notes Computer Science. Springer-Verlag. To appear.

[dlV96] Wenceslas Fernandez de la Vega: The largest induced tree in a sparse random graph. Random Struct. Algorithms, 9(1–2):93–97, 1996.

[DMZ05] William Duckworth, David Manlove und Michele Zito: On the approximability of the maximum induced matching problem. J. Discrete Algorithms, 3(1):79–91, 2005.

[ESS86] Paul Erdős, Michael Saks und Vera Sós: Maximum Induced Trees in Graphs. Journal of Combinatorial Theory, Series B, 41(1):61–79, 1986.

[FLS09] Jacob Fox, Po–Shen Loh und Benny Sudakov: Large induced trees in K_r-free graphs. Journal of Combinatorial Theory, Series B, 99:494–501, 2009.

[GMW00] Carsten Gutwenger, Petra Mutzel und René Weiskircher: *Inserting an Edge Into a Planar Graph.* In: *Algorithmica*, Seiten 246–255. ACM Press, 2000.

[KLŠ10] Lukasz Kowalik, Borut Lužar und Riste Škrekovski: An improved bound on the largest induced forests for triangle-free planar graphs. Discrete Mathematics Theoretical Computer Science, 12(1):87–100, 2010.

[MŠ07] J. Matoušek und R. Šámal: Induced trees in triangle-free graphs. ArXiv e-prints, 2007.

[Pel04] Michael J. Pelsmajer: Maximum Induced Linear Forests in Outerplanar Graphs. Graphs and Combinatorics archive, 20:121–129, 2004.

[Pfe10] Florian Pfender: Rooted induced trees in triangle-free graphs. Journal of Graph Theory, 64(3):206–209, 2010.

[Sch90] Walter Schnyder: *Embedding planar graphs on the grid.* In: *Proc. 1st ACM-SIAM Sympos. Discrete Algorithms*, Seiten 138–148, 1990.

[Sco97] A. D. Scott: Induced trees in graphs of large chromatic number. Journal of Graph Theory, 24(4):297–311, 1997.

[Tam87] Roberto Tamassia: On embedding a graph in the grid with the minimum number of bends. SIAM Journal on Computing archive, 16:421 – 444, 1987.

[Tut63] William T. Tutte: *How to Draw a Graph.* In: *Proc. London Math. Soc.*, Band s3–13, Seiten 743–767, 1963.

[WBG+97] Emo Welzl, Giuseppe Di Battista, Ashim Garg, Giuseppe Liotta, Roberto Tamassia, Emanuele Tassinari und Francesco Vargiu: An Experimental Comparison of Four Graph Drawing Algorithms. Comput. Geom., 7:303–325, 1997.

[Zit00] Michele Zito: Linear Time Maximum Induced Matching Algorithm for Trees. Nord. J. Comput., 7(1):58–, 2000.

Abbildungsverzeichnis