Maximale Kreispackungen durch gute 3-Splits in Halin-Graphen

Dissertation

zur Erlangung des Grades eines

Doktors der Wirtschaftswissenschaften

der Technischen Universität Dortmund
an der Fakultät Wirtschaftswissenschaften
im Fachgebiet Operations Research
und Wirtschaftsinformatik

von

Christin Otto

Dortmund 2021

Erster Gutachter:	*Prof. Dr. Peter Recht* Technische Universität Dortmund
Zweite Gutachterin:	*Prof. Dr. Anja Fischer* Technische Universität Dortmund

Dissertation an der Fakultät Wirtschaftswissenschaften der Technischen Universität Dortmund in Dortmund

Inhaltsverzeichnis

Abbildungsverzeichnis **v**

Tabellenverzeichnis **vi**

Algorithmenverzeichnis **vii**

1 Einleitung **1**

2 Grundlagen **5**
2.1 Graphentheoretische Grundlagen 5
2.2 Stand der Forschung - Kreispackungen 7
2.3 Stand der Forschung - Graphzerlegungen 9

3 Kreispackungsproblem in den Anwendungen **13**
3.1 Das Index Coding Problem 13
3.2 Das Traffic Grooming Problem 18
3.3 Das Minimum Local Scenario Problem 24

4 Kreispackungen und Graphzerlegungen **36**
4.1 Kreispackungen in nicht zusammenhängenden Graphen 36
4.2 Kreispackungen in 1-zusammenhängenden Graphen 39
4.3 Kreispackungen in 2-zusammenhängenden Graphen 43
4.3.1 SPQR-Zerlegung eines 2-zusammenhängenden Graphen 45
4.3.2 Kreispackungsalgorithmus für 2-zusammenhängende Graphen . . . 52
4.4 Optimalität des Verfahrens für serienparallele Graphen 56
4.5 Implementierung und Auswertung 58

5 Zerlegung von Halin-Graphen **64**
5.1 Wheel-Zerlegung eines Halin-Graphen 66
5.2 Ergebnisse zu Fans und guten 3-Splits in Halin-Graphen 80

6 Kreispackungen in Halin-Graphen **85**

6.1 Maximale Kreispackungen in speziellen Halin-Graphen 85

6.2 Ein Algorithmus zur Bestimmung von Kreispackungen in Halin-Graphen . 102

7 Fazit **125**

Literaturverzeichnis **128**

Abbildungsverzeichnis

3.1 Instanz eines Index Coding Problems mit vier Empfängern. 14
3.2 Instanz eines Index Coding Problems mit vier Empfängern (a) und der dazu gehörige Abhängigkeitsgraph $\vec{G}$ (b). 15
3.3 Lichtwellenleiter, welcher Signale verschiedener Wellenlängen überträgt. . 19
3.4 Station zum Hinzufügen, Extrahieren und Weiteleiten von Signalen. . . . 19
3.5 Netzwerk mit farbig markierten Lichtwegen. 20
3.6 Bedarfsgraph G_d mit vier zu übermittelnden Signalen. 23
3.7 Minimale Auswahl von Lichtwegen für die Signalübermittlung entsprechend des Bedarfsgraphen G_d. 23
3.8 Genomgraph G des Genoms A. 27
3.9 Adjazenzgraph $AG(\tilde{A}, \tilde{B})$ der Genome $\tilde{A}, \tilde{B}$. 30
3.10 Adjazenzgraph $AG(\tilde{A}, \tilde{B})$ und Färbungsgraph $J(\tilde{A}, \tilde{B}, col)$ der Genome $\tilde{A}, \tilde{B}$. 33
3.11 Adjazenzgraph $AG(\tilde{A}', \tilde{B})$ und Färbungsgraph $J(\tilde{A}', \tilde{B}, col)$ der Genome $\tilde{A}', \tilde{B}$. 33

4.1 Ein Graph (a) und der zugehörige Tiefensuchebaum (b). 38
4.2 Ein Graph (a), sein Block-Graph (b) und sein BC-Baum (c). 40
4.3 Ein 2-zusammenhängender Graph (a), eine 2-Separation (b) und das Konzept der virtuellen Kanten (c). 45
4.4 Splitklassen E_1, E_2, E_3 zum Splitpaar (u, v) für $G = (V, E_1 \cup E_2 \cup E_3)$. . . 46
4.5 Ein 2-zusammenhängender Graph (a), die Zusammenfassung der Splitklassen (b) und die durch den Split entstehenden Splitgraphen (c). 46
4.6 Kein Tutte-Split (a), kein Tutte-Split (b), Tutte-Split (c). 47
4.7 Ein 2-zusammenhängender Graph G (a), die 3-Zusammenhangskomponenten von G (c) und der zugehörige SPR-Baum $\mathcal{T}(G)$ (b). 49
4.8 Die Bestimmung der 3-Zusammenhangskomponenten $G_1, \ldots, G_5$ von G in vier Schritten. 51

4.9 Generierte serienparallele Graphen mit $m = 50$ Kanten und einer Wahrscheinlichkeit von p für eine serielle Komposition. 59
4.10 Laufzeit zur Bestimmung der Kreispackungszahl über alle Graphen (a) und je Wahrscheinlichkeit p (b). 62
4.11 Durchschnittliche Anzahl kantendisjunkter Kreise je tausend Kanten (a) und durchschnittliche Anzahl gefundener Kreise je Sekunde (b). 63

5.1 Ein Halin-Graph $G = T \cup C$. Der Kreis C wird durch die gestrichelten Kanten dargestellt, die übrigen Kanten induzieren den Baum T. 64
5.2 Beispiel eines Wheel-Graphen. 65
5.3 Beispiel eines Twirl-Graphen. 66
5.4 Konstruktion der einfachen 3-Zerlegung. 67
5.5 Auswirkungen einer (nicht zyklischen) 3-Separation. 68
5.6 Graph G mit 3-Separation $\{G_1, G_2\}$ und zyklischer 3-Separation $\{\tilde{G}_1, \tilde{G}_2\}$. 68
5.7 3-Separatoren induzieren K_3 [Gar89]. 69
5.8 3-Separation $\{\tilde{G}_1, \tilde{G}_2\}$ bzw. $\{\bar{G}_1, \bar{G}_2\}$ und 3-Split $\tilde{S}$ bzw. $\bar{S}$ von G. 70
5.9 Die vier verschiedenen 3-Split-Typen. 70
5.10 Verdeutlichung der Eigenschaften unterschiedlicher 3-Splits. 71
5.11 Halin-Graph G (a) und die zugehörige Zerlegung (b) mit Splitkomponenten $G'_i, i = 1, \ldots, 7$. 73
5.12 Halin-Graph G (a), die zugehörige Zerlegung (c) und der zugehöriger W-Baum (b). 76
5.13 Halin-Graph G und Fans $F_1, \ldots, F_4$. 80
5.14 Halin-Graph G und der verkleinerte Graph $G \times F_1$. 81
5.15 Halin-Graph G und der erweiterte Fan $\tilde{F}_1$. 82
5.16 Skizze eines Halin-Graphen G 83

6.1 Beispiel eines kubischen Caterpillar-Halin-Graph 86
6.2 Kubischer Necklace-Halin-Graph N_5 86
6.3 Kubischer Necklace-Halin-Graph N_2 87
6.4 Kubischer Necklace-Halin-Graph N_k mit $k \geqslant 3$ und k gerade 88
6.5 Kubischer Necklace-Halin-Graph N_k mit $k \geqslant 3$ und k ungerade 88
6.6 Beispiel eines (nicht kubischen) Necklace-Halin-Graphen 89
6.7 Beispiel eines Graphen G und seines Dualgraphen G^* 90
6.8 Beispiel eines Caterpillar-Halin-Graphen mit $\delta_G(v)$ gerade für alle $v \in V(P)$. 97

6.9 Schematische Darstellung der drei Varianten eines Caterpillar-Halin-Graph G mit $P = (v_1, \ldots, v_N)$ und der jeweiligen einfachen 3-Separation $\{G \times F_N, G_{\mu_N}\}$. 99
6.10 Caterpillar-Halin-Graph G mit zugehöriger Zerlegung $\mathcal{G} = \{G'_1, \ldots, G'_4\}$, wobei G'_1 und G'_4 ungerade Wheels sind. 102
6.11 Halin-Graph G mit Nummerierung der Kanten $e \in C$ durch $\{1, 2, \ldots, 15\}$ (a), die zugehörige Zerlegung $\mathcal{G} = \{G_{\mu_1}, \ldots, G_{\mu_7}\}$ mit Nummerierung der Kanten $\tilde{e} \in C_{\mu_i}$ (c) und der zugehörige W-Baum (b). 105
6.12 Halin-Graph G (a) mit Nummerierung der Kanten, die zugehörige Zerlegung (c) mit Nummerierung der Kanten und der zugehörige W-Baum (b). 112
6.13 Die maximale Kreispackung des Halin-Graphen G wird induziert durch $\mathcal{Z} = \{B_{14}, B_{11}, B_5, B_7, B_9, B_1, B_3\}$. 114
6.14 Halin-Graph G (a) mit Nummerierung der Kanten, die zugehörige Zerlegung (c) mit Nummerierung der Kanten und der zugehörige W-Baum (b). 122
6.15 Die maximale Kreispackung des Halin-Graphen G wird induziert durch $\mathcal{Z} = \{B_{13}, B_1, B_5, B_7, B_{11}, B_{15}, B_3, B_9\}$. 124

Tabellenverzeichnis

1.1 Aufbau der Arbeit. 4

3.1 Zuordnung der Lichtwegsequenzen zu den zu übermittelnden Signalen entsprechend G_d. 24

4.1 Durchschnittliche Laufzeiten und durchschnittliche Kreispackungszahl über alle Wahrscheinlichkeiten $p \in \{10\,\%, 20\,\%, \ldots, 90\,\%\}$. 61

6.1 Der Verlauf der Werte ℓ_j bei Anwendung von Algorithmus 6.2 auf Beispiel 6.20. 113
6.2 Die Iteration bei Anwendung von Algorithmus 6.2 auf Beispiel 6.20. . . . 114
6.3 Der Verlauf der Werte ℓ_j bei Anwendung von Algorithmus 6.3 auf Beispiel 6.27. 123
6.4 Iterationen bei Anwendung von Algorithmus 6.3 auf Beispiel 6.27. 123
6.5 Iterationen der Funktion SucheVergrösserung$(\mathcal{Z}, Z, \mathcal{L})$ bei Anwendung von Algorithmus 6.3 auf Beispiel 6.27. 124

Algorithmenverzeichnis

3.1 Berechnung einer Lösung des Complementary Index Coding Problems [Cha+11] 17

4.1 Tiefensuche (DFS=Depth-first search) 37
4.2 Zusammenhangskomponenten eines Graphen (DFS-ZSHG) 39
4.3 2-Zusammenhangskomponenten eines Graphen (DFS-2-ZSHG) [Gut10] . . 42
4.4 Greedy-Algorithmus zur Berechnung einer Kreispackung 43
4.5 Kreispackungsalgorithmus für 2-zusammenhängende Graphen 54

5.1 Zerlegung eines minimal 3-zusammenhängenden Graphen [Gar89] 77
5.2 Bestimmung eines guten 3-Splits [Gar89] 78

6.1 Kreispackungsalgorithmus entsprechend Satz 6.13 101
6.2 Kreispackungsalgorithmus für Halin-Graphen 108
6.3 Kreispackungsalgorithmus für Halin-Graphen entsprechend Satz 6.25 . . . 121

Kapitel 1

Einleitung

Die Suche nach einer Packung, Überdeckung oder Zerlegung eines Graphen durch kleinere Strukturen oder kleinere Graphen gehört zu den klassischen Fragestellungen der kombinatorischen Optimierung. Da es sich bei diesen Problemen im Allgemeinen um NP-schwere Probleme handelt, gibt es unzählige Forschungsarbeiten, die verschiedene Ausprägungen des Problems, verschiedene Eigenschaften von Graphen und spezielle kleinere Strukturen betrachten. So wird speziell bei den Packungsproblemen unter anderem zwischen knotendisjunkten und kantendisjunkten Packungsproblemen unterschieden, gerichtete und ungerichtete Graphen betrachtet und die kleineren Strukturen, aus welchen die Packung des Graphen bestehen soll, auf besonders schöne oder symmetrische Strukturen wie Dreiecke, Kreise, Wege etc. beschränkt. Frühe Arbeiten zu solchen Problemstellungen lassen sich laut Yuster [Yus07], der 2007 eine Übersicht zu diesen Themenbereichen veröffentlich hat, bereits 1847 von Kirkman [Kir47] finden. Dieser zeigte, dass K_n genau dann in Kreise der Länge 3 zerlegt werden kann, wenn $n \equiv 1, 3 \pmod 6$. Im Jahr 1892 beschreibt Lucas eine Konstruktion von Walecki, die zu einer Zerlegung des K_n in Hamiltonkreise führt [Luc92]. Erste Forschungsarbeiten zur Matchingtheorie lassen sich laut einer Übersicht im gleichnahmigen Buch von Plummer und Lovász [PL86] in den Arbeiten von Petersen [Pet91] 1891 zu regulären Graphen und Kőnig [Kőn15] 1915 zu bipartiten Graphen finden. Deutlich später, im Jahr 1947, liefert Tutte [Tut47] eine notwendige und hinreichende Bedingung für die Existenz eines perfekten Matchings in beliebigen Graphen. Eine der bekanntesten bisher nicht bewiesenen Fragestellungen ist die 1959 aufgestellte Vermutung von Erdős und Gallai [EG59], dass die Kantenmenge eines jeden Graphen mit n Knoten in $O(n)$ Kreise und Kanten zerlegt werden kann. In den folgenden Jahren veröffentliche Erdős zusammen mit Pósa und Dirac weitere Forschungsarbeiten zu knoten- und kantendisjunkten Kreisen in Graphen [DE63; EP62;

EP65], wobei diese in den Arbeiten als *unabhängige* bzw. *kantenunabhängige Kreise* bezeichnet wurden. Eine deutliche Steigerung der Anzahl an Forschungsergebnissen findet dann im letzten Drittel des 20. Jahrhunderts statt, insbesondere durch die Entwicklungen in der Informatik. Das bereits genannte Buch *Matching Theory* [PL86] wird beispielsweise 1986 veröffentlicht, außerdem entwickeln sich verschiedene Themenbereiche innerhalb der kombinatorischen Optimierung wie die *kombinatorische Designtheorie* (siehe zum Beispiel [CM87]), in welcher die Forschung sich ebenfalls mit Packungsproblemen in Graphen befasst.

In dieser Arbeit soll ein bestimmter Typ von Packungsproblemen betrachtet werden: die Packung eines Graphen mit einer maximalen Anzahl kantendisjunkter Kreise. Dieser Problemtyp wird auch maximales Kreispackungsproblem genannt, wobei hierunter im Grunde zwei Fragestellungen zusammengefasst werden. Zum einen die Frage nach der maximalen Anzahl an kantendisjunkten Kreisen in einem Graphen, zum anderen die Frage nach der tatsächlichen Kreispackung, die eine maximale Anzahl an kantendisjunkten Kreisen beschreibt. Beide Probleme zählen zu den NP-schweren Problemen [CPR03; DT92; Kri+07] und sind bisher noch nicht allzu stark erforscht. Begriffliche Verwechselungsgefahr besteht mit einem anderen Problemtyp, der in der deutschsprachigen Literatur ebenfalls Kreispackungsproblem genannt wird aber zu den geometrischen Packungsproblemen gehört. In der englischsprachigen Literatur wird daher zwischen Cycle Packing und Circle Packing unterschieden. Das Circle Packing beschreibt die geometrische Problemstellung der Anordnung von Kreisen innerhalb eines vorgegebenen Gebietes, sodass keine zwei Kreise einander überlappen und die Kreise sich gegenseitig berühren. Das Cycle Packing bezieht sich auf die für diese Arbeit relevante graphentheoretische Problemstellung, der Packung von Kreisen in Graphen.

Die grundlegende Idee dieser Arbeit besteht darin, einen Graphen G auf bestimmte Weise in „kleinere Graphen“ G_i zu zerlegen, maximale Kreispackungen dieser G_i zu bestimmen und daraufhin einen Bezug zur maximalen Kreispackung von G zu finden. Die Art der Zerlegung von G hängt dabei davon ab, wie „stark“ G „zusammenhängend“ ist. Dass ein solches Vorgehen sinnvoll sein kann, zeigen einfache Schlussfolgerungen für nicht zusammenhängende und einfach zusammenhängende Graphen sowie die Ergebnisse aus [Ott14] für 2-zusammenhängende Graphen. Weiterhin haben Heinrich et al. in [Hei+20] eine Zerlegung, welche eng verwandt mit der in [Ott14] verwendeten Zerlegung ist, für sogenannte $2,5$-zusammenhängende Graphen modifiziert und auf Grundlage dieser Zerlegung Aussagen zur Kreispackungszahl und zu Kreiszerlegungen von eulerschen Graphen entwickelt. Darüber hinaus gibt es zahlreiche NP-schwere Probleme, welche auf

Graphen mit beschränkter Baumweite in Polynomialzeit gelöst werden können, indem die zugehörige Baumzerlegung genutzt wird (siehe beispielsweise [Arn85], [ALS91] oder [Bod94]). In einer Baumzerlegung wird die Knotenmenge eines Graphen in Teilbäume (also „kleinere Graphen“) zerlegt, deren Adjazenzen wiederum durch einen Baum abgebildet werden.

Einen Einblick in die Umsetzung der Idee, die dieser Arbeit zugrunde liegt, soll die folgende Übersicht über die Struktur der Arbeit geben. Im Anschluss an das aktuelle einleitende Kapitel 1 werden in Kapitel 2 zunächst die graphentheoretischen und forschungsbezogenen Grundlagen dieser Arbeit erläutert. Dazu gehört neben dem Überblick über bisherige Arbeiten zu maximalen Kreispackungen auch ein Überblick über die Literatur, die sich mit Zerlegungen von 3-zusammenhängenden Graphen beschäftigt. In Kapitel 3 werden drei praktische Anwendungsbeispiele vorgestellt, deren formale Behandlung zu Kreispackungsproblemen führt. Zum einen spielen Kreispackungen eine Rolle bei der Bündelung von Daten in drahtlosen Netzwerken. Beim Lösungsverfahren für das sogenannte *Index Coding Problem* wird die Bestimmung einer maximalen kantendisjunkten Kreispackung verwendet, um Einsparungen bei der Übermittlung von Datenpaketen zu finden. Das daraufhin vorgestellte *Traffic Grooming Problem* weist Parallelen zum Index Coding Problem auf, da es die gebündelte Übertragung von Daten in optischen Netzwerken behandelt. Bei einem Spezialfall dieses Problems wird die Bestimmung einer maximalen Kreispackung zur Lösung verwendet. Ein häufig genannter Anwendungsfall für Kreispackungsprobleme sind sogenannte *Genome Rearrangement Probleme*, da Caprara die Beziehung zur maximalen Anzahl an kantendisjunkten alternierenden Kreisen in speziellen Graphen und einem speziellen Genome Rearrangement Problem genutzt hat, um zu zeigen, dass dieses NP-schwer ist [Cap97]. Hier wird ein anderes spezielles Genome Rearrangement Problem, das sogenannte *Minimum Local Scenario Problem* vorgestellt für dessen Lösung maximale kantendisjunkte Kreispackungen bestimmt werden.

Die weitere Struktur der Arbeit folgt der Eigenschaft des Zusammenhangs eines Graphen G (vergleiche Tabelle 1.1). In Kapitel 4 werden Grundlagen hinsichtlich sogenannter Separatoren und Zerlegungen von Graphen vertieft und insbesondere Schlussfolgerungen zu Kreispackungen für nicht zusammenhängende, 1-zusammenhängende und 2-zusammenhängende Graphen näher betrachtet. Nach der Erläuterung wie sich Aussagen zu Kreispackungen mithilfe einer Zerlegung von nicht zusammenhängenden Graphen in Komponenten sowie von einfach zusammenhängenden Graphen in höher zusammenhängende Teilgraphen herleiten lassen, werden ab Abschnitt 4.3 „nächst höher zusammenhängende“ Graphen betrachtet, genauer 2-zusammenhängende Graphen. Für diese

Kapitel	**Grad des Zusammenhangs von G**	**Zerlegung von G in Teilgraphen G_i**	**Kreispackung**
Kap. 4	G nicht zusammenhängend	G_i 1-Komponenten	$\nu(G) = \sum \nu(G_i)$ $\mathcal{Z}^*(G) = \bigcup \mathcal{Z}^*(G_i)$
Kap. 4	G 1-zusammenhängend	G_i Blöcke	$\nu(G) = \sum \nu(G_i)$ $\mathcal{Z}^*(G) = \bigcup \mathcal{Z}^*(G_i)$
Kap. 4	G 2-zusammenhängend	G_i serielle/parallele/ rigide Komponenten	$\nu(G) \overset{\text{Alg}}{\rightsquigarrow} \nu(G_i)$ $\mathcal{Z}^*(G) \overset{\text{Alg}}{\rightsquigarrow} \mathcal{Z}^*(G_i)$
Kap. 5&6	G minimal 3-zusammenhängend (speziell Halin)	G_i Wheel/Twirl/zykl. 4-zusammenhängende Komponenten	$\nu(G) \overset{\text{Alg}}{\rightsquigarrow} \nu(G_i)$ $\mathcal{Z}^*(G) \overset{\text{Alg}}{\rightsquigarrow} \mathcal{Z}^*(G_i)$

Tabelle 1.1: Aufbau der Arbeit.

wurde in [Ott14] ein auf SPQR-Bäumen basierender Algorithmus zur Bestimmung einer unteren Schranke für die Größe einer maximalen Kreispackung entwickelt, sowie für spezielle 2-zusammenhängende Graphen (genauer serienparallele Graphen) gezeigt, dass dieser Algorithmus eine maximale Kreispackung liefert. In Vorbereitung auf das folgende Kapitel 5 werden die in [Ott14] erarbeiteten Ergebnisse noch einmal erläutert. Ziel der Arbeit ist es, eine Erkenntnis zu Kreispackungen in Graphen der nächsten „Zusammenhangsstufe“, also 3-zusammenhängenden Graphen, zu gewinnen. Darüber hinaus wird in Abschnitt 4.5 die Implementierung des Kreispackungsalgorithmus für serienparallele Graphen mithilfe des *Open Graph Drawing Frameworks* [Chi+13] vorgestellt und ausgewertet. Die Zerlegung von 3-zusammenhängenden Graphen erweist sich hinsichtlich der Möglichkeit Rückschlüsse von Kreispackungen in den Komponenten der Zerlegung auf Kreispackungen im Ausgangsgraphen zu ziehen als deutlich unübersichtlicher als bei den zuvor betrachteten 2-zusammenhängenden Graphen. Daher werden in dieser Arbeit verschiedene Zwischenschritte betrachtet: Halin-Graphen sind minimal 3-zusammenhängende Graphen und besitzen ähnlich wie serienparallele Graphen einige besonders schöne Eigenschaften, auf deren Grundlage in Kapitel 5 Ergebnisse zur Bestimmung der Kreispackungszahl und zur Bestimmung von maximalen Kreispackungen in Halin-Graphen entwickelt werden.

Kapitel 2

Grundlagen

In diesem Kapitel wird die graphentheoretische Grundlage für die folgenden Kapitel geschaffen, indem die für die Arbeit wesentlichen Definitionen und Notationen erläutert werden. Darüber hinaus wird der Stand der Forschung in Bezug auf Kreispackungen und Graphzerlegungen skizziert.

2.1 Graphentheoretische Grundlagen

Sei $G = (V(G), E(G))$ ein endlicher, einfacher und ungerichteter Graph mit Knotenmenge $V(G)$ und Kantenmenge $E(G)$. Sofern keine Missverständnisse entstehen können, wird die Notation in manchen Fällen auf $V = V(G)$ bzw. $E = E(G)$ verkürzt. Die Kardinalität der Knotenmenge $|V|$ bestimmt die *Ordnung* des Graphen, die Kardinalität der Kantenmenge $|E|$ bestimmt die *Größe* des Graphen. Ein Graph $G' = (V', E')$ heißt *Teilgraph* von G ($G' \subseteq G$), falls $V' \subseteq V$ und $E' \subseteq E$. Zwei Teilgraphen $G' = (V', E')$ und $G'' = (V'', E'')$ werden *kantendisjunkt* genannt, falls $E' \cap E'' = \varnothing$. Ein Teilgraph $G' = (V', E') \subset G$ heißt durch $E' \subset E$ ($G' = G|_{E'}$) *induziert*, falls V' aus allen Knoten besteht, die inzident mit Kanten in E' sind. Analog wird $G' = (V', E') \subset G$ durch $V' \subset V$ ($G' = G|_{V'}$) *induziert*, falls E' aus allen Kanten $e \in E$ besteht, deren Endknoten beide in V' enthalten sind. Die Notation ist $G \backslash V' := G|_{V \backslash V'}$ bzw. $G \backslash E' := G|_{E \backslash E'}$. Außerdem wird die Notation $V(E(G)) := V(G|_E)$ verwendet. Für $u \in V$ entspricht der *Knotengrad* $\delta_G(u)$ der Anzahl seiner inzidenten Kanten in G.

Ein *Kantenzug* ist eine Folge von Kanten $(e_1, \ldots, e_r)$ mit $e_i = (v_{i-1}, v_i) \in E(G)$. Handelt

es sich um eine Folge paarweise verschiedener Kanten $(e_1, \ldots, e_r)$ mit $e_i = (v_{i-1}, v_i) \in E(G)$ und paarweise verschiedenen Knoten $v_0, \ldots, v_r \in V(G)$ heißt W *Weg* und hat die Länge $r \geqslant 0$. In manchen Fällen wird W auch v_0-v_r-Weg genannt, um den Anfangs- und Endknoten des Weges hervorzuheben. Ein *Kreis* C der Länge $r \geqslant 2$ ist eine Folge $(e_1, \ldots, e_{r-1}, e_r)$, sodass $(e_1, \ldots, e_{r-1})$ ein Weg der Länge $r-1$ ist und $e_r = (v_{r-1}, v_0)$. Da W als Teilgraph von G interpretiert werden kann, wird in manchen Fällen W als induziert durch seine Kantenmenge $E(W)$ beschrieben. Der *Umfang* eines Graphen G bezeichnet die Länge eines kürzestens Kreises in G. Ein Graph G heißt *zusammenhängend*, falls für jedes Knotenpaar $v, w \in V$ ein v-w-Weg in G exisitert. Ein kreisfreier Graph heißt *Wald*, ein zusammenhängender und kreisfreier Graph (eine Zusammenhangskomponente eines Waldes) heißt *Baum*.

Ein Kantenzug in einem zusammenhängenden Graphen, der jede Kante genau einmal enthält, heißt *Eulerzug*. Ein zusammenhängender Graph, in welchem ein geschlossener Eulerzug existiert, wird *eulersch* genannt.

Eine *Kreispackung* der Kardinalität z in G ist eine Menge $\mathcal{Z} = \{C_1, \ldots, C_z\}$ von paarweise kantendisjunkten Kreisen. Eine Kreispackung $\mathcal{Z}^*$ mit maximaler Kardinalität wird *maximale Kreispackung* genannt. Die Kardinalität $|\mathcal{Z}^*|$ wird mit $\nu(G)$ bezeichnet, wobei $\nu(G)$ *Kreispackungszahl* genannt wird.

Eine Menge $K \subset V$ mit $|K| = k$ und $k \geqslant 0$ wird k-*Separator* von G genannt, falls $G|_{V \setminus K}$ nicht zusammenhängend ist. Ein Graph G heißt k-*zusammenhängend*, falls kein $(k-1)$-Separator in G existiert. Ein Teilgraph G' heißt maximal k-zusammenhängend, wenn $G' = G''$ für jeden k-zusammenhängenden Teilgraph $G'' \subseteq G$ mit $G' \subseteq G''$ gilt. Die maximal 1-zusammenhängenden Teilgraphen von G werden 1-*Komponenten* genannt. Die maximal 2-zusammenhängenden Teilgraphen von G werden *Blöcke* genannt. Gibt es Teilgraphen G_1, G_2 von G, sodass $G = G_1 \cup G_2$ mit $|V(G_1) \cap V(G_2)| = k, E(G_1) \cap E(G_2) = \varnothing$ und $|E(G_1)| \geqslant k, |E(G_2)| \geqslant k$, so heißt G k-*separabel* und $\{G_1, G_2\}$ wird k-*Separation* von G genannt. Ein Graph G heißt *minimal* k-*zusammenhängend*, falls dieser Graph k-zusammenhängend ist aber $G|_{E \setminus e}$ für jede Kante $e \in E(G)$ $(k-1)$-separabel ist.

Ein Graph G heißt *planar*, falls er kreuzungsfrei in der Ebene gezeichnet werden kann. Eine solche Darstellung wird *planare Einbettung* genannt. Die in der Ebene maximalen zusammenhängenden Flächen der planaren Einbettung eines planaren Graphen G ohne Kanten und Knoten heißen *Facetten*. Die eine Facette B begrenzenden Kanten

werden auch als *Rand* dieser Facette bezeichnet und durch $\mathcal{R}(B)$ beschrieben. Weiterhin induziert der Rand einer Facette einen Kreis, dieser wird als *Facettenkreis* bezeichnet.

2.2 Stand der Forschung - Kreispackungen

Wie bereits in Kapitel 1 für allgemeine (Kreis-)Packungsprobleme erläutert, wird unter anderem zwischen knotendisjunkten und kantendisjunkten Packungsproblemen, gerichteten und ungerichteten Graphen und speziellen Strukturen, mit welchen eine Kreispackung gefunden werden soll, unterschieden. Der folgende Abschnitt soll einen Überblick einiger Arbeiten zu verschiedenen Ausprägungen des Kreispackungsproblems liefern.

Ein großer Teil der Forschungsarbeiten zu Kreispackungsproblemen bezieht sich auf knotendisjunkte Kreispackungen. Einen wesentlichen Anteil daran hat unter anderem das bekannte Theorem von Erdös und Pósa, welches besagt, dass eine Funktion $f(k) = \mathcal{O}(k \log k)$ existiert, sodass für jedes $k \in \mathbb{N}$ jeder ungerichtete Graph entweder mindestens k knotendisjunkte Kreise enthält oder eine Knotenmenge X mit $|X| \geqslant f(k)$, sodass $G \backslash X$ ein Wald ist [EP65]. Die Vielzahl von Arbeiten, die auf dieses Theorem folgen und verschiedenste Variationen wie beispielsweise Kreispackungen mit kürzesten Kreisen oder Kreisen ungerader Länge in gerichteten und ungerichteten Graphen betrachten, kann hier nicht vollständig zusammengefasst werden. Arbeiten aus den letzten zwei Jahren wären exemplarisch [BHJ20; Kaw+20; Lok+19]. Darüber hinaus sind knotendisjunkte Kreispackungen auch in Bezug auf die praktische Anwendung im Rahmen von Nierenaustauschprogrammen vermehrt untersucht worden. Der Austausch von Spenderorganen zwischen zwei, drei oder mehr verschiedenen Spender-/Empfängerpaaren lässt sich auf ein Kreispackungsproblem mit knotendisjunkten Kreisen zurückführen. So befassen sich einige Arbeiten mit verschiedenen Modellformulierungen und Algorithmen, zum Beispiel [Bir+21; BMR09; Car+20; Con+13; Dic+16; KAV14; Lin+19; LM20; Ped14; XW18].

Kantendisjunkte Kreispackungen wurden in der jüngeren Literatur aus verschiedenen Perspektiven betrachtet. In [Hor11] hat Horsley die maximale Anzahl kantendisjunkter Kreise der Länge m in einem vollständigen Graphen gezeigt. Weiterhin haben Bryant et al. [BHP14] gezeigt, wann genau ein vollständiger Graph in kantendisjunkte Kreise mit festen Längen $m_1, \ldots, m_t$ zerlegt werden kann. Im Falle eines Multigraphen haben Bryant et al. [Bry+15] notwendige und hinreichende Bedingungen für die Existenz einer solchen Zerlegung formuliert. Darauf aufbauend hat Cameron [Cam17] eine vollstän-

dige Charakterisierung für die Existenz einer kantendisjunkten Kreispackung in einem vollständigen Multigraphen mit Kreisen mit festen Längen $m_1, \ldots, m_t$ formuliert. Weitere Ergebnisse zu maximalen kantendisjunkten Kreispackungen in speziellen Graphen gibt es beispielsweise für Petersen Graphen [Spr15] oder polyhedrale Graphen [NS18; RS14; Ste21]. In [Har+10b] wurden maximale kantendisjunkte Kreispackungen eines Graphen auf maximale kantendisjunkte Kreispackungen in Teilgraphen zurückgeführt: Falls der Graph einen k-Separator enthält, kann die Kreispackungszahl des Graphen mithilfe der Kreispackungszahlen von höchstens $2^{\binom{k}{2}+1}$ Teilgraphen mit geringerer Ordnung bestimmt werden. Die *zyklomatische Zahl* $r(G)$ beschreibt die minimale Anzahl von Kanten, die aus einem Graphen G entfernt werden müssen, um einen Wald zu erzeugen und kann über die Formel $r(G) = |E(G)| - |V(G)| + \kappa(G)$ berechnet werden, wobei $\kappa(G)$ die Anzahl der Zusammenhangskomponenten in G beschreibt. In [Har+10a] wird gezeigt, dass für jedes $k = r(G) - \nu(G) \geqslant 0$ eine endliche Menge an Graphen existiert, sodass G mithilfe einfacher Graphoperationen aus einem dieser Graphen konstruiert werden kann. Für Graphen mit geradem Knotengrad wird in [Rec+16] ein Kriterium formuliert, welches die Maximalität einer Kreispackung indiziert. Dieses Kriterium führt zur Betrachtung des Problems als Kürzeste-Wege-Problem, für dessen Lösung ein Ansatz mit dynamischer Programmierung vorgestellt wird. Verwendet wird ein A^*-Kürzeste-Wege-Algorithmus auf einem speziellen Netzwerk. In den letzten Jahren wurden außerdem eine Reihe von Ergebnissen zu kantendisjunkten Kreispackungen in sogenannten *Turniergraphen* veröffentlicht. Ein *Turniergraph* ist ein gerichteter Graph $\vec{G}$, der genau eine Kante zwischen jedem Knotenpaar enthält. Zunächst wurden von Yuster und Akaria Schranken für die maximale Zahl von kantendisjunkten Kreisen der Länge drei in regulären und nahezu regulären Turniergraphen entwickelt [Aka14; AY15; Yus13]. Bessy et al. zeigen, dass die Bestimmung maximaler kantendisjunkter Kreispackungen auch für Packungen mit Kreisen der Länge drei NP-schwer ist [BBT18]. Die zeitlich folgenden Arbeiten beschäftigen sich dann mit der parametrisierten Komplexität dieser Problemstellung [Bes+19; Kri+18; SS20]. Jacob und Krithika untersuchten kantendisjunkte Kreise in bipartiten Turniergraphen und zeigten, dass die Frage nach k kantendisjunkten Kreisen in solchen Graphen in $2^{\mathcal{O}(k \log k)} n^{\mathcal{O}(1)}$ Zeit beantwortet werden kann [JK20]. Zuletzt haben Heinrich et al. in [Hei+20] eine Zerlegung von [HT73a] modifiziert, welche eng verwandt mit der in [Ott14] verwendeten SPQR-Zerlegung ist. Analog zu den 3-Zusammenhangskomponenten in [HT73a] wurden in [Hei+20] eine Zerlegung von 2-zusammenhängenden Graphen in sogenannte 2,5-zusammenhängende Komponenten definiert und auf Grundlage dieser Zerlegung Aussagen zur Kreispackungszahl und zu Kreiszerlegungen von eulerschen Graphen entwickelt. Heinrich et al. nennen einen 2-zusammenhängenden Graphen $G = (V, E)$ *2,5-zusammenhängend*, wenn $G \backslash \{v\}$ und $G \backslash \{e\}$ für jeden beliebigen Kno-

ten $v \in V$ und jede beliebige Kante $e \in E$ zusammenhängend sind. Die Modifikation der verwendeten Zerlegung von [HT73a] bezieht sich auf eine Färbung sogenannter virtueller Kanten, mit der die 2-Separatoren von Separatoren, welche aus einem Knoten und einer Kante bestehen, unterschieden werden können. Ähnlich zu den Ergebnissen für serienparallele Graphen in [Ott14] führen Heinrich et al. die Kreispackungszahl eines eulerschen Graphen G auf die Kreispackungszahlen der Komponenten der Zerlegung zurück.

In Bezug auf die Approximierbarkeit der Kreispackungszahl wurden ebenfalls einige wesentliche Ergebnisse veröffentlicht. Zunächst wurde in [CR02] gezeigt, dass die maximale Anzahl kantendisjunkter Kreise der Länge 3 in Polynomialzeit gefunden werden kann, wenn der Knotengrad des Graphen höchstens 4 beträgt. Für Graphen G mit einem maximalen Knotengrad größer als 4 ist das Problem jedoch APX-schwer und für planare Graphen mit einem maximalen Knotengrad größer als 4 ist es NP-schwer. Diese Ergebnisse greifen Rautenbach und Regen in [RR09] auf und zeigen, dass für Graphen mit einem Umfang von $g = 4$ (bzw. $g = 5$) die Kreispackungszahl für kantendisjunkte Packungen mit Kreisen der Länge 4 (bzw. 5) in Polynomialzeit bestimmt werden kann, aber dass die Bestimmung maximaler kantendisjunkter Kreise der Länge g für Graphen mit einem Umfang von $g \geqslant 6$ APX-schwer ist, sogar dann, wenn der maximale Knotengrad des Graphen mit 3 beschränkt wird. In [CPR03] wurde eine Greedy-Heuristik für das allgemeine kantendisjunkte Kreispackungsproblem auf einem Graphen $G = (V, E)$ mit $|V| = n$ vorgestellt und gezeigt, dass es sich bei dieser um einen $\mathcal{O}(\log n)$-Approximationsalgorithmus handelt. In [Kri+07] wurde gezeigt, dass eine Variante dieses Algorithmus eine Approximationsrate von $\mathcal{O}(\sqrt{\log n})$ hat. Dass diese Approximationsrate das nahezu bestmögliche ist, wurde in [FS11] gezeigt. Approximationsalgorithmen für einige verwandte Probleme wurden in [NY04] (gerichtete Graphen) und [CTW09] (Kreise der Länge 3) vorgestellt.

2.3 Stand der Forschung - Graphzerlegungen

Wie in der Einleitung beschrieben besteht die grundlegende Idee dieser Arbeit darin, einen Graphen G auf bestimmte Weise in „kleinere Graphen“ G_i zu zerlegen, maximale Kreispackungen $\mathcal{Z}^*(G_i)$ dieser G_i zu bestimmen und daraufhin einen Bezug zur maximalen Kreispackung $\mathcal{Z}^*(G)$ von G zu finden. Entsprechend der Tabelle 1.1 gibt es bereits Ergebnisse für nicht zusammenhängende, 1-zusammenhängende und 2-zusammenhängende

Graphen, sodass im folgenden Schritt 3-zusammenhängende Graphen betrachtet werden sollten. Aus diesem Grund folgt nun ein Überblick zum Stand der Forschung zu Zerlegungen für 3-zusammenhängende Graphen.

Seit den 80er Jahren wurden einige Ergebnisse zu derartigen Zerlegungen veröffentlicht. Cornuéjols et al. zeigten, dass Halin-Graphen, welche (minimal) 3-zusammenhängend sind, eindeutig zerlegt werden können, sodass jede Komponente der Zerlegung ein spezieller „kleinerer“ Halin-Graph, ein sogenanntes *Wheel* ist [CNP83]. Auf Basis dieser Zerlegung entwickelten sie einen Algorithmus, der das Traveling Salesman Problem auf Halin-Graphen in Polynomialzeit löst. In [CNP85] weiten sie dieses Ergebnis auf weitere 3-zusammenhängende Graphen mit speziellen Eigenschaften aus. Eine Zerlegung von minimal 3-zusammenhängenden Graphen, deren Komponenten drei verschiedenen, speziellen Graphen entsprechen (Wheels, sogenannten *Twirls* und *zyklisch* 4*-zusammenhängende Graphen*), wurde von Gardner 1989 in [Gar89] publiziert (siehe auch [CGW93]). Dabei heißt ein minimal 3-zusammenhängender Graph *zyklisch* 4*-zusammenhängend*, wenn keine 3-Separation $\{G_1, G_2\}$ existiert, sodass sowohl G_1 als auch G_2 einen Kreis enthalten. Diese Zerlegung wurde in Anlehnung an [CE80] entwickelt, wobei Cunningham und Edmonds in [CE80] erwähnt, dass seine Arbeit und die von Tutte [Tut66] zur gleichen kanonischen Zerlegung eines 2-zusammenhängenden Graphen führen, auch wenn die Herangehensweise eine andere ist. Da die zu Tutte äquivalenten Ergebnisse von Hopcroft und Tarjan [HT73a] auch Grundlage der von Battista und Tamassia in Di Battista und Tamassia vorgestellten SPQR-Zerlegung für 2-zusammenhängende Graphen waren, bestehen auch einige Ähnlichkeiten zwischen der SPQR-Zerlegung für 2-zusammenhängende Graphen und der von Gardner vorgestellten Zerlegung für minimal 3-zusammenhängende Graphen. Außerdem sind beide Zerlegungen eindeutig. 1991 haben Kanevsky, Tamassia, Di Battista und Chen unabhängig von Gardner selbst eine Zerlegung von 3-zusammenhängenden Graphen in Anlehnung an die SPQR-Zerlegung entwickelt und diese *FWRT-Zerlegung* genannt [Kan+91]. Die FWRT-Zerlegung beschreibt eine Zerlegung eines 3-zusammenhängenden Graphen, deren Komponenten ebenfalls drei verschiedenen, speziellen Graphen entsprechen (Wheels, sogenannten *Flowers* und *rigiden Komponenten* entsprechend der SPQR-Zerlegung). Einige der dort beschrieben Ergebnisse können laut [Kan+91] auf k-zusammenhängende Graphen erweitert werden. Bereits 1989 haben Fouquet und Thuillier Ergebnisse zu einer eindeutigen Zerlegung 3-zusammenhängender kubischer Graphen zur Publikation eingereicht, diese wurden allerdings erst 1993 in [FT93] veröffentlicht. Auch diese Zerlegung wurde in Anlehnung an [CE80] entwickelt und stellt einen Spezialfall der Zerlegung von Gardner dar. D'souza veröffentlicht 2016 in [Dso16] eine Zerlegung von 3-zusammenhängenden Graphen in

quasi-4-zusammenhängende Komponenten. Ein 3-zusammenhängender Graph G wird in [Dso16] *quasi-4-zusammenhängend* genannt, wenn für jede 3-Separation $\{G_1, G_2\}$ von G entweder G_1 oder G_2 genau vier Knoten besitzt. Hierbei können jedoch zwei verschiedene Graphen zur gleichen Zerlegung führen, dementsprechend ist eine Rekonstruktion des Ursprungsgraphen nicht eindeutig möglich. Insbesondere in Bezug auf die enthaltenen Kreise ist die Zerlegung nicht eindeutig. Durch die Operationen zur Erzeugung der quasi 4-zusammenhängenden Komponenten wird die Anzahl der Kreise vervielfacht und kann nicht eindeutig auf die Anzahl der Kreise im Ursprungsgraphen zurückgeführt werden. Im gleichen Jahr wird die Arbeit von Grohe zur Zerlegung eines Graphen in quasi-4-zusammenhängende Komponenten veröffentlicht, wobei diese Komponenten zwar identisch zu D'souza benannt aber anders definiert wurden [Gro16]. Ein 3-zusammenhängender Graph G wird in [Gro16] *quasi-4-zusammenhängend* genannt, wenn für jede 3-Separation $\{G_1, G_2\}$ von G entweder G_1 oder G_2 aus einem einzigen Knoten bestehen. Grohe zeigt außerdem, dass die quasi-4-zusammenhängenden Komponenten den von Robertson und Seymour vorgestellten „hoch zusammenhängenden Regionen“ (der Ordnung 4) eines Graphen, sogenannten *Tangles*, entsprechen [RS91]. Diese Art der Graphzerlegung wird insbesondere für Theorien zur Struktur von Graphen mit ausgeschlossenen Minoren benutzt. Unabhängig von den vorangegangenen Ergebnissen haben Karpov und Pastor ab 2003 (zum Teil einzeln, zum Teil gemeinsam) eine ganze Reihe von Ergebnissen zur Struktur und Zerlegung von k-zusammenhängenden Graphen veröffentlicht [Kar05; Kar16; Kar20; KP03; KP12; Pas18]. In [KP12] beschreiben Karpov und Pastor eine Zerlegung von 3-zusammenhängenden Graphen durch 3-Separatoren in kleinere Strukturen, die sie *Komplexe* nennen. Die Autoren selbst bezeichnen diese Zerlegung jedoch als sehr kompliziert und versuchen in den darauffolgenden Veröffentlichungen diese zu vereinfachen. Karpov stellt in [Kar16] eine vereinfachte Zerlegung von minimal k-zusammenhängenden Graphen durch paarweise unabhängige k-Separatoren vor, wobei ein k-Separator in [Kar16] aus Kanten und Knoten besteht und mindestens eine Kante enthält. Damit erinnert die Wahl der Separatoren an die von [Kan+91] verwendeten Separatoren. Pastor wiederum beschreibt in [Pas18] abweichend von den bisherigen Ergebnissen eine Zerlegung von 3-zusammenhängenden Graphen in *zyklisch 4-Kanten-zusammenhängende* Graphen, wobei die Definition solcher zyklisch 4-Kanten-zusammenhängenden Graphen nicht identisch zu der Definition von zyklisch 4-zusammenhängenden Graphen von Gardner ist. Pastor nennt einen 3-zusammenhängenden Graphen *zyklisch 4-Kanten-zusammenhängend*, falls bei Entfernung von drei beliebigen Kanten des Graphen mindestens eine zusammenhängende Komponente einen Kreis enthält. Weiterhin heißt ein 3-zusammenhängender Graph genau dann *4-Kanten-zusammenhängend*, wenn der Graph trotz Entfernen von drei beliebigen Kanten zusammenhängend bleibt oder ein Graph

mit zwei zusammenhängenden Komponenten entsteht, von denen eine aus einem einzigen Knoten besteht. Karpov hingegen stellt in [Kar20] weitere Vereinfachungen der Ergebnisse aus [KP12] vor. Eine für 3-zusammenhängende planare Graphen entwickelte Zerlegung ist die Zerlegung von Eppstein und Reed [ER19], die eine spezielle Teilmenge von 3-Separatoren nutzt. Sie zeigen außerdem, dass diese speziellen Teilmengen mithilfe der SPQR-Zerlegung auch für die Menge von 2-Separatoren eines 2-zusammenhängenden Graphen bestimmt werden können.

Aufgrund der größeren Parallelen zur SPQR-Zerlegung wird in dieser Arbeit bei Betrachtung von 3-zusammenhängenden Graphen in Kapitel 5 auf die Zerlegung von Gardner zurückgegriffen. Zunächst folgt jedoch eine Einführung zu Kreispackungen in nicht zusammenhängenden, 1-zusammenhängenden und 2-zusammenhängenden Graphen, siehe Kapitel 4. In Kapitel 4 wird dazu jedoch zunächst die Situation für k-zusammenhängende Graphen G mit $k \leqslant 2$ rekapituliert.

Kapitel 3

Kreispackungsproblem in den Anwendungen

In diesem Kapitel werden praktische Problemstellungen betrachtet, deren Modellierung zur Bestimmung maximaler kantendisjunkter Kreispackungen führt. Zunächst werden zwei Probleme in Netzwerken vorgestellt. In Abschnitt 3.1 wird die Übertragung von Daten in drahtlosen Netzwerken betrachtet und in Abschnitt 3.2 wird die Übertragung von Daten in optischen Netzwerken betrachtet. Abschnitt 3.3 erläutert die häufig in Verbindung mit Kreispackungsproblemen angegebene Anwendung des Genome Rearrangement und Sorting by Reversals.

3.1 Das Index Coding Problem

Eine Problemstellung, in welcher die Bestimmung einer maximalen kantendisjunkten Kreispackung Teil eines Lösungsverfahrens ist, ist das sogenannte *Index Coding Problem*. Diese Problemstellung wurde von Birk und Kol [BK98] formuliert und ist ein spezielles sogenanntes *Network Coding Problem*. Eine Instanz dieser Problemstellung besteht aus einem drahtlosen Netzwerk mit einem Sender und mehreren Empfängern, in welchem Daten ausschließlich vom Sender zum Empfänger weitergegeben werden können. Ein Datenaustausch zwischen den Empfängern ist nicht möglich. Bei einer gegebenen Menge von Datenblöcken, sogenannten *Paketen*, benötigt jeder Empfänger nun ein einzelnes dieser Pakete, welches der Sender über das drahtlose Netzwerk an den jeweiligen Empfänger senden muss. Über dieses benötigte Paket hinaus ist dem Sender aber bekannt,

dass die Empfänger andere Pakete bereits erhalten und zwischengespeichert haben und diese damit vom benötigten Paket unterscheiden können. Mit dieser Zusatzinformation besteht die Möglichkeit in einer Sendung mehrere Pakete zusammenzufassen. Dieser Schritt nennt sich *Kodierung*. Im Schritt der *Dekodierung* kann der Empfänger nun das benötigte Paket unter Nutzung seiner ihm bereits bekannten Pakete aus der Sendung herausfiltern. Das Problem besteht nun in der Bestimmung einer möglichst geringen Anzahl von Sendungen, sodass jeder Empfänger sein benötigtes Paket erhält und aus der Sendung herausfiltern kann. Wie in [Bar+06; ECS07] gezeigt, gehört das Index Coding Problem zur Klasse der NP-schweren Probleme und ist darüber hinaus auch schwer zu approximieren [LS08].

Abbildung 3.1 zeigt ein Beispiel eines Index Coding Problems mit einem Sender und vier Empfängern. Jeder Empfänger benötigt ein Paket und hat aus vorherigen Sendungen die unter „Kennt“ angegeben Pakete bereits zwischengespeichert und kann sie daher von den benötigten Paketen unterscheiden.

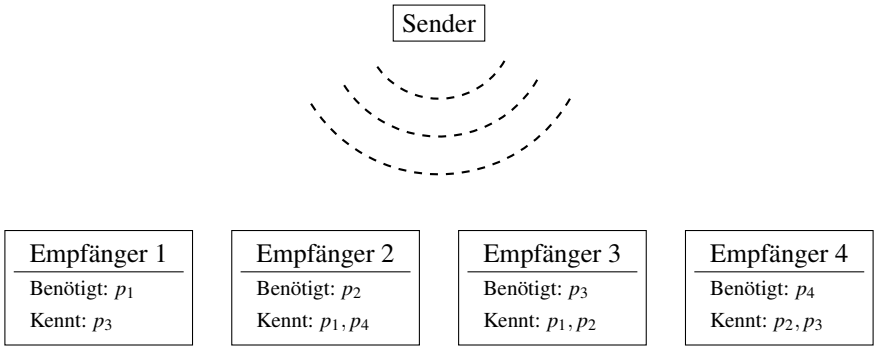


Abbildung 3.1: Instanz eines Index Coding Problems mit vier Empfängern.

In diesem Beispiel ist leicht zu erkennen, dass zwei Übertragungsrunden genügen, um alle Empfänger mit ihren benötigten Paketen zu versorgen. In der ersten Sendung werden die Pakete p_1 und p_3 zusammen versendet $(p_1 + p_3)$, in der zweiten Sendung werden die Pakete p_2 und p_4 zusammen versendet. Mit der Sendung $p_1 + p_3$ erhalten Empfänger 1 und Empfänger 3 ihre benötigten Pakete. Empfänger 1 hat das Paket p_3 bereits zwischengespeichert und kann Paket p_1 damit herausfiltern. Im Vergleich zur Sendungsvariante, bei der alle Pakete einzeln gesendet werden ohne die Zusatzinformation im Zwischenspeicher der Empfänger zu nutzen, können also zwei Sendungen eingespart werden.

Das hier beschriebene Index Coding Problem enthält eine Einschränkung, die jedoch vor dem Hintergrund eines drahtlosen Netzwerkes einfach umzusetzen ist. Jeder Empfänger

darf in dieser Problemstellung nur ein einziges Paket nachfragen. Angenommen ein Empfänger benötigt mehrere Pakete, so kann dieser Empfänger in der Problemformulierung einfach durch mehrere Empfänger ersetzt werden, welche nur ein Paket nachfragen und die gleichen Pakete als Zusatzinformation besitzen wie der ursprüngliche Empfänger [Bar+06].

Das Index Coding Problem lässt sich entsprechend [Cha+11] nun wie folgt formalisieren. Gegeben sei ein Sender, eine Menge $R = \{r_1, \ldots, r_n\}$ von Empfängern in einem drahtlosen Netzwerk und eine Menge von Paketen $P = \{p_1, \ldots, p_n\}$ mit $p_i \in GF(q)^n$, welche vom Sender an die Empfänger übermittelt werden müssen. Unter einem *Galois field* $GF(q)$ versteht man einen endlichen Körper der Restklassen ganzer Zahlen modulo q mit q Elementen. Jedem Empfänger $r_i \in R$ ist ein Tupel $(w_i, H(r_i))$ zugeordnet, wobei $w_i \in P$ das von r_i benötigte Paket beschreibt und die Menge $H(r_i) \subseteq P$ die bereits vom Empfänger r_i zwischengespeicherten Pakete enthält. In jeder der insgesamt μ Übertragungsrunden übermittelt der Sender eine Sendung x_j mit $1 \leqslant j \leqslant \mu$, wobei die Sendung x_j das Ergebnis einer Kodierung einer Teilmenge der Pakete aus P ist mit $x_j = \sum_{i=1}^{n} g_j^i \cdot p_i$. Dabei entspricht g_j einer linearen Kodierungsfunktion für Übertragungsrunde j bzw. g_j^i dem Kodierungskoeffizienten für Übertragungsrunde j und Paket p_i mit $g_j = \{g_j^i\} \in GF(q)^n$. Die Dekodierung vom benötigten Paket w_i von Empfänger r_i erfolgt dann mit einer ebenfalls linearen Dekodierungsfunktion q_i mit $w_i = q_i(x_1, \ldots, x_\mu, H(r_i))$ für $1 \leqslant i \leqslant n$. Ziel des Index Coding Problems ist es eine Menge von Kodierungsfunktionen $g_1, \ldots, g_\mu$ und Dekodierungsfunktionen $q_1, \ldots, q_n$ zu finden, sodass die Anzahl an Sendungen μ minimal ist.

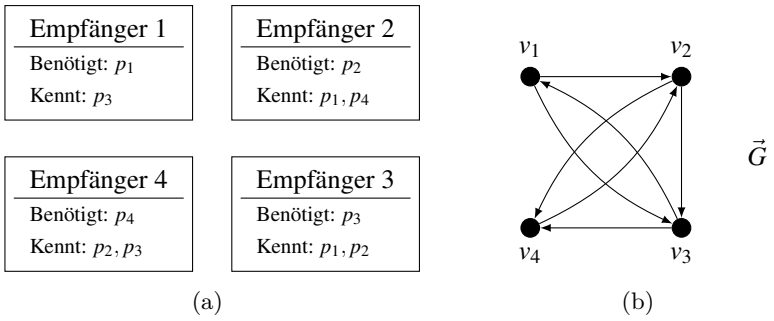


Abbildung 3.2: Instanz eines Index Coding Problems mit vier Empfängern (a) und der dazu gehörige Abhängigkeitsgraph $\vec{G}$ (b).

Üblicherweise wird zur Lösung dieser Problemstellung ein Graph modelliert, welcher die Beziehung zwischen dem benötigten Paket w_i seines Empfängers r_i und den vorhandenen Paketen $H(r_j)$ eines Empfängers r_j beschreibt. Dieser Graph $\vec{G} = (V, \vec{E})$ heißt

Abhängigkeitsgraph, wobei $V = \{v_1, \ldots, v_n\}$ ist und der Knoten v_i den Empfänger r_i repräsentiert. Die Kantenmenge $\vec{E}$ beschreibt dann entsprechend die Beziehung zwischen den Empfängern mit $\vec{E} = \{(i,j) \mid w_i \in H(r_j)\}$, das heißt $\vec{G}$ enthält eine Kante (i,j) genau dann, wenn Empfänger r_j das von Empfänger r_i benötigte Paket w_i bereits zwischengespeichert hat [Cha+11]. Abbildung 3.2(b) zeigt den zur Beispielinstanz aus 3.2(a) gehörenden Abhängigkeitsgraphen $\vec{G}$.

Eine komplementäre Formulierung zum Index Coding Problem ergibt sich aus der Beobachtung, dass ohne Kodierung alle n Pakete in einzelnen Sendungen übermittelt werden müssen. Anstatt also die Anzahl benötigter Sendungen μ zu minimieren, kann äquivalent die durch Kodierung eingesparte Anzahl an Sendungen $n - \mu$ maximiert werden [Cha+11]. Diese Problemstellung wird als *Complementary Index Coding Problem* bezeichnet. Der später vorgestellte Approximationsalgorithmus basiert auf einem Bezug zwischen der Einsparung von Sendungen entsprechend dem Entscheidungsziel beim Complementary Index Coding Problem und Kreisen im zugehörigen Abhängigkeitsgraphen. Betrachte hierzu erneut Abbildung 3.2. Wie zuvor beschrieben besteht die Lösung ohne Index Coding in der einzelnen Übermittlung aller Pakete. Mithilfe einer Kodierung können stattdessen die Pakete p_1 und p_3 sowie die Pakete p_2 und p_4 in jeweils einer Sendung übermittelt werden. Diese Lösung kann mithilfe des Abhängigkeitsgraphen $\vec{G}$ bestimmt werden. Betrachte hierzu alle Kreise in $\vec{G}$. $\vec{G}$ enthält die vier Kreise $\vec{C}_1 = (v_1, v_3, v_1)$, $\vec{C}_2 = (v_2, v_4, v_2)$, $\vec{C}_3 = (v_2, v_3, v_4, v_2)$ und $\vec{C}_4 = (v_2, v_3, v_1, v_2)$. Dass durch das Zusammenfassen der Pakete der Empfänger in den Kreisen $\vec{C}_1$ und $\vec{C}_2$ jeweils eine Sendung eingespart werden kann, ist bereits bekannt. Der Kreis $\vec{C}_3$ kann jedoch ebenso eine Einsparung implizieren. Benötigt werden die Pakete p_2, p_3 und p_4. Der jeweilige Nachfolger eines Empfängers (bezogen auf den Kreis) besitzt das benötigte Paket bereits. Anstatt diese drei Pakete nun einzeln zu übermitteln, genügt es zwei Sendungen mit $x_1 = p_2 + p_3$ und $x_2 = p_3 + p_4$ zu senden. Mit der Sendung x_1 kann Empfänger r_3 das benötigte Paket dekodieren, mit der Sendung x_2 kann Empfänger r_4 das benötigte Paket dekodieren. Empfänger r_2 wiederum erhält sein Paket aus der Summe beider Sendungen. Da $p_i \in GF(q)^n$ ist, gilt $p_2 + p_3 + p_3 + p_4 = p_2 + p_4$. Die Anzahl der Sendungen kann so auf zwei Sendungen für drei Empfänger reduziert werden. Insgesamt würden bei dieser Variante drei Sendungen für vier Empfänger benötigt werden, da Paket p_1 für Empfänger r_1 einzeln übermittelt werden muss. Diese Lösung ist also schlechter als die Lösung, welche durch die Kreise C_1 und C_2 impliziert wird. Die Idee zur Lösung des Complementary Index Coding Problems besteht daher zunächst in der Bestimmung einer maximalen knotendisjunkten Kreispackung im Abhängigkeitsgraphen $\vec{G}$.

Lemma 3.1 [Cha+11]
Sei α eine Menge von knotendisjunkten Kreisen im Abhängigkeitsgraph $\vec{G} = (V, \vec{E})$. Dann kann eine zulässige Lösung des Complementary Index Coding Problems mit Zielfunktionswert $|\alpha|$ konstruiert werden.

Das Problem eine maximale knotendisjunkte Kreispackung zu bestimmen gehört allerdings selbst zu den NP-schweren Problemen [SV05]. Äquivalent dazu ist jedoch die Bestimmung einer kantendisjunkten Kreispackung im sogenannten *Vertex Split Graph*. Zu einem gegebenen Graphen $\vec{G} = (V, \vec{E})$ wird der entsprechende *Vertex Split Graph* $G' = (V', E')$ wie folgt konstruiert: Für jeden Knoten $v \in V$ werden zwei Knoten v_{in}, v_{out} in V' und eine (ungerichtete) Kante (v_{in}, v_{out}) in E' eingefügt. Weiterhin wird für jede Kante $(u, v) \in \vec{E}$ eine (ungerichtete) Kante (u_{out}, v_{in}) in E' ergänzt. Für das kantendisjunkte Kreispackungsproblem existiert ein einfacher Approximationsalgorithmus von Krivelevich et al. [Kri+07], welcher später in Kapitel 4 näher erläutert wird. Daher ist dieser Approximationsalgorithmus ein wesentlicher Teil des Verfahrens für das Complementary Index Coding Problem [Cha+11].

Nun kann der Approximationsalgorithmus von Chaudhry et al. [Cha+11] für das Complementary Index Coding Problem entsprechend in Algorithmus 3.1 formuliert werden. Wie oben entwickelt, wird zu einem gegebenen Complementary Index Coding Problem

Algorithmus 3.1 Berechnung einer Lösung des Complementary Index Coding Problems [Cha+11]

Eingabe: Instanz des Complementary Index Coding Problems.
Ausgabe: Kodierung $\mathcal{X} = \{x_1, \ldots, x_{n-s}\}$.
1: $X \leftarrow \{p_1, \ldots, p_n\}$
2: Bestimme Abhängigkeitsgraph $\vec{G}$ zur Instanz des Complementary Index Coding Problems.
3: Bestimme Vertex Split Graph G' zu $\vec{G}$.
4: Bestimme kantendisjunkte Kreispackung $\{C_1, \ldots, C_s\}$ in G'.
5: Überführe die kantendisjunkte Kreispackung $\{C_1, \ldots, C_s\}$ in G' in eine gerichtete, knotendisjunkte Kreispackung $\{\vec{C}_1, \ldots, \vec{C}_s\}$ in $\vec{G}$.
6: **for all** $\vec{C}_i \in \{\vec{C}_1, \ldots, \vec{C}_s\}$ **do**
7: Sei $C_i = (v_{i_1}, \ldots, v_{i_k}, v_{i_1})$ für ein $k \in \mathcal{N}$.
8: $\mathcal{X} \leftarrow (\mathcal{X} \backslash \{p_{i_1}, \ldots, p_{i_k}\}) \cup \{p_{i_1} + p_{i_2}, \ldots, p_{i_{k-1}} + p_{i_k}\}$
9: **end for**
10: **return** X

der zugehörige Abhängigkeitsgraph $\vec{G}$ und darauffolgend der Vertex Split Graph G' bestimmt. Mithilfe des Approximationsalgorithmus von Krivelevich et al. wird dann eine kantendisjunkte Kreispackung in G' bestimmt, die im Anschluss wiederum in eine gerichtete, knotendisjunkte Kreispackung in $\vec{G}$ überführt wird. Für jeden der gefundenen

Kreise werden die den Empfängern entsprechenden Pakete zu Sendungen mit einer Ersparnis von einer Sendung zusammengefasst.

Satz 3.2 [Cha+11]
Algorithmus 3.1 bestimmt eine Lösung des Complementary Index Coding Problems mit Approximationsgüte von $\Omega(\sqrt{n} \cdot \log n \cdot \log \log n)$.

Das Problem der maximalen Kreispackungen wird hier also zur Bestimmung von Einsparungen bei der Übermittlung der Datenpakete verwendet. Die Approximationsgüte von Algorithmus 3.1 ist mit dem Faktor $\sqrt{n}$ dabei im Wesentlichen abhängig von der Approximationsgüte des Algorithmus zur Bestimmung einer maximalen Kreispackung. Eine Verbesserung dieser Approximationsgüte wäre daher wesentlich für die Verbesserung der Approximationsgüte von Algorithmus 3.1.

Darüber hinaus gibt es Ansätze spezielle Graphen für das Index Coding Problem zu betrachten, in der Hoffnung das Problem auf diesen Graphen besser lösen zu können. So wurde beispielsweise in [BL11] festgestellt, dass das Index Coding Problem mit einem außenplanaren Abhängigkeitsgraphen effizient gelöst werden kann. Außenplanare Graphen wiederum sind spezielle serienparallele Graphen, für welche das maximale Kreispackungsproblem, wie in [OR19] gezeigt, ebenfalls effizient gelöst werden kann.

3.2 Das Traffic Grooming Problem

Ein weiteres Problem, bei dessen Lösung maximale Kreispackungen eine Rolle spielen, ist das *Traffic Grooming Problem* in optischen Netzwerken. Auch hier geht es im Kern um die gebündelte Übertragung von Daten, ähnlich wie beim Index Coding Problem. Allerdings handelt es sich beim betrachteten Netzwerk um ein optisches und kein drahtloses Netzwerk. Um das Traffic Grooming in optischen Netzwerken zu verstehen, muss zunächst die Art der Datenübertragung genauer betrachtet werden. In optischen Netzwerken werden sogenannte *Lichtwellenleiter* zur Übertragung genutzt. Ein solcher Lichtwellenleiter ist beispielsweise das in diesem Kontext bekannte Glasfaserkabel, mit dem Daten mittels Lichtwellen übertragen werden können. Um nun nicht jeden Datensatz einzeln mithilfe eines Lichtwellenleiters übertragen zu müssen, wurden weitere Verfahren zur Steigerung der Kapazität eines Lichtwellenleiters, sogenannte *Multiplexverfah-*

ren, entwickelt. Hierbei wird ein gemeinsames Übertragungsmedium (in diesem Fall ein einzelner Lichtwellenleiter) verwendet, um mehrere verschiedene Signale zu übermitteln [Jah09]. Beim *Wellenlängenmultiplexsystem (WDM)* werden unterschiedliche Wellenlängen innerhalb eines Lichtwellenleiters zur Übertragung verschiedener Daten genutzt. So entstehen innerhalb eines Lichtwellenleiters wiederum verschiedene (virtuelle) *Lichtwellenkanäle*. Werden sehr viele dicht beieinander liegende Wellenlängen innerhalb eines Lichtwellenleiters verwendet, spricht man vom sogenannten *Dichten Wellenlängenmultiplexsystem (DWDM)*. Dieses stellt eine Weiterentwicklung des WDM-Verfahrens dar. Die Bauteile, die eine solche Einspeisung verschiedener Wellenlängen und später auch die Extraktion dieser ermöglichen, nennen sich *Multiplexer* bzw. *Demultiplexer*. Abbildung 3.3 zeigt verschiedene farbig markierte Signale (verschiedene Wellenlängen), die gleichzeitig durch den Lichtwellenleiter übertragen werden.

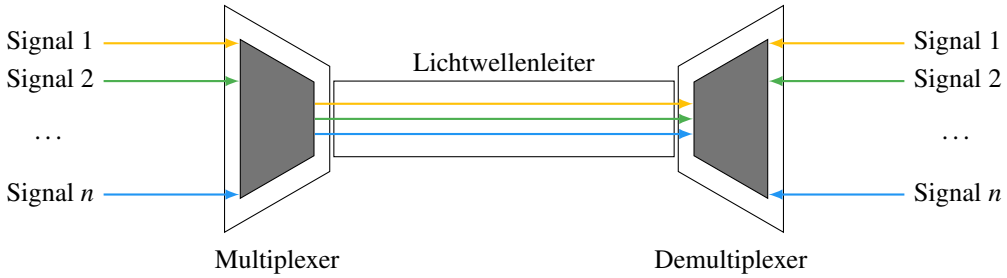


Abbildung 3.3: Lichtwellenleiter, welcher Signale verschiedener Wellenlängen überträgt.

Ein optisches Netzwerk besteht nun aus verschiedenen Stationen, welche mit Lichtwellenleitern untereinander verbunden sind. Über jeden Lichtwellenleiter können dabei unter Nutzung des WDM-Systems N verschiedene Wellenlängen übertragen werden. Die Multiplexer an den Stationen erlauben es Signale hinzuzufügen, zu extrahieren oder weiterzuleiten (vgl. Abbildung 3.4).

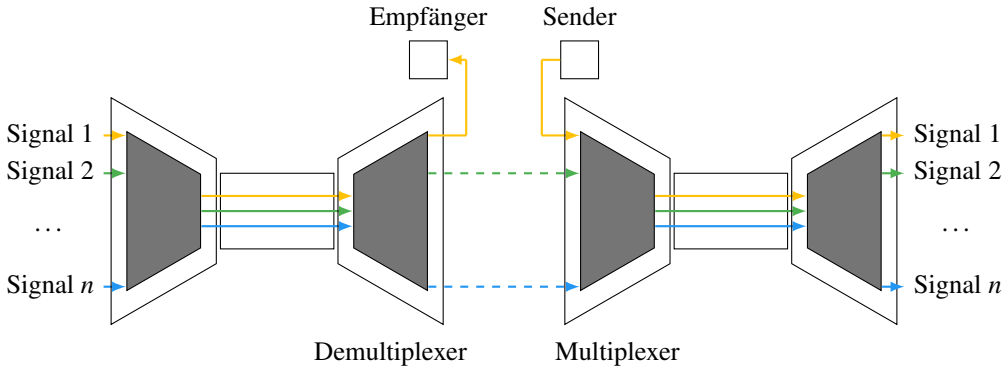


Abbildung 3.4: Station zum Hinzufügen, Extrahieren und Weiteleiten von Signalen.

Nun gibt es eine weitere Art von Multiplexverfahren, mit welchem die Kapazität des Netzwerks abermals erhöht werden kann. Die Kapazität eines (virtuellen) Lichtwellenkanals innerhalb eines Lichtwellenleiters ist in der Regel deutlich größer als die Kapazität, die ein einzelnes Signal belegt. Um die Kapazität eines Lichtwellenkanals besser auszunutzen, wurde das Verfahren des *Traffic Groomings* entwickelt, welches mehrere Signale gebündelt über einen gemeinsamen Lichtwellenkanal überträgt [AZ11].

Eine Instanz des Traffic Grooming Problems besteht nun aus einem optischen Netzwerk mit einer Menge von Stationen, welche über Lichtwellenleiter miteinander verbunden sind, und einer Menge von Signalen, welche von einer Station zu einer anderen übermittelt werden sollen. Ein Signal wird dabei über eine Sequenz von sogenannten *Lichtwegen* geleitet. Ein Lichtweg beschreibt eine Zuordnung eines (virtuellen) Lichtwellenkanals zu einer Sequenz von Lichtwellenleitern [AZ11]. An der Start- und Endstation eines jeden Lichtwegs können über entsprechende Multiplexer Signale extrahiert oder gebündelt werden.

Abbildung 3.5 zeigt ein Beispielnetzwerk mit einem gelb markierten Lichtweg, welcher einer Zuordnung eines Lichtwellenkanals zu den Lichtwellenleitern zwischen Station 1 und Station 2 sowie Station 2 und Station 4 entspricht. Der blau markierte Lichtweg entspricht einer Zuordnung eines Lichtwellenkanals zum Lichtwellenleiter zwischen Station 4 und Station 5. Der grün markierte Lichtweg entspricht einer Zuordnung eines Lichtwellenkanals zu den Lichtwellenleitern zwischen Station 3 und Station 2 sowie Station 2 und Station 4. Gegeben seien nun zwei Signale d_1, d_2, wobei d_1 von Station 1 zu

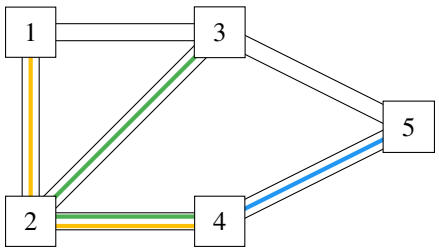


Abbildung 3.5: Netzwerk mit farbig markierten Lichtwegen.

Station 4 gesendet werden soll und d_2 von Station 1 zu Station 5. Mithilfe des Traffic Grooming-Verfahrens werden die Signale nun gebündelt über den gelben Lichtweg von Station 1 zu Station 4 übertragen und das Signal d_2 wird über den blauen Lichtweg weiter zu Station 5 übermittelt.

Bei gegebener Instanz beschreibt das Traffic Grooming Problem nun die Bestimmung einer zulässigen Menge von Lichtwegen und die Zuordnung der gegebenen Signale zu einer Sequenz von Lichtwegen unter Berücksichtigung der Kapazitäten bei Minimierung der Netzwerkkosten. Eine Menge von Lichtwegen heißt zulässig, wenn

- jeder Lichtweg unabhängig vom Lichtwellenleiter den gleichen Lichtwellenkanal (die gleiche Wellenlänge, also in den Abbildungen die gleiche Farbe) verwendet,
- jeweils zwei Lichtwege, welche den selben Lichtwellenleiter nutzen, unterschiedlichen Lichtwellenkanälen zugeordnet sind und
- für jedes Signal eine Sequenz von Lichtwegen existiert, mit welcher dieses von der Start- zur Endstation übermittelt werden kann.

Die Netzwerkkosten setzen sich aus Kosten für die benötigte Technik an der Start- und Endstation eines jeden Lichtwegs und Nutzungskosten der Lichtwellenkanäle zusammen.[AZ11] Das Traffic Grooming Problem ist NP-vollständig [CM00].

Offensichtlich kann das Traffic Grooming Problem auf einem Graphen formuliert werden (vgl. [AZ11]). Gegeben sei ein ungerichteter Graph $G = (V, E, c)$ und eine Menge $\mathcal{D}$ von zu übermittelnden Signalen. Die Knotenmenge V beschreibt die Stationen des optischen Netzwerks, die Kantenmenge E die Verbindungen dieser Stationen mit Lichtwellenleitern. Die Kostenfunktion $c\colon E \to \mathbb{N}$ beschreibt die Nutzungskosten der Lichtwellenkanäle eines Lichtwellenleiters. Jedes Signal $d \in \mathcal{D}$ wird durch ein Knotenpaar $d = (u, v) \in V \times V$ beschrieben, welches dem Start- und Endknoten des Signals entspricht. Jedes Signal d hat einen Kapazitätsbedarf von $b_d \in \mathbb{N}$ an der Gesamtkapazität B eines Lichtwellenkanals. Die zu übermittelnden Signale können über einen *Bedarfsgraphen* $G_d = (V, \mathcal{D})$ beschrieben werden. Da mehrere Signale denselben Start- und Endknoten besitzen können, ist G_d ein Multigraph. Gesucht ist nun eine Menge von Wegen $\mathcal{P}$ in G, welche den einzelnen Lichtwegen im Netzwerk entsprechen, und eine binäre Relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{D}$, welche die Zuordnung der Lichtwege zu den Signalen beschreibt. Sei nun $G_p = (V, E_p)$ analog zum Bedarfsgraph der sogenannte *Lichtweggraph*, wobei E_p für jeden Lichtweg $p \in \mathcal{P}$ mit Startknoten u und Endknoten v eine Kante $e = (u, v)$ enthält. Sei weiterhin $\mathcal{P}_{uv}$ die Menge aller Lichtwege mit Startknoten u und Endknoten v sowie $\mathcal{D}_{uv} := \{d \mid pRd, p \in \mathcal{P}_{uv}\}$ die Menge aller Signale, welche über einen Lichtweg $p \in \mathcal{P}_{uv}$ übermittelt werden. Dann muss die binäre Relation R die folgenden Eigenschaften erfüllen:

1. Für jede Kante $(u, v) \in G_p$ muss die Kapazitätsbeschränkung $\sum_{d \in \mathcal{D}_{uv}} b_d \leqslant B \cdot |\mathcal{P}_{uv}|$ eingehalten werden.

2. Für jedes Signal $d = (u, v) \in \mathcal{D}$ müssen die Kanten, welche in G_p einen Lichtweg mit pRd repräsentieren, einen Weg in G_p bilden.

Die Netzwerkkosten $|\mathcal{P}| + \sum_{e \in E} c(e)p(e)$ mit $p(e) := |\{p \in \mathcal{P} \mid e \in p\}|$ sollen minimiert werden. Diese Kosten setzen sich zusammen aus Kosten von einer Einheit für jeden Lichtweg $p \in \mathcal{P}$, welche die Kosten der benötigten Technik an Start- und Endstation eines jeden Lichtwegs beschreiben und Nutzungskosten $c(e)$ für jeden Lichtwellenkanal.

Relevant wird das maximale Kreispackungsproblem in Bezug auf eine Lösung des Traffic Grooming Problems nun beim Betrachten eines Spezialfalls. Gegeben sei ein Netzwerk und eine Menge von zu übermittelnden Signalen $\mathcal{D}$, wobei jedes Signal d einen Kapazitätsbedarf von $b_d = \frac{B}{2}$ hat. Weiterhin wird die Zielfunktion angepasst. Im Folgenden wird nur die Anzahl der Lichtwege $|\mathcal{P}|$ minimiert, welche die Kosten für die benötigte Technik an Start- und Endstation eines jeden Lichtwegs repräsentieren. Selbst für dieses vereinfachte Problem wurde in [AZ11] gezeigt, dass es APX-schwer ist.Nun kann jedoch für dieses Problem mithilfe der Kreispackungszahl des Bedarfsgraphen eine minimale Anzahl von Lichtwegen bestimmt werden, welche für die optimale Lösung des beschriebenen Problems benötigt werden. Sei $D = |\mathcal{D}|$ die Anzahl der unterschiedlichen Signale, welche übermittelt werden sollen, und $\nu(G_d)$ die Kreispackungszahl des Bedarfsgraphen G_d, dann werden in der optimalen Lösung des Traffic Grooming Problems genau

$$OPT = D - \nu(G_d)$$

Lichtwege benötigt [AZ11].

Gegeben sei ein Netzwerk mit 5 Stationen wie beispielsweise in Abbildung 3.5. Zu diesem Netzwerk sei der Bedarfsgraph G_d aus Abbildung 3.6 mit der Signalmenge $\mathcal{D} = \{d_1 = (1, 3), d_2 = (1, 4), d_3 = (1, 5), d_4 = (3, 4)\}$ und $b_{d_i} = \frac{B}{2}$ gegeben. Offensichtlich gilt für diesen Graphen $\nu(G_d) = 1$ mit dem Kreis $C = \{1, 3, 4, 1\}$, sodass eine optimale Lösung mit $OPT = D - \nu(G_d) = 4 - 1 = 3$ Lichtwegen auskommen müsste. Zunächst ist anzumerken, dass die Einsparung eines Lichtwegs durch einen Kreis im Bedarfsgraphen einfach nachzuvollziehen ist. Gegeben sei ein Kreis $C = \{(v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n), (v_n, v_1)\}$ in einem Bedarfsgraphen. Jede Kante (v_i, v_j) entspricht einem Signal, welches von Station v_i zu Station v_j übermittelt werden muss. Für jede der ersten $n - 1$ Verbindungen

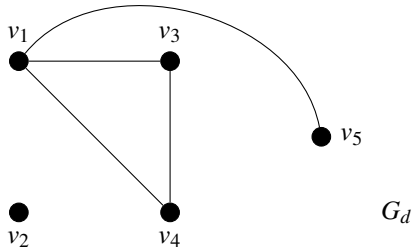


Abbildung 3.6: Bedarfsgraph G_d mit vier zu übermittelnden Signalen.

muss ein einzelner Lichtweg genutzt werden, da die Signale weder an einer gemeinsamen Station starten noch an einer gemeinsamen Station enden. Die einzige Ausnahme ist das n-te Signal, welches zwischen Station v_1 und Station v_n übermittelt werden soll. Anstatt einen eigenen Lichtweg zur Übermittlung zu verwenden, kann es mithilfe einer Sequenz der $n-1$ anderen Lichtwege übertragen werden. Für das Beispiel aus Abbildung 3.6 kann das Signal, welches von 1 zu 4 übertragen werden soll, also über einen Lichtweg von 1 zu 3 und im Anschluss gebündelt mit einem weiteren Signal über einen Lichtweg von 3 zu 4 übermittelt werden.

Nun stellt sich jedoch die Frage, ob die Lichtwege so konstruiert werden können, dass die Kapazität der Lichtwellenkanäle eingehalten wird. Wie zuvor beschrieben hat jedes Signal einen Kapazitätsbedarf von $b_d = \frac{B}{2}$, was gleichbedeutend damit ist, dass jeder Lichtwellenkanal genau zwei gebündelte Signale gleichzeitig übertragen kann. Betrachte hierfür Abbildung 3.7 mit den Bedarfen aus Abbildung 3.6.

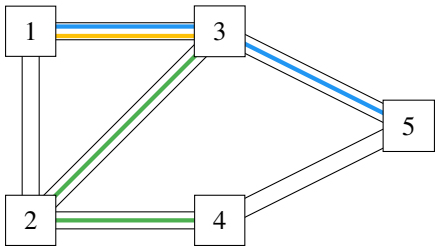


Abbildung 3.7: Minimale Auswahl von Lichtwegen für die Signalübermittlung entsprechend des Bedarfsgraphen G_d.

Zu sehen sind, wie zuvor berechnet, drei verschiedenfarbige Lichtwege. Die Sequenzen aus Lichtwegen für die einzelnen Signale lassen sich einfach wie in Tabelle 3.1 über die Farben beschreiben. Dabei werden offensichtlich zwei Signale über den gelben Lichtweg

übermittelt und ebenfalls zwei Signale über den grünen Lichtweg. Der blaue Lichtweg übermittelt nur ein Signal. Bei einer Kapazität von B je Lichtweg und einem Kapazitätsbedarf von $b_{d_i} = \frac{B}{2}$ für jedes Signal $d_i, i = 1, \ldots, 4$ wird die Kapazitätsbeschränkung damit eingehalten. Die Einhaltung der Kapazitätsbeschränkung ist ebenfalls über die Be-

Signal	Lichtwegsequenz
$d_1 = (1, 3)$	gelb
$d_2 = (1, 4)$	gelb, grün
$d_3 = (1, 5)$	blau
$d_4 = (3, 4)$	grün

Tabelle 3.1: Zuordnung der Lichtwegsequenzen zu den zu übermittelnden Signalen entsprechend G_d.

trachtung der kantendisjunkten Kreise zu erklären. Wird jedes Signal einzeln über einen eigenen Lichtweg übermittelt, wird die Kapazitätsbeschränkung offensichtlich eingehalten. Enthält G_d einen Kreis, kann, wie zuvor beschrieben, genau ein Lichtweg eingespart werden, sodass ein zusätzliches Signal über die anderen Lichtwege des Kreises übermittelt werden muss. Diese Lichtwege sind bezüglich ihrer Kapazität nun ausgelastet. Enthält G_d einen weiteren kantendisjunkten Kreis, wird ebenfalls ein Signal mit den anderen Signalen des Kreises gebündelt, die aber keine Lichtwege eines anderen Signals (außer der dem Kreis zugehörigen) verwenden. So werden immer höchstens zwei Signale auf einem Lichtweg gebündelt.

3.3 Das Minimum Local Scenario Problem

Das Anwendungsgebiet, welches häufig im Kontext mit Untersuchungen zu maximalen Kreispackungen in Graphen genannt wird, sind *Genome Rearrangement Probleme*. Dieser Problemtyp soll eine Schätzung für die evolutionäre Distanz zweier gegebener Genome liefern, indem eine minimale Anzahl von Umordnungen bestimmt wird, mit denen das eine Genom in das andere überführt werden kann. Als *Genom* wird die Gesamtheit aller vererbbaren Informationen einer Zelle, das heißt die gesamte DNA eines Lebewesens bezeichnet. Die DNA liegt in der Zelle in ein oder mehreren *Chromosomen* vor, daher wird zwischen *unichromosomalen* und *multichromosomalen* Genomen unterschieden. Es gibt sowohl *lineare* Chromosomen, deren Enden *Telomere* genannt werden, als auch *zirkuläre* Chromosomen. Begrenzte Abschnitte der DNA innerhalb eines Chromosoms werden *Gene* genannt. Da sich die Bausteine der DNA, die *Nukleotide*, nur in einem

Bestandteil unterscheiden und es von diesem auch nur vier verschiedene Varianten gibt, kann mit einem Alphabet von vier Buchstaben entsprechend der Variante des Nukleotids die DNA-Sequenz eindeutig beschrieben werden. Die unterschiedlichen Bestandteile der Nukleotide sind *Adenin (A), Guanin (G), Thymin (T)* und *Cytosin (C)*. Eine Folge von Nukleotiden bildet also ein Gen und eine Folge von Genen wiederum den DNA-Strang. In einer Zelle verbinden sich *komplementäre* DNA-Stränge zu einer *Doppelhelix*. Dabei ergibt sich der komplementäre DNA-Strang aus den jeweiligen Komplementen der unterschiedlichen Nukleotid-Bestandteile. Adenin verbindet sich mit Thymin und Guanin verbindet sich mit Cytosin. Die durch bestimmte Anfangs- und Endgruppen definierte Leserichtung beider DNA-Einzelstränge ist dabei gegenläufig zueinander. [Sch17]

Umordnungen und Mutationen bezeichnen Arten von Veränderungen des Erbguts. Dabei beschreiben Mutationen ausschließlich lokale Veränderungen der DNA auf Nukleotid-Ebene. Der Vergleich einzelner Gene auf Nukleotid-Ebene ist daher eine Problemstellung aus dem Bereich der Sequenzanalyse, bei welcher die minimale Zahl an lokalen Mutationen bestimmt werden soll, um ein Gen in ein anderes zu überführen. Hierfür wird die oben beschriebene Darstellung eines Gens als Zeichenkette genutzt und für zwei gegebene Zeichenketten die minimale Anzahl *lokaler Operationen* wie das Einfügen, Löschen oder Ersetzen einzelner Zeichen gesucht, um die eine Zeichenkette in die andere zu überführen.

Umordnungen bezeichnen dagegen globale Veränderungen wie beispielsweise Vertauschungen von ganzen Abschnitten eines oder mehrerer Chromosome, Inversionen verschiedener Abschnitte und einige weitere und entstehen, wenn ein Chromosom an mehreren Stellen zerbricht. Beim Genome Rearrangement Problem wird nun versucht die Unterschiede, die durch solche Umordnungen zwischen zwei Genomen entstanden sind, zu bestimmen und zu messen. Hierfür wird für Genome eine Folge von bestimmten Abschnitten betrachtet, welche in beiden Genomen vorhanden sind. Zur Vereinfachung wird in den weiteren Erläuterungen angenommen, dass es sich bei den betrachteten Abschnitten um Gene handelt und im Fall von multichromosomalen Genomen die Folge der Gene einer zuvor festgelegten Reihenfolge der Chromosomen folgt. Da weiterhin angenommen wird, dass die Gene unterscheidbar sind, können die Gene eines Genoms durch eine Folge von Ziffern [Gus97] oder alternativ auch Buchstaben beschrieben werden [BMS06]. Zusätzlich kann die Orientierung bzw. Leserichtung der Gene über ein entsprechendes Vorzeichen beschrieben werden. Ist die Orientierung für alle Gene bekannt, wird auch von *signierten* Genomen gesprochen, im anderen Fall heißt ein Genom *unsigniert*. Im Folgenden werden ausschließlich signierte Genome betrachtet. Ein Chro-

mosom bestehend aus benachbarten Genen a, b und c (in dieser Reihenfolge) könnte also über die Folge $(a, -b, c)$ beschrieben werden, wobei das negative Vorzeichen vor Gen b bedeutet, dass dieses Gen eine gegensätzliche Leserichtung besitzt. Die Menge $A = \{(a, -b, c), (d, e, d), (-f, -g, h)\}$ beschreibt dann beispielsweise ein Genom bestehend aus einer Menge von drei Chromosomen. Eine Instanz des Genome Rearrangement Problems besteht in diesem Kontext aus zwei signierten, unichromosomalen oder multichromosomalen Genomen A, B, welche aus derselben Menge von Genen bestehen, und einer Menge von sogenannten *Umordnungsoperationen*. Gesucht ist dann eine Folge mit einer minimalen Anzahl von Umordnungsoperationen, welche ein Genom in das andere überführt. Diese minimale Anzahl beschreibt dann die *Distanz* zweier Genome.

In der Forschung wird das Genome Rearrangement Problem häufig auf Probleme mit nur einem Typ von Umordnungsoperationen beschränkt. Das liegt unter anderem daran, dass einzelne solcher Probleme bereits NP-schwer sind, wie beispielsweise Genome Rearrangement Probleme für unsignierte Genome, bei welchen ausschließlich der Typ *Inversion* als Umordnungsoperation zulässig sind (*Sorting by Reversals*) [Cap97]. Im entsprechenden Beweis nutzt Caprara die Beziehung zur maximalen Anzahl an kantendisjunkten alternierenden Kreisen in speziellen Graphen, daher wird dieses spezielle Genome Rearrangement Problem häufig im Rahmen von Anwendungen für Kreispackungsprobleme erwähnt [Bal03; CPR03; Spr15].

Im Folgenden soll ein Genome Rearrangement Problem mit anderen Typen von Umordnungsoperationen vorgestellt werden, für welches es ebenfalls Aussagen gibt, die die Anzahl kantendisjunkter Kreise in bestimmten Graphen verwenden. Dafür muss zunächst jedoch erläutert werden, wie ein Genom mithilfe eines Graphen dargestellt werden kann. In [BMS06] stellen Bergeron et al. einen vereinheitlichten Ansatz zur Notation eines Genoms und einer Modellierung als Graph vor, der im Folgenden auch unter Verwendung von [Zak08] erläutert werden soll. Gegeben sei ein beliebiges multichromosomales, signiertes Genom bestehend aus linearen und zirkulären Chromosomen. Sei weiterhin a ein beliebiges Gen dieses Genoms. Da hier nur signierte Genome betrachtet werden, hat jedes Gen eine Orientierung. Diese startet am *Schwanz (engl. tail)* und endet am *Kopf (engl. head)*. Dies sind die *Extremitäten* eines Gens. Entsprechend wird der *Schwanz* des Gens a mit a_t bezeichnet und der *Kopf* mit a_h. Die Lage der Gene auf einem Chromosom wird über ein- und zweielementige Mengen der Extremitäten, *Telomere* und *Adjazenzen* genannt, beschrieben. Ist eine Extremität des Gens a nicht benachbart zu einer Extremität eines anderen Gens, wird dies durch das Telomer $\{a_t\}$ bzw. $\{a_h\}$ beschrieben. Ist das Gen a benachbart zu einem Gen b, kann dies je nach Signatur der Gene durch

die Adjazenzen $\{a_t, b_t\}, \{a_t, b_h\}, \{a_h, b_t\}$ und $\{a_h, b_h\}$ dargestellt werden. In dieser Form der Darstellung wird ein Genom mit n Genen dann durch eine Menge von Telomeren und Adjazenzen beschrieben, sodass jeder Schwanz und jeder Kopf eines jeden Gens genau einmal in dieser Menge enthalten sind. Aus dieser Notation kann ein *Genomgraph* konstruiert werden.

Sei A ein Genom und $\tilde{A}$ die Darstellung von A durch Telomere und Adjazenzen. Sei weiterhin $A_{Gene} := \{\{a_t, a_h\} \mid a \text{ ist Gen}\}$ die Menge aller Gene beschrieben durch ihre Extremitäten. Dann heißt $G = (V, E)$ mit $V = \tilde{A}$ und $(u, v) \in E \Leftrightarrow \exists a \in A_{Gene}$ mit $|a \cap (u \cup v)| = 2$ *Genomgraph* zu A. Jede Kante des Genomgraphen entspricht somit einem Gen, da jede Extremität eines Gens in genau einem Telomer oder einer Adjazenz enthalten ist. Auf diese Weise erhält man einen Graph mit Knoten, die ausschließlich Knotengrad eins oder zwei besitzen, und Zusammenhangskomponenten (Wege und Kreise), die die linearen und zirkulären Chromosomen repräsentieren.

Beispiel 3.3
Sei $A = \{(a, -b, c), (d, e, d), (-f, -g, h)\}$ ein Genom mit den Genen $\{a, b, c, d, e, f, g, h\}$. Überführt man A in die Darstellung durch Adjazenzen und Telomere ergibt sich

$$\tilde{A} = \{\{a_t\}, \{a_h, b_h\}, \{b_t, c_t\}, \{c_h\}, \{d_h, e_t\}, \{e_h, d_t\}, \{f_h\}, \{f_t, g_h\}, \{g_t, h_t\}, \{h_h\}\}.$$

Der Graph G in Abbildung 3.8 ist der Genomgraph zu $\tilde{A}$ mit zwei linearen und einem zirkulären Chromosom.

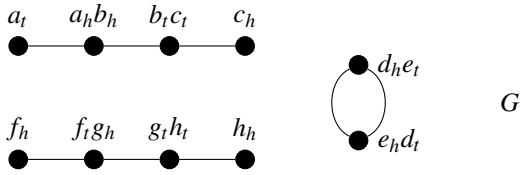


Abbildung 3.8: Genomgraph G des Genoms A.

Das Genome Rearrangement Problem, welches nun betrachtet werden soll, fasst verschiedene Typen von Umordnungen in einer Operation zusammen. Die Operation heißt *Double Cut and Join-Operation* (kurz DCJ-Operation). Sie kann allgemein auf Graphen mit Knoten, die ausschließlich einen Knotengrad von eins oder zwei besitzen angewendet werden. In Bezug auf die hier betrachteten Genomgraphen trennt diese Operation höchstens zwei Adjazenzen und fügt die entstehenden oder bestehende Telomere zu höchstens

zwei Adjazenzen zusammen. Wie in [SS17] seien p, q, r, s die Extremitäten verschiedener Gene eines Genoms. Dann ist eine *DCJ-Operation* eine Umordnung der Adjazenzen und Telomere entsprechend der folgenden Regeln:

1. $\{p, q\}, \{r, s\} \rightarrow \{p, r\}, \{q, s\}$ oder $\{p, s\}, \{q, r\}$
2. $\{p, q\}, \{r\} \rightarrow \{p, r\}, \{q\}$ oder $\{q, r\}, \{p\}$
3. $\{p, q\} \rightarrow \{p\}, \{q\}$
4. $\{p\}, \{q\} \rightarrow \{p, q\}$

Abhängig davon, ob die DCJ-Operation auf verschiedenen, demselben, linearen oder zirkulären Chromosomen angewendet wird, werden durch diese Regeln unterschiedliche Genomumordnungen beschrieben.

Regel 1: Fall 1.1 Es werden zwei Adjazenzen auf verschiedenen linearen Chromosomen verwendet und neu kombiniert. In diesem Fall findet ein Austausch der Genfolgen zwischen der jeweiligen Adjazenz und dem Telomer des Chromosoms statt (Translokation).

Fall 1.2 Es werden zwei Adjazenzen auf denselben Chromosomen verwendet und neu kombiniert. In diesem Fall findet eine Inversion der Genfolge zwischen den Adjazenzen statt (Inversion) oder eine neu kombinierte Adjazenz bildet ein neues zirkuläres Chromosom (Exzision).

Fall 1.3 Es werden zwei Adjazenzen eines linearen und eines zirkulären Chromosoms bzw. verschiedener zirkulärer Chromosomen verwendet und neu kombiniert. In diesem Fall wird das zirkuläre Chromosom in das lineare bzw. das andere zirkuläre Chromosom integriert (Integration).

Regel 2: Fall 2.1 Es werden eine Adjazenz und ein Telomer auf verschiedenen linearen Chromosomen verwendet und neu kombiniert. In diesem Fall findet ein Austausch der Genfolge zwischen der Adjazenz und dem Telomer des Chromosoms und dem Gen des Telomers statt (Translokation).

Fall 2.2 Es werden eine Adjazenz und ein Telomer auf demselben linearen Chromosomen verwendet und neu kombiniert. In diesem Fall findet eine Inversion der Genfolge zwischen der Adjazenz und dem Telomer statt (Inversion) oder die Genfolge zwischen der Adjazenz und dem Telomer bildet ein neues zirkuläres

Chromosom (Exzision).

Fall 2.3 Es werden eine Adjazenz eines zirkulären Chromosoms und ein Telomer eines linearen Chromosoms verwendet und neu kombiniert. In diesem Fall wird das zirkuläre Chromosom linearisiert und an das Telomer angehängt (Fusion).

Regel 3: Fall 3.1 Es wird eine Adjazenz eines linearen Chromosoms verwendet und getrennt. In diesem Fall entstehen zwei neue lineare Chromosomen (Fission).

Fall 3.2 Es wird eine Adjazenz eines zirkulären Chromosoms verwendet und getrennt. In diesem Fall entsteht ein lineares Chromosom (Linearisierung).

Regel 4: Fall 4.1 Es werden zwei Telomere verschiedener linearer Chromosomen verwendet und neu kombiniert. In diesem Fall werden die beiden Telomere zu einem neuen linearen Chromosom verschmolzen (Fusion).

Fall 4.2 Es werden zwei Telomere desselben linearen Chromosoms verwendet und verschmolzen. In diesem Fall wird das lineare Chromosom zu einem zirkulären Chromosom verschmolzen (Zirkularisierung).

Wird nun der Genomgraph G eines Genoms $\tilde{A}$ mit dem Genomgraph G' nach Anwendung einer DCJ-Operation verglichen, lässt sich die folgende Aussage formulieren.

Lemma 3.4 [Zak08]
Die Anwendung einer DCJ-Operation auf die Adjazenzen und Telomere eines Genoms $\tilde{A}$ ändert die Anzahl von Kreisen oder Wegen (und die Anzahl der Zusammenhangskomponenten) im Genomgraph G um höchstens eins.

Eine Instanz des Genome Rearrangement Problems mit DCJ-Operation besteht aus zwei signierten, unichromosomalen oder multichromosomalen Genomen $\tilde{A}$ und $\tilde{B}$ mit der gleichen Menge von n Genen. Als Umordnungsoperation ist die DCJ-Operation erlaubt. Gesucht ist dann eine Folge mit einer minimalen Anzahl von DCJ-Operationen, welche Genom $\tilde{A}$ in Genom $\tilde{B}$ überführt. Diese minimale Anzahl beschreibt dann die *DCJ-Distanz* $d_{DCJ}(\tilde{A}, \tilde{B})$ zweier Genome $\tilde{A}, \tilde{B}$. Zur Berechnung der DCJ-Distanz stellen Bergeron et al. in [BMS06] eine einfache Datenstruktur vor: Seien $\tilde{A}, \tilde{B}$ zwei Genome

mit der gleichen Menge von n Genen. Es sei $V_1 = \tilde{A}$ und $V_2 = \tilde{B}$. Dann nennt man den Multigraphen $AG(\tilde{A}, \tilde{B}) = (V_1 \cup V_2, \tilde{E})$ mit $(u, v) \in \tilde{E} \Leftrightarrow u \cap v \neq \varnothing$ *Adjazenzgraph*.

Bemerkung 3.5
Der Graph $AG(\tilde{A}, \tilde{B})$ ist bipartit.

Beispiel 3.6
Seien $\tilde{A}, \tilde{B}$ zwei Genome.

$$\tilde{A} = \{\{a_t\}, \{a_h, b_h\}, \{b_t, c_t\}, \{c_h\}, \{d_h, e_t\}, \{e_h, d_t\}, \{f_h\}, \{f_t, g_h\}, \{g_t, h_t\}, \{h_h\}\}$$
$$\tilde{B} = \{\{a_t\}, \{a_h, d_h\}, \{d_t, e_h\}, \{e_t\}, \{b_h, c_t\}, \{c_h, b_t\}, \{g_h\}, \{g_t, f_h\}, \{f_t, h_h\}, \{h_t\}\}$$

Der Graph $AG(\tilde{A}, \tilde{B})$ in Abbildung 3.9 ist der Adjazenzgraph zu $\tilde{A}$ und $\tilde{B}$.

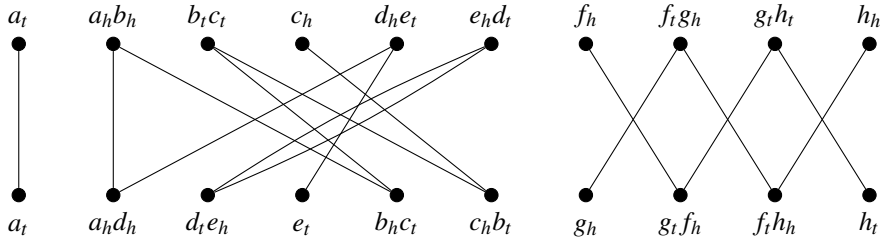


Abbildung 3.9: Adjazenzgraph $AG(\tilde{A}, \tilde{B})$ der Genome $\tilde{A}, \tilde{B}$.

Da jede Extremität in genau zwei Knoten enthalten ist, haben alle Knoten des bipartiten Graphen Knotengrad eins oder zwei. Auch bei diesem Graphen entsprechen die Zusammenhangskomponenten also Kreisen und Wegen. Kreise in bipartiten Graphen enthalten eine gerade Anzahl an Kanten. Auf Grundlage dieser Eigenschaften wurden in [Zak08] zwei Aussagen formuliert, welche zur Entwicklung einer unteren Schranke für die DCJ-Distanz zweier Genome beitragen:

Lemma 3.7 [Zak08]
Seien $\tilde{A}$ und $\tilde{B}$ zwei Genome, welche jedes der n Gene genau einmal enthalten. Sei $c(AG(\tilde{A}, \tilde{B})$ die Anzahl der Kreise im Adjazenzgraph und $ung(AG(\tilde{A}, \tilde{B}))$ die Anzahl ungerader Wege. Dann gilt

$$\tilde{A} = \tilde{B} \quad \Leftrightarrow \quad n = c(AG(\tilde{A}, \tilde{B})) + \frac{1}{2} ung(AG(\tilde{A}, \tilde{B}))$$

Offensichtlich besteht ein Adjazenzgraph zweier identischer Genome $\tilde{A}$ und $\tilde{B}$ aus Kreisen der Länge 2, welche die identischen Adjazenzen repräsentieren, und ungeraden Wegen der Länge 1, welche die identischen Telomere repräsentieren. Analog zu Lemma 3.4 kann eine Aussage über Auswirkungen der DCJ-Operation auf die Anzahl der ungeraden Wege im Adjazenzgraph getroffen werden.

Lemma 3.8 [Zak08]
Die Anwendung einer DCJ-Operation auf die Adjazenzen und Telomere eines Genoms $\tilde{A}$ ändert die Anzahl von ungeraden Wegen im Adjazenzgraph $AG(\tilde{A}, \tilde{B})$ um $-2, 0, 2$.

Aus Lemma 3.4, 3.7 und 3.8 konnten Zakotnik in [Zak08] eine untere Schranke für die DCJ-Distanz auf Basis der Kreise und ungeraden Wege im Adjazenzgraph entwickeln. Zusammen mit einem Greedy-Algorithmus, welcher das Genome Rearrangement Problem mit DCJ-Operation löst und die untere Schranke realisiert, ergibt sich die folgende Aussage.

Satz 3.9 [Zak08]
Seien $\tilde{A}$ und $\tilde{B}$ zwei Genome, welche jedes der n Gene genau einmal enthalten. Sei $c(AG(\tilde{A}, \tilde{B})$ die Anzahl der Kreise im Adjazenzgraph und $ung(AG(\tilde{A}, \tilde{B}))$ die Anzahl ungerader Wege. Dann gilt

$$d_{DCJ}(\tilde{A}, \tilde{B}) = n - \left(c(AG(\tilde{A}, \tilde{B})) + \frac{1}{2} ung(AG(\tilde{A}, \tilde{B})) \right)$$

Die in Satz 3.9 verwendeten Kreise sind aufgrund der Definition von $AG(\tilde{A}, \tilde{B})$ kantendisjunkt, da $AG(\tilde{A}, \tilde{B})$ bipartit ist und $\delta(v) \leqslant 2$ für alle $v \in V(AG(\tilde{A}, \tilde{B}))$. Erst in einer Weiterentwicklung des Genome Rearrangement Problems mit DCJ-Operation werden explizit kantendisjunkte Kreise benötigt. In [SS17] bzw. [SS18] stellen Simonaitis und Swenson ein Genome Rearrangement Problem vor, bei welchem aus biologischen Gründen unwahrscheinliche DCJ-Operationen minimiert werden sollen. Hierfür wird eine Gewichtung von DCJ-Operationen eingeführt, welche Adjazenzen und Telomere markiert, die mit höherer Wahrscheinlichkeit Teil einer Umordnungsoperation sind als andere. Verwendet wird eine Färbung der Adjazenzen und Telomere, welche eine binäre Gewichtung dieser repräsentieren soll. Sei $\tilde{A}$ ein Genom. Eine *Färbung* von $\tilde{A}$ mit einer Menge von Farben Δ ist eine Funktion $col\colon \tilde{A} \to \Delta$, welche die Adjazenzen und Telomere von $\tilde{A}$ in

Teilmengen verschiedener Farben aufteilt, und wird für $p \in \tilde{A}$ als Paar $(p, col(p))$ dargestellt. Die Färbung definiert die *Kosten* einer DCJ-Operation: Eine DCJ-Operation heißt *lokal* und hat Kosten Null, falls sie auf Adjazenzen und Telomere mit der gleichen Farbe angewendet wird. Ist die Farbe der verwendeten Adjazenzen und Telomere nicht identisch, heißt die DCJ-Operation *nicht lokal* und hat Kosten von Eins. Die Gesamtkosten einer Folge von DCJ-Operationen ergeben sich aus der Summe der einzelnen Kosten. Eine DCJ-Operation gefärbter Adjazenzen und Telomere ist wie folgt definiert.

Seien p, q, r, s die Extremitäten verschiedener Gene eines Genoms und seien x, y, z drei verschiedene Farben. Dann ist eine *DCJ-Operation mit Färbung* eine Umordnung gefärbter Adjazenzen und Telomere entsprechend der folgenden Regeln:

1. $(\{p, q\}, x), (\{r, s\}, y) \rightarrow (\{p, r\}, x), (\{q, s\}, y)$ oder $(\{p, r\}, y), (\{q, s\}, x)$
2. $(\{p, q\}, x), (\{r\}, y) \rightarrow (\{p, r\}, x), (\{q\}, y)$ oder $(\{p, r\}, y), (\{q\}, x)$
3. $(\{p, q\}, x) \rightarrow (\{p\}, x), (\{q\}, z)$ oder $(\{p\}, z), (\{q\}, x)$ für irgendeine Farbe z
4. $(\{p\}, x), (\{q\}, y) \rightarrow (\{p, q\}, x)$ oder $(\{p, q\}, y)$

Analog zur Berechnung der DCJ-Distanz in polynomieller Zeit in [BMS06] wurde in [SSB16] gezeigt, dass auch die DCJ-Distanz mit Färbung (dort *Minimum Local Scenario Parsimonious Problem (MLPS)* genannt) in polynomieller Zeit bestimmt werden kann.

Zur Vereinfachung der folgenden Erläuterungen werden nun ausschließlich signierte Genome mit zirkulären Chromosomen betrachtet. Die Menge $\tilde{A}$ besteht in diesem Fall dann ausschließlich aus einer Menge von Adjazenzen (die Menge der Telomere ist leer).

Eine Instanz des *Minimum Local Scenario Problems (MLS-Problem)* besteht nun aus zwei signierten, unichromosomalen oder multichromosomalen Genomen $\tilde{A}$ und $\tilde{B}$, welche ausschließlich aus Adjazenzen bestehen, einer Färbung von $\tilde{A}$ und der DCJ-Operation mit Färbung. Gesucht ist nun eine kostenminimale Folge von DCJ-Operationen mit Färbung, welche $\tilde{A}$ in $\tilde{B}$ überführt. Im Folgenden wird nur noch von DCJ-Distanz gesprochen. Für die Entwicklung von Aussagen zur Lösung des MLS-Problems wird in [SS17] neben dem Adjazenzgraph ein weiterer Graph genutzt.

Seien $\tilde{A}, \tilde{B}$ zwei Genome mit der gleichen Menge von n Genen. Sei $col\colon \tilde{A} \Rightarrow \Delta$ eine Färbung von $\tilde{A}$. Der *Färbungsgraph* $J(\tilde{A}, \tilde{B}, col) = (\Delta, \bar{E})$ ist ein Multigraph, dessen Knotenmenge der Menge der Farben entspricht. Für jedes $p \in \tilde{B}$ enthält der Graph eine Kante $(x, y) \in \bar{E}$, sodass x und y die Farben der zu p adjazenten Knoten von A im

Adjazenzgraphen $AG(\tilde{A}, \tilde{B})$ sind.

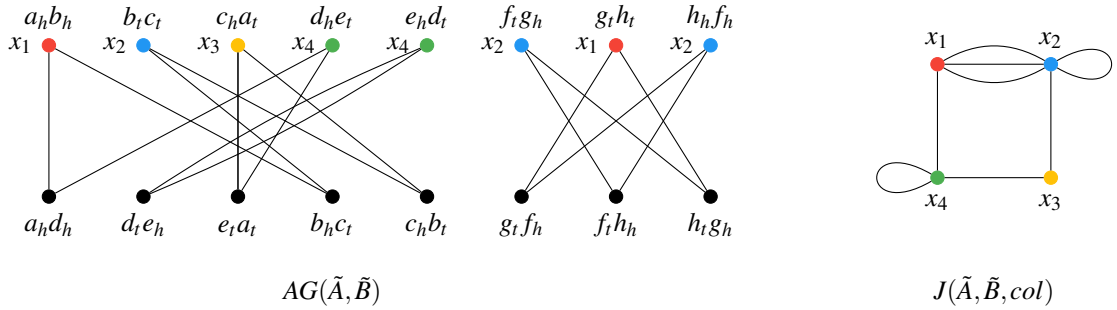
Beispiel 3.10

Seien $\tilde{A}$ und $\tilde{B}$ zwei Genome. Sei $col\colon \tilde{A} \Rightarrow \Delta$ eine Färbung von $\tilde{A}$.

$$A = \{(\{a_h, b_h\}, x_1), (\{b_t, c_t\}, x_2), (\{c_h, a_t\}, x_3), (\{d_h, e_t\}, x_4), (\{e_h, d_t\}, x_4), (\{f_t, g_h\}, x_2), (\{g_t, h_t\}, x_1), (\{h_h, f_h\}, x_2)\}$$

$$B = \{\{a_h, d_h\}, \{d_t, e_h\}, \{e_t, a_t\}, \{b_h, c_t\}, \{c_h, b_t\}, \{g_t, f_h\}, \{f_t, h_h\}, \{h_t, g_h\}\}$$

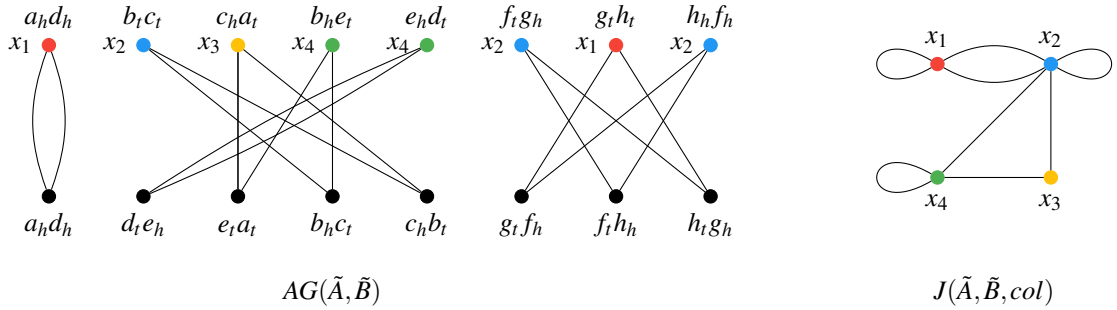
Der Graph $AG(\tilde{A}, \tilde{B})$ in Abbildung 3.10 ist der Adjazenzgraph zu $\tilde{A}$ und $\tilde{B}$. Der Graph $J(\tilde{A}, \tilde{B}, col)$ ist ein zugehöriger Färbungsgraph. Die Anwendung einer DCJ-Operation



$AG(\tilde{A}, \tilde{B})$ $J(\tilde{A}, \tilde{B}, col)$

Abbildung 3.10: Adjazenzgraph $AG(\tilde{A}, \tilde{B})$ und Färbungsgraph $J(\tilde{A}, \tilde{B}, col)$ der Genome $\tilde{A}, \tilde{B}$.

auf $(\{a_h, b_h\}, x_1)$ und $(\{d_h, e_t\}, x_4)$ führt zur Umordnung $(\{a_h, d_h\}, x_1)$ und $(\{b_h, e_t\}, x_4)$ und überführt das Genom $\tilde{A}$ in ein Genom $\tilde{A}'$. Der neue Adjazenzgraph $AG(\tilde{A}', \tilde{B})$ und der neue Färbungsgraph $J(\tilde{A}', \tilde{B}, col)$ ist in Abbildung 3.11 dargestellt.



$AG(\tilde{A}, \tilde{B})$ $J(\tilde{A}, \tilde{B}, col)$

Abbildung 3.11: Adjazenzgraph $AG(\tilde{A}', \tilde{B})$ und Färbungsgraph $J(\tilde{A}', \tilde{B}, col)$ der Genome $\tilde{A}', \tilde{B}$.

Wird eine DCJ-Operation auf ein Genom $\tilde{A}$ angewandt, ändert sich natürlich nicht nur der zugehörige Adjazenzgraph, sondern auch der entsprechende Färbungsgraph. Gilt in Folge einer Sequenz von DCJ-Operationen $\tilde{A} = \tilde{B}$, so besteht der Adjazenzgraph $AG(\tilde{A}, \tilde{B})$ wie oben beschrieben aus Kreisen der Länge zwei (und für allgemeine Genome ungeraden Wegen der Länge 1). Der Färbungsgraph $J(\tilde{A}, \tilde{B}, col)$ wiederum besteht dann ausschließlich aus einer Menge von Knoten mit je einer Schlaufe. Ein solcher Färbungsgraph wird *terminal* genannt. [SS17]

Wie im Beispiel 3.10 erkennbar, verursachen DCJ-Operationen ohne Kosten keine Änderungen im Färbungsgraph. Aus dieser Beobachtung heraus konnte in [SS17] gezeigt werden, dass die Gesamtkosten des MLS-Problems direkt mit der Überführung des Färbungsgraphs in einen terminalen Graph zusammenhängen.

Lemma 3.11 [SS17]
Seien $\tilde{A}, \tilde{B}$ zwei Genome mit der gleichen Menge von n Genen. Sei $col\colon \tilde{A} \to \Delta$ eine Färbung von $\tilde{A}$. Für jede Folge von DCJ-Operationen mit Gesamtkosten w, welche $\tilde{A}$ in $\tilde{B}$ überführt, gibt es eine Folge von höchstens w DCJ-Operationen, welche $J(\tilde{A}, \tilde{B}, col)$ in einen terminalen Graph überführt.

Nun gilt es unter Verwendung dieser Aussage eine Abschätzung für die Gesamtkosten w zu finden, mit denen $\tilde{A}$ in $\tilde{B}$ überführt werden kann. An dieser Stelle werden erneut maximale kantendisjunkte Kreispackungen bzw. die Kreispackungszahl eines Graphen relevant. Bezeichne $w(J)$ die minimale Anzahl von DCJ-Operationen, die benötigt werden, um einen Färbungsgraph in einen terminalen Graph zu überführen.

Satz 3.12 [SS17]
Sei $J(\tilde{A}, \tilde{B}, col) = (\Delta, \bar{E})$ ein Färbungsgraph. Dann gilt $w(J) = |\bar{E}| - \nu(J)$.

Da $\tilde{A}, \tilde{B}$ ausschließlich Adjazenzen und keine Telomere enthalten, bestehen die Zusammenhangskomponenten des Adjazenzgraphs aus Kreisen gerader Länge. Da im Färbungsgraph die Knoten den Farben entsprechen und über Kanten miteinander verbunden werden, falls im Adjazenzgraphen zwei gefärbte Adjazenzen aus $\tilde{A}$ über einen Weg miteinander verbunden sind, führt ein Kreis im Adjazenzgraph zu einem Kreis im Färbungsgraph. Wenn man die Schlingen im Färbungsgraph, die aus Kreisen der Länge zwei im Adjazenzgraph resultieren, nicht berücksichtigt, ist ein Färbungsgraph offensichtlich eulersch.

Für die Überführung eines Kreises C der Länge $l > 1$ im Färbungsgraph in eine Schlaufe, werden $l-1$ DCJ-Operationen benötigt. Da im Eulergraphen die Anzahl der Kanten aller Kreise gleich der Anzahl aller Kanten im Graphen ist, sind $|E| - \nu(J)$ DCJ-Operationen zur Überführung eines Färbungsgraphen in einen terminalen Graphen notwendig.

Kapitel 4

Kreispackungen und Graphzerlegungen

Dieses Kapitel dient als Einstieg in das Thema Graphzerlegungen und Kreispackungen. Anhand von nicht zusammenhängenden, 1-zusammenhängenden und 2-zusammenhängenden Graphen können wesentliche Überlegungen zur Beziehung zwischen einer Zerlegung dieser Graphen und der Bestimmung einer maximalen Kreispackung bzw. der Bestimmung der Kreispackungszahl erläutert werden. Die hier erarbeiteten Ergebnisse sind die Motivation für die in dieser Arbeit in Kapitel 5 folgenden Ausarbeitungen zu 3-zusammenhängenden Graphen.

4.1 Kreispackungen in nicht zusammenhängenden Graphen

Sei $G = (V, E)$ ein (nicht notwendigerweise zusammenhängender) Graph. Falls G nicht zusammenhängend ist, kann G in 1-zusammenhängende Teilgraphen G_i zerlegt werden und es gilt entsprechend der Motivation aus Kapitel 1 für die Kreispackungszahl $\nu(G)$ und die maximale Kreispackung $\mathcal{Z}^*(G)$, dass

$$\nu(G) = \sum \nu(G_i) \text{ und } \mathcal{Z}^*(G) = \bigcup \mathcal{Z}^*(G_i)$$

Die Zerlegung von G in solche einfach zusammenhängenden Komponenten G_i kann mit effizienten Algorithmen bestimmt werden. So entwickelten Hopcroft und Tarjan [HT73b] 1973 einen auf Tiefensuche basierenden Algorithmus zur Bestimmung aller Zusammenhangskomponten eines Graphen, welcher in diesem Abschnitt erläutert werden soll. Zunächst wird hierfür die Tiefensuche beschrieben.

Die Tiefensuche ist ein Verfahren, welches mit einer Laufzeit von $\mathcal{O}(|V| + |E|)$ systematisch einen Graphen inspiziert und jeden Knoten bei seinem ersten Besuch markiert. Dabei setzt der Algorithmus die Erforschung noch nicht markierter Knoten immer ausgehend vom gerade markierten Knoten fort, arbeitet in diesem Sinne zunächst in die Tiefe. Algorithmus 4.1 beschreibt das Verfahren formal. Sei $G = (V, E)$ ein zusammenhängender, ungerichteter Graph. Für einen Knoten $v \in V$ gibt der Wert $Besucht(v)$ an, ob der Knoten bereits besucht wurde ($Besucht(v) = 1$) oder nicht ($Besucht(v) = 0$). Die Tiefensuchenummer $Nummer(v)$ ist ein Zähler, der eine Reihenfolge induziert, in welcher die Knoten des Graphen zum ersten Mal besucht werden. Durch Festlegung einer solchen Reihenfolge wird während des Verfahrens für jede ungerichtete Kante eine Richtung bestimmt und so ein gerichteter Graph $\vec{G} = (V, \vec{E})$ konstruiert. Die Kanten werden dabei in zwei Typen aufgeteilt, sogenannte *Baumkanten* und sogenannte *Rückwärtskanten*. Eine Kante $\{u, v\} \in E$ wird im Verfahren zur Baumkante $(u, v) \in \vec{E}$, falls $Nummer(u) < Nummer(v)$. Eine Kante $\{u, v\} \in E$ wird zur Rückwärtskante $(u, v) \in \vec{E}$, falls ein gerichteter Weg aus Baumkanten von v nach u in $\vec{G} = (V, \vec{E})$ existiert. Die ungerichteten Baumkanten induzieren einen Spannbaum von G. Eine *Arboreszenz mit Wurzel* w ist ein gerichteter Graph, welcher aus einem (ungerichteten) Baum entsteht, indem man für jede Kante eine Orientierung wählt, sodass für jeden Knoten $v \in V$ ein gerichteter Weg von w nach v existiert. Ist der zugrunde liegende Baum der Arboreszenz ein Spannbaum, spricht man von einer *aufspannenden Arboreszenz*. Die gerichteten Baumkanten induzieren eine aufspannende Arboreszenz.

Algorithmus 4.1 Tiefensuche (DFS=Depth-first search)

Eingabe: Graph $G = (V, E)$.
Ausgabe: Information je Knoten: besucht/nicht besucht und Tiefensuchenummer

```
i ← 0
for all v ∈ V do
    Besucht(v) ← 0 und Nummer(v) ← 0
end for
for all v ∈ V do
    if Besucht(v) = 0 then
        DFS(v)
    end if
end for

procedure DFS(v)
    i ← i + 1, Besucht(v) ← 1 und Nummer(v) ← i
    for all w mit {v, w} ∈ E do
        if Besucht(w) = 0 then
            DFS(w)
        end if
    end for
end procedure
```

Abbildung 4.1(a) zeigt einen Graphen, auf welchem eine Tiefensuche bereits durchgeführt wurde. Die Knotenbezeichnungen entsprechen der Reihenfolge, in welcher die Knoten zuerst besucht wurden. Die zugehörige durch die Baumkanten induzierte aufspannende Arboreszenz, der sogenannte *Tiefensuchebaum* $\vec{T} = (V, \vec{E}^*)$, wird in Abbildung 4.1(b) dargestellt. Zu diesem Tiefensuchebaum gehören die gestrichelt dargestellten Rückwärtskanten, auch wenn es sich nun streng genommen nicht mehr um einen Baum handelt.

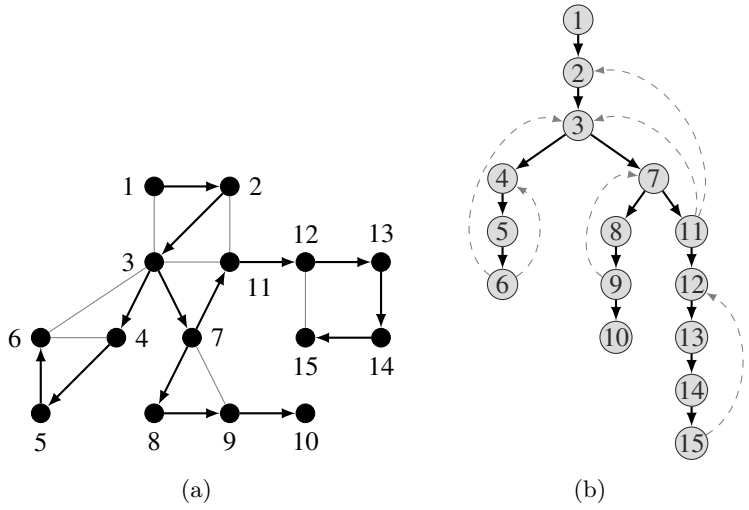


Abbildung 4.1: Ein Graph (a) und der zugehörige Tiefensuchebaum (b).

Der in [HT73b] formulierte Algorithmus zur Bestimmung der Zusammenhangskomponenten eines Graphen nutzt nun diese Tiefensuche und hat ebenfalls eine Laufzeit von $\mathcal{O}(|V| + |E|)$. Beginnend bei einem beliebigen Knoten werden alle von diesem Knoten mittels Tiefensuche erreichbaren Knoten besucht und mit dem Wert des aktuellen Zählers i markiert ($Komponente(v) \leftarrow i$). Kann kein weiterer Knoten mehr erreicht werden, ist eine Zusammenhangskomponente (mit der Nummer i) bestimmt, der Zähler wird erhöht und der Algorithmus beginnt erneut bei einem noch nicht markierten Knoten. Das Verfahren wird so lange fortgeführt, bis alle Knoten des Graphen markiert sind. Der jedem Knoten zugeordnete Zahlenwert gibt an, zu welcher Komponente dieser Knoten gehört. Das Maximum dieser Werte bestimmt dann die Anzahl der Zusammenhangskomponenten des Graphen. Algorithmus 4.2 beschreibt das Verfahren formal.

Algorithmus 4.2 Zusammenhangskomponenten eines Graphen (DFS-ZSHG)

Eingabe: Graph $G = (V, E)$.
Ausgabe: Information je Knoten: Komponentenzugehörigkeit

```
i ← 0
for all v ∈ V do
    Komponente(v) ← 0
end for
for all v ∈ V do
    if Komponente(v) = 0 then
        i ← i + 1
        DFS-ZSHG(v)
    end if
end for

procedure DFS-ZSHG(v)
    Komponente(v) ← i
    for all w mit {v, w} ∈ E do
        if Komponente(w) = 0 then
            DFS-ZSHG(w)
        end if
    end for
end procedure
```

4.2 Kreispackungen in 1-zusammenhängenden Graphen

Mit dem Wissen, dass ein Graph G effizient in seine 1-zusammenhängenden Komponenten G_i zerlegt werden kann, kann nun eine Zerlegung dieser 1-zusammenhängenden Komponenten G_i näher betrachtet und hinsichtlich ihrer Kreispackungen $\mathcal{Z}(G_i)$ untersucht werden. Sei G ein 1-zusammenhängender Graph. Wie in der Motivation aus Kapitel 1 propagiert, kann auch hier die Zerlegung von G in maximal 2-zusammenhängende Komponenten G_i, die sogenannten *Blöcke*, genutzt werden, um von einer maximalen Kreispackung und der Kreispackungszahl dieser Blöcke G_i Rückschlüsse auf eine maximale Kreispackung und die Kreispackungszahl von G zu ziehen. Zur Erläuterung eines solchen Vorgehens soll zunächst die Blockzerlegung von G genauer betrachtet werden. Die 1-Separatoren, welche G in seine Blöcke G_i zerfallen lassen, werden in der Literatur auch als *Schnittknoten* oder *Artikulationspunkte* bezeichnet. Ein Schnittknoten gehört zu mindestens zwei verschiedenen Blöcken. Jede Kante $e \in E$ gehört genau wie jeder Knoten $v \in V$, welcher kein 1-Separator ist, zu genau einem Block. Weiterhin enthalten zwei Blöcke G_i, G_j maximal einen gemeinsamen Knoten $\bar{v}$, welches dann der 1-Separator von G ist. Aufgrund dieser Eigenschaften lässt sich die Zerlegung eines 1-zusammenhängenden Graphen G in seine Blöcke G_i mithilfe eines sogenannten *Block-Graphen* $G_B = (V_B, E_B)$ darstellen (vergleiche Abbildung 4.2(b)). Es sei $V_B = V_c \cup B$, wobei V_c die Menge der

1-Separatoren ist und $B = \{b_1, \ldots, b_k\}$ die Knotenmenge, in welcher jeder Knoten b_i einen Block G_i repräsentiert. Weiterhin sei genau dann eine Kante $(c, b) \in E_B$, wenn der 1-Separator c in Block b enthalten ist. Da G 1-zusammenhängend ist, ist G_B ein Baum. Eine weitere Darstellungsvariante der Zerlegung ist der *BC-Baum* $\vec{G}_B = (V_B, \vec{E}_B)$ (vergleiche Abbildung 4.2(c)). Der BC-Baum ist eine Arboreszenz, welche aus dem Block-Graphen G_B mit einem beliebigen Block $b \in B$ als Wurzel entsteht.

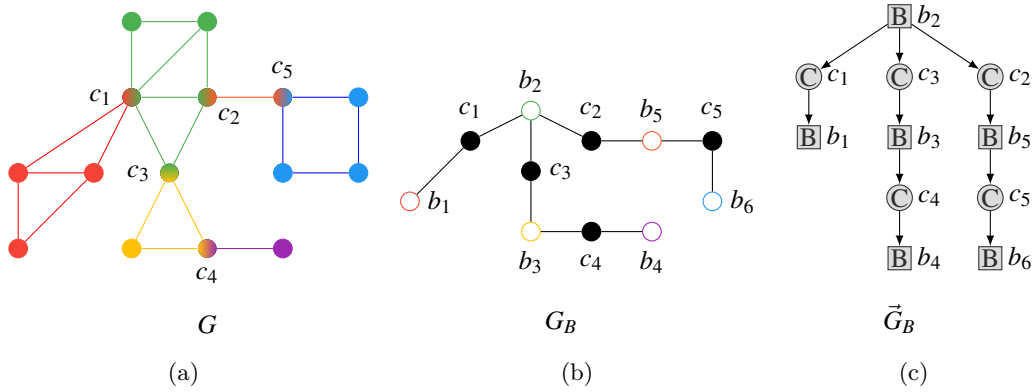


Abbildung 4.2: Ein Graph (a), sein Block-Graph (b) und sein BC-Baum (c).

Jeder der kantendisjunkten Kreise einer maximalen Kreispackung enthält jeweils nur Kanten eines einzigen Blocks:

Lemma 4.1

Sei G ein 1-zusammenhängender Graph und seien $G_1, \ldots, G_k$ die Blöcke von G. Sei weiterhin $\mathcal{Z}^*(G)$ eine maximale kantendisjunkte Kreispackung von G. Dann gilt $C \subseteq G_i$ für einen Kreis $C \in \mathcal{Z}^*(G)$ und einen Block $G_i \in \{G_1, \ldots, G_k\}$.

Beweis: Seien G_i und G_j zwei Blöcke mit gemeinsamem Schnittknoten $\bar{v}$. Angenommen es existiert ein $C \in \mathcal{Z}^*(G)$, sodass $C \cap E(G_i) \neq \varnothing$ und $C \cap E(G_j) \neq \varnothing$ ist. Dann ist C ein geschlossener Kantenzug der Form $C = (e_1, \ldots, e_\ell, e_{\ell+1}, \ldots, e_n)$ mit $\bar{v} \in e_h$ für $h = 1, \ell, \ell+1, n$ und kann in $\mathcal{Z}^*(G)$ durch zwei Kreise $C' = (e_1, \ldots, e_\ell)$ und $C'' = (e_{\ell+1}, \ldots, e_n)$ ersetzt werden. Das ist ein Widerspruch zur Maximalität von $\mathcal{Z}^*(G)$. □

Nach Lemma 4.1 gilt nun die folgende Beziehung.

$$\nu(G) = \sum \nu(G_i) \text{ und } \mathcal{Z}^*(G) = \bigcup \mathcal{Z}^*(G_i)$$

Auch für die Bestimmung aller Blöcke G_i eines 1-zusammenhängenden Graphen G entwickelten Tarjan [Tar71] bzw. Hopcroft und Tarjan [HT73b] bereits in den 70ern einen effizienten, auf Tiefensuche basierenden Algorithmus mit einer Laufzeit von $\mathcal{O}(|V|+|E|)$. Die diesem Algorithmus zugrunde liegende Idee wie 1-Separatoren mittels Tiefensuche bestimmt werden können, beschreibt das folgende Lemma 4.2. Sei $\vec{T} = (V, \vec{E})$ eine Arboreszenz. Dann heißt ein Knoten $v \in V$ genau dann *Nachfolger* eines Knotens $w \in V$, wenn ein w-v-Weg in $\vec{T}$ existiert. Umgekehrt ist w dann *Vorgänger* von v.

Lemma 4.2 [Tur96]
Sei $\vec{T}$ der Tiefensuchebaum eines zusammenhängenden Graphen G. Die Wurzel w von $\vec{T}$ ist genau dann ein 1-Separator in G, falls $|\{u \in V \mid \exists u - w - \text{Weg } W \in \vec{T}\}| \geqslant 2$ gilt. Ein Knoten $v \neq w$ von $\vec{T}$ ist genau dann ein 1-Separator in G, falls gilt

- v hat einen Nachfolger in $\vec{T}$,
- weder v noch ein Nachfolger von $v \in \vec{T}$ sind durch eine Rückwärtskante mit einem Vorgänger von v verbunden.

Um diese Charakterisierungen für die Bestimmung von 1-Separatoren bei einer Tiefensuche zu integrieren, muss für jeden Knoten $v \in G$ während des Verfahrens ein weiterer Wert, die sogenannte *Tiefensuchenummer*, bestimmt werden. Sei $\vec{T}$ der Tiefensuchebaum eines zusammenhängenden Graphen G. Dann wird die Tiefensuchenummer eines Knotens $v \in \vec{T}$ wie folgt bestimmt. Sei $\bar{V}$ die Menge aller Knoten $u \in \vec{T}$ für die ein v-u-Weg W existiert, welcher aus einer Folge von Baumkanten und einer einzigen Rückwärtskante besteht. Dann sei $MinNummer(v) := \min\{Nummer(u) \mid u \in \bar{V}\}$, das heißt die kleinste Nummer eines Knotens u, für den ein solcher Weg W existiert. Falls ein solcher Knoten nicht existiert, sei $MinNummer(v) = Nummer(v)$. Mithilfe dieser Werte kann dann für jeden Knoten überprüft werden, ob es sich um einen 1-Separator handelt.

Lemma 4.3 [Tar71]
Sei $\vec{T}$ der Tiefensuchebaum eines zusammenhängenden Graphen G. Seien weiterhin c, v, u Knoten von G, sodass $(c, v) \in \vec{T}$ gilt und u kein Nachfolger von $v \in \vec{T}$ ist. Falls $MinNummer(v) \geqslant Nummer(c)$, dann ist c ein 1-Separator von G und v und u liegen

in unterschiedlichen Blöcken G_i und G_j mit $i \neq j$ von G.

Algorithmus 4.3 2-Zusammenhangskomponenten eines Graphen (DFS-2-ZSHG) [Gut10]

Eingabe: Graph $G = (V, E)$.
Ausgabe: #Blöcke und Blöcke G_j für $j = 1, \ldots, \#\text{Blöcke}$

```
 1: i ← 0 und j ← 0
 2: for all v ∈ V do
 3:     Nummer(v) ← 0 und MinNummer(v) ← 0
 4:     Vorgänger(v) ← ∅
 5: end for
 6: for all v ∈ V do
 7:     if Nummer(v) = 0 then
 8:         DFS-2-ZSHG(v)
 9:     end if
10: end for
11:
12: procedure DFS-2-ZSHG(v)
13:     i ← i + 1
14:     Nummer(v) ← i und MinNummer(v) ← i
15:     Füge v zu Stapel hinzu
16:     for all w mit {v, w} ∈ E do
17:         if Nummer(w) = 0 then
18:             Vorgänger(w) ← v
19:             DFS-2-ZSHG(w)
20:             MinNummer(v) ← min{MinNummer(v), MinNummer(w)}
21:         else if Nummer(w) < Nummer(v) then
22:             MinNummer(v) ← min{MinNummer(v), Nummer(w)}
23:         end if
24:     end for
25:     if Vorgänger(v) ≠ ∅ und MinNummer(v) ⩾ Nummer(Vorgänger(v)) then
26:         repeat
27:             w ← oberstes Element von Stapel
28:             for all u mit {w, u} ∈ E do
29:                 if Nummer(w) > Nummer(u) then
30:                     Füge Kante {w, u} zum Block G_j hinzu
31:                 end if
32:             end for
33:         until v = w
34:         j = j + 1
35:     end if
36: end procedure
```

Die Überprüfung dieser Eigenschaften erfolgt nun entsprechend [Gut10] in Algorithmus 4.3. Mit der zugrunde liegenden Tiefensuche wird wie zuvor die Tiefesuchennummer $Nummer(v)$ eines jeden Knotens $v \in G$ bestimmt. Der Wert $MinNummer(v)$ wird nach der Initialisierung an drei Stellen im Algorithmus aktualisiert. Bei jedem rekursiven Aufruf der Tiefensuche DFS-2-ZSHG(v) für einen Knoten v erhält $MinNummer(v)$ zunächst den gleichen Wert wie $Nummer(v)$ (siehe Zeile 14). Wird während dieser Tiefensuche eine Rückwärtskante (v, u) gefunden, wird $MinNummer(v)$ in Zeile 22 aktualisiert

und erhält, falls sich $MinNummer(v)$ dadurch verkleinert, den Wert $Nummer(u)$. So ändern sich im Verlauf der Rekursion womöglich auch die Werte $MinNummer(v)$ und $MinNummer(u)$ einer Vorwärtskante (v, u): In Zeile 20 wird $MinNummer(v)$ aktualisiert, falls sich der Wert $MinNummer(u)$ ihres Nachfolgers u verkleinert hat. Die zur Menge G_j für $j = 1, \ldots, \#\text{Blöcke}$ hinzugefügten Kanten induzieren dann alle Blöcke des Graphen G.

Im nächsten Schritt sollen nun 2-zusammenhängende Graphen untersucht werden. Dazu werden in den folgenden Abschnitten die Ergebnisse aus [Ott14] bzw. [OR19] zu maximalen Kreispackungen in 2-zusammenhängenden Graphen zusammengefasst.

4.3 Kreispackungen in 2-zusammenhängenden Graphen

Ein bereits bekannter Approximationsalgorithmus zur Bestimmung einer Kreispackung in ungerichteten Graphen ist der in [CPR03] vorgestellte Greedy-Algorithmus sowie die entsprechende Modifikation in [Kri+07], der auf einer iterativen Suche nach kürzesten Kreisen in einem gegebenen Graphen $G = (V, E)$ basiert (siehe Algorithmus 4.4). In

Algorithmus 4.4 Greedy-Algorithmus zur Berechnung einer Kreispackung

Eingabe: Graph $G = (V, E)$.
Ausgabe: Kreispackung $\mathcal{Z}$ der Größe $\underline{\nu}(G)$.
1: $\mathcal{Z} \leftarrow \varnothing$ und $\underline{\nu}(G) \leftarrow 0$
2: **while** $G \neq \varnothing$ **do**
3: **for all** $v \in V$ mit $\delta(v) \leqslant 1$ **do**
4: lösche $v \in G$
5: **end for**
6: **for all** $v \in V$ mit $\delta(v) = 2$ **do**
7: ersetze $e' = (u, v)$ und $e'' = (v, w)$ durch $e = (u, w)$
8: **end for**
9: suche einen kürzesten Kreis $C \in G$
10: $\mathcal{Z} \leftarrow \mathcal{Z} \cup C$
11: $\underline{\nu}(G) \leftarrow \underline{\nu}(G) + 1$
12: **for all** $e \in C$ **do**
13: lösche $e \in G$
14: **end for**
15: **end while**
16: **return** Kreispackung $\mathcal{Z}$ und untere Schranke $\underline{\nu}(G)$ von $\nu(G)$.

jedem Durchlauf des Algorithmus werden zunächst alle Knoten gelöscht, welche nicht in einem Kreis enthalten sein können (Zeile 4), und der Graph weiter vereinfacht, indem für alle Knoten mit Knotengrad 2 die inzidenten Kanten zu einer Kante verschmolzen

werden (Zeile 6 bis 8). In dem so modifizierten Graphen wird ein kleinster Kreis gesucht, zur Packung hinzugefügt und dann aus G gelöscht (Zeile 9 bis 14). Dieses Vorgehen wird wiederholt, solange der Graph noch Kreise enthält. Die Menge der gelöschten kürzesten Kreise entspricht dann einer Kreispackung im Graphen. Dieser Algorithmus hat eine Approximationsrate von $\mathcal{O}(\log n)$ [CPR03]. Die Laufzeit kann durch $O(|V||E|^2 + |V|^2)$ abgeschätzt werden: Im schlechtesten Fall werden zum Löschen aller Knoten mit Knotengrad 1 und Verschmelzen aller Knoten mit Knotengrad 2 in jeder Iteration alle Knoten durchlaufen und ein kürzester Kreis kann in jeder Iteration in $O(|V||E|)$ Zeit bestimmt werden [IR78].

Der in [Ott14] entwickelte Algorithmus behält die Idee der Suche nach kürzesten Kreisen bei und nutzt zusätzlich die Struktur 2-zusammenhängender Graphen. So wurde im Beginn durch S. MacLane [Mac37] und später durch W. T. Tutte [Tut66] ein Verfahren zur eindeutigen Zerlegung von 2-zusammenhängenden Graphen in sogenannte 3-Zusammenhangskomponenten entwickelt. Diese Zerlegung nennt sich *SPQR-Zerlegung* [DT89] und kann in linearer Zeit bestimmt werden [Gut10; HT73a]. Für einen 2-zusammenhängenden Graphen G und seine SPQR-Zerlegung sucht der Algorithmus Kreise in bestimmten Komponenten der Zerlegung und definiert, wie komponentenübergreifende Kreise gefunden werden können. Letzteres ist ein entscheidender Unterschied im Vergleich zur Nutzung einer Zerlegung von nicht zusammenhängenden und 1-zusammenhängenden Graphen in Bezug auf maximale Kreispackungen. Ein Kreis C einer maximalen kantendisjunkten Kreispackung $\mathcal{Z}^*$ eines nicht zusammenhängenden Graphen bzw. eines 1-zusammenhängenden Graphen ist Teilmenge einer einzigen 1-Komponente bzw. eines einzigen Blocks. Es existiert kein Kreis in einer maximalen Kreispackung, der Kanten mehrerer 1-Komponenten bzw. mehrerer Blöcke enthält. Im Fall des nicht zusammenhängenden Graphen existiert ein solcher Kreis nicht, da es keinen Weg von einer 1-Komponente in eine andere 1-Komponente gibt. Im Fall des 1-zusammenhängenden Graphen wäre eine Kreispackung mit einem Kreis, welcher Kanten aus zwei Blöcken enthält nicht maximal, da dieser Kreis einen Schnittknoten enthält und so in zwei kleinere Kreise aufgeteilt werden könnte. Dank dieser Eigenschaft gilt für G nicht zusammenhängend ebenso wie im Fall G 1-zusammenhängend

$$\nu(G) = \sum \nu(G_i) \text{ und } \mathcal{Z}^*(G) = \bigcup \mathcal{Z}^*(G_i)$$

für die 1-Komponenten bzw. die Blöcke G_i von G. Für einen 2-zusammenhängenden Graphen G wird nun zunächst die zuvor angekündigte SPQR-Zerlegung vorgestellt und damit eine Zerlegung von G in kleinere Graphen G_i. Im Anschluss daran wird der Algo-

rithmus zur Berechnung einer Kreispackung auf Basis der SPQR-Zerlegung mit Erläuterung der Beziehung zwischen

$$\nu(G) \overset{\text{Alg}}{\leftrightsquigarrow} \nu(G_i) \text{ und } \mathcal{Z}^*(G) \overset{\text{Alg}}{\leftrightsquigarrow} \mathcal{Z}^*(G_i)$$

beschrieben.

4.3.1 SPQR-Zerlegung eines 2-zusammenhängenden Graphen

Sei G ein 2-zusammenhängender Graph, $\{G_1, G_2\}$ eine 2-Separation von G und $\{u, v\} = V(G_1) \cap V(G_2)$. Bei Betrachtung einer 2-Separation wie zum Beispiel in Abbildung 4.3(a) und 4.3(b) wird deutlich, dass sich bei einer Zerlegung von 2-zusammenhängenden Graphen nicht, wie womöglich aufgrund der Ergebnisse zuvor erwartet, kleinere 3-zusammenhängende Graphen ergeben. Aus diesem Grund hat Tutte [Tut66] das Konzept sogenannter *virtueller Kanten* entwickelt. Es handelt sich hierbei um eine Kante (u, v), welche jedem Teilgraphen einer 2-Separation $\{G_1, G_2\}$ mit $\{u, v\} = V(G_1) \cap V(G_2)$ hinzugefügt wird, um den jeweils anderen Teilgraphen zu repräsentieren. In Abbildung 4.3(c) sind die jeweiligen virtuellen Kanten gestrichelt dargestellt. Wie in Abbildung 4.3(c) erkenn-

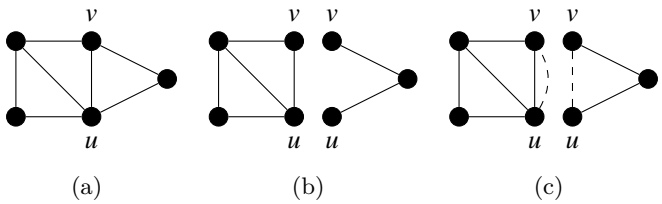


Abbildung 4.3: Ein 2-zusammenhängender Graph (a), eine 2-Separation (b) und das Konzept der virtuellen Kanten (c).

bar, entstehen durch die virtuellen Kanten unter anderem auch Graphen mit parallelen Kanten. Um eine Zerlegung mithilfe dieser virtuellen Kanten zu definieren, werden daher im Folgenden ausschließlich Multigraphen ohne Schleifen betrachtet.

Sei $G = (V, E)$ ein 2-zusammenhängender Multigraph ohne Schleifen. Da G parallele Kanten enthalten kann, wird neben dem Begriff eines 2-Separators nun eine weitere Definition benötigt. Zwei Knoten $\{u, v\} \subset V$ heißen *Splitpaar*, falls $\{u, v\}$ ein 2-Separator von G ist oder eine Kante $(u, v) \in E$ existiert. Sei $\{u, v\}$ ein solches Splitpaar. Dann kann die Kantenmenge E in sogenannte *Splitklassen* $E_1, \ldots, E_k$ partitioniert werden,

wobei sämtliche Kanten $(u, v) \in E$ in einer Splitklasse $E_{i'}$ für $i' = 1, \ldots, k$ enthalten sind und zwei Kanten $e, f \in E$ genau dann in einer Splitklasse E_i für $i = 1, \ldots, k$ mit $i \neq i'$ enthalten sind, wenn ein u-v-Weg $W \subset G$ mit $e, f \in W$ existiert. Jede Splitklasse $E_i, i = 1, \ldots, k$, induziert einen Teilgraphen G_i, wobei G_i entweder aus einer einzelnen Kante (u, v) besteht oder ein maximal zusammenhängender Teilgraph $G' \subseteq G$ ist mit $(u, v) \notin G'$ und $\{u, v\}$ ist kein 2-Separator von G'. Sei G ein 2-zusammenhängender Mul-

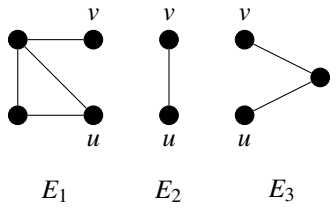


Abbildung 4.4: Splitklassen E_1, E_2, E_3 zum Splitpaar (u, v) für $G = (V, E_1 \cup E_2 \cup E_3)$.

tigraph und $\{u, v\}$ ein Splitpaar mit Splitklassen $E_1, \ldots, E_k$ in G. Sei $e = (u, v)$ eine virtuelle Kante. Seien weiterhin $E' := E_1 \cup \ldots \cup E_j$ und $\bar{E}' := E \backslash E'$, sodass $|E'| \geqslant 2$ und $|\bar{E}'| \geqslant 2$. Die Kantenmengen $E', \bar{E}'$ fassen also die Splitklassen $E_1, \ldots, E_k$ zu zwei Mengen zusammen, sodass jede dieser Mengen mindestens zwei Kanten enthält. Dann induzieren $E' \cup e$ und $\bar{E}' \cup e$ zwei sogenannte *Splitgraphen* $G'_1 = G|_{E' \cup \{e\}}, G'_2 = G|_{\bar{E}' \cup \{e\}}$ und $\{G'_1, G'_2\}$ heißt *Split* von G. Die farbigen Kanten in Abbildung 4.5 zeigen die unterschiedlichen Splitklassen und die entstehenden Splitgraphen werden in Abbildung 4.5(c) dargestellt. Da G'_1 und G'_2 wiederum 2-zusammenhängende Graphen sind, können auch

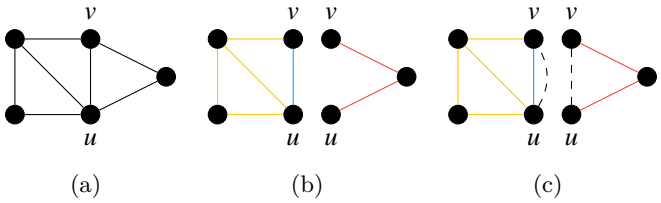


Abbildung 4.5: Ein 2-zusammenhängender Graph (a), die Zusammenfassung der Splitklassen (b) und die durch den Split entstehenden Splitgraphen (c).

diese mithilfe von Splits iterativ weiter zerlegt werden bis keiner der entstehenden Splitgraphen ein Splitpaar mehr enthält. Die auf diese Weise erzeugten Graphen $\{G'_1, \ldots, G'_k\}$ werden *Splitkomponenten* von G genannt. Dabei ist jede Kanten aus E in genau einer Splitkomponente enthalten und jede virtuelle Kante ist in genau zwei Splitkomponenten enthalten.

Bei den zuvor betrachteten nicht zusammenhängenden bzw. 1-zusammenhängenden Graphen war die jeweilige Zerlegung in 1-zusammenhängende Komponenten bzw. 2-zusammenhängende Blöcke eindeutig. Dies ist entsprechend der Definition eines Splits eines 2-zusammenhängenden Multigraphen G offensichtlich nicht der Fall, da nicht genauer spezifiziert wird, welches Splitpaar in welcher Reihenfolge für einen Split verwendet wird und wie die Splitklassen zu den beiden Teilmengen zusammengefasst werden sollen. Um eine eindeutige Zerlegung zu erhalten, wird ein spezieller Split benötigt: Ein Split $\{G'_1, G'_2\}$ mit $G'_1 = G|_{E' \cup \{e\}}, G'_2 = G|_{\bar{E}' \cup \{e\}}$ und $e = (u, v)$ heißt *Tutte-Split*, falls E' oder $\bar{E}'$ nur eine Splitklasse $E_{i*} \in \{E_1, \ldots, E_k\}$ enthält und $G|_{E'}$ oder $G|_{\bar{E}'}$ keinen 1-Separator enthält. Ein Tutte-Split ist in Abbildung 4.6(c) und tatsächlich auch in Abbildung 4.5(c) dargestellt. Die Abbildungen 4.6(a) und 4.6(b) zeigen zwei Splits, welche kein Tutte-Split sind. In Abbildung 4.6(a) enthalten beide Teilgraphen einen 1-Separator und in Abbildung 4.6(b) enthalten beide Teilgraphen mehr als eine Splitklasse. Wird G nun wie oben beschrieben mithilfe von Tutte-Splits sukzessive in seine Split-

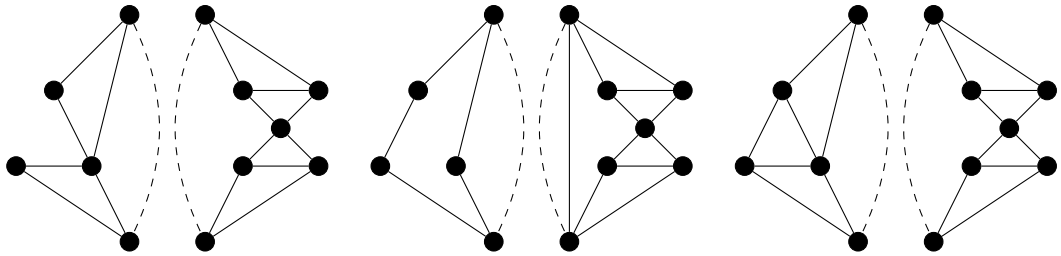


Abbildung 4.6: Kein Tutte-Split (a), kein Tutte-Split (b), Tutte-Split (c).

komponenten zerlegt, werden diese in der Literatur auch (manchmal missverständlich) 3-*Zusammenhangskomponenten* genannt, obwohl es sich wie im Folgenden erkennbar offensichtlich nicht unbedingt um 3-zusammenhängende Graphen handelt:

Satz 4.4 [Tut66]
Sei G ein 2-zusammenhängender Multigraph ohne Schleifen. Dann ist jede 3-Zusammenhangskomponente von G ein Graph des folgenden Typs:

- ein Kreis mit mindestens drei Kanten,
- ein Knotenpaar mit mindestens drei parallelen Kanten oder
- ein einfacher 3-zusammenhängender Graph.

Satz 4.5 [Tut66]
Sei G ein 2-zusammenhängender Multigraph ohne Schleifen. Dann sind die 3-Zusammenhangskomponenten von G eindeutig.

Tutte hat ebenfalls festgestellt, dass sich die Zerlegung von G in 3-Zusammenhangskomponenten entsprechend der BC-Bäume für 1-zusammenhängende Graphen als Baum repränsentieren lässt [Tut66]. Diesen Ansatz haben Di Battista und Tamassia weiter verfolgt und in [DT89] sogenannte *SPQR-Bäume* verwendet. Hier wird eine geringfügige Abwandlung dieser SPQR-Bäume entsprechend [Chi09] verwendet und vorgestellt. Ein SPR-Baum $\mathcal{T}(G) = (M, A)$ eines 2-zusammenhängenden Multigraphen G ohne Schleifen ist der kleinste Baum mit den folgenden Eigenschaften:

1. Jedem Knoten $\mu \in M$ wird ein Multigraph $G_\mu(V_\mu, E_\mu)$ zugeordnet, welcher *Skelett* von μ genannt wird.

2. Jeder Knoten $\mu \in M$ gehört entsprechend dem jeweils zugeordneten Skelett zu einem der folgenden drei Knotentypen:

   - μ ist ein S-Knoten, falls G_μ ein Kreis mit Länge $\geqslant 3$ ist,
   - μ ist ein P-Knoten, falls G_μ ein Bündel paralleler Kanten ist,
   - μ ist ein R-Knoten, falls G_μ ein einfacher 3-zusammenhängender Graph ist.

3. Es gibt genau dann eine Kante $(\mu, \mu') \in A$, wenn G_μ und $G_{\mu'}$ eine gemeinsame virtuelle Kante $e_{(\mu,\mu')} := (u, v)$ besitzen mit $u, v \in V$.

4. Durch Verschmelzen der Knoten μ, μ' für eine Kante $(\mu, \mu') \in A$ zu einem einzigen Knoten und durch Verschmelzen der zugehörigen Skelette zu $G_{(\mu,\mu')} := (G_\mu \backslash e_{(\mu,\mu')}) \cup (G_{\mu'} \backslash e_{(\mu,\mu')})$ kann der Graph G wieder hergestellt werden.

Beispiel 4.6
Ein Beispiel für einen 2-zusammenhängenden Graph ist der in Abbildung 4.7(a) dargestellte Graph G. Durch sukzessive Anwendung von Tutte-Splits entstehen die in Abbildung 4.7(c) gezeigten 3-Zusammenhangskomponenten von G. Entsprechend des Typs der jeweiligen 3-Zusammenhangskomponente und der durch die virtuellen Kanten repräsentierten Splits ergibt sich der SPR-Baum $\mathcal{T}(G)$ in Abbildung 4.7(b).

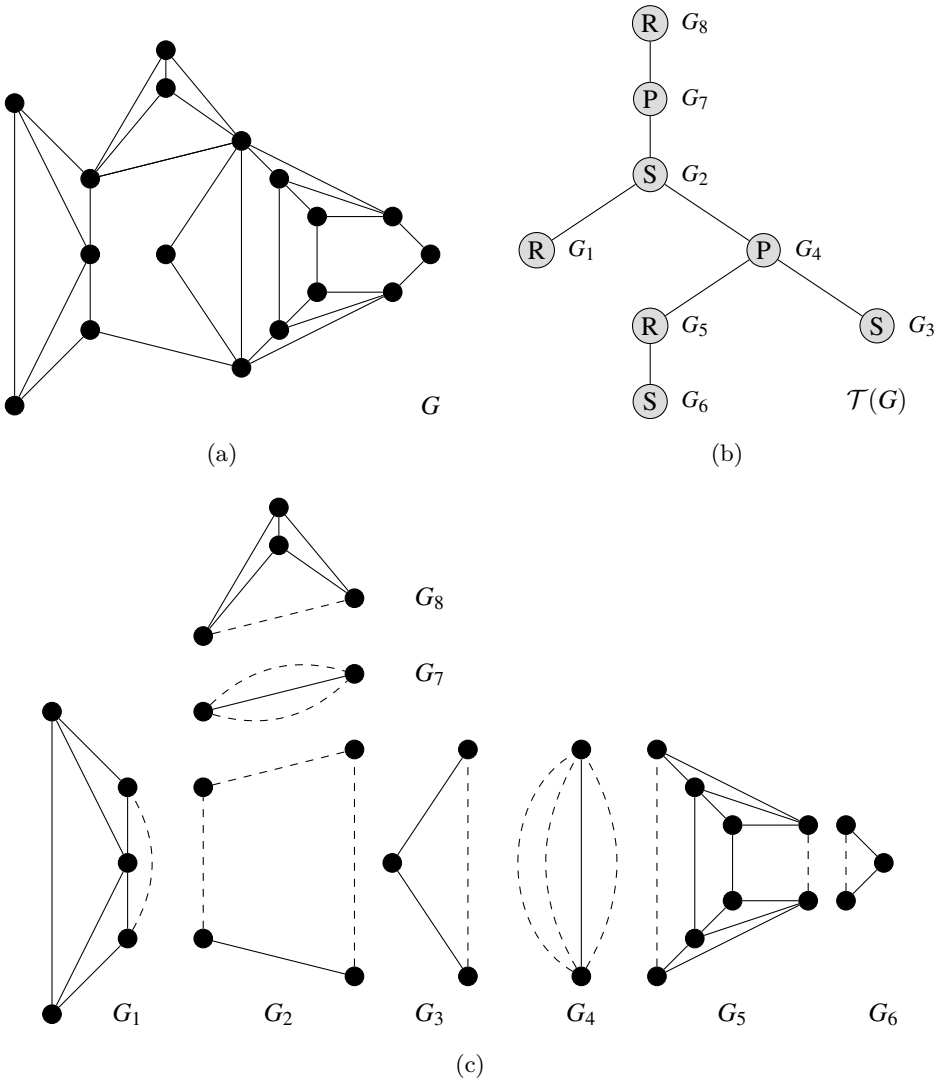


Abbildung 4.7: Ein 2-zusammenhängender Graph G (a), die 3-Zusammenhangskomponenten von G (c) und der zugehörige SPR-Baum $\mathcal{T}(G)$ (b).

Da sowohl die maximal 1-zusammenhängenden Komponenten (siehe Abschnitt 4.1) als auch die maximal 2-zusammenhängenden Komponenten (siehe Abschnitt 4.2) sehr einfach bestimmt werden können, ist nun natürlich das Verfahren zur Bestimmung der 3-Zusammenhangskomponenten und des SPR-Baums von Interesse. Es werden nun einzelne wesentliche Teile dieses Verfahrens entsprechend [Gut10] erläutert. Die Idee des von [HT73a] entwickelten und von [Gut10] verbesserten Algorithmus zur Bestimmung der 3-Zusammenhangskomponenten besteht darin, den Graphen durch irgendwelche Splits zu zerlegen und daraus die Tutte-Splits durch Verschmelzen zu erzeugen. Zwei Splitgraphen

$G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ mit gemeinsamer virtueller Kante werden zu einem Graphen G *verschmolzen*, indem G_1 und G_2 durch $G := (V_1 \cup V_2, (E_1 \cup E_2) \backslash \{e\})$ ersetzt werden.

Satz 4.7 [Gut10]
Sei G ein 2-zusammenhängender Multigraph mit Splitkomponenten $\{G'_1, \ldots, G'_k\}$. Es beschreibe $\{G_1, \ldots, G_N\}$ die Menge von Graphen, welche aus den Splitkomponenten von G durch Verschmelzen von Kantenbündeln mit gemeinsamen virtuellen Kanten und Verschmelzen von Kreisen mit gemeinsamen virtuellen Kanten entstehen, bis kein solches Verschmelzen mehr möglich ist. Dann sind die Graphen $G_1, \ldots, G_N$ die 3-Zusammenhangskomponenten von G.

Auf diese Art und Weise werden Splits, welche kein Tutte-Split waren, rückgängig gemacht, sodass das Verfahren zur Bestimmung von 3-Zusammenhangskomponenten und des zugehörigen SPR-Baums entsprechend [Gut10] in seiner Grobstruktur wie folgt zusammengefasst werden kann.

Sei $G = (V, E)$ ein 2-zusammenhängender Multigraph ohne Schleifen. Dann wird der SPR-Baum $\mathcal{T}(G)$ in den folgenden Schritten bestimmt.

1. Ersetze jedes Bündel von Multikanten in G durch eine virtuelle Kante und erzeuge damit für die Bündel von Multikanten die Splitgraphen $G'_1, \ldots, G'_\ell$, wobei jeder Splitgraph einem solchen Bündel mit virtueller Kante entspricht. Sei G' der entstehende einfache Graph.

2. Bestimme die Splitkomponenten $G'_{\ell+1}, \ldots, G'_{\ell'}$ von G'.

3. Verschmelze die Kantenbündel der Splitgraphen $G'_1, \ldots, G'_{\ell'}$ mit gemeinsamer virtueller Kante und verschmelze die Kreise der Splitgraphen $G'_1, \ldots, G'_{\ell'}$ mit gemeinsamer virtueller Kante bis kein solches Verschmelzen mehr möglich ist. Erhalte als Ergebnis die 3-Zusammenhangskomponenten $G_1, \ldots, G_N$ von G.

4. Konstruiere den SPR-Baum $\mathcal{T}$, wobei die 3-Zusammenhangskomponenten die Skelette der Knoten $\mu \in \mathcal{T}$ sind.

Beispiel 4.8

Ein Beispiel für die Bestimmung eines SPR-Baums $\mathcal{T}(G)$ eines 2-zusammenhängenden Graph G entsprechend dem zuvor beschriebenen Verfahren ist in Abbildung 4.8 dargestellt.

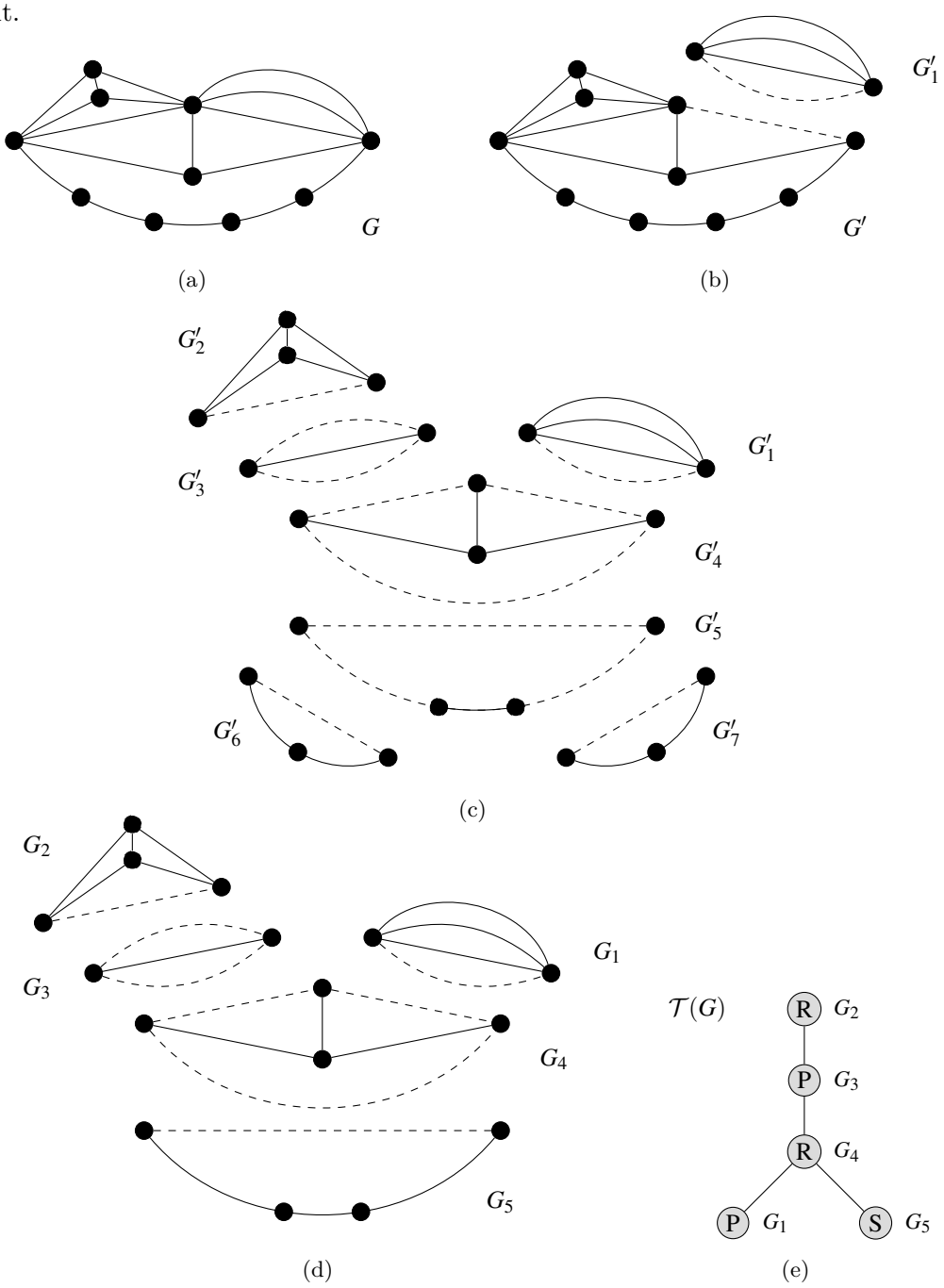


Abbildung 4.8: Die Bestimmung der 3-Zusammenhangskomponenten $G_1, \ldots, G_5$ von G in vier Schritten.

Im Graph G (a) wird entsprechend Schritt 1 zunächst das Bündel von Multikanten durch eine virtuelle Kante ersetzt, sodass die Graphen G' und G'_1 in (b) entstehen. Entsprechend Schritt 2 werden in (c) dann die Splitkomponenten $G'_2, \ldots, G'_7$ bestimmt. In (d) werden die Kreise G'_5, G'_6, G'_7 miteinander verschmolzen. Es entstehen die 3-Zusammenhangskomponenten $G_1, \ldots, G_5$. Zuletzt wird in (e) der zugehörige SPR-Baum $\mathcal{T}(G)$ konstruiert.

4.3.2 Kreispackungsalgorithmus für 2-zusammenhängende Graphen

Der nachfolgend dargestellte Algorithmus 4.5 zur Bestimmung einer Kreispackung für einen 2-zusammenhängenden Graphen G nutzt nun die zuvor definierte SPR-Zerlegung von G und den zugehörigen SPR-Baum $\mathcal{T}$. Dabei bestimmt der SPR-Baum $\mathcal{T}$ die Iteration, das heißt die Bearbeitungsreihenfolge innerhalb des Algorithmus. Iterativ wird dann eine Kreispackung von G bestimmt, deren Kreise aus Pfaden $\mathcal{P}_\mu$ für die Knoten $\mu \in \mathcal{T}$ konstruiert werden. Hierfür werden virtuelle Kanten von nicht virtuellen, *realen* Kanten unterschieden und die Mengen $\mathcal{P}_\mu = \{P(e) \mid e \text{ ist eine reale Kante in } E_\mu\}$ und $P(e) := e$ initialisiert. Die Menge $\mathcal{Z}$ beschreibt die Kreispackung und ist zu Beginn leer, der Wert $\underline{\nu}(G)$ beschreibt die untere Schranke der Kreispackungszahl $\nu(G)$ und hat zu Beginn den Wert null. Es werden nun sukzessive alle Blätter $\mu \in \mathcal{T}$ bearbeitet und im Anschluss an die Bearbeitung gelöscht. Ein Blatt $\mu \in \mathcal{T}$, welches ein S-Knoten, R-Knoten bzw. P-Knoten ist, wird *S-Blatt*, *R-Blatt* bzw. *P-Blatt* genannt. S-Blätter werden im Algorithmus stets zuerst bearbeitet. Daraufhin erst werden sämtliche R-Blätter inspiziert und, sobald weder weitere S- noch R-Blätter ausgewählt werden können, werden P-Blätter bearbeitet. Durch die einzige inzidente Kante $(\mu, \mu') \in \mathcal{T}$ wird für jedes Blatt $\mu \in \mathcal{T}$ ein eindeutiger Vorgänger $\mu' \in \mathcal{T}$ bestimmt. Dieser wird im Algorithmus durch Vorgänger(μ) addressiert. Die Kantenmenge E_μ enthält genau eine virtuelle Kante $\bar{e}_{(\mu,\mu')} = (u, v)$, welche den Teilgraphen von G repräsentiert, der nicht in G_μ enthalten ist. Der Algorithmus bestimmt nun kantendisjunkte Kreise bestehend aus realen Kanten in E_μ, da diese kantendisjunkten Kreisen in G entsprechen. Falls E_μ darüber hinaus einen u-v-Pfad aus realen Kanten enthält, wird die virtuelle Kante $\bar{e}_{(\mu,\mu')}$ in $E_{\mu'}$ durch eine reale Kante (u, v) ersetzt und der Pfad P_{uv} wird in $P((u, v))$ gespeichert, da solche Pfade wie auch die Kreise einem u-v-Pfad P_{uv} in G entsprechend. Existiert kein solcher u-v-Pfad in E_μ wird die virtuelle Kante $\bar{e}_{(\mu,\mu')}$ in $E_{\mu'}$ ersetzt.

Abhängig vom Typ des betrachteten Blattes μ variiert die Bearbeitung der Kantenmenge

$E_{\mu'}$ des Vorgängers μ' in $\mathcal{T}'$:

1_S Das Blatt μ ist ein S-Blatt. Falls die realen Kanten in E_μ einen u-v-Pfad in E_μ induzieren, wird $\bar{e}_{(\mu,\mu')} \in E_{\mu'}$ durch die reale Kante (u, v) ersetzt. Der u-v-Pfad, der durch $\bigcup\{E(P) \mid P \in \mathcal{P}_\mu\}$ induziert wird, wird in $P((u, v))$ gespeichert. Mit $\mathcal{P}_{\mu'}{=}\mathcal{P}_{\mu'} \cup P((u, v))$ wird die Menge der Pfade in μ ergänzt und zuletzt μ aus $\mathcal{T}$ gelöscht.

2_R Das Blatt μ ist ein R-Blatt. Der Algorithmus bestimmt die Kreispackungen $\mathcal{Z}_1$ und $\mathcal{Z}_2$ für die durch E_μ und $E_\mu \backslash \bar{e}_{(\mu,\mu')}$ induzierten Graphen. Es wird $\nu_\mu = |\mathcal{Z}_2|$ gesetzt, die entsprechenden kantendisjunkten Kreise in G werden zu $\mathcal{Z}$ hinzugefügt und die dazugehörigen Pfade aus $\mathcal{P}_\mu$ gelöscht. Falls $|\mathcal{Z}_1| = |\mathcal{Z}_2|$ gilt, wird $\bar{e}_{(\mu,\mu')} \in E_{\mu'}$ gelöscht. Falls $|\mathcal{Z}_1| > |\mathcal{Z}_2|$ gilt, existiert ein u-v-Pfad in E_μ, der nicht in einem der Kreise aus $\mathcal{Z}_2$ enthalten ist. Daher wird $\bar{e}_{(\mu,\mu')} \in E_{\mu'}$ durch die reale Kante (u, v) ersetzt. Der u-v-Pfad, der durch $\bigcup\{E(P) \mid P \in \mathcal{P}_\mu\}$ induziert wird, wird in $P((u, v))$ gespeichert. Mit $\mathcal{P}_{\mu'}{=}\mathcal{P}_{\mu'} \cup P((u, v))$ wird die Menge der Pfade in μ ergänzt und zuletzt μ aus $\mathcal{T}$ gelöscht.

3_P Das Blatt μ ist ein P-Blatt.

(i) Falls $|E_\mu|$ eine gerade Zahl ist, gibt es eine Kreispackung $\mathcal{Z}_P$ mit $\nu_\mu = \frac{|E_\mu|}{2} - 1$ Kreisen der Länge 2. Die entsprechenden kantendisjunkten Kreise in G werden zu $\mathcal{Z}$ hinzugefügt und die dazugehörigen Pfade aus $\mathcal{P}_\mu$ gelöscht. Dann gibt es eine Kante e in E_μ, die nicht in einem der Kreise von $\mathcal{Z}_P$ enthalten ist. Die Kante $\bar{e}_{(\mu,\mu')} \in E_{\mu'}$ wird durch die reale Kante (u, v) ersetzt und der u-v-Pfad, der durch e induziert wird, in $P((u, v))$ gespeichert. Mit $\mathcal{P}_{\mu'}{=}\mathcal{P}_{\mu'} \cup P((u, v))$ wird die Menge der Pfade in μ ergänzt und zuletzt μ aus $\mathcal{T}$ gelöscht.

(ii) Falls $|E_\mu|$ eine ungerade Zahl ist, gibt es eine Kreispackung $\mathcal{Z}_P$ mit $\nu_\mu = \frac{|E_\mu|-1}{2}$ Kreisen der Länge 2. Die entsprechenden kantendisjunkten Kreise in G werden zu $\mathcal{Z}$ hinzugefügt, $\bar{e}_{(\mu,\mu')} \in E_{\mu'}$ sowie μ aus $\mathcal{T}$ gelöscht.

Bei der so erfolgten Abarbeitung des SPR-Baums $\mathcal{T}$ wird der letzte Knoten des SPR-Baums $\mathcal{T}$ gesondert behandelt: Falls μ der letzte Knoten in $\mathcal{T}$ ist, wird eine Kreispackung in G_μ bestimmt und die kantendisjunkten Kreise in G werden zu $\mathcal{Z}$ hinzugefügt.

Algorithmus 4.5 Kreispackungsalgorithmus für 2-zusammenhängende Graphen

Eingabe: SPR-Zerlegung eines 2-zusammenhängenden Graphen G und SPR-Baum $\mathcal{T}_G$.
Ausgabe: Kreispackung $\mathcal{Z}$ und untere Schranke $\underline{\nu}(G)$ für die Kreispackungszahl $\nu(G)$.

$\mathcal{Z} \leftarrow \varnothing$, $\underline{\nu}(G) \leftarrow 0$ und $\mathcal{P}_\mu \leftarrow \varnothing \quad \forall \mu \in M$
while $\exists$ SPR-Knoten μ in $\mathcal{T}$ **do**
  for all S-Blätter μ **do**
    $\mu' :=$ Vorgänger(μ)
    if $\delta(v) = 2\ \forall v \in V_\mu$ **then**
      ersetze $\bar{e}_{(\mu,\mu')} \in E_{\mu'}$ durch eine reale Kante
      $\mathcal{P}_{\mu'} \leftarrow \mathcal{P}_{\mu'} \cup P(\bar{e}_{(\mu,\mu')})$
    end if
    $\mathcal{T} \leftarrow \mathcal{T} \backslash \mu$
  end for
  for all R-Blätter μ **do**
    $\mu' :=$ Vorgänger(μ)
    $\mathcal{Z}_1 \leftarrow$ Algorithmus 4.4(G_μ)
    $\mathcal{Z}_2 \leftarrow$ Algorithmus 4.4$(G_\mu \backslash \bar{e}_{(\mu,\mu')})$
    if $|\mathcal{Z}_1| == |\mathcal{Z}_2|$ **then**
      lösche $\bar{e}_{(\mu,\mu')} \in E_{\mu'}$
    else if $|\mathcal{Z}_1| > |\mathcal{Z}_2|$ **then**
      ersetze $\bar{e}_{(\mu,\mu')} \in E_{\mu'}$ durch eine reale Kante
      $\mathcal{P}_{\mu'} \leftarrow \mathcal{P}_{\mu'} \cup P(\bar{e}_{(\mu,\mu')})$
    end if
    $\nu_\mu \leftarrow |\mathcal{Z}_2|$ und $\mathcal{T} \leftarrow \mathcal{T} \backslash \mu$
    $\underline{\nu}(G) \leftarrow \underline{\nu}(G) + \nu_\mu$ und $\mathcal{Z} \leftarrow \mathcal{Z} \cup \mathcal{Z}_2$
  end for
  for all P-Blätter μ **do**
    $\mu' :=$ Vorgänger(μ)
    if $|E_\mu|$ ist nicht gerade **then**
      $\nu_\mu \leftarrow \frac{|E_\mu|-1}{2}$
      lösche $\bar{e}_{(\mu,\mu')}$ in $E_{\mu'}$
    else if $|E_\mu|$ ist gerade **then**
      $\nu_\mu \leftarrow \frac{|E_\mu|}{2} - 1$
      ersetze $\bar{e}_{(\mu,\mu')}$ in $E_{\mu'}$ durch eine reale Kante
      $\mathcal{P}_{\mu'} \leftarrow \mathcal{P}_{\mu'} \cup P(\bar{e}_{(\mu,\mu')})$
    end if
    $\underline{\nu}(G) \leftarrow \underline{\nu}(G) + \nu_\mu$ und $\mathcal{T} \leftarrow \mathcal{T} \backslash \mu$
    $\mathcal{Z} \leftarrow \mathcal{Z} \cup \{\{P^{(2i-1)}, P^{(2i)}\} \mid P^{(i)} \in \mathcal{P}_\mu \forall i = 1, \ldots, \nu_\mu\}$
  end for
  if μ ist letzter Knoten **then**
    if μ ist S-Blatt und $\delta(v) = 2\ \forall v \in V_\mu$ **then**
      $\nu_\mu \leftarrow 1$ und $\mathcal{Z} \leftarrow \mathcal{Z} \cup P|_{E(\bigcup_{e \in E_\mu} P(e))}$
    else if μ ist R-Blatt **then**
      $\mathcal{Z}_1 \leftarrow$ Algorithmus 4.4$(G_\mu), \nu_\mu \leftarrow |\mathcal{Z}_1|$ und $\mathcal{Z} \leftarrow \mathcal{Z} \cup \mathcal{Z}_1$
    else if μ ist P-Blatt **then**
      $\nu_\mu \leftarrow \lfloor \frac{|E_\mu|}{2} \rfloor$ und $\mathcal{Z} \leftarrow \mathcal{Z} \cup \{\{P^{(2i-1)}, P^{(2i)}\} \mid P^{(i)} \in \mathcal{P}_\mu \forall i = 1, \ldots, \nu_\mu\}$
    end if
    $\underline{\nu}(G) \leftarrow \underline{\nu}(G) + \nu_\mu$ und $\mathcal{T} \leftarrow \mathcal{T} \backslash \mu$
  end if
end while
return $\underline{\nu}(G)$ und $\mathcal{Z}$

Satz 4.9
Algorithmus 4.5 bestimmt eine Kreispackung $\mathcal{Z}$ von G mit Kardinalität

$$|\mathcal{Z}| = \sum_{\mu \in \mathcal{T}} \nu_\mu.$$

Beweis: Sei $\mathcal{T}$ der SPR-Baum von G.

Beim Bearbeiten eines S-Knotens gilt stets $\nu_\mu = 0$, da die realen Kanten in E_μ ohne die virtuellen Kanten keinen Kreis induzieren. Falls die realen Kanten aber einen u-v-Weg in E_μ induzieren, kann dieser u-v-Weg P_{uv} in G zu einem zusätzlichen Kreis C in $\mathcal{Z}$ führen. Aus diesem Grund wird die virtuelle Kante $\bar{e}_{(\mu,\mu')}$ in $E_{\mu'}$ durch eine reale Kante (u, v) ersetzt, der u-v-Weg P_{uv} in $P((u, v))$ gespeichert und μ aus $\mathcal{T}$ gelöscht. Der mögliche zusätzliche Kreis C wird bestimmt, sobald der Knoten μ' bearbeitet wird.

Beim Bearbeiten eines R-Knotens μ werden zwei Kreispackungen $\mathcal{Z}_1$ und $\mathcal{Z}_2$ für E_μ und $E_\mu \backslash \bar{e}_{(\mu,\mu')}$ bestimmt. Dabei induziert E_μ mindestens eine Kreispackung der Kardinalität $\nu_\mu = |\mathcal{Z}_2|$ in G. Falls $|\mathcal{Z}_1| > |\mathcal{Z}_2|$ führt $P((u, v))$ möglicherweise zu einem zusätzlichen Kreis C in $\mathcal{Z}$. Aus diesem Grund wird die virtuelle Kante $\bar{e}_{(\mu,\mu')}$ in $E_{\mu'}$ durch eine reale Kante (u, v) ersetzt und der mögliche zusätzliche Kreis C wird bestimmt, sobald der Knoten μ' bearbeitet wird.

Beim Bearbeiten eines P-Knotens μ bildet ein Paar realer Kanten in E_μ einen kantendisjunkten Kreis in G. Falls $|E_\mu|$ gerade ist, gibt es $\nu_\mu = \frac{|E_\mu|}{2} - 1$ solcher Paare. Der Weg P_{uv}, welcher durch die übrig bleibende reale Kante induziert wird, führt möglicherweise zu einem zusätzlichen Kreis C in $\mathcal{Z}$. Aus diesem Grund wird die virtuelle Kante $\bar{e}_{(\mu,\mu')}$ in $E_{\mu'}$ durch eine reale Kante (u, v) ersetzt und der mögliche zusätzliche Kreis C wird bestimmt, sobald der Knoten μ' bearbeitet wird. Falls $|E_\mu|$ ungerade ist, gibt es $\nu_\mu = \frac{|E_\mu|-1}{2}$ Paare realer Kanten, welche eine ebensolche Anzahl zusätzlicher Kreise in $\mathcal{Z}$ induzieren. □

Da bei Bearbeitung der R-Blätter Algorithmus 4.4 zur Bestimmung der Kreispackungen $\mathcal{Z}_1$ und $\mathcal{Z}_2$ verwendet wird, hat Algorithmus 4.5 die entsprechende Approximationsgüte $\mathcal{O}(\log n)$. Auch die Laufzeit von Algorithmus 4.5 wird von Algorithmus 4.4 dominiert und ist daher $O(|V||E|^2 + |V|^2)$.

4.4 Optimalität des Verfahrens für serienparallele Graphen

Wie zuvor festgestellt, liefert die SPR-Zerlegung drei Typen von Komponenten (Typ S, Typ P, Typ R). Bei den Komponenten vom Typ R handelt es sich um 3-zusammenhängende Graphen und für diese nutzt Algorithmus 4.5 den in Abschnitt 4.3 vorgestellten Algorithmus 4.4. Aus diesem Grund liefert Algorithmus 4.5 eine untere Schranke $\underline{\nu}(G)$ an die Kreispackungszahl $\nu(G)$. Für spezielle 2-zusammenhängende Graphen ist diese Schranke optimal.

Sei G ein Multigraph ohne Schleifen. G heißt *generalisierter serienparalleler Graph* [Kor94], wenn er durch die sequentielle Anwendung der folgenden drei Operationen auf den K_2 reduziert werden kann:

1. Ersetze alle parallelen Kanten $e_i = (u, v), i = 1, \ldots, k$, durch eine einzelne Kante $e = (u, v)$.

2. Verschmelze für jeden Knoten $v \in G$ mit $\delta(v) = 2$ den Kantenzug $(u, v), (v, w)$ in G zur Kante (u, w).

3. Lösche alle Knoten $v \in G$ mit $\delta(v) = 1$.

Enthält G keinen Knoten mit Knotengrad 1 und kann ausschließlich durch die Schritte 1 und 2 bereits auf den K_2 reduziert werden, heißt G *serienparalleler Graph.*

Lemma 4.10 [Kor94]
Ein außenplanarer Graph ist generalisiert serienparallel.

Satz 4.11
Sei G ein 2-zusammenhängender, generalisierter serienparalleler Multigraph ohne Schleifen. Dann ist

$$\nu(G) = \sum_{\mu \in \mathcal{T}} \nu_\mu,$$

das heißt Algorithmus 4.5 bestimmt eine maximale Kreispackung $\mathcal{Z}^*$ von G.

Beweis: Der Beweis erfolgt durch Induktion über die Anzahl N von Knoten im SPR-Baum $\mathcal{T}(G)$ von G.

Sei $N = 1$. Dann ist $\mathcal{T}$ entweder ein P-Knoten oder ein S-Knoten und G ist entsprechend entweder ein Bündel von r parallelen Kanten ($r \geqslant 3$) oder ein Kreis der Länge $\geqslant 3$. Im ersten Fall ist $\nu(G) = \lfloor \frac{r}{2} \rfloor$, im zweiten Fall ist $\nu(G) = 1$. Dies entspricht der Ausgabe von Algorithmus 4.5 (Schritt F).

Sei $N \geqslant 2$. Angenommen Algorithmus 4.5 bestimmt $\nu(G')$ für alle generalisiert serienparallelen Multigraphen G', für die $\mathcal{T}(G')$ höchstens $N-1$ Knoten hat. Sei nun G ein generalisiert serienparalleler Multigraph, sodass $\mathcal{T}(G)$ N Knoten besitzt. Nun wird Algorithmus 4.5 angewendet. Beim Betrachten des ersten Knotens $\mu \in \mathcal{T}(G)$ können die folgenden Fälle auftreten.

(a) μ ist ein S-Knoten. Dann bearbeitet Algorithmus 4.5 den Knoten μ entsprechend (1_S). Der Multigraph $G' = G \backslash (E_\mu \backslash \bar{e}_{(\mu,\mu')}) \cup (u, v)$ ist ebenfalls generalisiert serienparallel, da er durch Anwendung von Operation 2 auf G erzeugt werden kann, und $\mathcal{T}(G') = \mathcal{T}(G) \backslash \mu$, das heißt $\mathcal{T}(G')$ hat $N-1$ Knoten. Da die realen Kanten in E_μ einen u-v-Pfad in E_μ induzieren, gilt $\nu(G') = \nu(G)$. Nach Induktionsannahme gilt $\nu(G') = \sum_{\tilde{\mu} \in \mathcal{T}(G')} \nu_{\tilde{\mu}}$ und damit ist $\sum_{\tilde{\mu} \in \mathcal{T}(G)} \nu_{\tilde{\mu}} = \sum_{\tilde{\mu} \in \mathcal{T}(G')} \nu_{\tilde{\mu}} + \nu_\mu = \nu(G') + 0 = \nu(G)$.

(b) μ ist ein P-Knoten in $\mathcal{T}(G)$, das bedeutet alle Blätter sind P-Knoten.

(b1) Es existiert mindestens ein Blatt μ mit einer ungeraden Anzahl realer Kanten, das heißt $|E_\mu|$ ist gerade. Der Vorgänger μ' ist ein S-Knoten. Algorithmus 4.5 bearbeitet den Knoten μ entsprechend $(3_P(i))$. Der Multigraph $G' = G \backslash (E_\mu \backslash \bar{e}_{(\mu,\mu')}) \cup (u, v)$ ist generalisiert serienparallel, da er durch Anwendung von Operation 1 auf G erzeugt werden kann, und $\mathcal{T}(G') = \mathcal{T}(G) \backslash \mu$, das heißt $\mathcal{T}(G')$ hat $N-1$ Knoten. Außerdem gilt $\nu(G') = \nu(G) - (\frac{|E_\mu|}{2} - 1)$, da E_μ eine ungerade Anzahl realer Kanten besitzt und jedes Paar realer Kanten einem Kreis in G entspricht. Nach Induktionsannahme gilt $\nu(G') = \sum_{\tilde{\mu} \in \mathcal{T}(G')} \nu_{\tilde{\mu}}$ und damit ist $\sum_{\tilde{\mu} \in \mathcal{T}(G)} \nu_{\tilde{\mu}} = \sum_{\tilde{\mu} \in \mathcal{T}(G')} \nu_{\tilde{\mu}} + \nu_\mu = \nu(G') + (\frac{|E_\mu|}{2} - 1) = \nu(G)$.

(b2) Alle P-Knoten haben eine gerade Anzahl realer Kanten. Dann bearbeitet Algorithmus 4.5 ein Blatt μ entsprechend $(3_P(ii))$. Sei $\mu' = Vorgänger(\mu)$. Der

Knoten μ' ist ein S-Knoten. Angenommen μ' ist adjazent zu $k \geqslant 1$ P-Blättern $\mu_1, \ldots, \mu_k$ (sei $\mu_1 = \mu$). Sei $\hat{E}$ die Menge realer Kanten in $\bigcup_{i\in\{1,\ldots,k\}} E_{\mu_i} \cup E_{\mu'}$. Für den durch $\hat{E}$ induzierten Teilgraphen $\hat{G}$ gilt dann $\nu(\hat{G}) = \sum_{i\in\{1,\ldots,k\}} \nu_{\mu_i}$. Falls $E\backslash\hat{E} = \varnothing$ ist, gilt $\mathcal{T}(G) = \mathcal{T}(\hat{G})$ und $\nu(G) = \sum_{i\in\{1,\ldots,k\}} \nu_{\mu_i} = \sum_{\tilde{\mu}\in\mathcal{T}(G)} \nu_{\tilde{\mu}}$. Falls $E\backslash\hat{E} \neq \varnothing$ ist, gilt für den durch $E\backslash\hat{E}$ induzierten Graphen G', dass $\nu(G') = \nu(G) - \sum_{i\in\{1,\ldots,k\}} \nu_{\mu_i}$ ist. Nun bleibt noch zu zeigen, dass G' generalisiert serienparallel ist. In $\mathcal{T}(G)\backslash(\bigcup_{i\in\{1,\ldots,k\}} \mu_i \cup \mu')$ muss der Knoten μ' einen Vorgänger $\mu'' = Vorgänger(\mu)$ haben. Dieser Vorgänger μ'' ist ein P-Knoten und enthält mindestens zwei parallele Kanten mit Endknoten u'', v''. Eine dieser Kanten repräsentiert den Teilgraph $\hat{G}$. Da G generalisiert serienparallel ist, ist auch $G'' = G' \cup (u'', v'')$ generalisiert serienparallel, da G'' durch Anwendung von Operation 1 und 2 auf G erzeugt werden kann. Da es mindestens eine weitere virtuelle, zu (u'', v'') parallele Kante in $E_{\mu''}$ gibt, muss es einen Teilgraph $\tilde{G} \subset G$ geben, sodass $\tilde{G}$ zu einer parallelen Kante von (u'', v'') reduziert werden kann und $E(\tilde{G}) \cap E(\hat{G}) = \varnothing$ gilt. Entsprechend der Definition eines generalisiert serienparallelen Graphen ist $G''\backslash(u'', v'') = G'$ also generalisiert serienparallel. Offensichtlich gilt $\mathcal{T}(G') = \mathcal{T}(G)\backslash(\bigcup_{i\in\{1,\ldots,k\}} \mu_i \cup \mu')$. $\mathcal{T}(G')$ hat $N-(k+1)$ Knoten. Nach der Induktionsannahme gilt $\nu(G') = \sum_{\tilde{\mu}\in\mathcal{T}(G')} \nu_{\tilde{\mu}}$und $\nu(G) = \sum_{\tilde{\mu}\in\mathcal{T}(G')} \nu_{\tilde{\mu}} + \sum_{i\in\{1,\ldots,k\}} \nu_{\mu_i} = \sum_{\tilde{\mu}\in\mathcal{T}(G)} \nu_{\tilde{\mu}}$. □

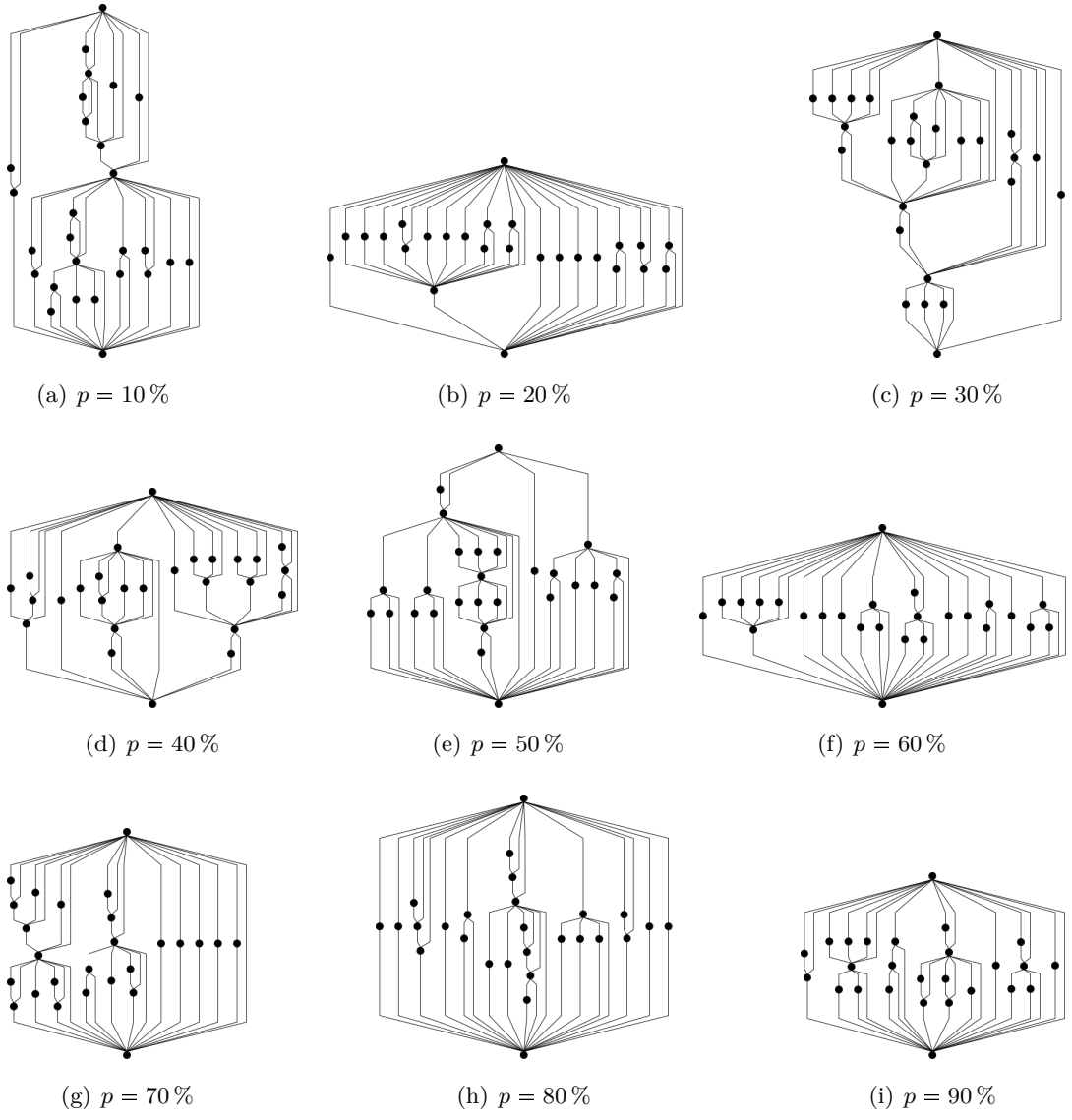
Der SPQR-Baum eines 2-zusammenhängenden Multigraphen kann in linearer Zeit bestimmt werden [GM01]. Gleiches gilt dann für den SPR-Baum [Chi09], sodass sich an obigen Satz 4.11 direkt die folgende Aussage anschließt.

Korollar 4.12
Falls G ein 2-zusammenhängender, generalisiert serienparalleler Graph ohne Schleifen ist, kann eine maximale Kreispackung von G in linearer Zeit bestimmt werden.

4.5 Implementierung und Auswertung

In diesem Abschnitt wird die Implementierung und Evaluation von praktischen Experimenten mit Algorithmus 4.5 beschrieben. Es soll getestet werden, wie sich die tatsächlichen Laufzeiten für Graphen unterschiedlicher Ordnung verhalten und ob der Algorithmus in der Praxis anwendbar ist. Der Algorithmus 4.5 wurde mithilfe des *Open Graph al-*

gorithms and Data structures Framework (OGDF) [Chi+13] implementiert (Stand 2015). Der Quellcode des Frameworks ist unter `https://ogdf.uos.de/` verfügbar. Es handelt sich hierbei um eine C++-Bibliothek, welche verschiedene Graphalgorithmen und Datenstrukturen enthält. Insbesondere stellt diese Bibliothek eine Linearzeitimplementierung der SPQR-Zerlegung bereit, die für Algorithmus 4.5 benötigt wird. Darüber hinaus umfasst die Bibliothek Algorithmen zum Bestimmen des Zusammenhangsgrads eines Graphen und Verfahren zur Graphgenerierung.



(a) $p = 10\,\%$ (b) $p = 20\,\%$ (c) $p = 30\,\%$

(d) $p = 40\,\%$ (e) $p = 50\,\%$ (f) $p = 60\,\%$

(g) $p = 70\,\%$ (h) $p = 80\,\%$ (i) $p = 90\,\%$

Abbildung 4.9: Generierte serienparallele Graphen mit $m = 50$ Kanten und einer Wahrscheinlichkeit von p für eine serielle Komposition.

Zur Auswertung des Algorithmus 4.5 wurden zunächst serienparallele Graphen generiert. Der Graphgenerator erzeugt einen serienparallen Graphen durch eine zufällige Folge serieller und paralleler Kompositionen. Dabei fließen in die Auswertungen Graphen der Größen $m = 500, 1.000, \ldots, 142.500$ Kanten ein. Für einzelne Auswertungen wurden auch Graphen der Größe $m = 1.000.000$ Kanten herangezogen. Weiterhin kann die Wahrscheinlichkeit für eine serielle Komposition gesteuert werden. Je Kantenzahl wurden daher wiederum 9 Graphen mit einer Wahrscheinlichkeit von $p = 10\,\%, 20\,\%, \ldots, 90\,\%$ für eine serielle Komposition generiert. Insgesamt beinhaltet das Testset daher 2.556 verschiedene Graphen. In Abbildung 4.9 sind exemplarisch alle generierten serienparallelen Graphen mit 50 Kanten dargestellt. Für die Zeichnung der mit OGDF erzeugten GraphML-Dateien wurde die Desktop-Version des Graph Editors *yED* genutzt.

Die Experimente wurden auf einem Computer ausgeführt, der mit einer Intel Core i7-4810MQ CPU (Taktfrequenz: 2,8 GHz-3,8 GHz, 4 CPU-Kerne/8 Hyperthreads und L1/L2-Cache: 64 KB/256 KB), 16 GB RAM ausgestattet ist. Bei dem verwendeten Betriebssystem handelt es sich um Windows 10. Der gesamte Code wurde mit dem Microsoft C/C++-Optimierungscompiler Version 19.28.29334 für x86 kompiliert.

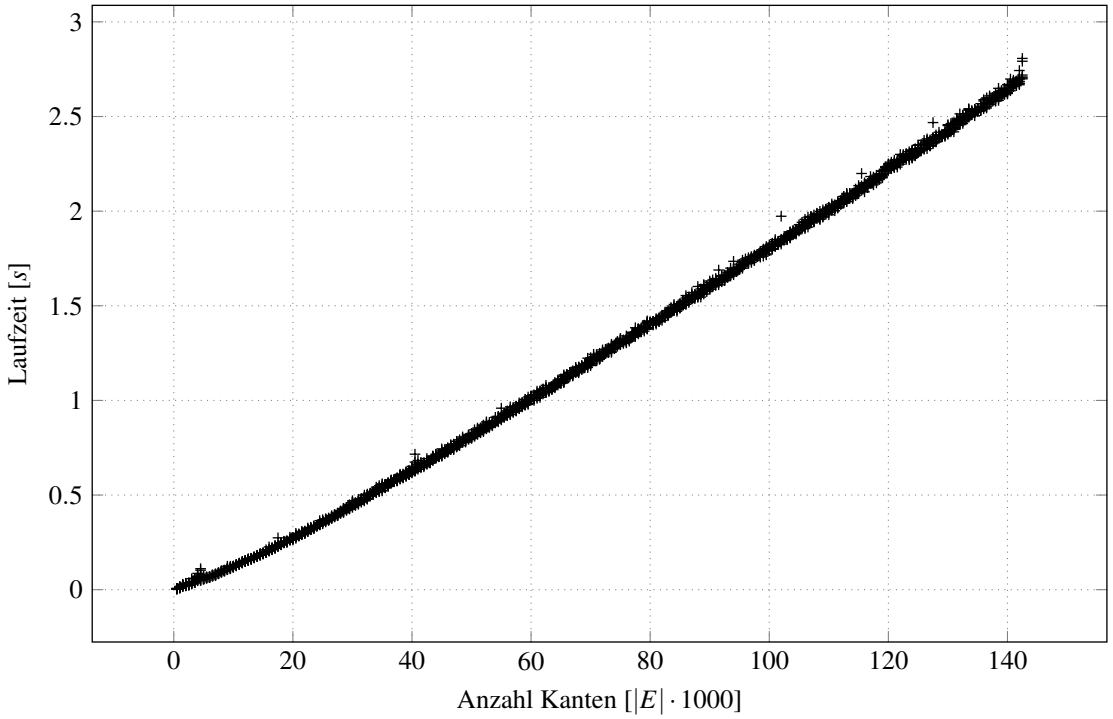
Zur Evaluation und Laufzeitanalyse der Implementierung wurden für jeden Graph verschiedene Zeiten gemessen und protokolliert: die Dauer der Graphgenerierung, die Dauer der Erzeugung bzw. des Schreibens einer GraphML-Datei für den Graphen und die Dauer der Berechnung einer Kreispackung. In Tabelle 4.1 sind die jeweiligen Durchschnittswerte aller Graphen mit Kantenzahl m (unabhängig von der Kompositionswahrscheinlichkeit p) dargestellt. Bereits die durchschnittlichen Berechnungsdauern zeigen, dass die Kreispackungszahl für serienparallele Graphen mit Algorithmus 4.5 in wenigen Sekunden bestimmt werden kann.

Die grafische Auswertung der Messdaten zeigt, dass die Laufzeit den theoretischen Resultaten entspricht. Abbildung 4.10(a) zeigt eine annähernd lineare Laufzeit (in Sekunden) für die Bestimmung der Kreispackungszahl mit Algorithmus 4.5 bei steigender Anzahl von Kanten. Die Berechnungsdauer hängt offensichtlich auch nicht davon ab, ob der generierte Graph mit einer höheren Wahrscheinlichkeit durch serielle Kompositionen generiert wurde (siehe Abbildung 4.10(b)). Die Abweichungen, wie in dem vergrößerten Ausschnitt von Abbildung 4.10(b) zu sehen, sind im Rahmen der Messgenauigkeit marginal. Interessant ist, dass die Anzahl kantendisjunkter Kreise bei den untersuchten serienparallelen Graphen gegen einen Wert von 247/248 Kreisen je tausend Kanten in G zu konvergieren scheint (vergleiche Abbildung 4.11(a)). Darüber hinaus sinkt die

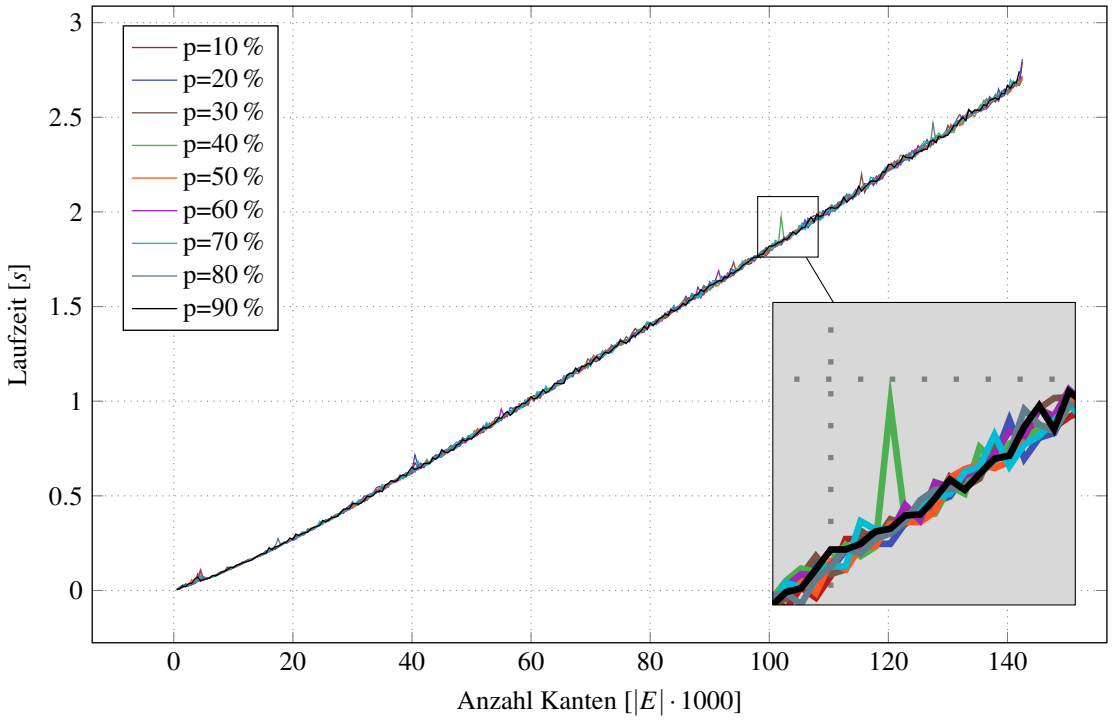
durchschnittliche Anzahl der von Algorithmus 4.5 in einer Sekunde berechneten Kreise von circa 15.000 Kreisen bei 60.000 Kanten auf circa 13.000 Kreise bei 140.000 Kanten (vergleiche Abbildung 4.11(b)).

$\|E\|$	∅ Generieren [s]	∅ Schreiben [s]	∅ Berechnen [s]	∅ $\nu(G)$
5000	0.1	0.0	0.1	1236.2
10000	0.5	0.0	0.1	2473.6
15000	1.4	0.1	0.2	3708.8
20000	2.8	0.1	0.3	4947.6
25000	4.8	0.1	0.4	6189.3
30000	7.2	0.1	0.5	7408.3
35000	10.1	0.2	0.5	8658.1
40000	13.6	0.2	0.6	9891.0
45000	17.7	0.2	0.7	11121.0
50000	22.5	0.2	0.8	12364.2
55000	28.1	0.3	0.9	13610.6
60000	34.5	0.3	1.0	14849.2
65000	42.1	0.3	1.1	16087.0
70000	50.6	0.3	1.2	17310.7
75000	60.5	0.4	1.3	18554.1
80000	71.0	0.4	1.4	19790.1
85000	83.4	0.4	1.5	21027.9
90000	97.1	0.4	1.6	22261.0
95000	113.7	0.5	1.7	23492.7
100000	130.5	0.5	1.8	24759.0
105000	148.8	0.5	1.9	25978.9
110000	171.6	0.5	2.0	27201.7
115000	196.1	0.6	2.1	28464.4
120000	226.5	0.6	2.2	29700.1
125000	255.2	0.6	2.3	30922.4
130000	289.8	0.6	2.4	32159.6
135000	326.7	0.7	2.5	33398.8
140000	368.7	0.7	2.6	34628.3
1000000	25934.0	5.0	17.8	247466.1

Tabelle 4.1: Durchschnittliche Laufzeiten und durchschnittliche Kreispackungszahl über alle Wahrscheinlichkeiten $p \in \{10\,\%, 20\,\%, \ldots, 90\,\%\}$.

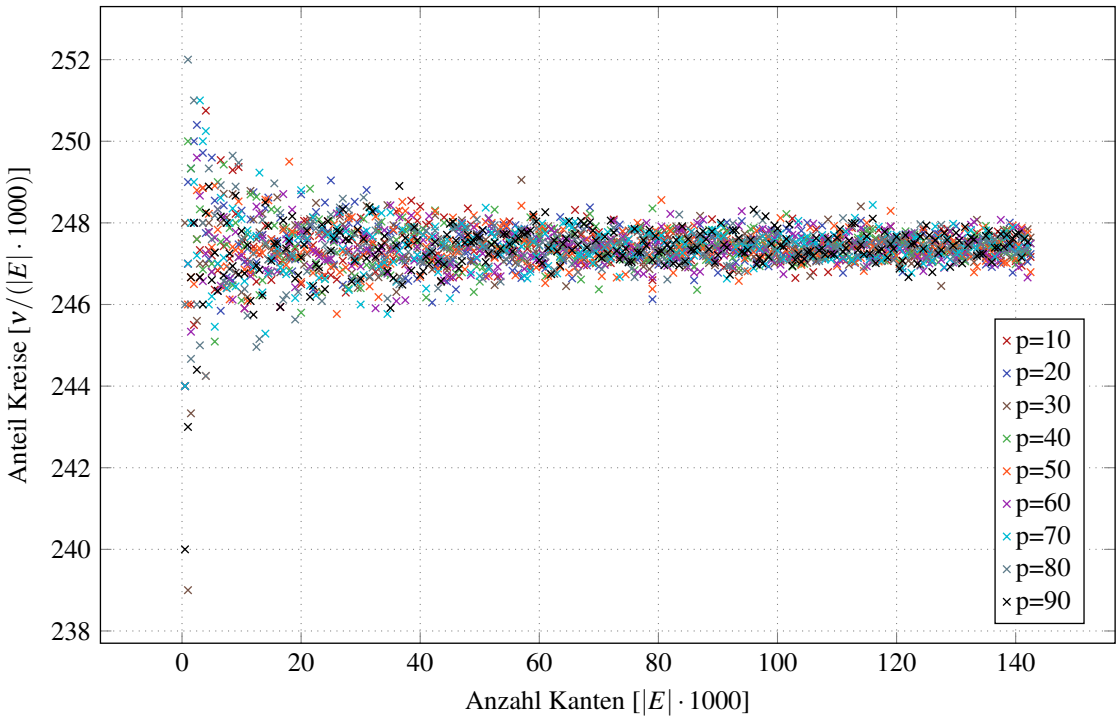


(a)

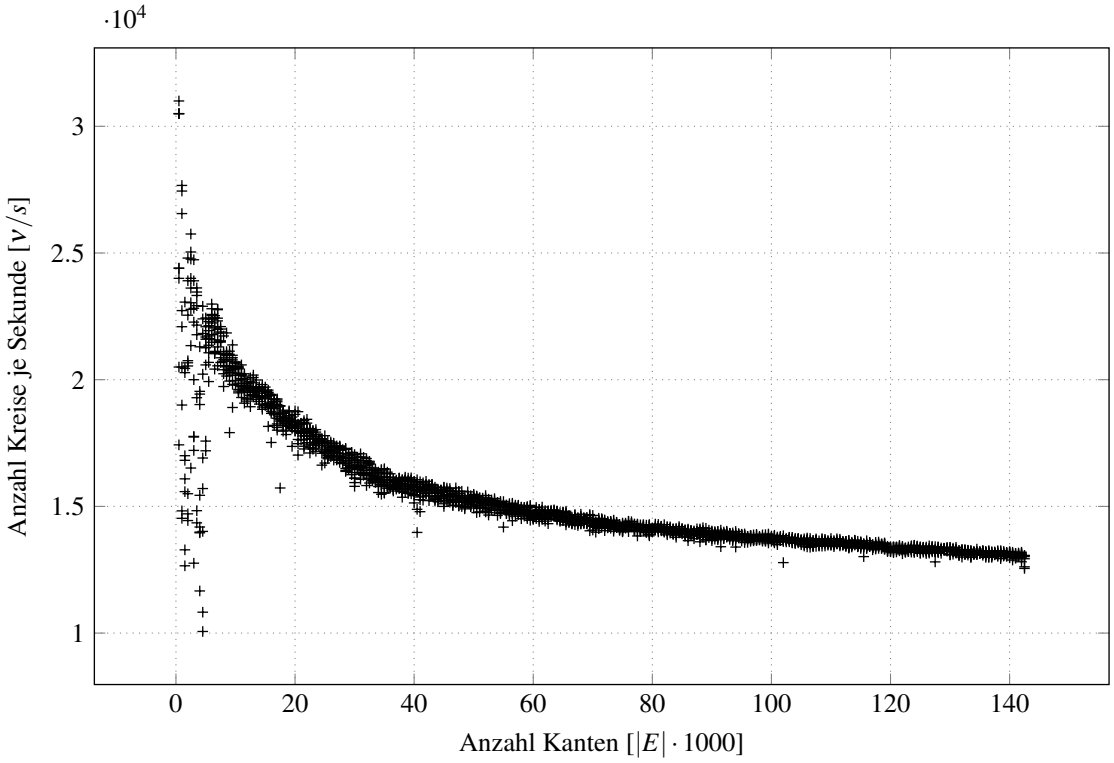


(b)

Abbildung 4.10: Laufzeit zur Bestimmung der Kreispackungszahl über alle Graphen (a) und je Wahrscheinlichkeit p (b).



(a)



(b)

Abbildung 4.11: Durchschnittliche Anzahl kantendisjunkter Kreise je tausend Kanten (a) und durchschnittliche Anzahl gefundener Kreise je Sekunde (b).

Kapitel 5

Zerlegung von Halin-Graphen

In diesem Kapitel werden nun spezielle 3-zusammenhängende Graphen betrachtet, die Halin-Graphen. Ein Graph G heißt *minimal* 3-*zusammenhängend*, falls G 3-zusammenhängend ist aber $G|_{E\setminus e}$ für jede Kante $e \in E(G)$ nicht 3-zusammenhängend ist. Ein Graph G heißt *Halin-Graph*, falls $|V(G)| \geqslant 4$, $\delta(v) \geqslant 3$ für alle Knoten $v \in V(G)$ und G aus der planaren Darstellung eines Baums T durch Verbinden seiner Blätter zu einem Kreis C konstruiert werden kann (vgl. Abbildung 5.1). Daher wird auch die Notation $G = T \cup C$ verwendet.

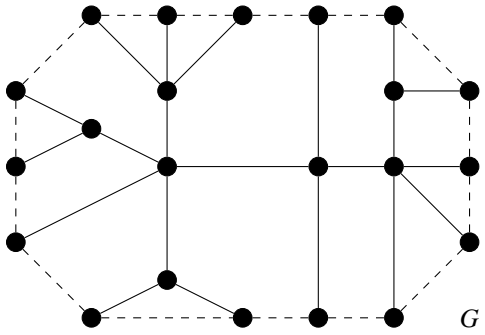


Abbildung 5.1: Ein Halin-Graph $G = T \cup C$. Der Kreis C wird durch die gestrichelten Kanten dargestellt, die übrigen Kanten induzieren den Baum T.

Ein einfaches Beispiel eines Halin-Graphen ist ein sogenanntes *Wheel*. Ein Wheel W_n ist ein Graph mit $n \geqslant 4$ Knoten $v_1, \ldots, v_n$ mit $\delta(v_1) = n - 1$ und $\delta(v_i) = 3, i = 2, \ldots, n$. Der Knoten v_1 (auch *Zentrum* genannt) ist adjazent zu allen Knoten, der Knoten v_i ist adjazent zu v_{i+1} für $i \in \{2, \ldots, n-1\}$ und v_n ist adjazent zu v_2. Die Kanten (v_1, v_i) für $i = 2, \ldots, n$ heißen *Speichen*. In der Notation für Halin-Graphen $W_n = T \cup C$ beschreibt

T einen *Stern* bestehend aus einem Zentrum und zugehörigen Speichen. Ein Wheel W_n heißt *gerade*, falls $n-1$ eine gerade Zahl ist, und *ungerade* im anderen Fall. Halin-Graphen (also auch Wheels) gehören zu den *planaren* Graphen. Ein Wheel W_n besitzt weiterhin exakt $n-1$ innere Facetten, die von genau drei Kanten umrandet werden und *Dreiecke* genannt werden. Ausschließlich die äußere Facette ist unbeschränkt. Abbildung 5.2 zeigt das gerade Wheel W_7 mit Zentrum v_1 und Speichen (v_1, v_i) für $i = 2, \ldots, 7$ sowie sechs Dreiecken, jeweils umrandet von zwei Speichen und einer weiteren zu C gehörenden Kante.

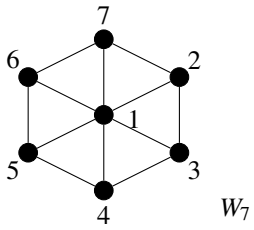


Abbildung 5.2: Beispiel eines Wheel-Graphen.

Einige ansonsten $\mathcal{NP}$-schwere Probleme auf Graphen können auf Halin-Graphen in polynomieller Zeit gelöst werden ([BPT05]), wie beispielsweise das Traveling Salesman Problem [CNP83], die Bestimmung eines gewichtsmaximalen Matchings [YYD07] und der chromatischen Zahl [ZL97]. Auch hinsichtlich des Problems der maximalen Kreispackungen haben Halin-Graphen eine Eigenschaft, die für die Entwicklung eines Lösungsalgorithmus genutzt werden soll. So haben Cornuéjols et al. zur Lösung eines TSP auf gewichteten Halin-Graphen ausgenutzt, dass ein Halin-Graph auf eine spezielle Weise in Wheels zerlegt werden kann [CNP83]. Mithilfe einer derartigen Wheel-Zerlegung soll nun im Rahmen dieser Arbeit ein Algorithmus zur Bestimmung der Kreispackungszahl eines Halin-Graphen entwickelt werden. Hierzu werden Resultate aus der Arbeit von Gardner genutzt, in welcher eine eindeutige Zerlegung minimal 3-zusammenhängender Graphen in Wheels, sogenannte *Twirls* und *zyklisch* 4-*zusammenhängende* Graphen entwickelt wurde [Gar89]. Ein *Twirl* ist ein vollständiger bipartiter Graph $K_{3,n}$ mit $n \geqslant 3$. Abbildung 5.3 zeigt den Twirl $K_{3,4}$. Ein minimal 3-zusammenhängender Graph heißt *zyklisch* 4-*zusammenhängend*, wenn keine 3-Separation $\{G_1, G_2\}$ existiert, sodass sowohl G_1 als auch G_2 einen Kreis enthalten.

Mit diesem Zerlegungsansatz können speziell Halin-Graphen, ähnlich wie in [CNP83], in Wheels zerlegt werden. Weiterhin bietet sich aber damit auch eine Möglichkeit, die Klasse der minimal 3-zusammenhängenden Graphen zu untersuchen und die gewonnenen

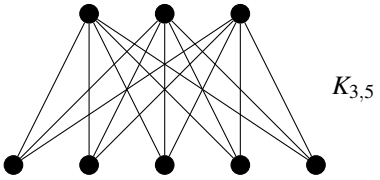


Abbildung 5.3: Beispiel eines Twirl-Graphen.

Zerlegungen für die Betrachtung von Kreispackungen nutzbar zu machen. Details dazu liefert der folgende Abschnitt.

5.1 Wheel-Zerlegung eines Halin-Graphen

Ähnlich wie die SPQR-Zerlegung im Fall eines 2-zusammenhängenden Graphen G soll nun eine Zerlegung für Halin-Graphen erläutert werden. Im Falle der 2-zusammenhängenden Graphen wurde dazu das Konzept der virtuellen Kanten entwickelt, um im zugehörigen Baum $\mathcal{T}$ darzustellen, auf welche Weise G zerlegt wurde und wie die so entstandenen Komponenten ursprünglich miteinander in Beziehung standen (siehe Abschnitt 4.3). Ein ähnliches Konzept soll zur Zerlegung minimal 3-zusammenhängender Graphen vorgestellt werden.

Es sei G ein 3-zusammenhängender Graph und es gebe in G eine 3-Separation $\{G_1, G_2\}$.[1] Es sei $\{s_1, s_2, s_3\} = V(G_1) \cap V(G_2)$. Ein Paar $\{G_1', G_2'\}$ heißt *einfache 3-Zerlegung* [CGW93, S. 8], falls es sich aus $\{G_1, G_2\}$ auf folgende Weise ergibt:

1. Für $i = 1, 2$ sei $V(G_i') = V(G_i) \cup \{x\}$. Der Knoten $x \in V(G_i')$ heißt *virtueller Knoten*.
2. Für $i = 1, 2$ sei $E(G_i') = E(G_i) \cup \{(s_j, x) \mid j = 1, \ldots, 3\}$.
3. Falls $\delta_{G_i'}(s_j) = 2$, kontrahiere in G_i' die beiden adjazenten Kanten $(v_k, s_j), (s_j, x)$ zur Kante (v_k, x).
4. Eine Kante $e \in E(G_i')$ heißt *virtuelle Kante*, falls $e = (s_j, x) \in E(G_i')$ für $i = 1, 2$.

Beispiel 5.1

Abbildung 5.4 zeigt die Konstruktion einer einfachen 3-Zerlegung $\{G_1', G_2'\}$ anhand eines

[1] Erinnerung: $\{G_1, G_2\}$ heißt *k-Separation von G*, falls $G = G_1 \cup G_2$ mit $|V(G_1) \cap V(G_2)| = k, E(G_1) \cap E(G_2) = \varnothing$ und $|E(G_1)| \geqslant k, |E(G_2)| \geqslant k$.

Beispielgraphen G mit 3-Separation $\{G_1, G_2\}$ und Separator $\{s_1, s_2, s_3\}$. Die virtuellen Kanten sind gestrichelt dargestellt.

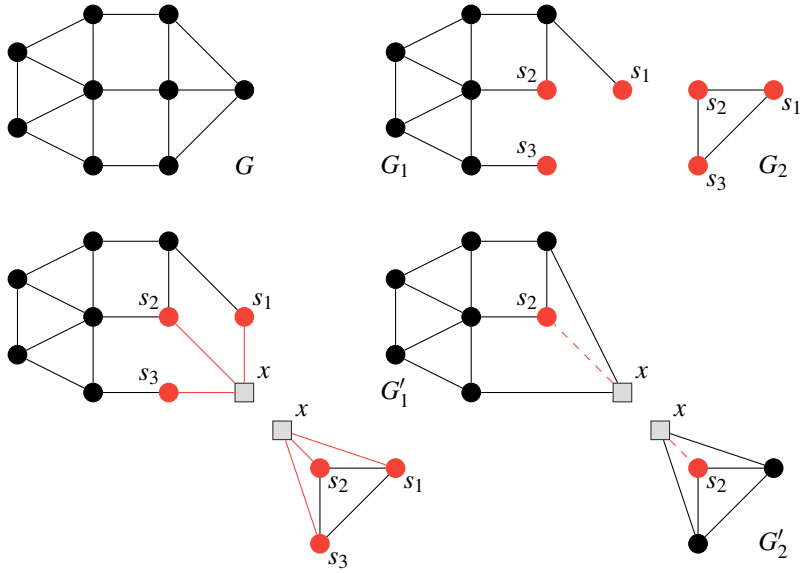


Abbildung 5.4: Konstruktion der einfachen 3-Zerlegung.

Wie man in Abschnitt 4.3 gesehen hat, mussten bei der Zerlegung von 2-zusammenhängenden Graphen G spezielle 2-Separationen (Splits) verwendet werden. Unter Verwendung derartiger Splits ergab sich dann eine eindeutige Zerlegung von G, deren Komponenten eine parallele oder serielle Struktur haben oder aber 3-zusammenhängend sind. Vor dem Hintergrund der Bestimmung einer maximalen Kreispackung zeigt sich, dass es für 3-zusammenhängende Graphen G notwendig wird, eine 3-Separation $\{G'_1, G'_2\}$ so durchzuführen, dass G'_1 und G'_2 bestimmte zusätzliche Eigenschaften erfüllen (siehe [CGW93] oder [Gar89, S. 26]). Die Forderung zusätzlicher Eigenschaften machen die beiden folgenden Beispiele deutlich.

Beispiel 5.2

Abbildung 5.5 zeigt eine 3-Separation $\{G_1, G_2\}$ eines minimal 3-zusammenhängenden Graphen G, bei welcher G_2 einem sogenannten *Claw-Graph*, dem vollständigen bipartiten Graph $K_{1,3}$, entspricht. Daraus ergibt sich die ebenfalls in Abbildung 5.5 dargestellte einfache 3-Zerlegung $\{G'_1, G'_2\}$ von G. Vor dem Hintergrund mithilfe einer Zerlegung von G kleinere Teilgraphen zu erzeugen (und diese dann einzeln weiter zu untersuchen) ist diese Art der 3-Separation offensichtlich nicht zielführend: So führt die 3-Separation in

diesem Beispiel zu einem Teilgraph G_1', welcher zum ursprünglichen Graphen G isomorph ist, und somit nicht zu einer gewünschten Verkleinerung.

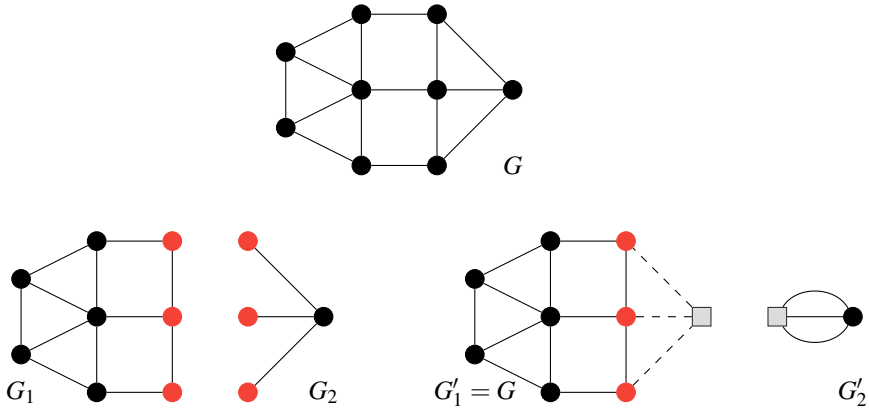


Abbildung 5.5: Auswirkungen einer (nicht zyklischen) 3-Separation.

Um zu verhindern, dass bei einer 3-Separation eines minimal 3-zusammenhängenden Graphen Claw-Graphen entstehen, werden nun zunächst sogenannte *zyklische* 3-*Separationen* betrachtet. Eine 3-Separation $\{\tilde{G}_1, \tilde{G}_2\}$ eines minimal 3-zusammenhängenden Graphen G wird *zyklische* 3-*Separation* genannt, falls sowohl $\tilde{G}_1$ als auch $\tilde{G}_2$ einen Kreis enthalten (siehe Abbildung 5.6). Minimal 3-zusammenhängende Graphen, die keine zyklische 3-Separation besitzen, werden als *zyklisch* 4-*zusammenhängend* bezeichnet. Dass

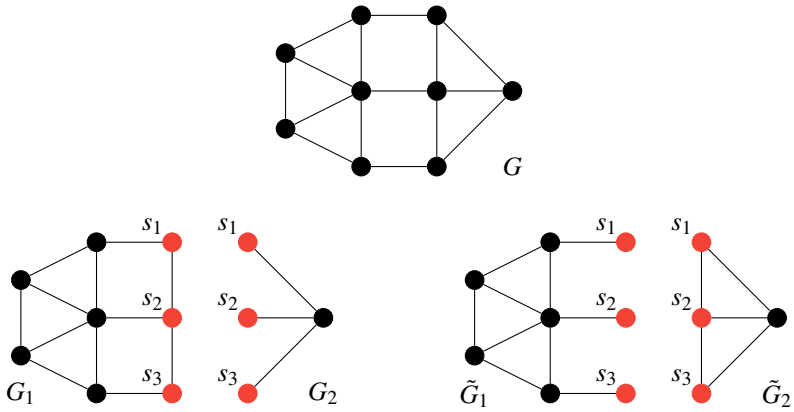


Abbildung 5.6: Graph G mit 3-Separation $\{G_1, G_2\}$ und zyklischer 3-Separation $\{\tilde{G}_1, \tilde{G}_2\}$.

aber auch durch zyklische 3-Separationen noch keine Verkleinerung von G garantiert wird, wird in folgendem Beispiel deutlich.

Beispiel 5.3

Abbildung 5.7 zeigt einen Graphen G mit der einfachen 3-Zerlegung $\{G'_1, G'_2\}$. Der durch $\{s_1, s_2, s_3\}$ induzierte Teilgraph ist offenbar isomorph zu K_3. Die Konstruktion der einfachen 3-Zerlegung führt zu drei virtuellen Kanten in G'_1 und G'_2, sodass die Kantenanzahl des Teilgraphen G'_1 der Kantenanzahl des Ausgangsgraphen G entspricht. So sind auch durch diese Zerlegung keine kleineren Komponenten G'_i entstanden.

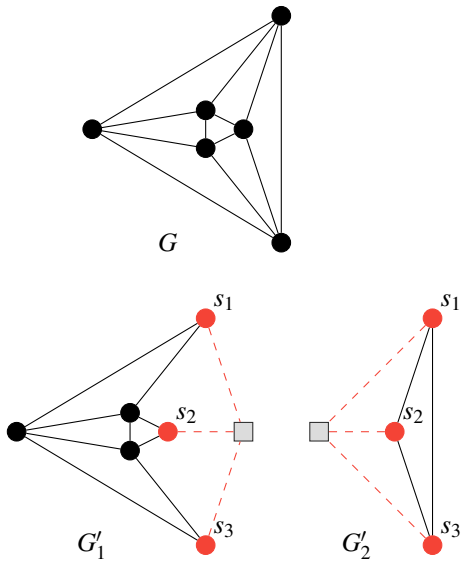


Abbildung 5.7: 3-Separatoren induzieren K_3 [Gar89].

Um Verkleinerungen zu garantieren, werden weitere Bedingungen an eine 3-Separation gestellt. Diese weiteren Einschränkungen sollen nun entsprechend [Gar89] eingeführt werden. Sei dafür $\{G_1, G_2\}$ eine zyklische 3-Separation bezüglich der 3-Separatoren $\{s_1, s_2, s_3\}$ eines minimal 3-zusammenhängenden Graphen G. Sei weiterhin S_3 die Menge aller Kanten, welche in G_1 oder G_2 inzident zu einem Knoten mit Knotengrad 1 sind, und seien S_1, S_2 die Mengen, welche sich durch $E(G_1 \backslash S_3)$ bzw. $E(G_2 \backslash S_3)$ ergeben. Dann wird das geordnete Tripel $S = (S_1, S_2; S_3)$ für $S_i \subset E(G)$ 3-*Split* von G genannt.

Beispiel 5.4

Abbildung 5.8 illustriert diesen Sachverhalt: Dort sind zwei 3-Separationen dargestellt. Die 3-Separationen $\{\tilde{G}_1, \tilde{G}_2\}$ (bezüglich des Separators $\{\tilde{s}_1, \tilde{s}_2, \tilde{s}_3\}$) und $\{\bar{G}_1, \bar{G}_2\}$ (bezüglich des Separators $\{\bar{s}_1, \bar{s}_2, \bar{s}_3\}$) führen zu 3-Splits $\tilde{S} = (\tilde{S}_1, \tilde{S}_2; \tilde{S}_3)$ und $\bar{S} = (\bar{S}_1, \bar{S}_2; \bar{S}_3)$. Zur besseren Kenntlichmachung sind die Kantenmengen farbig gekennzeichnet: Die Menge $\tilde{S}_1$ enthält die blauen Kanten aus $\tilde{G}_1$, die Menge $\tilde{S}_2$ enthält die grünen Kanten

aus $\tilde{G}_2$ und die Menge $\tilde{S}_3$ enthält die roten Kanten. Die Menge $C(\tilde{S})$ mit $C(\tilde{S}) := (\{\tilde{s}_1, \tilde{s}_2, \tilde{s}_3\}, \tilde{S}_3)$ heißt *Kontaktmenge*. Der zweite 3-Split $\bar{S}$ mit $C(\bar{S})$ ergibt sich analog.

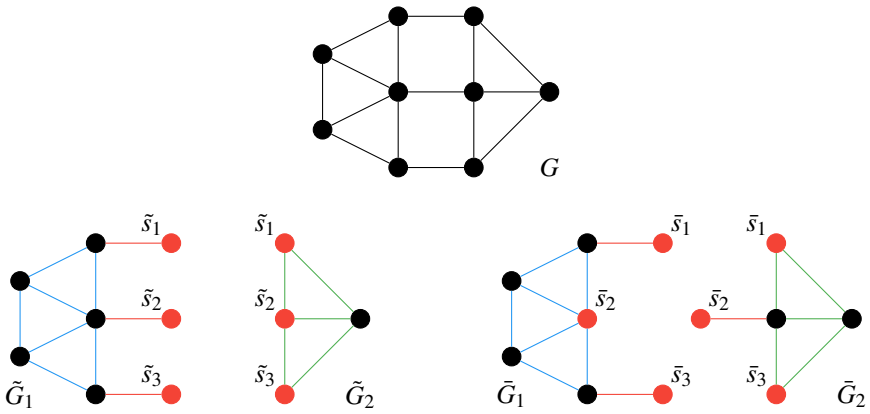


Abbildung 5.8: 3-Separation $\{\tilde{G}_1, \tilde{G}_2\}$ bzw. $\{\bar{G}_1, \bar{G}_2\}$ und 3-Split $\tilde{S}$ bzw. $\bar{S}$ von G.

Die in Beispiel 5.4 gezeigten 3-Separationen von G führen trotz unterschiedlicher 3-Separatoren offensichtlich zum gleichen 3-Split. Darüber hinaus induziert ein 3-Split eine eindeutige einfache 3-Zerlegung. Anhand der Anzahl von Kanten in der Kontaktmenge kann der 3-Split zusätzlich typisiert werden. Ein 3-Split ist vom *Typ* i, falls $|S_3| = i$ für $i \in \{0, 1, 2, 3\}$ gilt (siehe Abbildung 5.9).

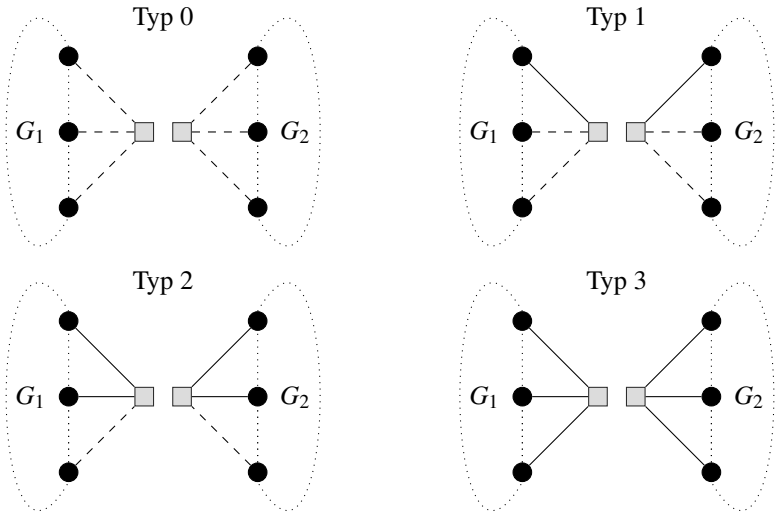


Abbildung 5.9: Die vier verschiedenen 3-Split-Typen.

Nun werden bestimmte Eigenschaften von 3-Splits definiert. Zwei 3-Splits $S = (S_1, S_2; S_3)$

und $S' = (S'_1, S'_2; S'_3)$ heißen *verschieden*, falls $S_1 \neq S'_1$ und $S_1 \neq S'_2$. Falls gilt, dass $S_i \backslash S'_j \neq \varnothing$ und $S'_j \backslash S_i \neq \varnothing$ für alle $i, j \in \{1, 2\}$, heißen S und S' *kreuzend*. Erfüllen zwei 3-Splits die Eigenschaften verschieden und nicht kreuzend zu sein, werden sie *kompatibel* genannt. Ein 3-Split, welcher mit jedem anderen kompatibel ist, wird dann ein *guter* 3-Split genannt. Die Betrachtung guter Splits verhindert die bereits beschriebenen 3-Kreise, welche zu Teilgraphen ohne verringerte Kantenanzahl führen.

Beispiel 5.5

Die beiden 3-Splits $\tilde{S}, \bar{S}$ aus Abbildung 5.8 sind offenbar nicht verschieden. Zwei sich kreuzende 3-Splits $\tilde{S}$ bzw. $\hat{S}$ werden beispielsweise durch die zyklischen 3-Separation $\{\tilde{G}_1, \tilde{G}_2\}$ (Abbildung 5.10(b)) bzw. $\{\hat{G}_1, \hat{G}_2\}$ (Abbildung 5.10(d)) induziert. Der durch $\{\bar{G}_1, \bar{G}_2\}$ (Abbildung 5.10(c)) induzierte 3-Split $\bar{S}$ ist hingegen nicht mit allen anderen 3-Splits kompatibel. Er kreuzt sich mit dem durch $\{\check{G}_1, \check{G}_2\}$ induzierten 3-Split $\check{S}$ (Abbildung 5.10(e)). Die 3-Separation $\{G_1, G_2\}$ in Abbildung 5.10(a) induziert einen guten 3-Split.

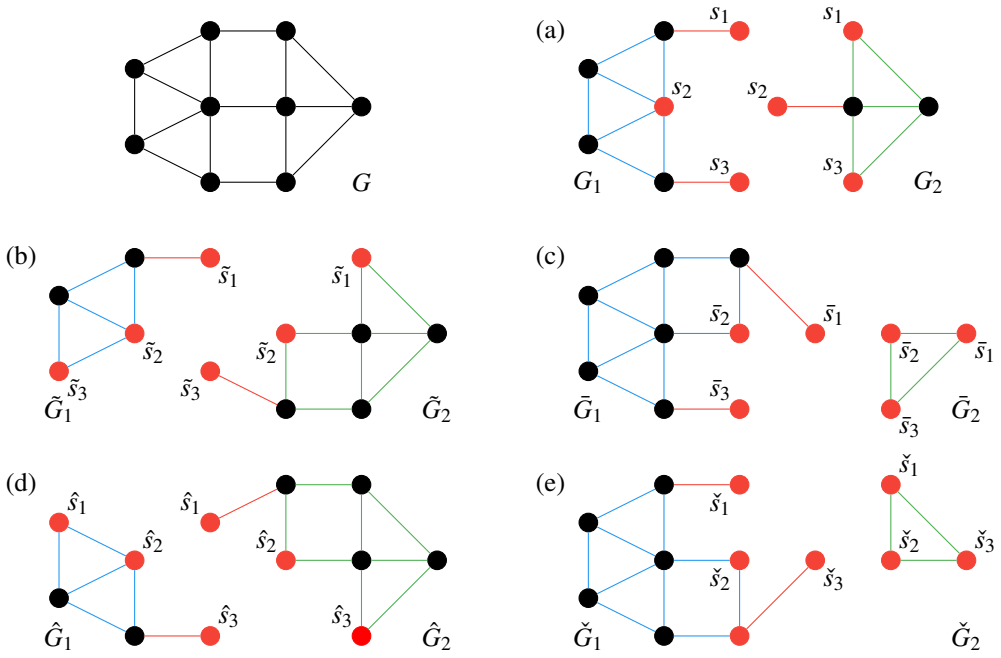


Abbildung 5.10: Verdeutlichung der Eigenschaften unterschiedlicher 3-Splits.

Vor dem Hintergrund einen minimal 3-zusammenhängenden Graphen G iterativ durch gute 3-Splits zu zerlegen, erhalten die folgenden Aussagen besondere Bedeutung.

Lemma 5.6 [CGW93]
Es sei $G = (V, E)$ ein minimal 3-zusammenhängender Graph. Die beiden folgenden Aussagen sind äquivalent

(i) G enthält keinen guten 3-Split

(ii) $G \simeq K_{3,n}$ $(n \geqslant 3)$ oder $G \simeq W_n$ $(n \geqslant 4)$ oder G ist zyklisch 4-zusammenhängend

Die grundlegende Idee besteht nun darin, für G eine zyklische 3-Separation $\{G_1, G_2\}$ anzugeben, derart, dass der induzierte 3-Split $S = \{S_1, S_2; S_3\}$ ein guter 3-Split ist. Da Teilgraphen G'_1, G'_2 der induzierten einfachen 3-Zerlegung nach [CGW93, S. 12] nun ebenfalls (minimal) 3-zusammenhängend sind, kann der Splitprozess für G'_1 bzw. G'_2 fortgesetzt werden, bis kein Teilgraph einen weiteren guten 3-Split erlaubt. Die im Laufe eines solchen Zerlegungsprozesses entstehenden Teilgraphen G'_i werden *Splitkomponenten* genannt. Die Zerlegung in Splitkomponenten ist nach Satz 5.7 von [CGW93] eindeutig.

Satz 5.7 [CGW93]
Es sei $G = (V, E)$ minimal 3-zusammenhängend. Dann gibt es eine eindeutige minimale Zerlegung $\mathcal{G} := \{G'_1, G'_2, \ldots, G'_N\}$ mit der Eigenschaft

$$G'_i \in \mathcal{G} \Leftrightarrow G'_i \simeq K_{3,n} \text{ oder } G'_i \simeq W_n \text{ oder } G'_i \text{ zyklisch 4-zusammenhängend.}$$

Dabei ist eine Zerlegung $\mathcal{G}$ *minimal* bezüglich einer Eigenschaft, wenn es keine Zerlegung $\mathcal{G}'$ mit $|\mathcal{G}'| < |\mathcal{G}|$, die diese Eigenschaft ebenfalls erfüllt. Im Folgenden wird eine solche Zerlegung verkürzt *die zu G zugehörige Zerlegung* $\mathcal{G}$ genannt. Für einen Halin-Graphen G kann Satz 5.7 noch spezifiziert werden.

Korollar 5.8
Es sei $G = (V, E)$ ein Halin-Graph und $\mathcal{G}$ die zu G zugehörige Zerlegung. Dann gilt

$$G'_i \in \mathcal{G} \Leftrightarrow G'_i \simeq W_n$$

Beispiel 5.9
Abbildung 5.11(a) zeigt einen Halin-Graph G. Die Zerlegung $\mathcal{G}$ von G resultiert in Kom-

ponenten $G'_i, i = 1, \ldots, 7$, entsprechend der Abbildung 5.11(b).

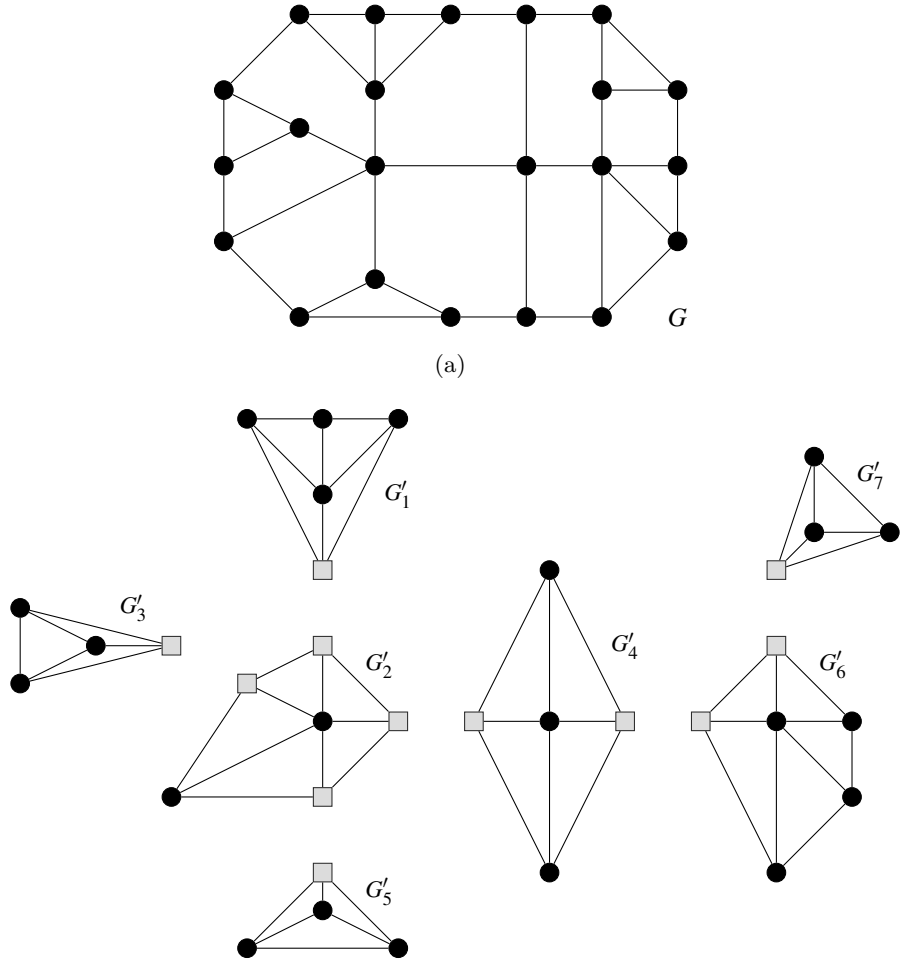


Abbildung 5.11: Halin-Graph G (a) und die zugehörige Zerlegung (b) mit Splitkomponenten $G'_i, i = 1, \ldots, 7$.

Durch die eindeutige Zerlegung $\mathcal{G} := \{G'_1, G'_2, \ldots, G'_N\}$ von G entsprechend Satz 5.7 wird ein weiterer Graph $\mathcal{T}(G) = (M, A)$ induziert. Hierbei entspricht ein Knoten $\mu \in M$ einer Splitkomponente G'_i und (μ, μ') ist eine Kante in A, wenn die zu μ und μ' gehörenden Splitkomponenten einen gemeinsamen virtuellen Knoten besitzen. Entsprechend des Typs der jeweiligen Splitkomponente kann jedem Knoten $\mu \in M$ ein Knotentyp zugeordnet werden:

- μ ist ein W-Knoten, falls G'_i ein Wheel ist,

- μ ist ein T-Knoten, falls G'_i ein Twirl ($K_{3,n}, n \geqslant 3$) ist,
- μ ist ein R-Knoten, falls G'_i ein zyklisch 4-zusammenhängender Graph ist.

Für eine solche Zerlegung $\mathcal{G}$ und den induzierten Graphen $\mathcal{T}(G)$ kann nun folgendes festgestellt werden:

Satz 5.10
Sei $G = (V, E)$ ein minimal 3-zusammenhängender Graph und $\mathcal{G} := \{G'_1, G'_2, \ldots, G'_N\}$ die zugehörige Zerlegung von G. Dann ist $\mathcal{T}(G) = (M, A)$ ein Baum.

Beweis: Da die Zerlegung $\mathcal{G} := \{G'_1, G'_2, \ldots, G'_N\}$ aus N Komponenten besteht und jede Komponente durch genau einen Knoten in $\mathcal{T}(G)$ repräsentiert wird, besitzt $\mathcal{T}(G)$ genau N Knoten. Nach Satz 5.7 ist die Zerlegung $\mathcal{G}$ eindeutig und keine der Komponenten $G'_i \in \mathcal{G}$ besitzt einen guten 3-Split. Der iterative Prozess, der zur Zerlegung $\mathcal{G}$ geführt hat, kann daher ohne Beschränkung der Allgemeinheit wie folgt beschrieben werden: Ein erster guter 3-Split von G führte zur Zerlegung $\{G'_1, H_1\}$, der zweite gute 3-Split von H_1 führte zur Zerlegung $\{G'_1, G'_2, H_2\}$, der i-te gute 3-Split von H_{i-1} führte zur Zerlegung $\{G'_1, G'_2, \ldots, G'_i, H_i\}$ und der $N-1$-te gute 3-Split von H_{N-2} führte zur Zerlegung $\{G'_1, G'_2, \ldots, G'_N\}$. Die Zerlegung $\mathcal{G} := \{G'_1, G'_2, \ldots, G'_N\}$ ist daher iterativ durch $N-1$ gute 3-Splits entstanden. Nach Konstruktion besitzt $\mathcal{T}(G)$ genau dann eine Kante $(\mu, \mu') \in A$, wenn die zu μ und μ' gehörenden Splitkomponenten G'_i und $G'_{i'}$ einen gemeinsamen virtuellen Knoten besitzen. Das ist dann der Fall, wenn G'_i und $G'_{i'}$ die Splitkomponenten eines guten 3-Splits sind. Da $\mathcal{G}$ iterativ durch $N-1$ gute 3-Splits entstanden ist, enthält $\mathcal{T}(G)$ $N-1$ Kanten und ist zusammenhängend. Also ist $\mathcal{T}(G)$ ein Baum. □

Analog zum SPQR-Baum für 2-zusammenhängende Graphen wird dieser Baum *WTR-Baum* genannt. Außerdem sei nun $\mathcal{G} = \{G'_{\mu_1}, \ldots, G'_{\mu_N}\}$, um den Bezug zwischen den Splitkomponenten der Zerlegung $\mathcal{G}$ und den Knoten von $\mathcal{T}$ zu verdeutlichen. Da eine solche Zerlegung $\mathcal{G}$ und der dazu gehörende WTR-Baum $\mathcal{T}$ später zur Bestimmung von Kreispackungen in G genutzt werden sollen, ist auch die Rekonstruktion von G aus der Zerlegung $\mathcal{G}$ von Bedeutung.

Satz 5.11
Sei $\mathcal{G} = \{G'_{\mu_1}, \ldots, G'_{\mu_N}\}$ die zugehörige Zerlegung eines minimal 3-zusammenhängenden Graphen G und $\mathcal{T}(G) = (M, A)$ der zugehörige Baum. Der Graph G kann wie folgt iterativ rekonstruiert werden:

1. Wähle eine Kante $(\mu, \mu') \in A$ und kontrahiere sie.

2. Verschmelze die Komponenten $G_\mu, G_{\mu'}$ zum Graphen $G_{(\mu,\mu')} = (V_{(\mu,\mu')}, E_{(\mu,\mu')})$ mit $E_{(\mu,\mu')} := E(G'_\mu)\backslash\{(s_j, x_{(\mu,\mu')}) \mid j = 1,2,3\} \cup E(G'_{\mu'})\backslash\{(v_k, x_{(\mu,\mu')}) \mid k = 1,2,3\}$ $\cup\{(s_j, v_k) \mid e_j = (s_j, x_{(\mu,\mu')}) \in E(G'_\mu)$ und $e_j = (v_k, x_{(\mu,\mu')}) \in E(G'_{\mu'})\ \forall j,k\}$ und $V_{(\mu,\mu')} := V(G'_\mu)\backslash\{x_{(\mu,\mu')}\} \cup V(G'_{\mu'})\backslash\{x_{(\mu,\mu')}\}$

3. Kontrahiere die Kante $(s_j, v_k) \in G_{(\mu,\mu')}$, falls $(s_j, x_{(\mu,\mu')})$ bzw. $(v_k, x_{(\mu,\mu')})$ virtuelle Kanten in G_μ bzw. $G_{\mu'}$ waren.

4. Lösche den virtuellen Knoten $x_{(\mu,\mu')}$. Starte dann erneut mit Schritt 1, falls $A \neq \varnothing$.

Beweis: Nach Konstruktion besitzt $\mathcal{T}(G)$ genau dann eine Kante $(\mu, \mu') \in A$, wenn die zu μ und μ' gehörenden Splitkomponenten G'_μ und $G'_{\mu'}$ einen gemeinsamen virtuellen Knoten besitzen. Das ist dann der Fall, wenn G'_μ und $G'_{\mu'}$ die Splitkomponenten eines guten 3-Splits sind. Die Splitkomponenten G'_μ und $G'_{\mu'}$ eines guten 3-Splits entstehen durch Konstruktion der einfachen 3-Zerlegung $\{G'_\mu, G'_{\mu'}\}$ aus einer 3-Separation $\{G_\mu, G_{\mu'}\}$. Der Ursprungsgraph $G_{(\mu,\mu')}$ einer 3-Separation $\{G_\mu, G_{\mu'}\}$ kann durch Zusammenführen der Graphen G_μ und $G_{\mu'}$ an den Knoten des 3-Separators rekonstruiert werden.

Die Zerlegung $\mathcal{G} = \{G'_{\mu_1}, \ldots, G'_{\mu_N}\}$ ist iterativ durch $N-1$ gute 3-Splits entstanden. Jeder dieser $N-1$ guten 3-Splits kann iterativ durch Revertierung der einfachen 3-Zerlegung auf eine 3-Separation und damit auf den Ausgangsgraphen der jeweiligen 3-Separation zurückgeführt werden: Durch Schritt 1 der Rekonstruktion wird jeder gute 3-Split einmal bearbeitet. Schritt 2 vereint Kanten- und Knotenmengen beider Komponenten G'_μ und $G'_{\mu'}$, löscht die mit dem virtuellen Knoten $x_{(\mu,\mu')}$ inzidenten Kanten, welche bei der Konstruktion der einfachen 3-Zerlegung ergänzt wurden, und verschmilzt die Komponenten an den Kanten $e_j \in G'_\mu$ und $e_j \in G'_{\mu'}$. In Schritt 3 werden alle virtuellen Kanten $e_j \in G_{(\mu,\mu')}$ kontrahiert. Schritt 4 löscht den in der einfachen 3-Zerlegung hinzugefügten virtuellen Knoten $x_{(\mu,\mu')}$. Der so entstandene Graph $G_{(\mu,\mu')}$ besitzt einen guten 3-Split, welcher zu den Splitkomponten $G_\mu, G_{\mu'}$ führt. □

Bemerkung 5.12

Da nach Korollar 5.8 für einen Halin-Graphen G und die zugehörige Zerlegung $\mathcal{G}$

$$G'_i \in \mathcal{G} \Leftrightarrow G'_i \simeq W_n$$

gilt, besteht $\mathcal{T}(G)$ ausschließlich aus W-Knoten. $\mathcal{T}(G)$ wird dann auch W-Baum genannt.

Die Abbildung 5.12(b) zeigt nun den W-Baum zum bereits bekannten Halin-Graphen und der zugehörigen Zerlegung.

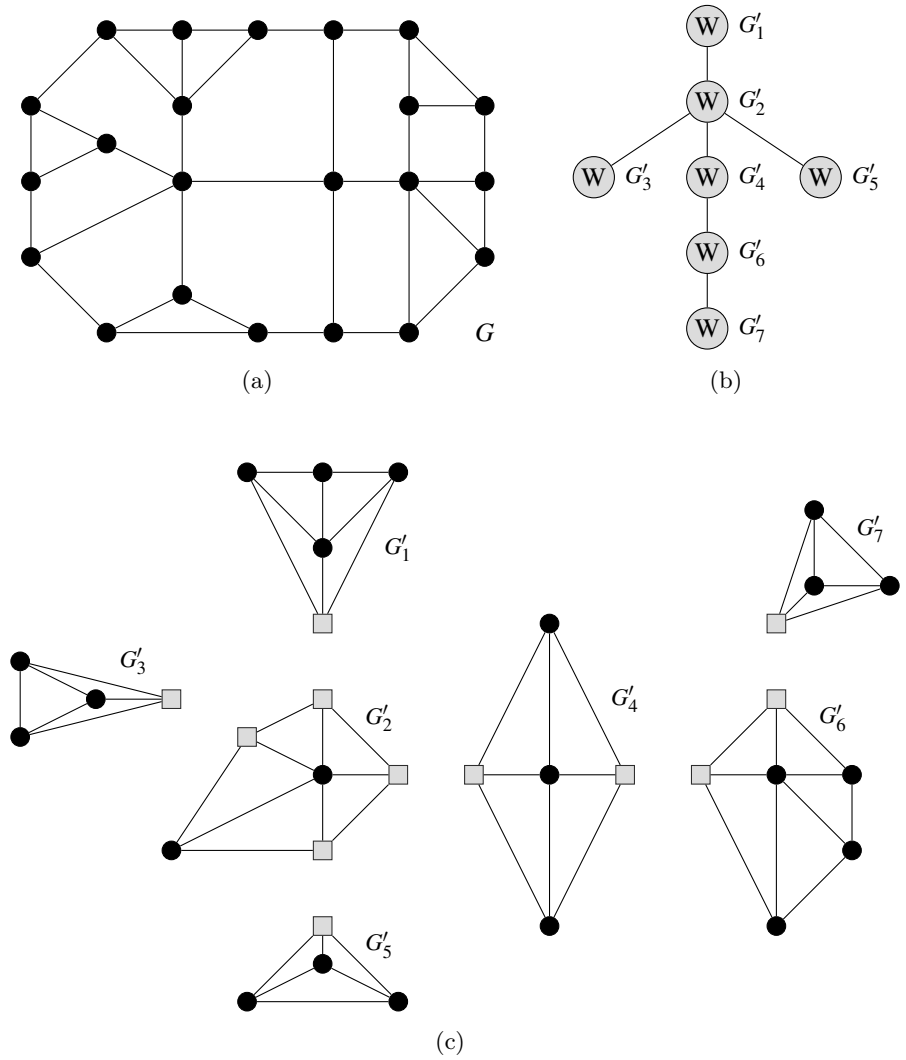


Abbildung 5.12: Halin-Graph G (a), die zugehörige Zerlegung (c) und der zugehöriger W-Baum (b).

In [Gar89] und [CGW93] wird ein Polynomialzeitalgorithmus zur Ermittlung der eindeutigen Zerlegung $\mathcal{G}$ eines minimal 3-zusammenhängenden Graphen entsprechend Satz 5.7 formuliert. Algorithmus 5.1 bestimmt sukzessive gute 3-Splits solange bis kein solcher mehr gefunden werden kann. Da nach [CGW93] die Zerlegung eines minimal 3-zu-

Algorithmus 5.1 Zerlegung eines minimal 3-zusammenhängenden Graphen [Gar89]

Eingabe: Ein minimal 3-zusammenhängenden Graph G.
Ausgabe: Zerlegung $\mathcal{G}$ von G entsprechend Satz 5.7.
1: $\mathcal{G} \leftarrow \{G\}$
2: **while** $\exists \tilde{G} \in \mathcal{G}$ mit $\tilde{G}$ besitzt einen guten 3-Split **do**
3: Konstruiere die zugehörige einfache 3-Zerlegung $\{\tilde{G}_1, \tilde{G}_2\}$
4: $\mathcal{G} \leftarrow \mathcal{G} \backslash \{G\} \cup \{\tilde{G}_1, \tilde{G}_2\}$
5: **end while**
6: $\mathcal{G}$ ist die Zerlegung von G

sammenhängenden Graphen G höchstens $N \geqslant \max\{1, 2|V(G)| - 10\}$ Elemente besitzt, muss in Algorithmus 5.1 höchstens $O(|V(G)|)$-mal ein guter 3-Split bestimmt werden. Für die Bestimmung eines guten 3-Splits wird in [CGW93] ebenfalls ein Verfahren angegeben (siehe Algorithmus 5.2). Für den Algorithmus wird eine weitere Version der Definition eines 3-Splits benötigt, die die Knotenmengen der Kanten in S_i verwendet. Sei $S = (S_1, S_2; S_3)$ ein 3-Split. Für $i = 1, 2$ sei $X_i := V(G|_{S_i})$, das heißt die Knotenmenge des durch S_i induzierten Graphen. Entsprechend [CGW93] gilt dann $G|_{X_i} = G|_{S_i}$ für $i = 1, 2$, sodass ein 3-Split S eindeutig durch $\{X_1, X_2\}$ bestimmt wird. Daher wird nun $\{X_1, X_2\}$ ebenfalls als 3-Split bezeichnet. Die Menge $C(X_i)$ bezeichne wiederum die Kontaktmenge, das heißt die Knotenmenge, die sich aus dem Schnitt $X_i \cap V(\tilde{G}|_{E(\tilde{G}) \backslash E(\tilde{G}|_{X_i})})$ ergibt.

Mithilfe von Algorithmus 5.2 wird zunächst nach guten 3-Splits gesucht, die zu Wheel- oder Twirl-Komponenten führen, da nach Satz 5.7 eine Zerlegung aus Wheels, Twirls oder zyklisch 4-zusammenhängenden Graphen besteht. Zur Identifikation einer Wheel-Komponente werden alle Kanten im Graphen $\tilde{G}$ untersucht, die möglicherweise der Speiche eines Wheels in der Zerlegung entsprechen könnten. Falls eine solche Kante existiert, wird in Zeile 4-6 untersucht, ob weitere adjazente Kanten existieren, die diese Kante so ergänzen können, dass sie einen Teilgraph eines Wheels induzieren. Die Menge X_1 eines potenziellen 3-Splits $\{X_1, X_2\}$ wird dann aus dem potenziellen Zentrum w eines Wheels und den adjazenten Knoten in W gebildet, die Menge X_2 aus den übrigen Knoten. Falls $|W| > 1$ und der 3-Split vom Typ 3 ist, ist $\{X_1, X_2\}$ ein guter 3-Split. Ist aber $|W| > 1$ und der 3-Split vom Typ 2, gibt es gegebenenfalls einen kreuzenden 3-Split. Dann ist $\{X_1, X_2\}$ kein guter 3-Split. Falls $|W| = 1$ ist, ist $\{X_1, X_2\}$ kein 3-Split. Existiert aber für $\tilde{G}|_{X_2}$ ein 2-Separator $\{w, y\}$, sodass y kein Element der Kontaktmenge ist, kann ein guter

Algorithmus 5.2 Bestimmung eines guten 3-Splits [Gar89]

Eingabe: Ein Graph $\tilde{G}$ mit $\tilde{G} \not\cong K_{3,n}$ $(n \geqslant 3)$ und $\tilde{G} \not\cong W_n$ $(n \geqslant 4)$.
Ausgabe: Ein guter 3-Split von $\tilde{G}$.

▷ Finde guten 3-Split, der eine Wheel-Komponente erzeugt:
for $(v, w) \in E(\tilde{G})$ mit $\delta(v) = 3$ und $\delta(w) \geqslant 4$ **do**
  $W \leftarrow \{v\}$
  for $u \in V(\tilde{G}) \backslash (W \cup \{w\})$ **do**
    if $\delta(u) = 3$ und u ist adjazent zu w und mindestens einem weiteren Knoten aus W **then**
      $W \leftarrow \{u\}$
    end if
  end for
  $X_1 \leftarrow W \cup \{w\}$ und $X_2 \leftarrow V(\tilde{G}) \backslash W$
  if $|W| > 1$ und $\{X_1, X_2\}$ ist Typ 3 **then**
    return $\{X_1, X_2\}$ ist ein guter 3-Split.
  else if ($|W| > 1$ und $\{X_1, X_2\}$ ist Typ 2) oder $|W| = 1$ **then**
    if $\tilde{G}|_{X_2}$ besitzt einen 2-Separator $\{w, y\}$ für $y \in X_2 \backslash C(X_2)$ **then**
      Wähle y so, dass für die zugehörige 2-Separation $\{\tilde{G}|_{\tilde{E}_1}, \tilde{G}|_{\tilde{E}_2}\}$ die Zahl $|\tilde{E}_1|$ minimal ist.
      Sei $\{Y_1, Y_2\}$ der von $\{\tilde{G}|_{\tilde{E}_1}, \tilde{G}|_{\tilde{E}-\tilde{E}_1}\}$ induzierte 3-Split.
      return $\{Y_1, Y_2\}$ ist ein guter 3-Split.
    end if
  end if
end for

▷ Finde guten 3-Split, der eine Twirl-Komponente erzeugt:
for $u, v \in V(\tilde{G})$ mit $\{(u, h_i) \mid i = 1, 2, 3\} \subset E(\tilde{G})$ und $\{(v, h_i) \mid i = 1, 2, 3\} \subset E(\tilde{G})$ **do**
  Wähle Komponente $\tilde{H}$ von $\tilde{G} \backslash \{h_1, h_2, h_3\}$ mit $|V(\tilde{H})| > 1$.
  $\tilde{E}_1 \leftarrow E(\tilde{H}) \cup \{(x, y) \in E(\tilde{G}) \mid x \in V(\tilde{H})$ und $y \in \{h_1, h_2, h_3\}\}$
  Sei $\{X_1, X_2\}$ der von $\{\tilde{G}|_{\tilde{E}_1}, \tilde{G}|_{E(\tilde{G}) \backslash \tilde{E}_1}\}$ induzierte 3-Split.
  return $\{X_1, X_2\}$ ist ein guter 3-Split.
end for

▷ Finde irgendeinen 3-Split:
for $v \in V(\tilde{G})$ mit $\delta(v) = 3$ **do**
  if $u, w \in V(\tilde{G})$ mit $\delta(u) = \delta(w) = 3$ und $(u, v), (w, v), (u, w) \in E(\tilde{G})$ **then**
    $X_1 \leftarrow \{u, v, w\}$ und $X_2 \leftarrow V(\tilde{G}) \backslash X_1$
    return $\{X_1, X_2\}$ ist ein guter 3-Split.
  end if
end for
for $v \in V(\tilde{G})$ **do**
  Bestimme, falls möglich, einen Tutte-Split $\{\tilde{G}'_1, \tilde{G}'_2\}$ in $\tilde{G} \backslash \{v\}$.
  Sei $\{u, w\}$ der 2-Separator zu $\{\tilde{G}'_1, \tilde{G}'_2\}$.
  if $\{u, w\} \neq \varnothing$ **then**
    Sei $\{\tilde{G}_1, \tilde{G}_2\}$ die durch den 3-Separator $\{u, v, w\}$ erzeugte 3-Separation.
    Sei $\{X_1, X_2\}$ der von $\{\tilde{G}_1, \tilde{G}_2\}$ induzierte 3-Split.
    return $\{X_1, X_2\}$ ist ein guter 3-Split.
  end if
end for

3-Split konstruiert werden. Enthält $\tilde{G}|_{X_2}$ mehrere solcher 2-Separatoren, wird derjenige gesucht, für den die Kantenmenge E_1 bei zugehöriger 3-Separation $\{\tilde{G}|_{\tilde{E}_1}, \tilde{G}|_{\tilde{E}_2}\}$ minimal ist. Nach Konstruktion von W enthält $\tilde{G}|_{\tilde{E}_1}$ außer w und y noch mindestens einen weiteren Knoten. Da $\tilde{G}$ 3-zusammenhängend ist, enthält $|E_1|$ mindestens drei Kanten. Daher ist $\{\tilde{G}|_{\tilde{E}_1}, \tilde{G}|_{\tilde{E}-\tilde{E}_1}\}$ eine 3-Separation und der induzierte 3-Split $\{Y_1, Y_2\}$ ist ein guter 3-Split. Konnte entsprechend Zeile 1-18 kein guter 3-Split gefunden werden, der zu einer Wheel-Komponente in der Zerlegung $\mathcal{G}$ führt, wird als nächstes nach einem guten 3-Split gesucht, der zu einem Twirl führt.

In den Zeilen 19-24 wird ein Teilgraph eines Twirls $H = K_{3,2}$ mit den Knoten $\{u, v\} \cup \{h_1, h_2, h_3\}$ mit $\delta(u) = \delta(v) = 3$ und $\delta(h_i) = 2$ für $i = 1, 2, 3$ gesucht. Falls ein solcher Twirl existiert, werden die Komponenten betrachtet, in die der Restgraph $\tilde{G}\backslash\{h_1, h_2, h_3\}$ zerfällt, damit eine 3-Separation konstruiert werden kann. Da $\tilde{G}$ minimal 3-zusammenhängend ist, gehören alle Komponenten, die nur einen Knoten enthalten zum gesuchten Twirl $H = K_{3,n}$. Der Graph H bildet dann den einen Graphen der 3-Separation zum 3-Separator $\{h_1, h_2, h_3\}$ und der Graph, induziert durch die Kanten der Komponente $\tilde{H}$ mit $|V(\tilde{H})| > 1$, bildet zusammen mit den Kanten, die sie mit $\{h_1, h_2, h_3\}$ verbinden (im Algorithmus bezeichnet mit E_1), den anderen Graphen der 3-Separation. Der induzierte 3-Split $\{X_1, X_2\}$ ist ein guter 3-Split. Konnte entsprechend Zeile 19-24 kein guter 3-Split gefunden werden, der zu einer Twirl-Komponente in der Zerlegung führt, existieren keine kreuzenden 3-Splits in $\tilde{G}$, sodass jeder 3-Split ein guter 3-Split ist. Daher wird als nächstes nach einem beliebigen 3-Split gesucht.

In den Zeilen 25-30 wird $\tilde{G}$ auf Dreiecke untersucht, da ein Dreieck einen 3-Split induziert. Falls $\tilde{G}$ auch kein Dreieck enthält, wird in den Zeilen 31-39 für beliebige Knoten $v \in V(\tilde{G})$ ein Tutte-Split in $\tilde{G}\backslash\{v\}$ gesucht. Gibt es einen solchen Tutte-Split mit 2-Separator $\{u, w\}$ in $\tilde{G}\backslash\{v\}$, ist $\{u, v, w\}$ ein 3-Separator für $\tilde{G}$ und der zugehörige 3-Split ein guter 3-Split. Sollte auch auf diese Weise kein guter 3-Split gefunden werden, ist $\tilde{G}$ ein zyklisch 4-zusammenhängender Graph und entspricht somit einer Komponente G_i in $\mathcal{G}$.

Das hier entsprechend [CGW93; Gar89] beschriebene Verfahren zur Bestimmung eines guten 3-Splits hat eine Laufzeit von $O(|V(\tilde{G})||E(\tilde{G})|)$, sodass das Verfahren zur Bestimmung der Zerlegung von G insgesamt $O(|V(G)|^2|E(G)|)$ Zeit benötigt.

Die in diesem Abschnitt definierte Zerlegung soll später zur Formulierung eines Verfahrens dienen, welches kantendisjunkte Kreispackungen in Halin-Graphen bestimmt.

5.2 Ergebnisse zu Fans und guten 3-Splits in Halin-Graphen

Mithilfe der Verkleinerung eines Halin-Graphen G zu Wheels konnten Cornuéjols et al. in dieser Klasse von Graphen bereits ein polynomielles Verfahren für ein anderes im Allgemeinen NP-schweres Problem entwickeln: das Traveling Salesman Problem [CNP83]. Das dort vorgestellte Vorgehen G zu verkleinern basiert auf der Kontraktion spezieller Teilstrukturen eines Halin-Graphen, sogenannter *Fans*. Da eine Beziehung zwischen der Kontraktion eines Fans und einer Zerlegung mithilfe guter 3-Splits besteht, soll nun erst das in [CNP83] vorgestellte Vorgehen zur Verkleinerung von G vorgestellt werden. Im Anschluss daran werden wichtige Ergebnisse aus [CNP83] erläutert, welche später zur Bestimmung einer maximalen kantendisjunkten Kreispackung verwendet werden können.

Es sei $G = T \cup C$ ein Halin-Graph und T enthalte mindestens zwei Knoten, die kein Blatt sind. Sei nun w ein Knoten in T mit $\delta_T(w) \neq 1$, welcher zu genau einem weiteren Knoten $v \in V(T)$ mit $\delta_T(v) \neq 1$ adjazent ist. Die Menge aller mit w adjazenten Blätter in T wird mit $C(w)$ bezeichnet. Der durch die Knotenmenge $\{w\} \cup C(w)$ induzierte Teilgraph von G heißt *Fan* (dt. Fächer), der Knoten w heißt *Zentrum* des Fans. Abbildung 5.13 zeigt einen Halin-Graphen und die Fans F_1, F_2, F_3, F_4. Die rot markierten Knoten sind die Zentren des jeweiligen Fans.

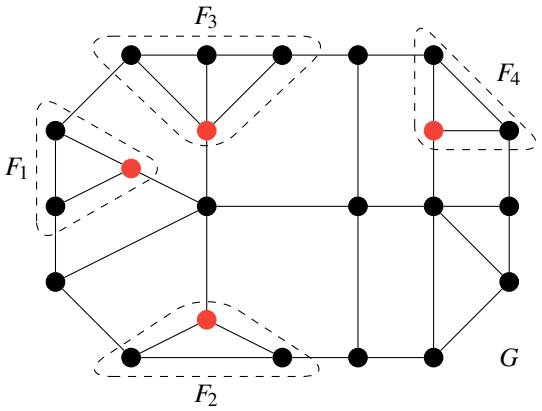


Abbildung 5.13: Halin-Graph G und Fans $F_1, \ldots, F_4$.

Es gilt die folgende Aussage:

Lemma 5.13 [CNP83]
Sei $G = T \cup C$ ein Halin-Graph, der kein Wheel ist. Dann besitzt G mindestens zwei Fans.

Auf Basis dieser Aussage haben Cornuéjols et al. in [CNP83] eine Verkleinerung eines Halin-Graphen definiert, welche nun erläutert werden soll. Sei $G = (V, E)$ ein Halin-Graph und $F \subseteq G$ ein Fan. Entsprechend [CNP83] sei $G \times F = (V(G \times F), E(G \times F))$ der Graph, der aus G durch Kontraktion von F zu einem Pseudoknoten w_F entsteht, genauer:

$$
\begin{aligned}
V(G \times F) :=& V(G) \backslash V(F) \cup \{w_F\}, \\
E(G \times F) :=& E(G) \backslash E(F) \\
&\cup \{(u, w_F) \mid (u, v) \in G \text{ mit } u \notin V(F) \text{ und } v \in V(F)\} \\
&\backslash \{(u, v) \mid u \notin V(F) \text{ und } v \in V(F)\}.
\end{aligned}
$$

Ein Beispiel einer solchen Verkleinerung zeigt Abbildung 5.14 anhand des Graphen G mit Fan F_1. Der Fan F_1 wird zu einem Pseudoknoten w_{F_1} kontrahiert, die Adjazenzen zwischen den Knoten des Fans und den übrigen Knoten des Graphen werden durch entsprechende Kanten zwischen dem Pseudoknoten und den übrigen Knoten des Graphen dargestellt.

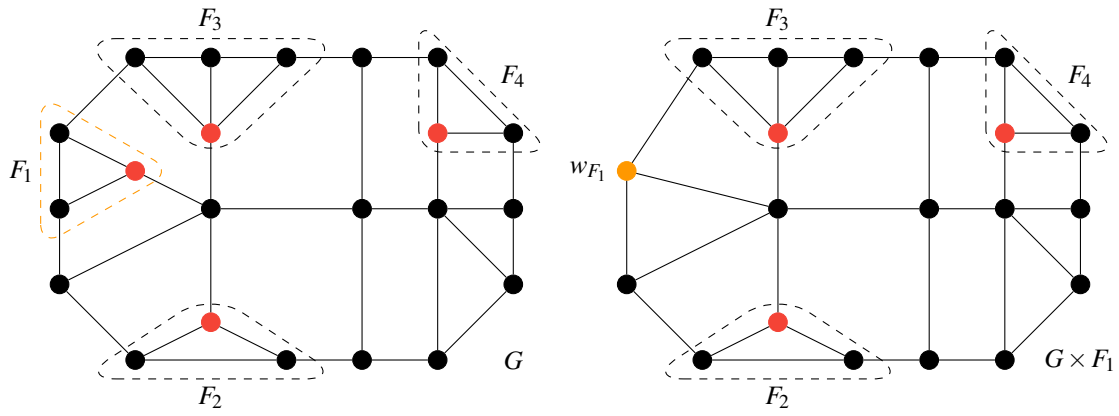


Abbildung 5.14: Halin-Graph G und der verkleinerte Graph $G \times F_1$.

Für einen im Vergleich zu G verkleinerten Graphen $G \times F$ gilt nun weiter die folgende Aussage.

Lemma 5.14 [CNP83]
Sei F ein Fan in einem Halin-Graph G. Dann ist $G' := G \times F$ ebenfalls ein Halin-Graph.

Bemerkung 5.15
Ist $G = T \cup C$ ein Halin-Graph und F ein Fan in G, so hat $G' = G \times F$ eine Darstellung $G' = T' \cup C'$. Offenbar ist $T' \subsetneq T$. Genauer gilt $T' = T \backslash C(w)$ für $F = \{w\} \cup C(w)$.

Sei $\mathcal{F}(G) = \{F \mid F \text{ ist Fan in } G\}$ die Menge aller Fans in G. Sei $\tilde{\mathcal{F}} = \{F_1, \ldots, F_k\} \subseteq \mathcal{F}(G)$. Dann ist $G \times \tilde{\mathcal{F}} := (((G \times F_1) \times F_2) \times \ldots) \times F_k$. Bei gegebener Menge $\mathcal{F}$ ist das Ergebnis $G \times \tilde{\mathcal{F}}$ unabhängig von der Reihenfolge der Operationen.

Betrachte nun einen Fan F mit Zentrum w in einem Halin-Graphen G. Der Teilgraph $\tilde{F}$ von G mit $\tilde{F} := F \cup \{\{v, w\} \in E(G) \mid v \notin F\} \cup \{\{v, \tilde{w}\} \in E(G) \mid v \notin F, \tilde{w} \in C(w)\}$ heißt *erweiterter Fan*. $\tilde{F}$ entspricht also dem Fan F ergänzt um die drei Kanten, die F mit dem restlichen Halin-Graphen verbinden (siehe Abbildung 5.15).

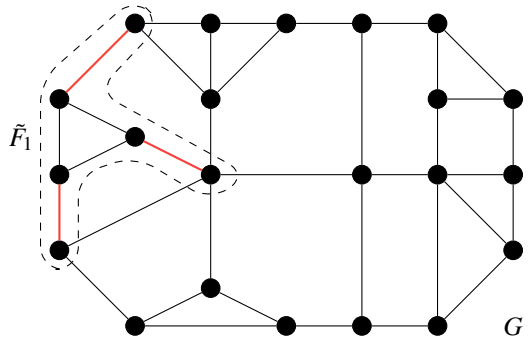


Abbildung 5.15: Halin-Graph G und der erweiterte Fan $\tilde{F}_1$.

Mithilfe des Begriffs eines erweiterten Fans kann nun eine Aussage zur Beziehung zwischen der Kontraktion eines Fans und einer Zerlegung mitihlfe guter 3-Splits getroffen werden.

Satz 5.16
Sei F ein Fan in einem Halin-Graph $G = T \cup C$ und $\tilde{F}$ der entsprechend erweiterte Fan. Dann ist $S = (E(F), E(G \backslash \tilde{F}); E(\tilde{F} \backslash F))$ ein guter 3-Split in G.

Beweis: Betrachte zur begleitenden Illustration der folgenden Erläuterungen Abbil-

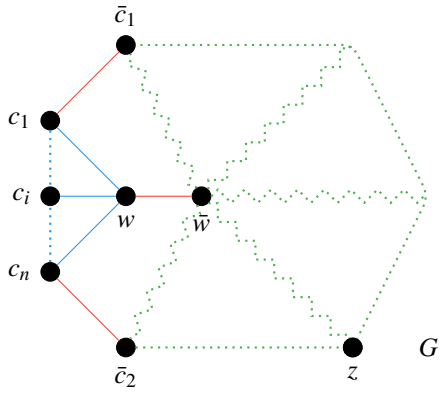


Abbildung 5.16: Skizze eines Halin-Graphen G

dung 5.16. Angenommen es existiert ein 3-Split $\tilde{S}$ mit zyklischer 3-Separation $\{\tilde{G}_1, \tilde{G}_2\}$ derart, dass S und $\tilde{S}$ kreuzend sind. Sei $e_1 \in S_1 \backslash \tilde{S}_1$ und $e_2 \in S_1 \backslash \tilde{S}_2$. Daraus folgt $\{\tilde{s}_1, \tilde{s}_2, \tilde{s}_3\} \cap C(w) \neq \varnothing$. O.B.d.A. sei $\tilde{s}_1 \in C(w)$. Angenommen $w \notin \{\tilde{s}_2, \tilde{s}_3\}$, dann gehört w entweder zu $V(\tilde{G}_1)$ oder zu $V(\tilde{G}_2)$ (aber nicht zu beiden Knotenmengen). O.B.d.A. sei $w \in V(\tilde{G}_1)$. Dann ist aber $C(w) \subset V(\tilde{G}_1)$ und somit $\bar{c}_1, \bar{c}_2 \in V(\tilde{G}_1)$. Das ist ein Widerspruch zur Annahme, dass S und $\tilde{S}$ kreuzende 3-Splits sind.

O.B.d.A. sei $\tilde{s}_2 = w$. Da $w \in T$, muss $\tilde{s}_3 \in C$ gelten.

Fall 1: Sei $\tilde{s}_3 \in V(\tilde{F}) \backslash \{w, \tilde{s}_2\}$. Dann ist aber $\tilde{S}_1 \subset S_1$ oder $\tilde{S}_2 \subset S_1$, das heißt, es gilt $\tilde{S}_1 \backslash S_1 = \varnothing$ oder $\tilde{S}_2 \backslash S_1 = \varnothing$. Das ist ein Widerspruch zur Annahme, dass S und $\tilde{S}$ kreuzend sind.

Fall 2: Sei $\tilde{s}_3 \notin V(\tilde{F})$, dann ist $\tilde{s}_3 \in C \backslash V(F)$. Dann liegen $\bar{c}_1$ und $\bar{c}_2$ in verschiedenen Zusammenhangskomponenten der Zerlegung $\{G_1, G_2\}$. O.B.d.A. sei $\bar{c}_1 \in \tilde{G}_1$ und $\bar{c}_2 \in \tilde{G}_2$. Betrachte den eindeutig bekannten Weg $P(\bar{c}_1, \bar{c}_2)$ in G, der ausschließlich aus Kanten von T besteht. $P(\bar{c}_1, \bar{c}_2)$ muss mindestens einen inneren Knoten $\bar{w} \in T$ enthalten. Offenbar muss $w \notin P(\bar{c}_1, \bar{c}_2)$ gelten, da F sonst kein Fan wäre. Damit ist aber $\bar{c}_2$ von $\bar{c}_1$ aus in $\tilde{G}_1$ erreichbar, das heißt $\bar{c}_2 \in \tilde{G}_1$. Das ist ein Widerspruch zu der Annahme, dass $\tilde{S}$ ein 3-Split ist. □

Somit kann festgehalten werden, dass ein Fan F in einem Halin-Graphen G einen guten 3-Split induziert, welcher immer vom Typ 3 ist.

Korollar 5.17
Sei G ein Halin-Graph und $\mathcal{T}(G)$ der zu einer Zerlegung $\mathcal{G}$ von G gehörende Baum. Dann gibt es für jeden Fan $F \subset G$ einen Knoten $\mu \in V(\mathcal{T}(G))$ mit $F \subset G_\mu$ und G_μ enthält keine virtuelle Kante.

Damit kann also jeder Fan eines Halin-Graphen G einem Blatt μ und dementsprechend auch einem Graphen G_μ zugeordnet werden.

Korollar 5.18
Sei G ein Halin-Graph und $\mathcal{T}(G)$ der zu einer Zerlegung $\mathcal{G}$ von G gehörende Baum. Sei $F \subset G$ der Fan mit $F \subset G_\mu$. Dann gilt

(i) für den Halin-Graphen $G \times F$ ist $\mathcal{T}(G \times F) = \mathcal{T}(G) \backslash \{\mu\}$.

(ii) Ist $w \in V(G)$ Zentrum von F, so ist w Zentrum von G_μ.

Nachdem nun die Beziehungen zwischen einem Fan im Halin-Graphen G, einem guten 3-Split in G und einer Splitkomponente der Zerlegung $\mathcal{G}$ erläutert wurden, kann nun eine weitere Eigenschaft für einen Fan F und eine Splitkomponente G_μ mit $F \subset G_\mu$ beschrieben werden.

Bemerkung 5.19
Sei G ein Halin-Graph und $\mathcal{T}(G)$ der zu einer Zerlegung $\mathcal{G}$ von G zugehörige Baum. Sei $F \subset G$ der Fan mit $F \subset G_\mu$. Wenn F eine (un-)gerade Anzahl von Facetten besitzt, besitzt G_μ ebenfalls eine (un-)gerade Anzahl von Facetten.

Kapitel 6

Kreispackungen in Halin-Graphen

Im vorherigen Kapitel konnten vielversprechende Eigenschaften einer Zerlegung von Halin-Graphen aufgezeigt werden. Nun werden spezielle Halin-Graphen betrachtet, für die mithilfe der Erkenntnisse aus Kapitel 5 eine maximale Kreispackung bzw. die Kreispackungszahl bestimmt werden soll. Hierfür werden zunächst einige Definitionen benötigt.

6.1 Maximale Kreispackungen in speziellen Halin-Graphen

Ein Graph $G = (V, E)$ heißt *kubisch*, falls $\delta(v) = 3$ für alle $v \in V$ gilt. Ein Baum T heißt *Caterpillar*, falls der Teilgraph P von T, der dadurch entsteht, dass sämtliche Blätter in T entfernt werden, einen Weg $P = (v_1, v_2, \ldots, v_k)$ der Länge $k \geqslant 2$ beschreibt. Wir nennen P den zu T gehörigen Weg. Einen Halin-Graphen $G = T \cup C$ mit T Caterpillar nennen wir *Caterpillar-Halin-Graphen*. Existiert zu einem Caterpillar-Halin-Graph eine Nummerierung der Knoten in C mit $C = (u_0, u_1, \ldots, u_{|C|-1}, u_0)$, sodass für jeweils zwei Kanten $(v_i, u_r), (v_j, u_s) \in E(G)$ mit $i < j$ auch $r < s$ gilt, heißt G *Necklace-Halin-Graph*.

Ist $G = T \cup C$ ein kubischer Caterpillar-Halin-Graph mit zugehörigem Weg $P = (v_1, \ldots, v_k)$, dann gibt es $u_0, u_1, \ldots, u_k, u_{k+1}$ mit der Eigenschaft

- $V(G) = V(P) \cup V(C) = \{v_1, \ldots, v_k, u_0, \ldots, u_{k+1}\}$ sowie

- $E(G) = \{(u_i, v_i) \mid 1 \leqslant i \leqslant k\} \cup \{(u_0, v_1), (u_{k+1}, v_k)\} \cup E(C)$.

In einem kubischen Caterpillar-Halin-Graph ist für $2 \leqslant i \leqslant k-1$ also jeder Knoten v_i adjazent zu genau einem Blatt u_i von T. Die Knoten v_1 und v_k sind jeweils zu genau zwei Blättern von T adjazent.

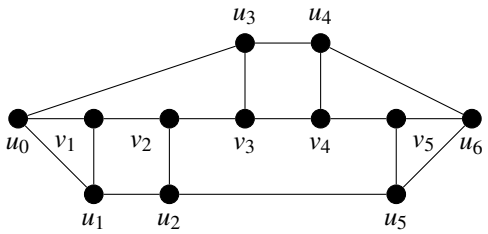


Abbildung 6.1: Beispiel eines kubischen Caterpillar-Halin-Graph

Ist $G = T \cup C$ ein kubischer Necklace-Halin-Graph mit zugehörigem Weg $P = (v_1, \ldots, v_k)$, dann gibt es $u_0, u_1, \ldots, u_k, u_{k+1}$ mit der Eigenschaft

- $V(G) = V(P) \cup V(C) = \{v_1, \ldots, v_k, u_0, \ldots, u_{k+1}\}$ sowie

- $E(G) = \{(u_i, v_i) \mid 1 \leqslant i \leqslant k\} \cup \{(u_i, u_{i+1}) \mid 0 \leqslant i \leqslant k\} \cup \{(v_i, v_{i+1}) \mid 1 \leqslant i \leqslant k-2\} \cup \{(u_0, u_{k+1})\{(u_0, v_1), (u_{k+1}, v_k)\}$.

In einem kubischen Necklace-Halin-Graph ist für $2 \leqslant i \leqslant k-1$ also jeder Knoten v_i adjazent zu genau einem Blatt u_i von T, wobei eine planare Darstellung angegeben werden kann, sodass die Blätter sich alle unterhalb des Weges P befinden. Die Knoten v_1 und v_k sind jeweils zu genau zwei Blättern von T adjazent. Wir bezeichnen einen kubischen Necklace-Halin-Graph, dessen zugehöriger Weg P die Länge k besitzt, mit N_k. Abbildung 6.2 zeigt den kubischen Necklace-Halin-Graphen N_5. Offenbar besitzt N_k

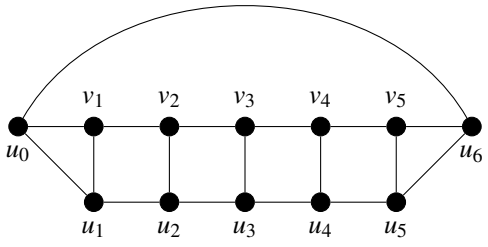


Abbildung 6.2: Kubischer Necklace-Halin-Graph N_5

die Ordnung $2k + 2$.

Der *Girth* $\gamma(G)$ eines Graphen G bezeichnet die Länge eines kürzesten Kreises in G. Darüber hinaus bezeichnet $\gamma_2(G)$ bzw. $\gamma_3(G)$ die Länge eines zweit- bzw. drittkürzesten Kreises in G.

Satz 6.1
Sei $N_k = T \cup C$ mit $k \geqslant 2$ ein kubischer Necklace-Halin-Graph der Ordnung n. Dann ist

$$\nu(N_k) = \left\lfloor \frac{n+2}{4} \right\rfloor = \left\lfloor \frac{2(k+2)}{4} \right\rfloor = \left\lfloor \frac{k+2}{2} \right\rfloor.$$

Beweis: Für $k = 2$ besitzt T nach Definition $k + 2 = 4$ Blätter und N_2 somit 6 Knoten. Die folgende Abbildung 6.3 zeigt den Graphen N_2.

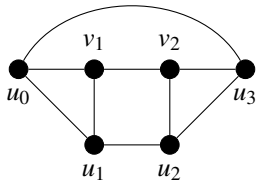


Abbildung 6.3: Kubischer Necklace-Halin-Graph N_2

Offensichtlich ist $\nu(N_2) = \left\lfloor \frac{n+2}{4} \right\rfloor = \left\lfloor \frac{8}{4} \right\rfloor = 2$.

Sei nun $k \geqslant 3$. Offenbar ist $n = 2k + 2$.

(i) Ist $\hat{C}$ ein Kreis in N_k, so enthält $\hat{C}$ mindestens eine Kante $e \in E(C)$. Da $|E(C)| = k + 2$ gilt und N_k kubisch ist, kann es höchstens $\left\lfloor \frac{|E(C)|}{2} \right\rfloor$ kantendisjunkte Kreise in N_k geben, das heißt

$$\nu(N_k) \leqslant \left\lfloor \frac{k+2}{2} \right\rfloor = \left\lfloor \frac{k+1}{2} + \frac{1}{2} \right\rfloor = \left\lfloor \frac{2(k+1)}{4} + \frac{1}{2} \right\rfloor = \left\lfloor \frac{n}{4} + \frac{1}{2} \right\rfloor = \left\lfloor \frac{n+2}{4} \right\rfloor.$$

(ii) Nun ist zu zeigen, dass auch $\nu(N_k) \geqslant \left\lfloor \frac{k+2}{2} \right\rfloor$ gilt.

Fall 1: Es sei k eine gerade Zahl. Betrachte folgende in Abbildung 6.4 rot markierte

kantendisjunkte Kreispackung. Die markierte Kreispackung $\mathcal{Z}$ enthält zwei

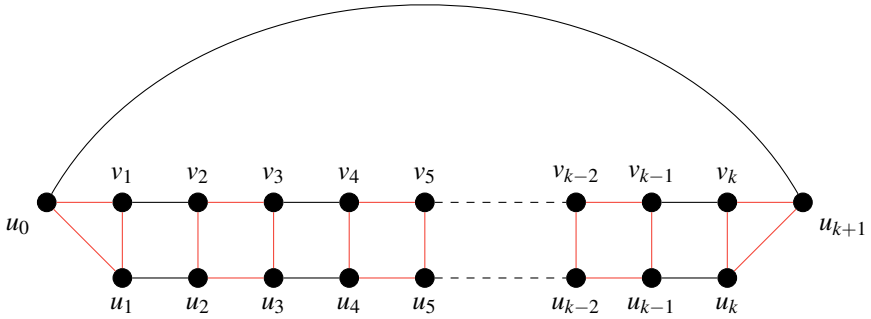


Abbildung 6.4: Kubischer Necklace-Halin-Graph N_k mit $k \geqslant 3$ und k gerade

Kreise der Länge drei und $\frac{k-2}{2}$ Kreise der Länge vier. Damit lässt sich $|\mathcal{Z}|$ abhängig von n wie folgt bestimmen.

$$|\mathcal{Z}| = 2+\frac{k-2}{2} = \frac{k+2}{2} = \left\lfloor \frac{k+2}{2} \right\rfloor = \left\lfloor \frac{2(k+1)}{4} + \frac{1}{2} \right\rfloor = \left\lfloor \frac{n}{4} + \frac{1}{2} \right\rfloor = \left\lfloor \frac{n+2}{4} \right\rfloor$$

Fall 2: Es sei k eine ungerade Zahl. Betrachte folgende in Abbildung 6.5 rot markierte kantendisjunkte Kreispackung. Die markierte Kreispackung $\mathcal{Z}$ enthält

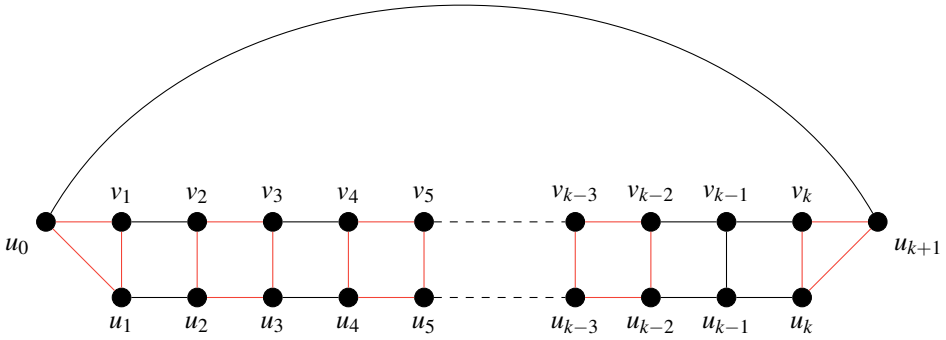


Abbildung 6.5: Kubischer Necklace-Halin-Graph N_k mit $k \geqslant 3$ und k ungerade

zwei Kreise der Länge drei und $\frac{k-3}{2}$ Kreise der Länge vier. Damit lässt sich $|\mathcal{Z}|$ abhängig von n wie folgt bestimmen.

$$|\mathcal{Z}| = 2 + \frac{k-3}{2} = \frac{k+1}{2} = \left\lfloor \frac{k+1}{2} + \frac{1}{2} \right\rfloor = \left\lfloor \frac{k+2}{2} \right\rfloor = \left\lfloor \frac{n+2}{4} \right\rfloor$$

Für die Kreispackungszahl von N_k muss also folgendes gelten:

$$\nu(N_k) \geqslant |\mathcal{Z}| = \left\lfloor \frac{n+2}{4} \right\rfloor$$

Aus (i) $\nu(N_k) \leqslant \left\lfloor \frac{n+2}{4} \right\rfloor$ und (ii) $\nu(N_k) \geqslant \left\lfloor \frac{n+2}{4} \right\rfloor$ folgt dann $\nu(N_k) = \left\lfloor \frac{n+2}{4} \right\rfloor$. □

Nun betrachten wir einen (nicht notwendigerweise kubischen) Necklace-Halin-Graphen. Ein Beispiel eines solchen Graphen zeigt Abbildung 6.6.

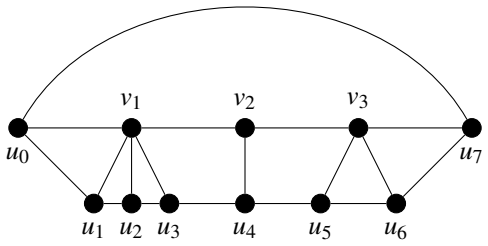


Abbildung 6.6: Beispiel eines (nicht kubischen) Necklace-Halin-Graphen

Auch in einer solchen Situation kann die Kreispackungszahl und eine maximale Kreispackung unmittelbar angegeben werden.

Satz 6.2
Sei $N_k = T \cup C$ ein Necklace-Halin-Graph der Ordnung n. Dann ist

$$\nu(N_k) = \left\lfloor \frac{n-k}{2} \right\rfloor.$$

Beweis: Ist $\hat{C}$ ein Kreis in N_k, so enthält $\hat{C}$ mindestens eine Kante $e \in E(C)$. Da $|E(C)| = n - k$ gilt und $\delta(v) = 3$ für $v \in C$, kann es höchstens $\left\lfloor \frac{|E(C)|}{2} \right\rfloor$ kantendisjunkte Kreise in N_k geben, das heißt

$$\nu(N_k) \leqslant \left\lfloor \frac{n-k}{2} \right\rfloor.$$

Seien $P = (v_1, \ldots, v_k)$ der zu N_k gehörende Weg und $C = (u_0, \ldots, u_{n-k-1})$ der äußere Kreis in N_k. Da N_k ein Halin-Graph ist, gibt es zu jedem Knoten $u_i \in C$ genau einen Knoten $a(u_i) \in V(P)$.

Fall 1: Die Zahl $n-k$ ist gerade. Betrachte die Kreispackung

$$\mathcal{Z} = \{(u_i, u_{i+1}, a(u_{i+1}), a(u_i), u_i) \mid i = 0, 2, \ldots, n-k-2\}.$$

Es handelt sich um eine kantendisjunkte Kreispackung mit Kardinalität $|\mathcal{Z}| = \frac{n-k}{2}$.

Fall 2: Die Zahl $n-k$ ist ungerade. Betrachte die Kreispackung

$$\mathcal{Z} = \{(u_i, u_{i+1}, a(u_{i+1}), a(u_i), u_i) \mid i = 0, 2, \ldots, n-k-3\}.$$

Es handelt sich um eine kantendisjunkte Kreispackung mit Kardinalität $|\mathcal{Z}| = \left\lfloor \frac{n-k}{2} \right\rfloor$.

Damit gilt die Behauptung. □

Die Kreispackungszahlen für Necklace-Halin-Graphen sind offenbar einfach zu bestimmen. Im nächsten Schritt soll die größere Klasse der Caterpillar-Halin-Graphen betrachtet werden. Zur Bestimmung der Kreispackungszahl $\nu(G)$ als auch einer maximalen Kreispackung $\mathcal{Z}^*(G)$ für kubische Caterpillar-Halin-Graphen G wird nun ein zu G modifizierter dualer Graph G^* verwendet. Es sei G ein planarer Graph zusammen mit einer planaren Darstellung. Es sei $\mathcal{B}^+(G)$ die Menge der Facetten von G. Der Graph $G^* = (V^*, E^*)$ heißt *Dualgraph* von G, falls $V^* = \mathcal{B}^+(G)$ und $(B, B') \in E^*$ genau dann, wenn B und B' eine gemeinsame Kante in G haben. Die Einbettung eines 3-zusammenhängenden Graphen und sein Dualgraph können in $O(|V|)$ Zeit bestimmt werden [EK05]. Den zum Graphen G aus Abbildung 6.1 gehörenden Dualgraph G^* zeigt Abbildung 6.7 (roter Graph).

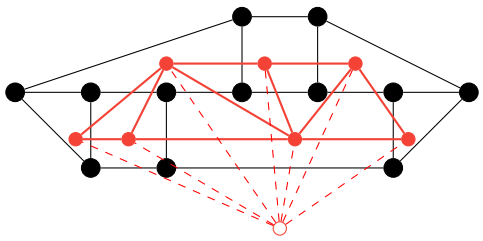


Abbildung 6.7: Beispiel eines Graphen G und seines Dualgraphen G^*

Nun soll ein modifizierter Dualgraph G^*_{mod} betrachtet werden. Dazu sei G ein planarer Graph zusammen mit einer planaren Darstellung. Ist $B_a \in \mathcal{B}^+(G)$ die äußere Facette von G, so heißt der Graph $G^*_{mod} = G^* \backslash \{B_a\}$ der *innere Dualgraph* von G. Der innere

duale Graph G^*_{mod} entsteht also durch Löschen des Knotens B_a in G^*, welcher die äußere Facette in G repräsentiert, und der mit ihm inzidenten Kanten. Zu dem in Abbildung 6.7 schwarz gekennzeichneten Graphen G entspricht der innere duale Graph G^*_{mod} dem roten Graphen ohne den nicht farbig ausgefüllten Knoten und die gestrichelt gekennzeichneten inzidenten Kanten. Sei nun $\mathcal{B}(G) := \mathcal{B}^+(G)\backslash\{B_a\}$ die Menge der inneren Facetten von G.

In einer planaren Darstellung eines kubischen Caterpillar-Halin-Graphen G trennt P die Menge der inneren Facetten $\mathcal{B}(G)$ in G und damit auch die Menge der Knoten $\mathcal{B}(G) \in V(G^*_{mod})$ in zwei disjunkte Teilmengen. Wir bezeichnen mit $\mathcal{B}_s(G)$ die Menge der Facetten oberhalb von P und mit $\mathcal{B}_t(G)$ die Menge der Facetten unterhalb von P. Ist $B \in \mathcal{B}(G)$ eine Facette, so bezeichne $V(B)$ die durch B induzierten Knoten in G.

Satz 6.3
Der innere duale Graph G^*_{mod} eines kubischen Caterpillar-Halin-Graphen G ist ein serienparalleler Graph.

Beweis: Sei $G = T \cup C$ ein kubischer Caterpillar-Halin-Graph zusammen mit einer planaren Darstellung. Offensichtlich ist G^*_{mod} planar. Da G kubisch ist, entspricht jede Facette $B^* \in \mathcal{B}^*(G^*_{mod})$ einem Dreieck. Da zudem T Caterpillar mit zugehörigem Weg $P = (v_1, \ldots, v_k)$, $k \geqslant 2$, ist, gibt es mindestens zwei Knoten B_1 und B_2 in G^*_{mod} mit $\delta_{G^*_{mod}}(B_1) = \delta_{G^*_{mod}}(B_2) = 2$. Weiterhin gilt $\mathcal{B}(G) = \mathcal{B}_s(G) \dot\cup \mathcal{B}_t(G)$ mit $\mathcal{B}_s(G), \mathcal{B}_t(G) \neq \varnothing$, sodass für jede Facette $B^* \in \mathcal{B}^*(G^*_{mod})$ entweder $|V(B^*) \cap \mathcal{B}_s(G)| = 2$ und $|V(B^*) \cap \mathcal{B}_t(G)| = 1$ gilt oder $|V(B^*) \cap \mathcal{B}_s(G)| = 1$ und $|V(B^*) \cap \mathcal{B}_t(G)| = 2$. Damit lässt sich G^*_{mod} wie folgt auf den K_2 reduzieren:

1. Wähle den Knoten $B_1 \in G^*_{mod}$. Wegen $\delta_{G^*_{mod}}(B_1) = 2$ ist dieser Knoten adjazent zu einem Knoten $B_s \in \mathcal{B}_s(G)$ und $B_t \in \mathcal{B}_t(G)$.

2. Verschmelze den Kantenzug (B_s, B_1, B_t) in G^*_{mod} zur Kante (B_s, B_t).

3. Lösche eine der beiden parallelen Kanten $(B_s, B_t) \in G^*_{mod}$. Dadurch reduziert sich der Knotengrad von B_s und B_t um eins, sodass entweder $\delta_{G^*_{mod}}(B_s) = 2$ oder $\delta_{G^*_{mod}}(B_t) = 2$ gilt.

4. Setze $B_1 := B$ für $B \in \{B_s, B_t\}$ mit $\delta_{G^*_{mod}}(B) = 2$ und starte erneut mit Schritt 1.

Damit ist G^*_{mod} serienparallel. □

Nun kann das Problem der Bestimmung einer maximalen Kreispackung auf kubischen Caterpillar-Halin-Graphen G auf eine andere Problemstellung auf dem modifizierten dualen Graphen zurückgeführt werden. Sei $G = (V, E)$ ein Graph. Eine Knotenteilmenge $I \subseteq V$ von G heißt *stabile* bzw. *unabhängige Menge* in G, wenn keine zwei Knoten aus I adjazent in G sind. Die *Unabhängigkeitszahl* $\alpha(G)$ bezeichne die maximale Größe von I in G. Für einen Halin-Graphen G gilt $\nu(G) = \alpha(G^*)$ entsprechend [Ste21]. Darüber hinaus wurde in [Tol89] gezeigt, dass die maximale stabile Menge eines serienparallenen Graphen in linearer Zeit bestimmt werden kann. Es ergibt sich somit

Korollar 6.4
Sei G ein kubischer Caterpillar-Halin-Graph. Dann kann

(i) $\nu(G)$ in linearer Zeit bestimmt werden,

(ii) $\mathcal{Z}(G)$ in linearer Zeit bestimmt werden und

(iii) $\nu(G^*_{mod})$ und $\mathcal{Z}(G^*_{mod})$ können in linearer Zeit bestimmt werden.

Die Aussage von Korollar 6.4(iii) ist eine unmittelbare Konsequenz aus der Verwendung von Algorithmus 4.5. Außerdem stellen wir folgendes fest:

Korollar 6.5
Sei G ein kubischer Caterpillar-Halin-Graph und $\mathcal{G}$ die zugehörige Zerlegung von G. Dann gilt

$$G'_i \in \mathcal{G} \Leftrightarrow G'_i \simeq W_4.$$

Beweis: Nach Korollar 5.8 gilt $G'_i \in \mathcal{G} \Leftrightarrow G'_i \simeq W_n$ für einen Halin-Graphen G. Da G ein kubischer Caterpillar-Halin-Graph ist, ist auch jedes G_i kubisch. Damit gilt $G'_i \simeq W_4$.□

Im nächsten Schritt können diese Ergebnisse auf allgemeine Caterpillar-Halin-Graphen

erweitert werden.

Satz 6.6

Der innere duale Graph G^*_{mod} eines Caterpillar-Halin-Graphen G ist serienparallel.

Beweis: Sei $G = T \cup C$ ein Caterpillar-Halin-Graph zusammen mit einer planaren Darstellung und zugehörigem Weg $P = (v_1, \ldots, v_k)$. Offensichtlich ist G^*_{mod} planar. Seien $B, B' \in \mathcal{B}(G)$ zwei beliebige benachbarte innere Facetten in G. Dann sind B, B' adjazente Knoten in G^*_{mod}. Es sind zwei Fälle zu unterscheiden:

Fall 1: B, B' besitzen eine gemeinsame Kante $(v_i, u_j) \in E(T) \backslash E(P)$. Betrachte nun alle entsprechenden Kanten $(B, B') \in E(G^*_{mod})$. Da T Caterpillar ist, bilden diese Kanten einen Kreis C^* in G^*_{mod}.

Fall 2: B, B' besitzen eine gemeinsame Kante $(v_i, v_{i+1}) \in P$. Betrachte nun alle entsprechenden Kanten $(B, B') \in E(G^*_{mod})$. Es gilt $(B, B') \notin C^*$, aber, da T Caterpillar ist, gilt $B, B' \in C^*$.

Wiederum da T Caterpillar mit zugehörigem Weg $P = (v_1, \ldots, v_k)$, $k \geqslant 2$, ist, gibt es mindestens zwei Wege $C^*_1, C^*_2 \subset C^*$ mit $\delta_{G^*_{mod}}(B_{1,i}) = \delta_{G^*_{mod}}(B_{2,j}) = 2$ für alle Knoten $B_{1,i}, B_{2,j} \in C^*_1, C^*_2$. Damit lässt sich G^*_{mod} wie folgt auf K_2 reduzieren:

1. Wähle den Weg $C^*_1 \subset G^*_{mod}$. Wegen $\delta_{G^*_{mod}}(B_{1,i}) = 2$ sind der Startknoten $B_{1,1}$ bzw. der Endknoten $B_{1,|C^*_1|}$ des Weges adjazent zu zwei Knoten $B, B' \in C^*$ mit $(B, B') \in E(G^*_{mod}) \backslash E(C^*_1)$.

2. Verschmelze den Kantenzug (B, C^*_1, B') in G^*_{mod} zur Kante (B, B').

3. Lösche eine der beiden parallelen Kanten $(B, B') \in G^*_{mod}$. Entsprechend Fall 2 besitzen $B, B' \in \mathcal{B}(G)$ eine gemeinsame Kante $(v_i, v_{i+1}) \in P$. Dadurch gilt entweder $\delta_{G^*_{mod}}(B) = 3$ oder $\delta_{G^*_{mod}}(B') = 3$. Durch Löschen der Kante $(B, B') \in G^*_{mod}$ reduziert sich der Knotengrad von B und B' um eins, sodass nun entweder $\delta_{G^*_{mod}}(B) = 2$ oder $\delta_{G^*_{mod}}(B') = 2$ gilt.

4. Sei C^*_{neu} ein Weg in C^*, der die Kante (B, B') enthält. Setze $C^*_1 := C^*_{neu}$ und starte erneut mit 1.

Damit ist G^*_{mod} serienparallel. □

Korollar 6.7
Sei G ein Caterpillar-Halin-Graph. Dann können $\nu(G)$ und $\mathcal{Z}(G)$ in linearer Zeit bestimmt werden.

Ähnlich wie in Korollar 6.4(iii) ergibt sich bei Anwendung von Algorithmus 4.5 auch Korollar 6.8.

Korollar 6.8
Die Kreispackungszahl und eine maximale Kreispackung des inneren Dualgraphen G^*_{mod} eines Caterpillar-Halin-Graphen G können nach Satz 4.11 mit Algorithmus 4.5 in linearer Zeit bestimmt werden.

Für Caterpillar-Halin-Graphen G ist die Bestimmung von $\nu(G)$ durch die Bestimmung einer maximalen stabilen Menge von G^* also einfach. Liegt ein Caterpillar-Halin-Graph G vor, der die Eigenschaft besitzt, dass in der zugehörigen Zerlegung $\mathcal{G}$ sämtliche Splitkomponenten G_i isomorph zu geraden Wheels sind, so lässt sich $\nu(G)$ über die Kreispackungszahlen $\nu(G_i)$ bestimmen. Dies soll als nächstes nachgewiesen werden. Zunächst werden einige Eigenschaften, die bereits für allgemeine Halin-Graphen gezeigt wurden, für Caterpillar-Halin-Graphen spezifiziert. Mit Lemma 5.14 wurde festgestellt, dass die Fanreduktion $G \times F$ eines Halin-Graphen G selbst wieder ein Halin-Graph ist. Das folgende Lemma zeigt, dass speziell für Caterpillar-Halin-Graphen G der entstehende Halin-Graph $G \times F$ selbst wieder ein Caterpillar-Halin-Graph ist.

Lemma 6.9
Sei $G = T \cup C$ ein Caterpillar-Halin-Graph mit zugehörigem Weg $P = (v_1, \ldots, v_k)$.

1. Dann besitzt G genau zwei Fans F_1 und F_k.

2. Für $i \in \{1, k\}$ ist $G'_i = G \times F_i$ ebenfalls ein Caterpillar-Halin-Graph mit Darstellung $G'_i = T'_i \cup C'_i$ und $T'_i = T \backslash C(v_i)$.

3. Ist P'_i der zu T' gehörige Weg und $\delta_G(v)$ eine gerade Zahl für alle $v \in V(P)$, so ist $\delta_{G'_i}(v)$ ebenfalls eine gerade Zahl für alle $v \in V(P'_i)$.

Beweis: Zu 1: Es sei $P = (v_1, \ldots, v_k)$ der zu T gehörige Weg. Da $k \geqslant 2$ enthält P genau zwei Blätter v_1 und v_k. Da P ein Weg ist, ist $v_1 \in V(T)$ zu genau einem Knoten $v_2 \in V(T)$, der kein Blatt ist, adjazent. Ebenso ist $v_k \in V(T)$ zu genau einem Knoten $v_{k-1} \in V(T)$, der kein Blatt in T ist, adjazent. Damit gibt es genau zwei Fans $F_1 = \{v_1\} \cup C(v_1)$ und $F_k = \{v_k\} \cup C(v_k)$.

Zu 2: Nach Lemma 5.14 ist $G_i' = G \times F_i$ für $i \in \{1, k\}$ ein Halin-Graph und hat damit eine Darstellung $G_i' = T_i' \cup C_i'$. Für $V(F_i) = \{v_i\} \cup C(v_i)$ gilt $T_i' = T \backslash C(v_i)$. Da T Caterpillar ist, folgt dann direkt T_i' Caterpillar und $P_i' = P \backslash \{v_i\}$. Damit ist G_i' also selbst wieder ein Caterpillar-Halin-Graph.

Zu 3: Folgt sofort aus $V(P_i') \subset V(P)$. □

Lemma 6.10
Sei $G = T \cup C$ ein Caterpillar-Halin-Graph mit zugehörigem Weg $P = (v_1, \ldots, v_k)$. Es sei $\mathcal{T}(G) = (M, A)$ der zu der Zerlegung $\mathcal{G}$ von G gehörige W-Baum. Dann gilt

1. $\mathcal{T}(G)$ ist ein Weg mit k Knoten,

2. für $P = (v_1, \ldots, v_k)$ und $\mathcal{G} = \{G_{\mu_1}, \ldots, G_{\mu_k}\}$ ist der Knoten v_i Zentrum des Wheels G_{μ_i} und $\delta_G(v_i) = \delta_{G_{\mu_i}}(v_i)$ für alle $i = 1, \ldots, k$ und

3. jedes Skelett $G_\mu \in \mathcal{G}$ enthält maximal zwei virtuelle Knoten.

Beweis: Der Beweis von Aussage 1 und 2 erfolgt durch Induktion über die Anzahl $k \geqslant 2$ von Knoten in P.

Sei $k = 2$. Nach Lemma 6.9 besitzt G genau zwei Fans. Sei $P = (v_1, v_2)$. Dann sind die Knoten v_1 und v_2 jeweils Zentrum dieser Fans. Sei F_2 der durch $v_2 \cup C(v_2)$ induzierte Fan. Nach Lemma 5.16 induziert F_2 einen guten Split. Nach Lemma 6.9 ist $G' = G \times F_2$ ebenfalls ein Caterpillar-Halin-Graph mit $|P'| = 1$, also ein Wheel. Da ein Wheel keinen guten Split besitzt, ist $M = \{\mu_1, \mu_2\}$ und $A = \{(\mu_1, \mu_2)\}$, das heißt $\mathcal{T}(G)$ ist ein Weg mit zwei Knoten. Also gilt Aussage 1.

Darüber hinaus ist $\mathcal{G} = \{G_{\mu_1}, G_{\mu_2}\}$, wobei $F_1 \subset G_{\mu_1} = G'$ und $F_2 \subset G_{\mu_2}$ ist. Nach

Korollar 5.18 ist v_i dann auch Zentrum von G_{μ_i} für $i = 1, 2$. Außerdem gilt $\delta_G(v_i) = \delta_{G_{\mu_i}}(v_i)$ für $i = 1, 2$, da nach Korollar 5.17 in der Konstruktion der einfachen 3-Zerlegung von Halin-Graphen die G_{μ_i} keine virtuellen Kanten enthalten, das heißt es gilt Aussage 2.

Sei $k \geqslant 3$ und es sei angenommen die Aussagen 1 und 2 gelten für jeden Caterpillar-Halin-Graphen $G' = T' \cup C'$ mit $P' = (v_1, \ldots, v_{k'})$, $k' \leqslant k-1$. Betrachte nun einen Caterpillar-Halin-Graphen $G = T \cup C$ mit $P = (v_1, \ldots, v_k)$. G besitzt nach Lemma 6.9 genau zwei Fans. Die Knoten v_1 und v_k sind jeweils Zentrum eines Fans. Sei F_k der durch $v_k \cup C(v_k)$ induzierte Fan. Nach Lemma 5.16 induziert F_k einen guten Split. Dann gibt es nach Korollar 5.17 einen Knoten $\mu_k \in V(\mathcal{T}(G))$ mit $F_k \subset G_{\mu_k}$. Die zugehörige einfache 3-Separation ist dann $\{G \times F_k, G_{\mu_k}\}$. Setze $G' := G \times F_k$. Nach Lemma 6.9 ist G' ebenfalls ein Caterpillar-Halin-Graph mit $|P'| = k-1$. Für G' ist $\mathcal{T}(G') = (M', A')$ ein Weg mit $k-1$ Knoten und die Zerlegung $\mathcal{G}'$ ist gegeben durch $\mathcal{G}' = \{G'_{\mu_1}, \ldots, G'_{\mu_{k-1}}\}$. Weiterhin ist v_i Zentrum von G'_{μ_i} und $\delta_{G'}(v_i) = \delta_{G'_{\mu_i}}(v_i)$ für $i = 1, \ldots, k-1$. Nach Korollar 5.18 ist $V(\mathcal{T}(G)) := V(\mathcal{T}(G')) \cup \{\mu_k\}$. Angenommen $\mathcal{T}(G)$ ist kein Weg mit k Knoten. Da $\mathcal{T}(G)$ aber ein Baum ist, muss dann $(\mu_k, \mu_i) \in A(\mathcal{T}(G))$ für $i \in \{2, \ldots, k-2\}$ gelten. Da $\mathcal{T}(G')$ ein Weg mit $k-1$ Knoten ist und damit genau zwei Blätter μ_1 und μ_{k-1} besitzt, muss $\mathcal{T}(G)$ genau drei Blätter besitzen. Da $\mathcal{G}$ eindeutig ist und jeder Fan $F \subset G$ einen guten Split induziert, muss G drei solcher Fans enthalten. Das ist ein Widerspruch zur Annahme, dass G ein Caterpillar-Halin-Graph ist. Also gilt Aussage 1.

Ohne Beschränkung der Allgemeinheit sei $A(\mathcal{T}(G)) := A(\mathcal{T}(G')) \cup \{(\mu_{k-1}, \mu_k)\}$. Damit ist $\mathcal{T}(G) = (M, A)$ ein Weg mit k Knoten. Setze $G_{\mu_i} := G'_{\mu_i}$ für $i = 1, \ldots, k-1$. Dann ist $\mathcal{G} = \{G_{\mu_1}, \ldots, G_{\mu_k}\}$ mit v_i Zentrum von G_{μ_i} und $\delta_G(v_i) = \delta_{G_{\mu_i}}(v_i)$ für $i = 1, \ldots, k-1$. Weiterhin ist v_k das Zentrum von F_k und nach Korollar 5.18 ebenfalls das Zentrum von G_{μ_k}. Da G_{μ_k} nach Korollar 5.17 keine virtuelle Kante enthält, gilt $\delta_G(v_k) = \delta_{G_{\mu_k}}(v_k)$ und damit Aussage 2.

Die Aussage 3 folgt mit der Eigenschaft von $\mathcal{T}$ direkt aus Aussage 1. □

Da nun für einen Caterpillar-Halin-Graphen G und seine Zerlegung $\mathcal{G}$ der Knotengrad des Zentrums v_i einer Splitkomponente G_{μ_i} identisch zum Knotengrad des Knotens v_i in G ist, können nun Caterpillar-Halin-Graphen, die in gerade Wheels zerlegt werden können, spezifiziert werden.

Korollar 6.11
Sei $G = T \cup C$ ein Caterpillar-Halin-Graph und $\mathcal{T}(G) = (M, A)$ der zu der Zerlegung $\mathcal{G}$ von G zugehörige W-Baum. Genau dann, wenn $\delta_G(v)$ eine gerade Zahl für alle $v \in V(P)$ ist, ist G_{μ_i} ein gerades Wheel für alle $i = 1, \ldots, k$.

Beweis: Folgt direkt aus Lemma 6.10 Teil 2. □

Ein Caterpillar-Halin-Graph G, für den $\delta_G(v)$ eine gerade Zahl für alle $v \in V(P)$ ist, ist in Abbildung 6.8 illustriert.

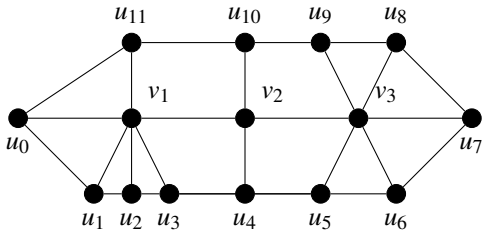


Abbildung 6.8: Beispiel eines Caterpillar-Halin-Graphen mit $\delta_G(v)$ gerade für alle $v \in V(P)$.

Für solche Graphen werden nun Aussagen zur Kreispackungszahl hergeleitet. Dafür wird die Existenz spezieller maximaler Kreispackungen in Halin-Graphen benötigt.

Lemma 6.12 [Ste21]
Sei G ein Halin-Graph zusammen mit einer planaren Darstellung. Sei $\mathcal{B}(G)$ die Menge der inneren Facetten von G. Dann existiert eine maximale kantendisjunkte Kreispackung $\mathcal{Z}^* = \{C_1, \ldots, C_{\nu(G)}\}$, in welcher jeder Kreis C_i eine Facette $B_i \in \mathcal{B}(G)$ induziert, $i = 1, \ldots, \nu(G)$.

Ein solcher Facetten-induzierender Kreis C_i wird *Facettenkreis* genannt. Zur besseren Lesbarkeit wird im Folgenden bei der Schreibweise nicht zwischen Facetten und Facettenkreisen unterschieden. Abhängig vom Kontext soll hervorgehen, ob mit der Bezeichnung B_i jeweils die Facette oder der Facettenkreis gemeint ist.

Satz 6.13
Sei $G = T \cup C$ ein Caterpillar-Halin-Graph mit zugehörigem Weg P und $\mathcal{T}(G) = (M, A)$

mit $M = \{\mu_1, \ldots, \mu_N\}$ der zu der Zerlegung $\mathcal{G} = \{G_{\mu_1}, \ldots, G_{\mu_N}\}$ von G zugehörige W-Baum. Weiterhin sei $\delta_G(v)$ eine gerade Zahl für alle $v \in V(P)$. Dann gilt

$$\nu(G) = \sum_{\mu \in \mathcal{T}(G)} \nu(G_\mu) - (N-1).$$

Beweis: Sei $v_i \in V(P)$ das Zentrum des Wheels $G_{\mu_i} \in \mathcal{G}$. Da $\delta_G(v_i)$ eine gerade Zahl ist, ist G_{μ_i} ein gerades Wheel (Korollar 6.11). Der weitere Beweis erfolgt durch Induktion über N.

Sei $N = 2$. Dann ist $\mathcal{G} = \{G_{\mu_1}, G_{\mu_2}\}$ und $\mathcal{T} = (M, A)$ mit $M = \{\mu_1, \mu_2\}$ und $A = \{(\mu_1, \mu_2)\}$. und $\nu(G) = \nu(G_{\mu_1}) - 0$. Da G_{μ_1} und G_{μ_2} keine virtuellen Kanten enthalten, entspricht jede innere Facette $\bar{B} \in \mathcal{B}(G_{\mu_1})$ und $\tilde{B} \in \mathcal{B}(G_{\mu_2})$ einer inneren Facette $B \in \mathcal{B}(G)$. Darüber hinaus gibt es zwei benachbarte Paare von Facetten $\bar{B}^{(1)}, \bar{B}^{(2)} \subset G_{\mu_1}$ und $\tilde{B}^{(1)}, \tilde{B}^{(2)} \subset G_{\mu_2}$, sodass sie jeweils denselben benachbarten Facetten $B^{(1)}, B^{(2)} \in G$ entsprechen. G_{μ_1} und G_{μ_2} sind gerade Wheels. Seien $\mathcal{Z}^*(G_{\mu_1})$ und $\mathcal{Z}^*(G_{\mu_2})$ maximale Kreispackungen von G_{μ_1} und G_{μ_2} mit $\bar{B}^{(1)} \in \mathcal{Z}^*(G_{\mu_1})$ und $\tilde{B}^{(1)} \in \mathcal{Z}^*(G_{\mu_2})$. Dann induziert $\mathcal{Z}^*(G_{\mu_1}) \cup \mathcal{Z}^*(G_{\mu_2})$ eine maximale Kreispackung $\mathcal{Z}^*(G)$ und $\nu(G) = \nu(G_{\mu_1}) + \nu(G_{\mu_2}) - 1$.

Sei $N \geqslant 3$ und es sei angenommen, dass die Aussage für alle Caterpillar-Halin-Graphen $G' = T' \cup C'$ mit $\mathcal{T}(G') = (M', A')$ gilt, die die Eigenschaft $|M'| = N - 1$ haben. Das heißt, für einen solchen Graphen G' gilt $\nu(G') = \sum_{\mu \in \mathcal{T}(G')} \nu(G_\mu) - (N-2)$.

Sei nun $G = T \cup C$ ein Caterpillar-Halin-Graph mit $\mathcal{T}(G) = (M, A)$ mit $|M| = N$. Sei F_N der durch $\{v_N\} \cup C(v_N)$ induzierte Fan. Nach Lemma 5.16 induziert F_N einen guten Split. Dann gibt es nach Korollar 5.17 einen Knoten $\mu_N \in V(\mathcal{T}(G))$ mit $F_N \subset G_{\mu_N}$. Die zugehörige einfache 3-Separation ist dann $\{G \times F_N, G_{\mu_N}\}$. Setze $G' := G \times F_N$. Nach Lemma 6.9 ist G' ebenfalls ein Caterpillar-Halin-Graph mit $|P'| = N - 1$. Für G' ist $\mathcal{T}(G') = (M', A')$ ein Weg mit $N - 1$ Knoten und die Zerlegung $\mathcal{G}'$ ist gegeben durch $\mathcal{G}' = \{G'_{\mu_1}, \ldots, G'_{\mu_{N-1}}\}$. Weiterhin ist v_i Zentrum von G'_{μ_i} und $\delta_{G'}(v_i) = \delta_{G'_{\mu_i}}(v_i)$ für $i = 1, \ldots, k-1$. Nach Korollar 5.18 ist $V(\mathcal{T}(G)) := V(\mathcal{T}(G')) \cup \{\mu_N\}$. Nach Lemma 6.10 ist $A(\mathcal{T}(G)) = A(\mathcal{T}(G')) \cup \{(\mu_{N-1}, \mu_N)\}$ und $G'_{\mu_i} = G_{\mu_i}$ für $i = 1, \ldots, N-1$. Betrachte nun die Graphen G' und G_{μ_N} (siehe auch Abbildung 6.9(a)-6.9(c)). Die drei schematischen Darstellungen in Abbildung 6.9 illustrieren die drei verschiedenen Konstellationen der beiden zu F_N benachbarten (in der Abbildung grau hinterlegten) Facetten.

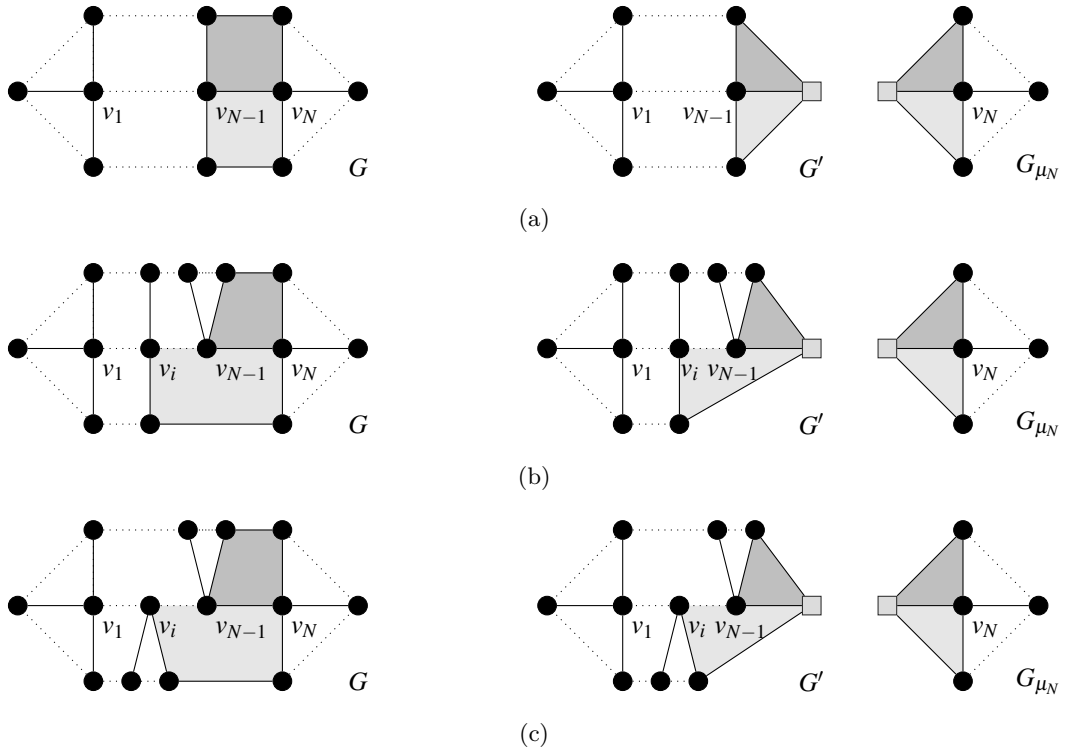


Abbildung 6.9: Schematische Darstellung der drei Varianten eines Caterpillar-Halin-Graph G mit $P = (v_1, \ldots, v_N)$ und der jeweiligen einfachen 3-Separation $\{G \times F_N, G_{\mu_N}\}$.

Da G' und G_{μ_N} keine virtuellen Kanten enthalten, entspricht jede innere Facette $\bar{B} \in \mathcal{B}(G')$ und $\tilde{B} \in \mathcal{B}(G_{\mu_N})$ einer Facette $B \in \mathcal{B}(G)$. Darüber hinaus gibt es zwei benachbarte Paare von Facetten $\bar{B}^{(1)}, \bar{B}^{(2)} \subset G'$ und $\tilde{B}^{(1)}, \tilde{B}^{(2)} \subset G_{\mu_N}$, sodass sie jeweils denselben benachbarten Facetten $B^{(1)}, B^{(2)} \in G$ entsprechen (vgl. mit den grau hinterlegten Facetten in Abbildung 6.9). Weiterhin existiert nach Lemma 6.12 für Halin-Graphen eine maximale Kreispackung, in welcher jeder Kreis eine innere Facette induziert. Sei $\mathcal{Z}_G(G')$ bzw. $\mathcal{Z}_G(G_{\mu_N})$ die von $\mathcal{Z}(G')$ bzw. $\mathcal{Z}(G_{\mu_N})$ induzierte Kreispackung in G. Sei nun $\mathcal{Z}^*(G')$ bzw. $\mathcal{Z}^*(G_{\mu_N})$ eine maximale Kreispackung von G' bzw. G_{μ_N}. Zu zeigen ist nun:

(i) Es existiert genau ein Kreis $\hat{B} \in \mathcal{Z}^*_G(G') \cap \mathcal{Z}^*_G(G_{\mu_N})$ und

(ii) $\mathcal{Z}^*(G) := \mathcal{Z}^*_G(G') \cup \mathcal{Z}^*_G(G_{\mu_N})$ ist eine maximale Kreispackung von G mit $|\mathcal{Z}^*(G)| = |\mathcal{Z}^*(G')| + |\mathcal{Z}^*(G_{\mu_N})| - 1$.

Zu (i): Zunächst kann immer eine maximale Kreispackung $\mathcal{Z}^*_(G')$ mit $\bar{B}^{(1)} \in \mathcal{Z}^*(G')$

oder $\bar{B}^{(2)} \in \mathcal{Z}^*(G')$ gefunden werden: Falls $\bar{B}^{(j)} \notin \mathcal{Z}^*(G')$ für $j = 1, 2$ ist, müssen die benachbarten Facetten von $\bar{B}^{(j)}$ in $\mathcal{Z}^*(G')$ enthalten sein (ansonsten wäre $\mathcal{Z}^*(G')$ nicht maximal). In einem solchen Fall ersetze eine der benachbarten Facetten in der maximalen Kreispackung durch die Facette $\bar{B}^{(1)}$ bzw. $\bar{B}^{(2)}$. Falls $\bar{B}^{(1)} \in \mathcal{Z}^*(G')$ gilt, kann ohne Beschränkung der Allgemeinheit gefordert werden, dass $\tilde{B}^{(1)} \in \mathcal{Z}^*(G_{\mu_N})$ ist, da G_{μ_N} ein gerades Wheel ist. Das gleiche gilt, falls $\bar{B}^{(2)} \in \mathcal{Z}^*(G')$. Dann existiert genau ein Kreis $\hat{B} \in \mathcal{Z}_G^*(G') \cap \mathcal{Z}_G^*(G_{\mu_N})$.

Zu (ii): Offensichtlich ist $\mathcal{Z}^*(G)$ eine kantendisjunkte Kreispackung aus Facettenkreisen von G. Angenommen $\mathcal{Z}^*(G)$ ist nicht maximal, dann gibt es eine Kreispackung $\hat{\mathcal{Z}}^*(G)$ mit $|\hat{\mathcal{Z}}^*(G)| = |\mathcal{Z}^*(G)| + 1$. Sei $\hat{\mathcal{Z}}^*(G) = \{B_1, \ldots, B_{|\mathcal{Z}^*(G)|+1}\}$. Jede dieser Facetten B_i wird mindestens einmal durch eine Facette $\bar{B}_{\bar{i}} \in \mathcal{B}(G')$ oder $\tilde{B}_{\tilde{i}} \in \mathcal{B}(G_{\mu_N})$ repräsentiert. Damit induziert die Kreispackung $\hat{\mathcal{Z}}^*(G)$ Kreispackungen $\hat{\mathcal{Z}}^*(G')$ und $\hat{\mathcal{Z}}^*(G_{\mu_N})$ und es muss gelten $|\hat{\mathcal{Z}}^*(G')| > |\mathcal{Z}^*(G')|$ oder $|\hat{\mathcal{Z}}^*(G_{\mu_N})| > |\mathcal{Z}^*(G_{\mu_N})|$. Dies ist ein Widerspruch zur Maximalität von $\mathcal{Z}^*(G')$ und $\mathcal{Z}^*(G_{\mu_N})$. Damit ist $\mathcal{Z}^*(G) := Z_G^*(G') \cup Z_G^*(G_{\mu_N})$ eine maximale Kreispackung von G und $|\mathcal{Z}^*(G)| = |\mathcal{Z}^*(G')| + |\mathcal{Z}^*(G_{\mu_N})| - 1$.

Insgesamt ergibt sich damit

$$\begin{aligned}\nu(G) &= \nu(G') + \nu(G_{\mu_N}) - 1 = \sum_{\mu \in \mathcal{T}(G')} \nu(G_\mu) - (N-2) + \nu(G_{\mu_N}) - 1 \\ &= \sum_{\mu \in \mathcal{T}(G)} \nu(G_\mu) - (N-1).\end{aligned}$$ □

Bemerkung 6.14

Für die in Satz 6.13 betrachteten Graphen kann zusammen mit der Kreispackungszahl $\nu(G)$ auch die zugehörige maximale Kreispackung $\mathcal{Z}^*(G)$ angegeben werden, da der Beweis von Satz 6.13 konstruktiv ist. Sei wie im dortigen Beweis $\mathcal{Z}^*(G')$ bzw. $\mathcal{Z}^*(G_{\mu_N})$ eine maximale Kreispackung von G' bzw. G_{μ_N}.

Fall 1: Es existiert genau ein Kreis $\hat{B} \in \mathcal{Z}_G^*(G') \cap \mathcal{Z}_G^*(G_{\mu_N})$. Dann setze $\mathcal{Z}^*(G) := \mathcal{Z}_G^*(G') \cup \mathcal{Z}_G^*(G_{\mu_N})$.

Fall 2: Es existieren jeweils ein Kreis $\bar{B}_i \in \mathcal{Z}_G^*(G')$ und ein Kreis $\tilde{B}_i \in \mathcal{Z}_G^*(G_{\mu_N})$ mit $E(\bar{B}_i) \cap E(\tilde{B}_i) = \{v_{N-1}, v_N\}$. Dann setze $\mathcal{Z}^*(G_{\mu_N}) := \mathcal{B}(G_{\mu_N}) \backslash \mathcal{Z}^*(G_{\mu_N})$ und $\mathcal{Z}^*(G) := \mathcal{Z}_G^*(G') \cup \mathcal{Z}_G^*(G_{\mu_N})$.

Fall 3: Fall 2 gilt nicht und es existieren jeweils ein Kreis $\bar{B}_i \in \mathcal{Z}^*_G(G')$ und $\tilde{B}_i \in \mathcal{Z}^*_G(G_{\mu_N})$ mit $E(\bar{B}_i) \cap E(\tilde{B}_i) \neq \emptyset$. Dann setze $\mathcal{Z}^*(G) := \mathcal{Z}^*_G(G') \backslash \{\bar{B}_i\} \cup \mathcal{Z}^*_G(G_{\mu_N})$.

Algorithmus 6.1 beschreibt die Konstruktion einer maximalen Kreispackung und die Berechnung der Kreispackungszahl für Caterpillar-Halin-Graphen G (sofern $\delta_G(v)$ eine gerade Zahl ist für alle $v \in V(P)$) entsprechend Satz 6.13 und Bemerkung 6.14. Die Laufzeit des Algorithmus ist $O(|E(G)|^2 + |V(G)|)$ und wird daher von der Laufzeit von Algorithmus 5.2 zur Bestimmung der Zerlegung dominiert, welcher $O(|V(G)|^2|E(G)|)$ Zeit benötigt.

Algorithmus 6.1 Kreispackungsalgorithmus entsprechend Satz 6.13

Eingabe: Zerlegung $\mathcal{G} = \{G_{\mu_1}, \ldots, G_{\mu_N}\}$ von G und zugehöriger W-Baum $\mathcal{T}(G) = (M, A)$.
Ausgabe: Maximale Kreispackung $\mathcal{Z}^*(G)$ und Kreispackungszahl $\nu(G)$.

$\mathcal{Z}^*(G) \leftarrow \emptyset$, $\nu(G) \leftarrow 0$
Wähle ein Blatt $\mu \in \mathcal{T}$.
while $\mu \neq \emptyset$ **do**
    $\nu(G_\mu) \leftarrow \frac{\delta_{G_\mu}(v)}{2}$ für v Zentrum von G_μ
    $\nu(G) \leftarrow \nu(G) + \nu(G_\mu)$
    Bestimme $\mathcal{Z}^*(G_\mu)$.
    if $\mathcal{Z}^*(G) == \emptyset$ oder $\mathcal{Z}^*(G) \cap \mathcal{Z}^*(G_\mu) \neq \emptyset$ **then**
        $\mathcal{Z}^*(G) \leftarrow \mathcal{Z}^*(G) \cup \mathcal{Z}^*(G_\mu)$
    else
        $\mathcal{Z}^*(G_\mu) \leftarrow \mathcal{B}(G_\mu) \backslash \mathcal{Z}^*(G_\mu)$
        $\mathcal{Z}^*(G) \leftarrow \mathcal{Z}^*(G) \cup \mathcal{Z}^*(G_\mu)$
    end if
    if $\exists\ \mu'$ mit $\{\mu, \mu'\} \in \mathcal{T}$ **then**
        $\mathcal{T} \leftarrow \mathcal{T} \backslash \mu$
        $\mu \leftarrow \mu'$
    else
        $\mu \leftarrow \emptyset$
    end if
end while
$\nu(G) \leftarrow \nu(G) - (N - 1)$
return $\mathcal{Z}^*(G)$ und $\nu(G)$

Das nachfolgende Beispiel zeigt, dass die Bedingung $\delta_G(v) \equiv 0 \pmod 2$ für alle $v \in V(P)$ essentiell ist.

Beispiel 6.15

Die Knoten v_1 und v_4 des in der Abbildung 6.10 dargestellten Caterpillar-Halin-Graphen haben einen ungeraden Knotengrad. Die Wheels G'_1 und G'_4 seiner Zerlegung $\mathcal{G} = \{G_1, \ldots, G_4\}$ sind dementsprechend ungerade. Es gilt $\nu(G) = 4$, aber $\sum_{\mu \in \mathcal{T}(G)} \nu(G_\mu) - (N - 1) = 3$.

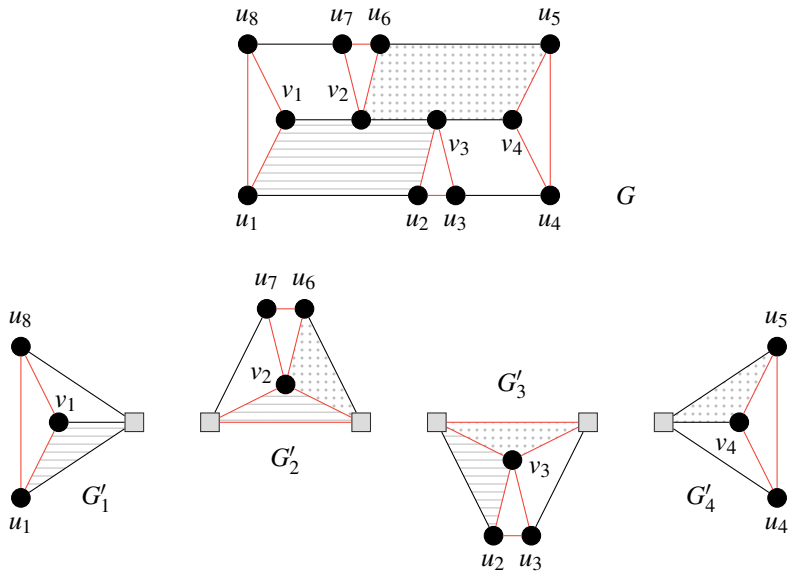


Abbildung 6.10: Caterpillar-Halin-Graph G mit zugehöriger Zerlegung $\mathcal{G} = \{G'_1, \ldots, G'_4\}$, wobei G'_1 und G'_4 ungerade Wheels sind.

Beispiel 6.15 verdeutlicht zudem, dass offenbar Situationen auftreten können, in denen bestimmte Facetten eines Caterpillar-Halin-Graphen G (siehe schraffierte bzw. gepunktete Facetten) in mehr als zwei Wheels der Zerlegung $\mathcal{G}$ vorkommen können (hier in G'_1, G'_2, G'_3 bzw. in G'_2, G'_3, G'_4). Ein Verfahren, welches derartige Situationen mit berücksichtigt, soll als nächstes angegeben werden.

6.2 Ein Algorithmus zur Bestimmung von Kreispackungen in Halin-Graphen

Zunächst werden die Beziehungen zwischen den Facetten eines Halin-Graphen G und den Facetten einer Zerlegung $\mathcal{G}$ von G betrachtet.

Bemerkung 6.16

Sei $G = T \cup C$ ein Halin-Graph und $\mathcal{G} = \{G_{\mu_1}, \ldots, G_{\mu_N}\}$ die Zerlegung von G. Sei weiterhin $\mathcal{B}$ die Menge der Facetten von G. Dann repräsentiert jede Facette $\tilde{B}$ eines Wheels G_{μ_i} eine Facette $B \in \mathcal{B}$ von G und umgekehrt wird jede Facette $B \in \mathcal{B}$ mindestens einmal durch eine Facette $\tilde{B}$ in mindestens einem Wheel G_{μ_i} repräsentiert.

Wie das Beispiel 6.15 illustriert, kann es allerdings vorkommen, dass eine Facette $B \subset G$ in mehreren Splitkomponenten als Facette repräsentiert wird. Für einen Halin-Graphen $G = T \cup C$ mit Zerlegung $\mathcal{G} = \{G_{\mu_1}, \ldots, G_{\mu_N}\}$ heißt $\tilde{B} \subset G_{\mu_i}$ eine (zu x_{μ_i} gehörende) *Kontaktfacette* von G_{μ_i}, falls $x_{\mu_i} \in V(\tilde{B})$ gilt. Für solche Kontaktfacetten kann Folgendes festgestellt werden.

Lemma 6.17
Es sei $G = T \cup C$ ein Halin-Graph, $S = (S_1, S_2; S_3)$ ein guter 3-Split in G und $B_i \in \mathcal{B}(G)$ mit $S_3 \cap E(B_i) \neq \varnothing$ für $i = 1, 2$. Dann gibt es in $\mathcal{G}$ zwei Komponenten G_{μ_1}, G_{μ_2} mit dem virtuellen Knoten $x_{(\mu_1,\mu_2)}$ derart, dass B_1, B_2 in G_{μ_1} durch die zu $x_{(\mu_1,\mu_2)}$ gehörenden Kontaktfacetten $\tilde{B}_1, \tilde{B}_2$ und in G_{μ_2} durch die zu $x_{(\mu_1,\mu_2)}$ gehörenden Kontaktfacetten $\tilde{B}'_1, \tilde{B}'_2$ repräsentiert werden.

Sind umgekehrt $G_{\mu_1}, G_{\mu_2} \in \mathcal{G}$ mit dem virtuellen Knoten $x_{(\mu_1,\mu_2)}$, dann repräsentieren die zu $x_{(\mu_1,\mu_2)}$ in G_{μ_1} gehörenden Kontaktfacetten $\tilde{B}_1, \tilde{B}_2$ bzw. die zu $x_{(\mu_1,\mu_2)}$ in G_{μ_2} gehörenden Kontaktfacetten $\tilde{B}'_1, \tilde{B}'_2$ dieselben zwei Facetten B_1, B_2 in G.

Beweis: Ein guter 3-Split $S = (S_1, S_2; S_3)$ in G induziert eine einfache 3-Zerlegung $\{G_1, G_2\}$ von G. Entsprechend der Definition einer einfachen 3-Zerlegung besitzen G_1, G_2 einen virtuellen Knoten x. Da $\mathcal{G}$ wie in Abschnitt 5.1 beschrieben Ergebnis eines Splitprozesses mit guten 3-Splits ist, gibt es zwei Splitkomponenten G_{μ_1}, G_{μ_2} mit dem virtuellen Knoten $x_{(\mu_1,\mu_2)} := x$. Da G ein Halin-Graph ist, gilt $S_3 = \{e_1, e_2, e_3\}$ mit $e_1, e_2 \in C$ und $e_3 \in T$. Aufgrund der einfachen 3-Zerlegung werden die Facetten $B_i \in \mathcal{B}(G)$, für die $S_3 \cap E(B_i) \neq \varnothing$ für $i = 1, 2$ ist, in G_{μ_1} durch die zu $x_{(\mu_1,\mu_2)}$ gehörenden Kontaktfacetten $\tilde{B}_1, \tilde{B}_2$ und in G_{μ_2} durch die zu $x_{(\mu_1,\mu_2)}$ gehörenden Kontaktfacetten $\tilde{B}'_1, \tilde{B}'_2$ repräsentiert.

Ebenfalls aufgrund der einfachen 3-Zerlegung gilt: Sind umgekehrt $G_{\mu_1}, G_{\mu_2} \in \mathcal{G}$ mit dem virtuellen Knoten $x_{(\mu_1,\mu_2)}$, dann repräsentieren die zu $x_{(\mu_1,\mu_2)}$ in G_{μ_1} gehörenden Kontaktfacetten $\tilde{B}_1, \tilde{B}_2$ bzw. die zu $x_{(\mu_1,\mu_2)}$ in G_{μ_2} gehörenden Kontaktfacetten $\tilde{B}'_1, \tilde{B}'_2$ dieselben zwei Facetten B_1, B_2 in G. □

Die besondere Struktur eines Halin-Graphen kann nun genutzt werden, um $\mathcal{B}(G)$ geeignet zu nummerieren.

Lemma 6.18
Sei $G = T \cup C$ ein Halin-Graph und $\mathcal{B}$ die Menge der Facetten von G. Dann gilt $|C| = |\mathcal{B}| - 1$.

Beweis: Sei $B_a \in \mathcal{B}$ die äußere Facette von G. Da T ein Baum ist und jede Kante $e = (u, v) \in C$ zwei Blätter $u, v \in T$ verbindet, grenzt jede Facette $B \in \mathcal{B} \backslash B_a$ an genau eine Kante $e \in C$ und jede Kante $e \in C$ begrenzt genau eine Facette $B \in \mathcal{B} \backslash B_a$. Daher gilt $|C| = |\mathcal{B}| - 1$. □

Eine jede (innere) Facette B eines Halin-Graphen G kann nach Lemma 6.18 also eindeutig über seine zugehörige Kreiskante $e \in C \cap E(B)$ identifiziert werden. Im Folgenden sei daher der Kreis C durch $e_1, e_2, \ldots, e_{|\mathcal{B}|-1}$ so nummeriert, dass e_i, e_{i+1} (bzw. $e_{|\mathcal{B}|-1}$ und e_1) einen gemeinsamen Knoten besitzen. Die (inneren) Facetten seien entsprechend ihrer Kreiskante e_j mit B_j nummeriert. Für $\mu \in M$ korrespondieren die einzelnen Facetten $\tilde{B}$ von G_μ jeweils mit einer Facette B von G. Die Nummerierung in $\mathcal{B}$ induziert daher eine Nummerierung der Facetten in G_μ. Diese induzierte Nummierung soll im Folgenden verwendet werden, wenn einzelne Facetten in G_μ betrachtet werden. Abbildung 6.11 veranschautlicht eine solche Nummerierung.

Grundlage des Algorithmus ist die Darstellung des Halin-Graphen G als $G = T \cup C$, dessen Zerlegung $\mathcal{G} = \{G_{\mu_1}, \ldots, G_{\mu_N}\}$ sowie der zugehörige W-Baum $\mathcal{T} = (M, A)$. In $\mathcal{T}$ werden sukzessive Blätter bearbeitet und nach ihrer Bearbeitung aus $\mathcal{T}$ gelöscht. Dies erfolgt so lange, bis alle Knoten einmal bearbeitet wurden. In jedem Iterationsschritt wird das zum Knoten $\mu \in M$ gehörende Wheel G_μ betrachtet und als Ergebnis einer solchen Betrachtung eine bestehende kantendisjunkte Kreispackung $\mathcal{Z}$ gegebenenfalls vergrößert. Am Ende des Verfahrens liegt dann eine Kreispackung $\mathcal{Z}$ von G der Kardinalität $|\mathcal{Z}|$ vor. Blätter μ mit ungeradem Wheel G_μ werden priorisiert bearbeitet.

Für die Bearbeitung eines Knotens $\mu \in M$ werden folgende Informationen benötigt. Der Vektor $l = (\ell_1, \ldots, \ell_{|\mathcal{B}|-1}) \in \mathbb{N}^{|\mathcal{B}|-1}$ liefert Informationen, welche Facetten in G für eine Kreispackung $\mathcal{Z}$ (noch) in Frage kommen können. Solange $\ell_j > 0$ ist, ist B_j ein Kandidat für die *Vergrößerung* einer bestehenden Kreispackung $\mathcal{Z}$. Falls $\ell_j = 0$ ist, so kommt B_j dafür nicht mehr in Frage. Zu Beginn des Verfahrens entspricht ℓ_j der Anzahl derjenigen Komponenten $G_\mu \in \mathcal{G}$, die $\tilde{B}_j$ enthalten. Zu Beginn des Verfahrens gilt $\mathcal{Z} = \varnothing$. Bei der Bearbeitung eines Blattes μ werden drei spezielle, von G_μ abhängige, Kreispackungen $\bar{Z}, Z_{j_1}, Z_{j_2}$ bestimmt. Aufgrund dieser Kreispackungen wird entschieden, ob bestimmte

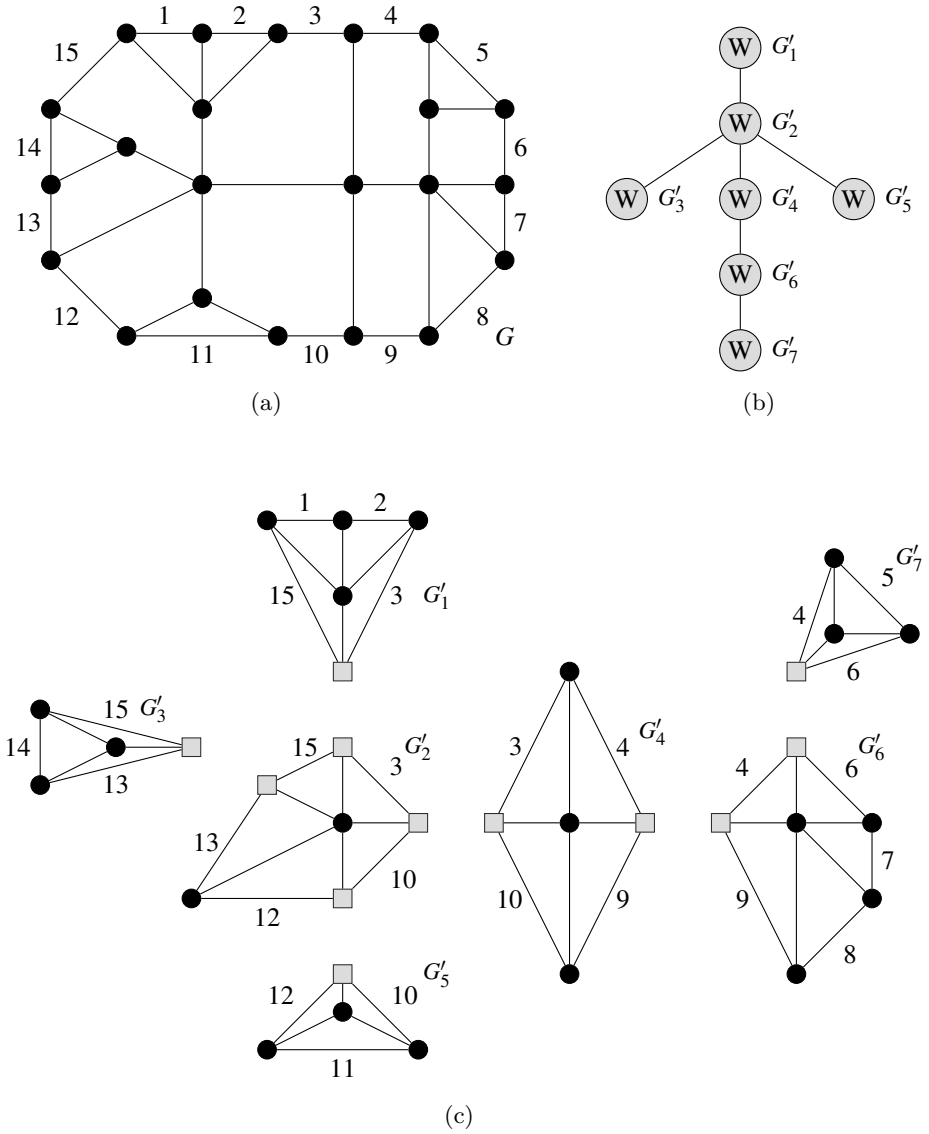


Abbildung 6.11: Halin-Graph G mit Nummerierung der Kanten $e \in C$ durch $\{1, 2, \ldots, 15\}$ (a), die zugehörige Zerlegung $\mathcal{G} = \{G_{\mu_1}, \ldots, G_{\mu_7}\}$ mit Nummerierung der Kanten $\tilde{e} \in C_{\mu_i}$ (c) und der zugehörige W-Baum (b).

Kreise aus diesen Kreispackungen unmittelbar (d.h. in der aktuellen k-ten Iteration, in der μ bearbeitet wird) zur bestehenden Kreispackung $\mathcal{Z}$ hinzugefügt werden können oder das Hinzufügen erst später erfolgen kann. Im ersten Fall wird $\mathcal{Z}$ unmittelbar um diese Kreise erweitert. Falls Kreise aus diesen drei Kreispackungen nicht unmittelbar zur Vergrößerung von $\mathcal{Z}$ beitragen können, werden sie für eine potentielle Vergrößerung in einer späteren Phase des Verfahrens gespeichert. Dazu wird in der k-ten Iteration

($k \leqslant N$) ein Paar $L_k = \{\bar{Z}_{j_1}, \bar{Z}_{j_2}\}$ benutzt. $\bar{Z}_{j_1}$ und $\bar{Z}_{j_2}$ enthalten die Information, welche Kreise für eine potentielle Vergrößerung in einer späteren Phase des Verfahrens in Frage kommen. Die Indizes j_1, j_2 beschreiben die Nummern zweier Kontaktfacetten $\tilde{B}_{j_1}, \tilde{B}_{j_2} \in G_\mu$. Zu Beginn wird die Menge $\mathcal{L}$ aller Paare L_k mit $\mathcal{L} = \varnothing$ initialisiert. Immer dann, wenn $\mathcal{Z}$ unmittelbar zum Zeitpunkt k (bei der Bearbeitung von μ) vergrößert wird, muss geprüft werden, ob auch in früheren Phasen des Verfahrens in $\mathcal{L}$ gespeicherte Kreispackungen ebenfalls zur Vergrößerung von $\mathcal{Z}$ beitragen können und der Menge $\mathcal{L}$ entnommen werden können.

Um dieses Vorgehen detaillierter zu beschreiben, betrachten wir nun den k-ten Schritt im Verfahren ($k \leqslant N$). Dann sei μ ein Blatt in $\mathcal{T}$. Also existiert ein eindeutiger Knoten $\mu' \in M$, sodass $(\mu, \mu') \in A$. G_μ enthält exakt einen virtuellen Knoten $x_{(\mu,\mu')}$ und mit $\tilde{B}_{j_1}, \tilde{B}_{j_2}$ die beiden zu $x_{(\mu,\mu')}$ gehörenden Kontaktfacetten. Bestimme nun für die drei Graphen

$$\begin{aligned} \bar{G}_\mu &= G_\mu \backslash \{e_j \mid \ell_j \neq 1\} \\ G_{\mu,j_1} &= G_\mu \backslash (\{e_j \mid \ell_j = 0\} \cup \{e_{j_1}\}) \\ G_{\mu,j_2} &= G_\mu \backslash (\{e_j \mid \ell_j = 0\} \cup \{e_{j_2}\}) \end{aligned}$$

jeweils maximale Kreispackungen $\bar{Z}, Z_{j_1}, Z_{j_2}$ (offenbar gilt stets $\bar{G}_\mu \subset G_{\mu,j_1}$ und $\bar{G}_\mu \subset G_{\mu,j_2}$).

(i) Falls $|\bar{Z}| = |Z_{j_1}| = |Z_{j_2}|$ gilt, setze $\mathcal{Z} := \mathcal{Z} \cup \{B_j \mid \tilde{B}_j \in \bar{Z}\}$ und setze $\ell_j := 0$ für alle j mit der Eigenschaft, dass $\tilde{B}_j$ eine Facette von G_μ ist.

(ii) Falls $|Z_{j_1}| > |Z_{j_2}| = |\bar{Z}|$ gilt, setze $\mathcal{Z} := \mathcal{Z} \cup \{B_j \mid \tilde{B}_j \in Z_{j_1} \backslash \{\tilde{B}_{j_2}\}\}$, setze $\ell_j := 0$ für alle j mit der Eigenschaft, dass $\tilde{B}_j \neq \tilde{B}_{j_1}, \tilde{B}_{j_2}$ eine Facette von G_μ ist und setze $\ell_{j_i} := \max\{0, \ell_{j_i} - 1\}$ für $i = 1, 2$.

(iii) Falls $|Z_{j_2}| > |Z_{j_1}| = |\bar{Z}|$ gilt, setze $\mathcal{Z} := \mathcal{Z} \cup \{B_j \mid \tilde{B}_j \in Z_{j_2} \backslash \{\tilde{B}_{j_1}\}\}$, setze $\ell_j := 0$ für alle j mit der Eigenschaft, dass $\tilde{B}_j \neq \tilde{B}_{j_1}, \tilde{B}_{j_2}$ eine Facette von G_μ ist und setze $\ell_{j_i} := \max\{0, \ell_{j_i} - 1\}$ für $i = 1, 2$.

(iv) Falls $|Z_{j_1}| = |Z_{j_2}| > |\bar{Z}|$ gilt, setze $L_k := \{\bar{Z}_{j_1}, \bar{Z}_{j_2}\} := \{\{B_j \mid \tilde{B}_j \in Z_{j_1} \backslash \{\tilde{B}_{j_2}\}\}, \{B_j \mid \tilde{B}_j \in Z_{j_2} \backslash \{\tilde{B}_{j_1}\}\}\}$, setze $\ell_j := 0$ für alle j mit der Eigenschaft, dass $\tilde{B}_j \neq \tilde{B}_{j_1}, \tilde{B}_{j_2}$ eine Facette von G_μ ist und setze $\ell_{j_i} := \max\{0, \ell_{j_i} - 1\}$ für $i = 1, 2$.

Lösche μ aus $\mathcal{T}$. Wenn durch Fall (i),(ii) oder (iii) eine Vergrößerung von $\mathcal{Z}$ stattgefunden hat, überprüfe, ob $\mathcal{Z}$ weiter vergrößert werden kann: Solange ein $B_{j'} \in \mathcal{Z}$ und ein $L_{k'} \in \mathcal{L}$ mit $Z_{j'} \in L_{k'}$ existieren, setze $\mathcal{Z} := \mathcal{Z} \cup Z_{j'}$ und $\mathcal{L} := \mathcal{L} \backslash L_{k'}$.

Ist $k = N$ (das heißt ist μ der letzte zu bearbeitende Knoten in $\mathcal{T}$), setze $\mathcal{Z} := \mathcal{Z} \cup \{B_j \mid \tilde{B}_j \in \bar{Z}\}$ und setze $\ell_j := 0$ für alle j mit der Eigenschaft, dass $\tilde{B}_j$ eine Facette von G_μ ist.

Der finale Schritt des Algorithmus besteht in der die Bearbeitung der Menge $\mathcal{L}$. Solange $\mathcal{L} \neq \varnothing$ ist, wähle $L_{k*} \in \mathcal{L}$ mit $k^* = \max\{k^* \mid L_{k*} \in \mathcal{L}\}$, setze $\mathcal{Z} := \mathcal{Z} \cup Z$ für $Z \in L_{k*}$, $\mathcal{L} := \mathcal{L} \backslash L_{k*}$ und prüfe dann für $B_{j'} \in \mathcal{Z}$: Solange ein $B_{j'} \in \mathcal{Z}$ und ein $L_{k'} \in \mathcal{L}$ mit $Z_{j'} \in L_{k'}$ existieren, setze $\mathcal{Z} := \mathcal{Z} \cup Z_{j'}$ und $\mathcal{L} := \mathcal{L} \backslash L_{k'}$. Der Algorithmus terminiert, sobald $\mathcal{L} = \varnothing$ gilt.

Die Laufzeit der Funktion SucheVergrösserung$(\mathcal{Z}, \bar{Z}, \mathcal{L})$ kann durch $O(|E(G)|)$ abgeschätzt werden, da zum Beispiel mit Hilfe von perfekten Hashtabellen in konstanter Zeit geprüft werden kann, ob ein Element in einer Menge enthalten ist. Die Kardinalität der Mengen $\bar{Z}, Z_{neu}$ und $\mathcal{L}$ kann durch $O(|E(G)|)$ abgeschätzt werden, sodass die While-Schleife der Funktion im schlechtesten Fall $O(|E(G)|)$-mal durchlaufen wird. Weiterhin wird die Funktion von Algorithmus 6.2 $O(|M(\mathcal{T})|)$-mal aufgerufen, sodass sich hierfür eine Gesamtlaufzeit von $O(|E(G)||M(\mathcal{T})|)$ ergibt. Die Auswahl der Bearbeitungsreihenfolge der Knoten $\mu \in \mathcal{T}$ abhängig von der zugehörigen Splitkomponente G_μ kann für den gesamten Algorithmus in $O(|E(G)|)$ Zeit erfolgen. Für jede der $|M(\mathcal{T})|$ Splitkomponenten bestimmt der Algorithmus im Verlauf drei verschiedene Kreispackungen. Da die Zerlegung $\mathcal{G}$ nach [CGW93] höchstens $|M(\mathcal{T})| \geqslant \max\{1, 2|V(G)| - 10\}$ Elemente besitzt und für jeden der $|M(\mathcal{T})| - 1$ verschiedenen 3-Splits genau drei Kanten zweimal in der Zerlegung repräsentiert werden, gilt $\sum_{\mu \in \mathcal{T}} |E(G_\mu)| = O(|E(G)|)$. Somit kann die Berechnung der drei Kreispackungen für alle Splitkomponenten G_μ durch $O(|E(G)|)$ abgeschätzt werden. In jeder der $|M(\mathcal{T})|$ Iterationen erfolgt die Entscheidung, welcher Fall durchlaufen wird, sowie die Aktualisierung der Mengen und Zähler in konstanter Zeit. Insgesamt kann die Laufzeit von Algorithmus 6.2 dann durch $O(|E(G)||M(\mathcal{T})|)$ abgeschätzt werden und wird daher von der Laufzeit von Algorithmus 5.2 zur Bestimmung der Zerlegung dominiert, welcher $O(|V(G)|^2|E(G)|)$ Zeit benötigt.

Nun wird zunächst gezeigt, dass Algorithmus 6.2 eine zulässige Lösung für das Kreispackungsproblem auf Halin-Graphen bestimmt.

Algorithmus 6.2 Kreispackungsalgorithmus für Halin-Graphen

Eingabe: Zerlegung $\mathcal{G} = \{G_{\mu_1}, \ldots, G_{\mu_N}\}$ eines Halin-Graphen G mit Kantennummerierung, Häufigkeit ℓ_j aller Facetten $B_j \in \mathcal{B}(G)$ in der Zerlegung $\mathcal{G}$ und zugehöriger W-Baum $\mathcal{T}(G) = (M, A)$ mit $M = \{\mu_1, \ldots, \mu_N\}$.

Ausgabe: Kreispackung $\mathcal{Z}(G)$.

$\mathcal{Z}, \bar{Z}, Z_{j_1}, Z_{j_2}, \mathcal{L}, L \leftarrow \emptyset, k, k^* \leftarrow 0$
Wähle ein Blatt $\mu \in \mathcal{T}$ mit (falls möglich) ungeradem G_μ.
while $\mu \neq \emptyset$ **do**
  $k \leftarrow k + 1$
  $\{j_1, j_2\} \leftarrow \{j \mid B_j \in \mathcal{B}(G_\mu)$ und $B'_j \in \mathcal{B}(G_{\mu'})$ für $(\mu, \mu') \in \mathcal{T}\}$
  Bestimme maximale Kreispackung $\bar{Z}$ von $G_\mu \backslash \{e_j \mid \ell_j \neq 1\}$.
  Bestimme maximale Kreispackung Z_{j_1} von $G_\mu \backslash (\{e_j \mid \ell_j = 0\} \cup \{e_{j_1}\})$.
  Bestimme maximale Kreispackung Z_{j_2} von $G_\mu \backslash (\{e_j \mid \ell_j = 0\} \cup \{e_{j_2}\})$.
  if $|\bar{Z}| = |Z_{j_1}| = |Z_{j_2}|$ **then**
    $\mathcal{Z} \leftarrow$ SucheVergrösserung$(\mathcal{Z}, \{B_j \mid \tilde{B}_j \in \bar{Z}\}, \mathcal{L})$
    $\mathcal{Z} \leftarrow \mathcal{Z} \cup \{B_j \mid \tilde{B}_j \in \bar{Z}\}$
    $\ell_j \leftarrow 0$ $\forall j$ mit $\tilde{B}_j \in \mathcal{B}(G_\mu)$
  else if $|Z_{j_1}| > |Z_{j_2}| = |\bar{Z}|$ **then**
    $\mathcal{Z} \leftarrow$ SucheVergrösserung$(\mathcal{Z}, \{B_j \mid \tilde{B}_j \in Z_{j_1} \backslash \{\tilde{B}_{j_2}\}\}, \mathcal{L})$
    $\mathcal{Z} \leftarrow \mathcal{Z} \cup \{B_j \mid \tilde{B}_j \in Z_{j_1} \backslash \{\tilde{B}_{j_2}\}\}$
    $\ell_j \leftarrow 0$ $\forall j \neq j_1, j_2$ mit $\tilde{B}_j \in \mathcal{B}(G_\mu)$
    $\ell_{j_i} \leftarrow \max\{0, \ell_{j_i} - 1\}$ für $i = 1, 2$
  else if $|Z_{j_2}| > |Z_{j_1}| = |\bar{Z}|$ **then**
    $\mathcal{Z} \leftarrow$ SucheVergrösserung$(\mathcal{Z}, \{B_j \mid \tilde{B}_j \in Z_{j_2} \backslash \{\tilde{B}_{j_1}\}\}, \mathcal{L})$
    $\mathcal{Z} \leftarrow \mathcal{Z} \cup \{B_j \mid \tilde{B}_j \in Z_{j_2} \backslash \{\tilde{B}_{j_1}\}\}$
    $\ell_j \leftarrow 0$ $\forall j \neq j_1, j_2$ mit $\tilde{B}_j \in \mathcal{B}(G_\mu)$
    $\ell_{j_i} \leftarrow \max\{0, \ell_{j_i} - 1\}$ für $i = 1, 2$
  else if $|Z_{j_1}| = |Z_{j_2}| > |\bar{Z}|$ **then**
    $\bar{Z}_{j_1} \leftarrow \{B_j \mid \tilde{B}_j \in Z_{j_2} \backslash \{\tilde{B}_{j_1}\}\}, \bar{Z}_{j_2} \leftarrow \{B_j \mid \tilde{B}_j \in Z_{j_2} \backslash \{\tilde{B}_{j_1}\}\}$
    $L_k \leftarrow \{\bar{Z}_{j_1}, \bar{Z}_{j_2}\}$
    $\mathcal{L} \leftarrow \mathcal{L} \cup L_k$
    $\ell_j \leftarrow 0$ $\forall j \neq j_1, j_2$ mit $\tilde{B}_j \in \mathcal{B}(G_\mu)$
    $\ell_{j_i} \leftarrow \max\{0, \ell_{j_i} - 1\}$ für $i = 1, 2$
  end if
  $\mathcal{T} \leftarrow \mathcal{T} \backslash \mu$
  Wähle $\mu \in \mathcal{T}$ mit $\delta_{\mathcal{T}}(\mu) = 1$ und (falls möglich) ungeradem G_μ.
end while
Wähle $\mu \in \mathcal{T}$ mit $\delta_{\mathcal{T}}(\mu) = 0$.
Bestimme maximale Kreispackung $\bar{Z}$ von $G_\mu \backslash \{e_j \mid \ell_j \neq 1\}$.
$\mathcal{Z} \leftarrow$ SucheVergrösserung$(\mathcal{Z}, \{B_j \mid \tilde{B}_j \in \bar{Z}\}, \mathcal{L})$
$\mathcal{Z} \leftarrow \mathcal{Z} \cup \{B_j \mid \tilde{B}_j \in \bar{Z}\}$
$\ell_j \leftarrow 0$ $\forall j$ mit $\tilde{B}_j \in \mathcal{B}(G_\mu)$
while $\mathcal{L} \neq \emptyset$ **do**
  $k^* \leftarrow \max\{k^* \mid L_{k^*} \in \mathcal{L}\}$
  Wähle $\bar{Z} \in L_{k^*}$.
  $\mathcal{Z} \leftarrow \mathcal{Z} \cup \bar{Z}$
  $\mathcal{L} \leftarrow \mathcal{L} \backslash L_{k^*}$
  $\mathcal{Z} \leftarrow$ SucheVergrösserung$(\mathcal{Z}, \bar{Z}, \mathcal{L})$
end while
return $\mathcal{Z}$

```
function SucheVergrösserung(𝒵, Z, ℒ)
    Z_neu ← ∅
    while ∃B_j' ∈ Z ∪ Z_neu und ∃L ∈ ℒ mit Z_j' ∈ L do
        𝒵 ← 𝒵 ∪ Z_j'
        Z_neu ← Z_neu ∪ Z_j'
        ℒ ← ℒ\L
    end while
    return 𝒵 ∪ Z_neu
end function
```

Lemma 6.19

Sei $G = T \cup C$ ein Halin-Graph. Dann bestimmt Algorithmus 6.2 eine kantendisjunkte Kreispackung von G.

Beweis: Es ist zu zeigen, dass Algorithmus 6.2 eine zulässige Lösung für das Kreispackungsproblem auf G berechnet. Sei $\mathcal{G} = \{G_{\mu_1}, \ldots, G_{\mu_N}\}$ die Zerlegung von G. Entsprechend Bemerkung 6.16 repräsentiert jede Facette $\tilde{B}_j$ eines Wheels G_{μ_i} eine Facette $B_j \in G$ und somit den entsprechenden Facettenkreis. Die Elemente von $\mathcal{Z}$ sind also Kreise in G. Es bleibt zu zeigen, dass die (Facetten-)Kreise in $\mathcal{Z}$ kantendisjunkt sind. Dazu ziehen wir Größen $\ell_j^{(k)}$ heran. Die Größe $\ell_j^{(k)}$ beschreibt den Wert von ℓ_j zu Beginn von Iteration k. Bei der Bearbeitung von μ in Iteration $k \leqslant N$ liegt die folgende Situation vor:

Falls $\ell_j > 1$: Die Facette $B_j \in G$ wird durch Facetten $\tilde{B}_j$ repräsentiert, die in mehr als einer Splitkomponente G_{μ_i} vorkommen. Die Facette B_j ist (noch) kein Kandidat für die Kreispackung $\mathcal{Z}$.

Falls $\ell_j = 0$: Die Facette $B_j \in G$ oder eine Facette $B_{j'} \in G$ mit $E(B_j) \cap E(B_{j'}) \neq \varnothing$ wurde in einer Iteration $k' < k$ zur Kreispackung $\mathcal{Z}$ oder zur Menge späterer Kandidaten $L_{k'}$ hinzugefügt. Die Facette B_j darf nicht mehr zur Kreispackung $\mathcal{Z}$ hinzugefügt werden.

Falls $\ell_j = 1$: Die Facette $B_j \in G$ wird durch die Facette $\tilde{B}_j$ in genau einer Splitkomponente G_{μ_i} repräsentiert und es gibt keine Facette $B_{j'} \in G$ mit $E(B_j) \cap E(B_{j'}) \neq \varnothing$ und $B_{j'} \in \mathcal{Z}$. Die Facette B_j ist ein Kandidat für die Kreispackung $\mathcal{Z}$.

Betrachte nun zwei beliebige Facetten $B_j, B_{j'} \in \mathcal{Z}$ und die Kardinalität ihres Vorkommens $\ell_j, \ell_{j'}$. Angenommen B_j wurde in Iteration k zu $\mathcal{Z}$ hinzugefügt und $B_{j'}$ wurde in Iteration k' zu $\mathcal{Z}$ hinzugefügt. Sei $k' \leqslant kk$. Das heißt, es ist $\ell_j^{(k)} = 1$ und $\ell_{j'}^{(k')} = 1$. Es

können die folgenden drei Situationen auftreten:

1. Angenommen B_j wurde in Iteration k und $B_{j'}$ in Iteration k' bei Vergrößerung der Kreispackung $\mathcal{Z}$ durch Fall (i),(ii) oder (iii) zu $\mathcal{Z}$ hinzugefügt.

   Falls $k' = k$ ist, waren $\tilde{B}_{j'}$ und $\tilde{B}_j$ Facetten derselben Splitkomponente G_μ, welche in Iteration k bearbeitet wurde. Da $\mathcal{Z}$ um eine kantendisjunkte Kreispackung von $\bar{G}$, G_{μ,j_1} oder G_{μ,j_2} ergänzt wurde, sind $\tilde{B}_{j'}$ und $\tilde{B}_j$ kantendisjunkt in G_μ. Dann sind $B_{j'}$ und B_j auch kantendisjunkt in G.

   Falls $k' < k$ ist, war entweder $\tilde{B}_{j'} \in G_{\mu'}$ und $\tilde{B}_j \notin G_{\mu'}$ oder $\tilde{B}_j$ war Kontaktfacette in $G_{\mu'}$. Falls $\tilde{B}_{j'} \in G_{\mu'}$ und $\tilde{B}_j \notin G_{\mu'}$ war, gilt $E(\tilde{B}_j) \cap E(\tilde{B}_{j'}) \neq \varnothing$ und dann gilt auch $E(B_j) \cap E(B_{j'}) \neq \varnothing$. Falls $\tilde{B}_j$ Kontaktfacette in $G_{\mu'}$ war, muss ebenfalls $E(\tilde{B}_j) \cap E(\tilde{B}_{j'}) \neq \varnothing$ (und damit $E(B_j) \cap E(B_{j'}) \neq \varnothing$) gelten, da sonst $\ell_j^{(k+i)} = 0$ gesetzt worden wäre für $i \geqslant 1$ in Iteration $k + i$ des Algorithmus. Das ist ein Widerspruch zur Annahme $\ell_j^{(k)} = 1$.

2. Angenommen B_j wurde in Iteration k bei Vergrößerung der Kreispackung $\mathcal{Z}$ durch Fall (i),(ii) oder (iii) zu $\mathcal{Z}$ hinzugefügt und $B_{j'}$ wurde in Iteration k' bei Vergrößerung der Kreispackung $\mathcal{Z}$ durch Bearbeitung der Menge $\mathcal{L}$ zu $\mathcal{Z}$ hinzugefügt. Dann wurde $B_{j'}$ in Iteration $k'' < k'$ zur Menge $\mathcal{L}$ hinzugefügt.

   Dann muss $E(\tilde{B}_j) \cap E(\tilde{B}_{j'}) \neq \varnothing$ (und damit $E(B_j) \cap E(B_{j'}) \neq \varnothing$) gelten, da sonst $\ell_j^{(k''+i)} = 0$ gesetzt worden wäre für $i \geqslant 1$ in Iteration $k'' + i$ des Algorithmus. Das ist ein Widerspruch zur Annahme $\ell_j^{(k)} = 1$.

3. Angenommen $B_{j'}$ wurde in Iteration k' bei Vergrößerung der Kreispackung $\mathcal{Z}$ durch Fall (i),(ii) oder (iii) zu $\mathcal{Z}$ hinzugefügt und B_j wurde in Iteration k bei Vergrößerung der Kreispackung $\mathcal{Z}$ durch Bearbeitung der Menge $\mathcal{L}$ zu $\mathcal{Z}$ hinzugefügt. Dann wurde B_j in Iteration $k'' < k$ zur Menge $\mathcal{L}$ hinzugefügt.

   Falls $k'' < k' (\leqslant k)$ ist, muss $E(\tilde{B}_j) \cap E(\tilde{B}_{j'}) \neq \varnothing$ (und damit $E(B_j) \cap E(B_{j'}) \neq \varnothing$) gelten, da sonst $\ell_{j'}^{(k''+i)} = 0$ gesetzt worden wäre für $i \geqslant 1$ in Iteration $k'' + i$ des Algorithmus. Das ist ein Widerspruch zur Annahme $\ell_{j'}^{(k')} = 1$.

   Falls $k' \leqslant k'' (< k)$ ist, muss $E(\tilde{B}_j) \cap E(\tilde{B}_{j'}) \neq \varnothing$ (und damit $E(B_j) \cap E(B_{j'}) \neq \varnothing$)

gelten, da sonst $\ell_j^{(k'+i)} = 0$ gesetzt worden wäre für $i \geqslant 1$ in Iteration $k' + i$ des Algorithmus. Das ist ein Widerspruch zur Annahme $\ell_j^{(k)} = 1$.

4. Angenommen B_j wurde in Iteration k und $B_{j'}$ in Iteration k' bei Vergrößerung der Kreispackung $\mathcal{Z}$ durch Bearbeitung der Menge $\mathcal{L}$ zu $\mathcal{Z}$ hinzugefügt. Dann wurde B_j in Iteration $\bar{k} < k$ und $B_{j'}$ in Iteration $\bar{k}' < k'$ zur Menge $\mathcal{L}$ hinzugefügt.

   Falls $\bar{k} < \bar{k}' (< k' \leqslant k)$ ist, muss $E(\tilde{B}_j) \cap E(\tilde{B}_{j'}) \neq \varnothing$ (und damit $E(B_j) \cap E(B_{j'}) \neq \varnothing$) gelten, da sonst $\ell_{j'}^{(\bar{k}+i)} = 0$ gesetzt worden wäre für $i \geqslant 1$ in Iteration $\bar{k} + i$ des Algorithmus. Das ist ein Widerspruch zur Annahme $\ell_{j'}^{(k')} = 1$.

   Falls $\bar{k} = \bar{k}' (< k' \leqslant k)$ ist, gilt entweder $\tilde{B}_j \in L_{\bar{k}}$ sowie $\tilde{B}_{j'} \in L_{\bar{k}}$ und damit $E(\tilde{B}_j) \cap E(\tilde{B}_{j'}) \neq \varnothing$ (also auch $E(B_j) \cap E(B_{j'}) \neq \varnothing$). Andernfalls gilt $B_j \in \mathcal{Z}$ und $B_{j'} \notin \mathcal{Z}$ bzw. $B_j \notin \mathcal{Z}$ und $B_{j'} \in \mathcal{Z}$. Das ist ein Widerspruch zur Annahme $B_j, B_{j'} \in \mathcal{Z}$.

   Falls $\bar{k}' < \bar{k} (< k)$ ist, muss $E(\tilde{B}_j) \cap E(\tilde{B}_{j'}) \neq \varnothing$ (und damit $E(B_j) \cap E(B_{j'}) \neq \varnothing$) gelten, da sonst $\ell_j^{(\bar{k}'+i)} = 0$ gesetzt worden wäre für $i \geqslant 1$ in Iteration $\bar{k}' + i$ des Algorithmus. Das ist ein Widerspruch zur Annahme $\ell_j^{(k)} = 1$.

Damit sind $B_j, B_{j'} \in \mathcal{Z}$ kantendisjunkt und der Algorithmus berechnet eine zulässige Lösung. □

Beispiel 6.20

Der Ablauf von Algorithmus 6.2 soll anhand des Halin-Graphen G veranschaulicht werden, für den in Abschnitt 5.1 bereits die Zerlegung und der zugehörige W-Baum entwickelt wurden (siehe Abbildung 5.12). Abbildung 6.12 zeigt nun eben diesen Halin-Graphen $G = T \cup C$ und seine Zerlegung $\mathcal{G} = \{G'_1, \ldots, G'_7\}$ sowie die Nummerierung der Kanten $e \in C$. Durch die Nummerierung der Kreiskanten wird deutlich, dass jede Facette von G mindestens einmal durch eine Facette eines Wheels G_μ repräsentiert wird. Tabelle 6.1 zeigt die Häufigkeit ℓ_j einer solchen Repräsentation in der Zerlegung $\mathcal{G}$.

Der Baum $\mathcal{T}$ enthält als Blätter die Knoten $\mu_1, \mu_3, \mu_5, \mu_7$, wobei G'_1 ein gerades Wheel darstellt. Das zuerst ausgewählte Blatt ist μ_3. Die zugehörige Komponente G'_3 entspricht einem ungeraden Wheel. Die Facetten $\tilde{B}_{13}, \tilde{B}_{15}$ sind die Kontaktfacetten von G'_3 und es gilt $\ell_{13} = 2 > 0$ und $\ell_{15} = 3 > 0$. Der Algorithmus bestimmt $\bar{Z} = \{B_{14}\}$, $Z_{13} = \{B_{14}\}$

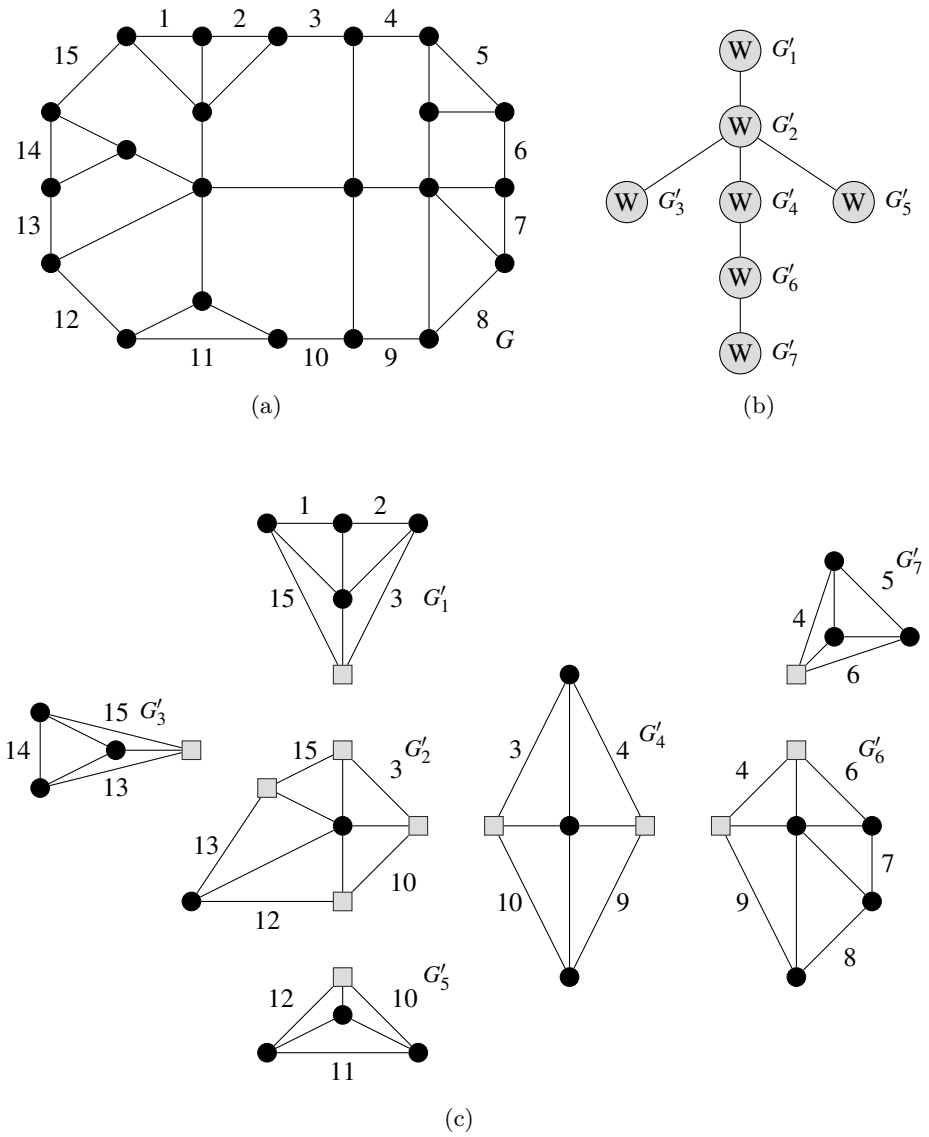


Abbildung 6.12: Halin-Graph G (a) mit Nummerierung der Kanten, die zugehörige Zerlegung (c) mit Nummerierung der Kanten und der zugehörige W-Baum (b).

oder $Z_{13} = \{B_{15}\}$ und $Z_{15} = \{B_{14}\}$ oder $Z_{15} = \{B_{13}\}$. Alle diese Kreispackungen besitzen Kardinalität 1, sodass Fall (i) durchlaufen und $\mathcal{Z}$ um $\bar{Z}$ zu $\mathcal{Z} = \{B_{14}\}$ erweitert wird. Der Algorithmus setzt daraufhin $\ell_{13} = \ell_{14} = \ell_{15} = 0$ wie in Tabelle 6.2 Iteration 1 angegebenen. Das Blatt μ_3 wird im Anschluss aus dem Baum $\mathcal{T}$ gelöscht. Da $\mathcal{L} = \varnothing$, wird $\mathcal{Z}$ in dieser Iteration nicht erneut vergrößert. Bei der Wahl von μ_5 in Iteration 2 und μ_7 in Iteration 3 ist das Vorgehen des Algorithmus analog zu Iteration 1. Auch hier ist der Verlauf der Iterationen in Tabelle 6.2 beschrieben. Auch die Blätter μ_5 und μ_7

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
ℓ_j	1	1	3	3	1	2	1	1	2	3	1	2	2	1	3
It. 1													0	$\underline{0}$	0
It. 2										0	$\underline{0}$	0			
It. 3				0	$\underline{0}$	0									
It. 4							$\underline{0}$	0	1						
It. 5			2						$\underline{0}$						
It. 6			1												
It. 7	$\underline{0}$	0	$\underline{0}$												

Tabelle 6.1: Der Verlauf der Werte ℓ_j bei Anwendung von Algorithmus 6.2 auf Beispiel 6.20.

werden im Anschluss an ihre jeweilige Bearbeitung aus $\mathcal{T}$ gelöscht. Da weiterhin $\mathcal{L} = \varnothing$ gilt, wird $\mathcal{Z}$ in keiner der beiden Iterationen zusätzlich vergrößert.

Danach ist μ_6 das einzige Blatt mit ungeradem Wheel. Die beiden Kontaktfacetten von μ_6 und μ_4' sind $\tilde{B}_4$ und $\tilde{B}_9$ mit $\ell_4 = 0$ und $\ell_9 = 2$. Der Algorithmus bestimmt $\bar{Z} = \{B_7\}$ oder $\bar{Z} = \{B_8\}$, $Z_4 = \{B_7, B_9\}$ und $Z_9 = \{B_7\}$ oder $Z_9 = \{B_8\}$, da $\ell_4 = 0$ ist und die Facette B_4 daher nicht mehr für eine Kreispackung ausgewählt werden darf. Da $|Z_4| > |\bar{Z}|$ und $|Z_4| \neq |Z_9|$ gilt, wird Fall (iii) durchlaufen. Für die Facette $\tilde{B}_9$ gilt $\ell_9 = 2$, es existiert also eine weitere Facette $\tilde{B}_9$ in G_4', sodass diese noch nicht zu $\mathcal{Z}$ hinzugefügt werden darf. $\mathcal{Z}$ wird um $Z_4 \backslash \{B_9\}$ zu $\mathcal{Z} = \{B_{14}, B_{11}, B_5, B_7\}$ erweitert. Der Wert von ℓ_9 wird um eins reduziert, sodass die Facette B_9 in einer späteren Iteration ausgewählt werden kann. Das Blatt μ_6 wird im Anschluss an seine Bearbeitung aus dem Baum $\mathcal{T}$ gelöscht. Da weiterhin $\mathcal{L} = \varnothing$ gilt, wird $\mathcal{Z}$ nicht zusätzlich vergrößert.

Nun enthält $\mathcal{T}$ kein Blatt mit ungeradem Wheel, sodass μ_4 gewählt wird. Der Ablauf von Iteration 5 entspricht Iteration 4. Nach dem Löschen von μ_4 wird μ_2 zu einem Blatt in $\mathcal{T}$ mit einem ungeraden Wheel G_2'. Allerdings gilt $\ell_{10} = \ell_{12} = \ell_{13} = \ell_{15} = 0$ und $\ell_3 = 2$, sodass der Algorithmus zwar Fall (iii) durchläuft, die Kreispackung $\mathcal{Z}$ in Iteration 6 aber nicht erweitert wird. Wieder ist $\mathcal{L} = \varnothing$.

In der letzten Iteration ist die Menge der Blätter in $\mathcal{T}$ leer, der Baum besteht aus dem Knoten μ_1. Der Algorithmus bestimmt $\bar{Z} = \{B_1, B_3\}$ und ergänzt $\mathcal{Z}$ zu $\mathcal{Z} = \{B_{14}, B_{11}, B_5, B_7, B_9, B_1, B_3\}$. Da $\mathcal{L} = \varnothing$ ist, ist $\mathcal{Z}$ die finale Kreispackung.

Durch den Algorithmus wurde die in Abbildung 6.13 farblich markierte Kreispackung bestimmt, welche durch die Facetten $\mathcal{Z} = \{B_{14}, B_{11}, B_5, B_7, B_9, B_1, B_3\}$ induziert wird.

	μ_i	$\mathcal{Z}$	ℓ_j
It. 1	μ_3	$\mathcal{Z} = \{B_{14}\}$	$\ell_{13} = \ell_{14} = \ell_{15} = 0$
It. 2	μ_5	$\mathcal{Z} = \{B_{14}, B_{11}\}$	$\ell_{10} = \ell_{11} = \ell_{12} = 0$
It. 3	μ_7	$\mathcal{Z} = \{B_{14}, B_{11}, B_5\}$	$\ell_4 = \ell_5 = \ell_6 = 0$
It. 4	μ_6	$\mathcal{Z} = \{B_{14}, B_{11}, B_5, B_7\}$	$\ell_7 = \ell_8 = 0, \ell_9 = 1$
It. 5	μ_4	$\mathcal{Z} = \{B_{14}, B_{11}, B_5, B_7, B_9\}$	$\ell_9 = 0, \ell_3 = 2$
It. 6	μ_2	$\mathcal{Z} = \{B_{14}, B_{11}, B_5, B_7, B_9\}$	$\ell_3 = 1$
It. 4	μ_1	$\mathcal{Z} = \{B_{14}, B_{11}, B_5, B_7, B_9, B_1, B_3\}$	$\ell_1 = \ell_2 = \ell_3 = 0$

Tabelle 6.2: Die Iteration bei Anwendung von Algorithmus 6.2 auf Beispiel 6.20.

Es gilt $|\mathcal{Z}| = 7$.

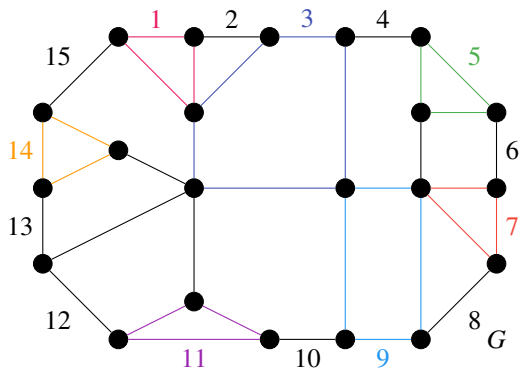


Abbildung 6.13: Die maximale Kreispackung des Halin-Graphen G wird induziert durch $\mathcal{Z} = \{B_{14}, B_{11}, B_5, B_7, B_9, B_1, B_3\}$.

Um zu zeigen, dass der Algorithmus auch eine optimale Lösung bestimmt, werden die folgenden Beobachtungen und Aussagen zu Facetten von Fans in einer maximalen Kreispackung benötigt.

Lemma 6.21
Sei $G = T \cup C$ ein Halin-Graph.

(i) Jede innere Facette $B \in \mathcal{B}(G)$ hat $|E(B)| - 1$ innere benachbarte Facetten.

(ii) Ist $F \subset G$ ein Fan und $B \in \mathcal{B}(F)$. Dann ist B eine Facette in G mit genau zwei inneren Nachbarfacetten.

Beweis: Zu (i): Klar, da es genau eine zu B benachbarte äußere Facette B_a gibt.

Zu (ii): Nach der Definition eines Fans entspricht jede Facette $B \in \mathcal{B}(F)$ einem Dreieck und ist mit der äußeren Facette B_a von G benachbart. Aus diesem Grund besitzt B genau zwei innere Nachbarfacetten. □

Dementsprechend ist es naheliegend, dass insbesondere die Facetten eines Fans zur Bestimmung einer maximalen Kreispackung herangezogen werden können. Wir betrachten (zunächst) den Fall, dass die Fans des Graphen jeweils aus einer geraden Anzahl von Facetten bestehen.

Lemma 6.22
Sei $G = T \cup C$ ein Halin-Graph und $\mathcal{F}(G) = \{F \mid F \text{ ist Fan in } G\}$. Sei $|\mathcal{B}(F)| = 2m_F, m_F \geqslant 1$ für jeden Fan $F \in \mathcal{F}(G)$. Dann gibt es eine maximale Kreispackung $\mathcal{Z}^*(G)$, sodass für jeden Fan $F \subset G$ mit $\mathcal{B}(F) = \{B_1, \ldots, B_{2m_F}\}$ und jedes $\ell \in \{1, \ldots, m_F\}$ entweder $B_{2\ell-1} \in \mathcal{Z}^*(G)$ oder $B_{2\ell} \in \mathcal{Z}^*(G)$ gilt.

Beweis: Nach Lemma 5.13 besitzt jeder Halin-Graph G mindestens zwei Fans [CNP83], das heißt es gilt $|\mathcal{F}(G)| \geqslant 2$. Nach Voraussetzung besitzt jeder Fan $F \in \mathcal{F}(G)$ eine gerade Zahl von Facetten $\{B_1, \ldots, B_{2m_F}\}$. Die Nummerierung der Facetten sei derart vorgenommen, dass $B_j \cap B_{j+1} \neq \varnothing$ für $j = 1, \ldots, 2m_F - 1$. Angenommen es gibt keine maximale Kreispackung mit $B_{2\ell-1} \in \mathcal{Z}^*(G)$ oder $B_{2\ell} \in \mathcal{Z}^*(G)$ für $\ell = 1, \ldots, m_F$.

Ohne Beschränkung der Allgemeinheit sei $B_{2\ell-1} \notin \mathcal{Z}^*(G)$ für $\ell = 1, \ldots, m_F$. Dann muss $B_{2\ell} \in \mathcal{Z}^*(G)$ mit $\ell = 1, \ldots, m_F - 2$ gelten, da $\mathcal{Z}^*(G)$ sonst nicht maximal wäre. Sei daher $B_{2\ell} \in \mathcal{Z}^*(G)$ mit $\ell = 1, \ldots, m_F - 2$. Da $B_{2m_F-1} \notin \mathcal{Z}^*(G)$ und $B_{2m_F} \notin \mathcal{Z}^*(G)$ gilt, muss eine zu B_{2m_F} benachbarte Facette aus $G \backslash F$ in $\mathcal{Z}^*(G)$ enthalten sein, da $\mathcal{Z}^*(G)$ sonst nicht maximal wäre. Dann existiert aber auch eine maximale Kreispackung $\mathcal{Z}^{**}(G)$ mit $B_{2m_F} \in \mathcal{Z}^{**}(G)$. Das ist ein Widerspruch zur Annahme, dass es eine solche Kreispackung nicht geben kann. □

Satz 6.23
Sei $G = T \cup C$ ein Halin-Graph mit Zerlegung $\mathcal{G} = \{G_{\mu_1}, \ldots, G_{\mu_N}\}$ und zugehörigem W-Baum $\mathcal{T} = (M, A)$. Sei $\delta_G(v)$ eine gerade Zahl für alle Knoten $v \in V(T) \backslash V(C)$ und $M' = \{\mu \in \mathcal{T} \mid \delta_{\mathcal{T}}(\mu) = 1\}$. Sei $\tilde{M} = \{\mu \in M' \mid \text{es existiert } (\mu, \mu') \in A \text{ mit } \delta_{\mathcal{T} \backslash M'}(\mu') = 1\}$

und $\tilde{\mathcal{F}} = \{F_\mu \mid \mu \in \tilde{M} \text{ und } F_\mu \subset G_\mu\}$. Dann gilt

$$\nu(G) = \begin{cases} \nu(G_\mu), \text{ falls } N = 1 \\ \sum_{\mu\in\mathcal{T}} \nu(G_\mu) - 1, \text{ falls } N = 2 \\ \sum_{\mu\in\tilde{M}} (\nu(G_\mu) - 1) + \nu(G \times \tilde{\mathcal{F}}), \text{ falls } N \geqslant 3 \end{cases}$$

Beweis: Der Beweis erfolgt durch Induktion über die Anzahl N von Knoten im W-Baum $\mathcal{T} = (M, A)$ von G. Ist $N = 1$, das heißt $\mathcal{G} = \{G\}$ und $\mathcal{T}$ besteht aus nur einem Knoten μ. Dann ist G ein gerades Wheel. Es gilt $\nu(G) = \nu(G_\mu)$. Das ist offensichtlich optimal.

Ist $N = 2$, ist $\mathcal{G} = \{G_{\mu_1}, G_{\mu_2}\}$ und $\mathcal{T} = (M, A)$ mit $M = \{\mu_1, \mu_2\}$ und $A = \{(\mu_1, \mu_2)\}$. Dann ist G ein Caterpillar-Halin-Graph, der die Eigenschaft besitzt, dass $\delta_G(v)$ für $v \in V(T)\backslash V(C)$ eine gerade Zahl ist. Nach Satz 6.13 gilt dann $\nu(G) = \sum_{\mu\in\mathcal{T}} \nu(G_\mu) - (N-1) = \sum_{\mu\in\mathcal{T}} \nu(G_\mu) - 1$.

Ist $N = 3$, ist $\mathcal{G} = \{G_{\mu_1}, G_{\mu_2}, G_{\mu_3}\}$ und $\mathcal{T} = (M, A)$ mit $M = \{\mu_1, \mu_2, \mu_3\}$ und $A = \{(\mu_1, \mu_3), (\mu_2, \mu_3)\}$. Es gilt $\mu_1, \mu_2 \in \tilde{M}$. Dann ist G ebenfalls ein Caterpillar-Halin-Graph, der die Eigenschaft besitzt, dass $\delta_G(v)$ für $v \in V(T)\backslash V(C)$ eine gerade Zahl ist. Wiederum mit Satz 6.13 ergibt sich dann

$$\begin{aligned} \nu(G) &= \sum_{\mu\in\mathcal{T}} \nu(G_\mu) - 2 \\ &= \nu(G_{\mu_1}) - 1 + \nu(G_{\mu_2}) - 1 + \nu(G_{\mu_3}) \\ &= \sum_{\mu\in\tilde{M}} (\nu(G_\mu) - 1) + \nu(G \times F_{\mu_1} \times F_{\mu_2}) \end{aligned}$$

Sei $N \geqslant 4$ und sei G ein Graph, der die Voraussetzungen des Satzes erfüllt. Da $\delta_G(v)$ eine gerade Zahl ist für alle Knoten $v \in V(T)\backslash V(C)$, ist auch $|\mathcal{B}(F_\mu)|$ eine gerade Zahl für jeden Fan $F_\mu \in \tilde{\mathcal{F}}$. Dann gibt es nach Lemma 6.22 eine maximale Kreispackung $\mathcal{Z}^*(G)$, sodass für jeden Fan $F_\mu \subset G$ mit $\{B_1, \ldots, B_{2m_{F_\mu}}\} \in F_\mu$ und $\mu \in \tilde{M}$ und jedes $\ell = 1, \ldots, m_{F_\mu}$ entweder $B_{2\ell-1} \in \mathcal{Z}^*(G)$ oder $B_{2\ell} \in \mathcal{Z}^*(G)$ gilt.

Der Graph $G' = G \times \tilde{\mathcal{F}}$ ist nach Lemma 5.14 ebenfalls ein Halin-Graph und besitzt nach Bemerkung 5.15 also eine Darstellung $G' = T' \cup C'$ mit $T' = T\backslash\{\bigcup_{\mu\in\tilde{M}} C(v_\mu) \mid \{v_\mu\} \cup C(v_\mu) \text{ induziert Fan } F_\mu \in \mathcal{F}_\mu\}$. Da der Baum T' durch Löschen einer Teilmenge von

Blättern von T entsteht, gilt $\delta_{G'}(v) \leqslant \delta_G(v)$ für alle Knoten $v \in V(C')$, der Knotengrad der Knoten, die zu Blättern in T adjazent sind, verringert sich also in T'. Insbesondere aber ändert sich der Knotengrad der Knoten, die nicht zu Blättern in T adjazent sind durch Löschen dieser Blätter nicht, das heißt, es gilt $\delta_{G'}(v) = \delta_G(v)$ für alle Knoten $v \in V(T')\backslash V(C')$. Daher ist $\delta_{G'}(v)$ eine gerade Zahl für alle Knoten $v \in V(T')\backslash V(C')$.

Es sei $\mathcal{Z}^*(G) = \bigcup_{\mu\in\tilde{M}} \mathcal{Z}^*(F_\mu) \cup \mathcal{Z}$ eine maximale Kreispackung von G, in der $\mathcal{Z}^*(F_\mu)$ eine maximale Kreispackung für den jeweiligen Fan $F_\mu \subset G_\mu$ ist. Eine solche Kreispackung existiert nach Lemma 6.22. Dann gilt:

$$\nu(G) = \sum_{\mu\in\tilde{M}} |\mathcal{Z}^*(F_\mu)| + |\mathcal{Z}| = \sum_{\mu\in\tilde{M}} (\nu(G_\mu) - 1) + |\mathcal{Z}|$$

Es wird nun gezeigt, dass $|\mathcal{Z}| = \nu(G')$ ist.

Da $G' = G \times \tilde{\mathcal{F}}$ durch Kontraktion aller Fans $F_\mu \in \tilde{\mathcal{F}}$ zu einem Knoten entsteht, gilt $\mathcal{B}(G') = \mathcal{B}(G)\backslash\mathcal{B}(\tilde{\mathcal{F}})$. Außerdem gilt $\mathcal{Z} \subset \mathcal{B}(G)\backslash\mathcal{B}(\tilde{\mathcal{F}})$ und $B_i \cap B_j = \varnothing$ für zwei beliebige Facetten $B_i, B_j \in \mathcal{Z}$. Daher ist $\mathcal{Z}$ eine Menge von kantendisjunkten Kreisen in G', somit gilt $|\mathcal{Z}| \leqslant \nu(G')$. Angenommen es ist $|\mathcal{Z}| < \nu(G')$. Dann gibt es eine Packung $\mathcal{Z}^*(G')$ mit $|\mathcal{Z}^*(G')| = \nu(G') > |\mathcal{Z}|$. Aber dann muss für mindestens ein $\mu \in \tilde{M}$ eine der Kontaktfacetten $B_\mu^{(1)}, B_\mu^{(2)}$ von G_μ zu $\mathcal{Z}^*(G')$ gehören. Ansonsten wäre $\bar{\mathcal{Z}}(G) := \bigcup_{\mu\in\tilde{M}} \mathcal{Z}^*(F_\mu) \cup \mathcal{Z}^*(G')$ eine kantendisjunkte Kreispackung mit $|\bar{\mathcal{Z}}(G)| > \nu(G)$. Sei $\tilde{M}_2 = \{\mu \in \tilde{M} \mid B_\mu^{(1)} \in \mathcal{Z}^*(G') \text{ oder } B_\mu^{(2)} \in \mathcal{Z}^*(G')\}$ und $\tilde{M}_1 = \tilde{M}\backslash\tilde{M}_2$. Nach dem eben Gesagten ist $\tilde{M}_2 \neq \varnothing$. Für $\mu \in \tilde{M}_2$ sei ohne Beschränkung der Allgemeinheit $B_\mu^{(1)} \in \mathcal{Z}^*(G')$. Sei $\mathcal{Z}^{**}(G_\mu)$ die maximale Kreispackung von G_μ, die $B_\mu^{(1)}$ enthält. Dann gilt

$$\mathcal{Z}(G) := \mathcal{Z}^*(G') \cup \bigcup_{\mu\in\tilde{M}_1} \mathcal{Z}^*(F_\mu) \cup \bigcup_{\mu\in\tilde{M}_2} (\mathcal{Z}^{**}(G_\mu)\backslash B_\mu^{(1)})$$

und dementsprechend ergibt sich

$$\begin{aligned} |\mathcal{Z}(G)| &= \nu(G') + \sum_{\mu\in\tilde{M}_1} (\nu(G_\mu) - 1) + \sum_{\mu\in\tilde{M}_2} (\nu(G_\mu) - 1) \\ &> |\mathcal{Z}| + \sum_{\mu\in\tilde{M}} (\nu(G_\mu) - 1) = \nu(G) \end{aligned}$$

Das ist ein Widerspruch zur Annahme, dass $\mathcal{Z}^*(G) = \bigcup_{\mu\in\tilde{M}} \mathcal{Z}^*(F_\mu) \cup Z$ eine maximale

Kreispackung von G und $\nu(G) = |\mathcal{Z}^*(G)|$ ist. Also muss gelten $|\mathcal{Z}| = \nu(G')$ und somit:

$$\nu(G) = \sum_{\mu \in \tilde{M}} (\nu(G_\mu) - 1) + \nu(G \times \tilde{\mathcal{F}}).$$

□

Entsprechend Satz 6.23 kann die Kreispackungszahl eines Halin-Graphen G mit Zerlegung $\mathcal{G}$ und W-Baum $\mathcal{T}$ für $N \geqslant 3$ konstruktiv bestimmt werden. Sei $G_{(1)} := G$, $\mathcal{T}_{(1)} := \mathcal{T}$ und $G_{(2)} := G_{(1)} \times \tilde{F}$. Falls auch $N - |\tilde{M}| \geqslant 3$ gilt, können wiederum für den Graphen $G_{(2)}$ die folgenden Mengen bestimmt werden:

$$\begin{aligned}
M'_{(2)} &:= \{\mu \text{ ist Blatt in } \mathcal{T}_{(2)} := \mathcal{T}_{(1)} \backslash \tilde{M}_{(1)}\} \text{ mit} \\
\tilde{M}_{(2)} &:= \{\mu \in M'_{(2)} \mid \text{es existiert } (\mu, \mu') \in A \text{ mit } \delta_{\mathcal{T}_{(1)} \backslash M'_{(1)}}(\mu') = 1\} \text{ und} \\
\tilde{\mathcal{F}}_{(2)} &:= \{F_\mu \mid \mu \in \tilde{M}_{(2)} \text{ und } F_\mu \subset G_\mu\}.
\end{aligned}$$

Wird diese Konstruktion fortgesetzt, kann G in $1 \leqslant q \leqslant \frac{N}{2}$ Schritten zu einem Caterpillar-Halin-Graph $G_{(q)}$ mit zugehörigem Weg $P_{(q)}$ der Länge 2 oder einem Wheel kontrahiert werden. Daher können die Mengen $M', \tilde{M}, \tilde{\mathcal{F}}$ aus Satz 6.23 für jeden Schritt r mit $r = 1, \ldots, q$ definiert werden. Sei $\mathcal{T}_{(1)} := \mathcal{T}$. Dann sei

$$\begin{aligned}
M'_{(r)} &:= \{\mu \text{ ist Blatt in } \mathcal{T}_{(r)} := \mathcal{T}_{(r-1)} \backslash \tilde{M}_{(r-1)}\} \text{ mit} \\
\tilde{M}_{(r)} &:= \{\mu \in M'_{(r)} \mid \text{es existiert } (\mu, \mu') \in A \text{ mit } \delta_{\mathcal{T}_{(r-1)} \backslash M'_{(r-1)}}(\mu') = 1\} \text{ und} \\
\tilde{\mathcal{F}}_{(r)} &:= \{F_\mu \mid \mu \in \tilde{M}_{(r)} \text{ und } F_\mu \subset G_\mu\} \text{ für } r = 1, \ldots, q.
\end{aligned}$$

Nun kann gezeigt werden, dass die Bestimmung der Kreispackungszahl $\nu(G)$ unabhängig von der in Satz 6.23 verwendeten Konstruktion und Satz 6.23 eine Verallgemeinerung von Satz 6.13 ist.

Satz 6.24

Sei $G = T \cup C$ ein Halin-Graph mit Zerlegung $\mathcal{G} = \{G_{\mu_1}, \ldots, G_{\mu_N}\}$ und $\mathcal{T} = (M, A)$ der zugehörige W-Baum. Sei $\delta_G(v)$ eine gerade Zahl für alle Knoten $v \in V(T) \backslash V(C)$. Dann gilt

$$\nu(G) = \sum_{\mu \in \mathcal{T}(G)} \nu(G_\mu) - (N - 1)$$

Beweis: Nach Satz 6.23 gilt

$$\nu(G) = \begin{cases} \nu(G_\mu), \text{ falls } N = 1 \\ \sum_{\mu\in\mathcal{T}} \nu(G_\mu) - 1, \text{ falls } N = 2 \\ \sum_{\mu\in\tilde{M}} (\nu(G_\mu) - 1) + \nu(G \times \tilde{\mathcal{F}}), \text{ falls } N \geqslant 3. \end{cases}$$

Offenbar ist die Aussage richtig für $N \leqslant 2$. Sei nun $N \geqslant 3$.

$$\begin{aligned} \nu(G) &= \sum_{\mu\in\tilde{M}_{(1)}} (\nu(G_\mu) - 1) + \nu(G \times \tilde{\mathcal{F}_{(1)}}) \\ &= \sum_{\mu\in\tilde{M}_{(1)}} (\nu(G_\mu) - 1) + \sum_{\mu\in\tilde{M}_{(2)}} (\nu(G_\mu) - 1) + \nu\big[(G \times \tilde{\mathcal{F}_{(1)}}) \times \tilde{\mathcal{F}_{(2)}}\big] \\ &\vdots \\ &= \sum_{1\leqslant r\leqslant q} \sum_{\mu\in\tilde{M}_{(r)}} (\nu(G_\mu) - 1) + \nu\left[\left(\big[(G \times \tilde{\mathcal{F}}_{(1)}) \times \tilde{\mathcal{F}}_{(2)}\big] \times \dots\right) \times \tilde{\mathcal{F}}_{(q)}\right] \end{aligned}$$

Es sind zwei Fälle zu unterscheiden:

Fall 1: Der Graph $\left(\big[(G \times \tilde{\mathcal{F}}_{(1)}) \times \tilde{\mathcal{F}}_{(2)}\big] \times \dots\right) \times \tilde{\mathcal{F}}_{(q)}$ stellt ein Wheel $G_{\bar{\mu}}$ dar (mit Zerlegung $\mathcal{G} = \{G_{\bar{\mu}}\}$) und $\bar{\mathcal{T}} = (\{\bar{\mu}\}, \varnothing)$. Nach Satz 6.23 gilt dann

$$\nu\left[\left(\big[(G \times \tilde{\mathcal{F}}_{(1)}) \times \tilde{\mathcal{F}}_{(2)}\big] \times \dots\right) \times \tilde{\mathcal{F}}_{(q)}\right] = \nu(G_{\bar{\mu}})$$

somit ergibt sich

$$\begin{aligned} \nu(G) &= \sum_{1\leqslant r\leqslant q} \sum_{\mu\in\tilde{M}_{(r)}} (\nu(G_\mu) - 1) + \nu\left[\left(\big[(G \times \tilde{\mathcal{F}}_{(1)}) \times \tilde{\mathcal{F}}_{(2)}\big] \times \dots\right) \times \tilde{\mathcal{F}}_{(q)}\right] \\ &= \sum_{\mu\in\mathcal{T}\setminus\{\bar{\mu}\}} (\nu(G_\mu) - 1) + \nu(G_{\bar{\mu}}) \\ &= \sum_{\mu\in\mathcal{T}\setminus\{\bar{\mu}\}} \nu(G_\mu) - (N-1) + \nu(G_{\bar{\mu}}) \\ &= \sum_{\mu\in\mathcal{T}} \nu(G_\mu) - (N-1) \end{aligned}$$

Fall 2: Der Graph $\left(\big[(G \times \tilde{\mathcal{F}}_{(1)}) \times \tilde{\mathcal{F}}_{(2)}\big] \times \dots\right) \times \tilde{\mathcal{F}}_{(q)}$ ist ein Graph $\bar{G}$ mit Zerlegung $\bar{\mathcal{G}} = \{\bar{G}_{\bar{\mu}_1}, \bar{G}_{\bar{\mu}_2}\}$ und W-Baum $\bar{\mathcal{T}} = (\bar{M}, \bar{A})$ und $\bar{M} = \{\bar{\mu}_1, \bar{\mu}_2\} = V\big(\mathcal{T}\setminus\bigcup_{1\leqslant r\leqslant q} \tilde{M}_{(r)}\big)$.

Nach Satz 6.23 gilt dann

$$\nu\left[\left(\left[(G \times \tilde{\mathcal{F}}_{(1)}) \times \tilde{\mathcal{F}}_{(2)}\right] \times \ldots\right) \times \tilde{\mathcal{F}}_{(q)}\right] = \sum_{\bar{\mu}_1,\bar{\mu}_2} \nu(G_\mu) - 1.$$

Dies bedeutet

$$\begin{aligned}
\nu(G) &= \sum_{1\leqslant r\leqslant q} \sum_{\mu\in\tilde{M}_{(r)}} (\nu(G_\mu) - 1) + \nu\left[\left(\left[(G \times \tilde{\mathcal{F}}_{(1)}) \times \tilde{\mathcal{F}}_{(2)}\right] \times \ldots\right) \times \tilde{\mathcal{F}}_{(q)}\right] \\
&= \sum_{1\leqslant r\leqslant q} \sum_{\mu\in\tilde{M}_{(r)}} (\nu(G_\mu) - 1) + \sum_{\bar{\mu}_1,\bar{\mu}_2} \nu(G_\mu) - 1 \\
&= \sum_{\mu\in\mathcal{T}\setminus\{\bar{\mu}_1,\bar{\mu}_2\}} (\nu(G_\mu) - 1) + \sum_{\bar{\mu}_1,\bar{\mu}_2} \nu(G_\mu) - 1 \\
&= \sum_{\mu\in\mathcal{T}\setminus\{\bar{\mu}_1,\bar{\mu}_2\}} \nu(G_\mu) - (N-2) + \sum_{\bar{\mu}_1,\bar{\mu}_2} \nu(G_\mu) - 1 \\
&= \sum_{\mu\in\mathcal{T}} \nu(G_\mu) - (N-1)
\end{aligned}$$

Somit ist die Aussage des Satzes bewiesen. □

Nun kann gezeigt werden, dass Algorithmus 6.2 für Graphen, die die Voraussetzungen von Satz 6.23 erfüllen, eine maximale Kreispackung ermittelt.

Satz 6.25
Sei $G = T \cup C$ ein Halin-Graph mit Zerlegung $\mathcal{G} = \{G_{\mu_1}, \ldots, G_{\mu_N}\}$ und $\mathcal{T} = (M, A)$ der zugehörige W-Baum. Sei $\delta_G(v)$ eine gerade Zahl für alle Knoten $v \in V(T)\setminus V(C)$. Dann bestimmt Algorithmus 6.2 eine maximale kantendisjunkte Kreispackung $\mathcal{Z}$.

Beweis: Sei $k \leqslant N$ und μ ein Blatt, welches in Iteration k bearbeitet wird. Dann werden für die drei Graphen

$$\begin{aligned}
\bar{G}_\mu &= G_\mu\setminus\{e_j \mid \ell_j \neq 1\} \\
G_{\mu,j_1} &= G_\mu\setminus(\{e_j \mid \ell_j = 0\} \cup \{e_{j_1}\}) \\
G_{\mu,j_2} &= G_\mu\setminus(\{e_j \mid \ell_j = 0\} \cup \{e_{j_2}\})
\end{aligned}$$

jeweils maximale Kreispackungen $\bar{Z}, Z_{j_1}, Z_{j_2}$ bestimmt. Da für alle Knoten $v \in V(T)\setminus V(C)$

$\delta_G(v)$ eine gerade Zahl ist, entspricht jede Splitkomponente $G_\mu \in \mathcal{G}$ einem geraden Wheel. Es gilt also $|Z_{j_1}| = |Z_{j_2}| > |\bar{Z}|$, das heißt es tritt Fall (iv) ein. Zu μ werden dann zwei disjunkte Kreispackungen $\bar{Z}_{j_1}, \bar{Z}_{j_2}$ der Größe $|\bar{Z}_{j_1}| = |\bar{Z}_{j_2}| = \nu(G_\mu) - 1$ ermittelt, von denen (später) genau eine ausgewählt wird. Algorithmus 6.2 bestimmt also eine kantendisjunkte Kreispackung $\mathcal{Z}$ mit maximaler Größe $|\mathcal{Z}| = \sum_{\mu \in \mathcal{T}} \nu(G_\mu) - (N-1)$. □

Bemerkung 6.26

Für Halin-Graphen $G = T \cup C$, die die Voraussetzungen von Satz 6.25 erfüllen, tritt ausschließlich Fall (iv) von Algorithmus 6.2 ein. Betrachte daher nun den entsprechenden Algorithmus 6.3. Die Laufzeit von Algorithmus 6.3 bleibt identisch zu Algorithmus 6.2 $O(|E(G)||M(\mathcal{T})|)$.

Algorithmus 6.3 Kreispackungsalgorithmus für Halin-Graphen entsprechend Satz 6.25

Eingabe: Zerlegung $\mathcal{G} = \{G_{\mu_1}, \ldots, G_{\mu_N}\}$ eines Halin-Graphen G mit Kantennummerierung, Häufigkeit ℓ_j aller Facetten $B_j \in \mathcal{B}(G)$ in der Zerlegung $\mathcal{G}$ und zugehöriger W-Baum $\mathcal{T}(G) = (M, A)$ mit $M = \{\mu_1, \ldots, \mu_N\}$.
Ausgabe: Kreispackung $\mathcal{Z}(G)$.

$\mathcal{Z}, Z_{j_1}, Z_{j_2}, \mathcal{L}, L \leftarrow \varnothing, k, k^* \leftarrow 0$
Wähle ein Blatt $\mu \in \mathcal{T}$.
while $\mu \neq \varnothing$ **do**
  $k \leftarrow k + 1$
  $\{j_1, j_2\} \leftarrow \{j \mid B_j \in \mathcal{B}(G_\mu)$ und $B'_j \in \mathcal{B}(G_{\mu'})$ für $(\mu, \mu') \in \mathcal{T}\}$
  Bestimme maximale Kreispackung Z_{j_1} von $G_\mu \backslash (\{e_j \mid \ell_j = 0\} \cup \{e_{j_1}\})$.
  Bestimme maximale Kreispackung Z_{j_2} von $G_\mu \backslash (\{e_j \mid \ell_j = 0\} \cup \{e_{j_2}\})$.
  $\bar{Z}_{j_1} \leftarrow \{B_j \mid \tilde{B}_j \in Z_{j_2} \backslash \{\tilde{B}_{j_1}\}, \bar{Z}_{j_2} \leftarrow \{B_j \mid \tilde{B}_j \in Z_{j_1} \backslash \{\tilde{B}_{j_2}\}\}$
  $L_k \leftarrow \{\bar{Z}_{j_1}, \bar{Z}_{j_2}\}$
  $\mathcal{L} \leftarrow \mathcal{L} \cup L_k$
  $\ell_j \leftarrow 0 \ \forall j \neq j_1, j_2$ mit $\tilde{B}_j \in \mathcal{B}(G_\mu)$
  $\ell_{j_i} \leftarrow \max\{0, \ell_{j_i} - 1\}$ für $i = 1, 2$
  $\mathcal{T} \leftarrow \mathcal{T} \backslash \mu$
  Wähle $\mu \in \mathcal{T}$ mit $\delta_{\mathcal{T}}(\mu) = 1$.
end while
Wähle $\mu \in \mathcal{T}$ mit $\delta_{\mathcal{T}}(\mu) = 0$.
Bestimme maximale Kreispackung $\bar{Z}$ von $G_\mu \backslash \{e_j \mid \ell_j \neq 1\}$.
$\mathcal{Z} \leftarrow$ SucheVergrösserung$(\mathcal{Z}, \{B_j \mid \tilde{B}_j \in \bar{Z}\}, \mathcal{L})$
$\mathcal{Z} \leftarrow \mathcal{Z} \cup \{B_j \mid \tilde{B}_j \in \bar{Z}\}$
$\ell_j \leftarrow 0 \ \forall j$ mit $\tilde{B}_j \in \mathcal{B}(G_\mu)$
while $\mathcal{L} \neq \varnothing$ **do**
  $k^* \leftarrow \max\{k^* \mid L_{k*} \in \mathcal{L}\}$
  Wähle $\bar{Z} \in L_{k*}$.
  $\mathcal{Z} \leftarrow \mathcal{Z} \cup \bar{Z}$
  $\mathcal{L} \leftarrow \mathcal{L} \backslash L_{k*}$
  $\mathcal{Z} \leftarrow$ SucheVergrösserung$(\mathcal{Z}, \bar{Z}, \mathcal{L})$
end while
return $\mathcal{Z}$

Beispiel 6.27

Abbildung 6.14 zeigt einen Halin-Graphen $G = T \cup C$, bei dem für jedes $v \in V(T)\backslash V(C)$ der Knotengrad $\delta_G(v)$ eine gerade Zahl ist, die Zerlegung $\mathcal{G} = \{G'_1, \ldots, G'_6\}$ sowie die Nummerierung der Kanten $e \in C$. Tabelle 6.3 zeigt in Zeile ℓ_j wie oft jede Facette B_j von G durch eine Facette $\tilde{B}_j$ in einer Komponente G'_μ der Zerlegung $\mathcal{G}$ repräsentiert wird.

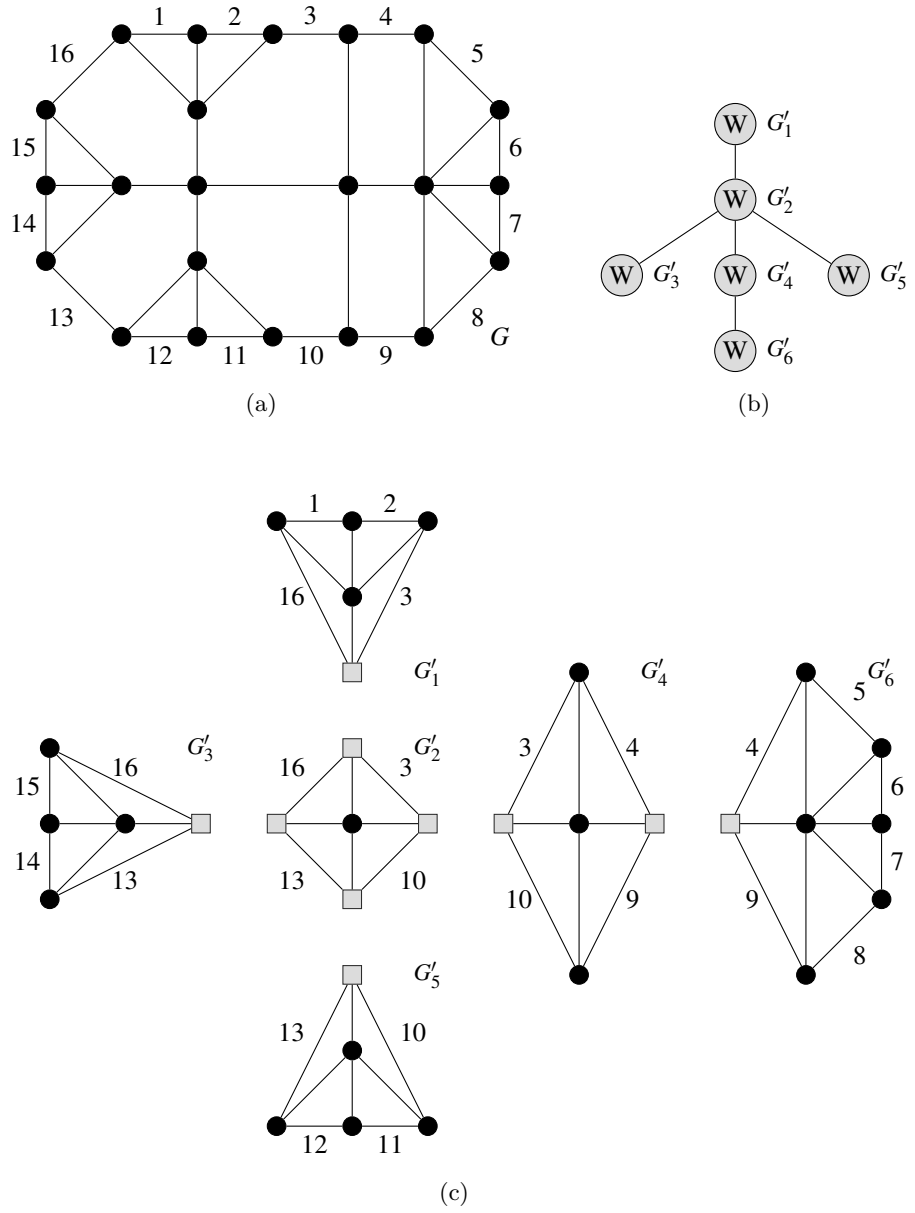


Abbildung 6.14: Halin-Graph G (a) mit Nummerierung der Kanten, die zugehörige Zerlegung (c) mit Nummerierung der Kanten und der zugehörige W-Baum (b).

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
ℓ_j	1	1	3	2	1	1	1	1	2	3	1	1	3	1	1	3
It. 1	0	0	2													2
It. 2													2	0	0	1
It. 3										2	0	0	1			
It. 4				1	0	0	0	0	1							
It. 5			1							1			0			0
It. 6			0	0					0	0						

Tabelle 6.3: Der Verlauf der Werte ℓ_j bei Anwendung von Algorithmus 6.3 auf Beispiel 6.27.

Der Baum $\mathcal{T}$ enthält als Blätter $\mu_i, i \in \{1,3,5,6\}$, wobei alle zugehörigen Splitkomponenten G'_i gerade Wheels sind. Das zuerst ausgewählte Blatt ist μ_1. Die Facetten $\tilde{B}_{16}, \tilde{B}_3$ sind die Kontaktfacetten von G'_1 und es gilt $\ell_{16} = 3 > 0$ und $\ell_3 = 3 > 0$. Der Algorithmus bestimmt $Z_{16} = \{B_1, B_3\}$ und $Z_3 = \{B_2, B_{16}\}$. $Z_{16}\backslash\{B_3\}$ sowie $Z_3\backslash\{B_{16}\}$ werden in $\mathcal{L}$ zur späteren Auswahl abgespeichert (vergleiche Tabelle 6.4). Der Algorithmus setzt daraufhin $\ell_1 = \ell_{12} = 0$ und $\ell_{16} = \ell_3 = 2$ wie in Tabelle 6.3 Iteration 1 angegebenen. Das Blatt μ_1 wird im Anschluss aus dem Baum $\mathcal{T}$ gelöscht. Bei der Wahl von μ_3 in Iteration 2, μ_5 in Iteration 3 und μ_6 in Iteration 4 ist das Vorgehen des Algorithmus identisch zu Iteration 1. Auch hier ist der Verlauf der Iterationen in den Tabellen 6.3 und 6.4 beschrieben. Auch die Blätter μ_3, μ_5 und μ_6 werden im Anschluss an die jeweilige Iteration aus $\mathcal{T}$ gelöscht. Nach Iteration 3 ist μ_2 ein Blatt in $\mathcal{T}$ und nach Iteration 4 ist auch μ_4 ein Blatt in $\mathcal{T}$. In Iteration 5 wird nun μ_2 ausgewählt. Erneut ist das Vorgehen identisch zu Iteration 1 und wird in Tabelle 6.2 beschrieben. Im Anschluss wird μ_2 aus $\mathcal{T}$ gelöscht.

	μ_i	L_k	$\mathcal{L}$
It. 1	μ_1	$L_1 = \{\bar{Z}_3 = \{B_1\}, \bar{Z}_{16} = \{B_2\}\}$	$\{L_1\}$
It. 2	μ_3	$L_2 = \{\bar{Z}_{16} = \{B_{14}\}, \bar{Z}_{13} = \{B_{15}\}\}$	$\{L_1, L_2\}$
It. 3	μ_5	$L_3 = \{\bar{Z}_{10} = \{B_{12}\}, \bar{Z}_{13} = \{B_{11}\}\}$	$\{L_1, L_2, L_3\}$
It. 4	μ_6	$L_4 = \{\bar{Z}_9 = \{B_5, B_7\}, \bar{Z}_4 = \{B_6, B_8\}\}$	$\{L_1, L_2, L_3, L_4\}$
It. 5	μ_2	$L_5 = \{\bar{Z}_3 = \{B_{13}\}, \bar{Z}_{10} = \{B_{16}\}\}$	$\{L_1, L_2, L_3, L_4, L_5\}$
It. 6	μ_4		$\{L_1, L_2, L_3, L_4, L_5\}$

Tabelle 6.4: Iterationen bei Anwendung von Algorithmus 6.3 auf Beispiel 6.27.

In der letzten Iteration ist die Menge der Blätter in $\mathcal{T}$ leer, der Baum enthält nur noch den Knoten μ_4. Der Algorithmus bestimmt $\bar{Z} = \{B_3, B_9\}$ (eine Alternative wäre $\bar{Z} = \{B_4, B_10\}$). Da $\mathcal{L} \neq \varnothing$ ist, wird $\mathcal{Z}$ mithilfe der Funktion SucheVergrösserung$(\mathcal{Z}, \bar{Z}, \mathcal{L})$ nun iterativ vergrößert. Zu der Facette $B_3 \in Z$ existiert eine Packung $\bar{Z}_3 \in L_5 \in \mathcal{L}$ mit

$\bar{Z}_3 = \{B_{13}\}$. Die Kreispackung $\mathcal{Z}$ wird zu $\mathcal{Z} = \{B_3, B_9, B_{13}\}$ erweitert, die Facette B_{13} wird zur Menge Z_{neu} hinzugefügt und die Menge L_5 wird aus $\mathcal{L}$ gelöscht. Für die Menge $Z \cup Z_{neu}$ wird in der nächsten Iteration dann erneut nach einer Erweiterung gesucht. Die weiteren Iterationen sind in Tabelle 6.5 dargestellt. Nach Abschluss dieser Iterationen wird $\mathcal{Z}$ um die Menge $\bar{Z} = \{B_3, B_9\}$ zu $\mathcal{Z} = \{B_{13}, B_1, B_5, B_7, B_{11}, B_{15}, B_3, B_9\}$ erweitert.

	$\mathcal{Z}$	Z_{neu}	$\mathcal{L}$
It. 6.1	$\varnothing$	$\varnothing$	$\{L_1, L_2, L_3, L_4, L_5\}$
It. 6.2	$\{B_{13}\}$	$\{B_{13}\}$	$\{L_1, L_2, L_3, L_4\}$
It. 6.3	$\{B_{13}, B_1\}$	$\{B_{13}, B_1\}$	$\{L_2, L_3, L_4\}$
It. 6.4	$\{B_{13}, B_1, B_5, B_7\}$	$\{B_{13}, B_1, B_5, B_7\}$	$\{L_2, L_3\}$
It. 6.5	$\{B_{13}, B_1, B_5, B_7, B_{11}\}$	$\{B_{13}, B_1, B_5, B_7, B_{11}\}$	$\{L_2\}$
It. 6.6	$\{B_{13}, B_1, B_5, B_7, B_{11}, B_{15}\}$	$\{B_{13}, B_1, B_5, B_7, B_{11}, B_{15}\}$	$\varnothing$

Tabelle 6.5: Iterationen der Funktion SucheVergrösserung$(\mathcal{Z}, Z, \mathcal{L})$ bei Anwendung von Algorithmus 6.3 auf Beispiel 6.27.

Der Algorithmus hat nun die in Abbildung 6.15 farblich markierte Kreispackung bestimmt, welche durch die Facetten $\mathcal{Z} = \{B_{13}, B_1, B_5, B_7, B_{11}, B_{15}, B_3, B_9\}$ induziert wird. Es gilt $\nu(G) = 8$.

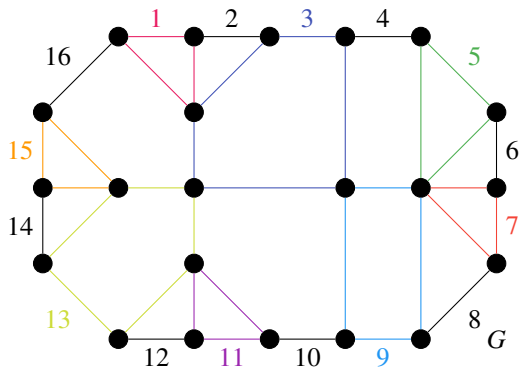


Abbildung 6.15: Die maximale Kreispackung des Halin-Graphen G wird induziert durch $\mathcal{Z} = \{B_{13}, B_1, B_5, B_7, B_{11}, B_{15}, B_3, B_9\}$.

Für Halin-Graphen, deren Zerlegung $\mathcal{G}$ ausschließlich Komponenten G_μ enthält, welche isomorph zu einem geraden Wheel sind, kann mithilfe von Algorithmus 6.2 eine optimale Lösung für das maximale kantendisjunkte Kreispackungsproblem bestimmt werden. Dieses wird auch für solche Halin-Graphen vermutet, deren Zerlegung $\mathcal{G}$ Komponenten G_μ enthält, welche isomorph zu einem ungeraden Wheel sind. Erste Beispielinstanzen stützen eine derartige Vermutung.

Kapitel 7

Fazit

In dieser Arbeit wurde das Problem der Bestimmung einer maximalen kantendisjunkten Kreispackung bzw. der Bestimmung der Kreispackungszahl von Graphen untersucht. Die Bedeutung dieser theoretischen Problemstellung wurde durch die Vorstellung dreier Anwendungen deutlich, deren Modellierung zur Bestimmung maximaler kantendisjunkter Kreispackungen führt. Da die Bestimmung solcher maximalen kantendisjunkten Kreispackungen sowie die Bestimmung der Kreispackungszahl NP-schwer sind, bestand die grundlegende Idee dieser Arbeit darin, einen Graphen G zunächst auf bestimmte Weise in „kleinere Graphen“ G_i zu zerlegen, maximale Kreispackungen dieser G_i zu bestimmen und daraufhin einen Bezug zur maximalen Kreispackung von G zu finden. Dieser systematische Ansatz ergab sich aus der Betrachtung von Kreispackungen in nicht zusammenhängenden, 1-zusammenhängenden und 2-zusammenhängenden Graphen.

In Vorbereitung auf den nächsten naheliegenden Schritt, die Betrachtung von 3-zusammenhängenden Graphen, erfolgte eine genaue Analyse der Ergebnisse für 2-zusammenhängende Graphen. Hierfür wurden die SPQR-Zerlegung für 2-zusammenhängende Graphen und ein Algorithmus zur Bestimmung einer kantendisjunkten Kreispackung unter Verwendung der SPQR-Zerlegung vorgestellt. Speziell für serienparallele Graphen konnten die bestehenden Ergebnisse im Rahmen dieser Arbeit um eine praktische Evaluation ergänzt werden: Der auf der SPQR-Zerlegung basierende Algorithmus, welcher für serienparallele Graphen in polynomieller Zeit eine optimale Lösung des Kreispackungsproblems bestimmt, wurde unter Verwendung des *Open Graph algorithms and Data structures Framework (OGDF)* implementiert und getestet. Ein wesentliches Ergebnis dieser Evaluation ist, dass das Verfahren auch für große serienparallele Graphen mit bis zu 1.000.000 Kanten skaliert.

Bei der Betrachtung von 3-zusammenhängenden Graphen wurde festgestellt, dass es für diese keine geeignete eindeutige Zerlegung entsprechend der SPQR-Zerlegung für 2-zusammenhängende Graphen gibt. Allerdings konnte eine entsprechende Zerlegung für minimal 3-zusammenhängende Graphen in der Literatur gefunden werden, welche vorgestellt und für die weitere Arbeit verwendet wurde. Da aufgrund der Ergebnisse für serienparallele Graphen die Hoffnung bestand, dass auch für eine Teilmenge der minimal 3-zusammenhängenden Graphen Verfahren zur Bestimmung von maximalen Kreispackungen mit polynomiellen Laufzeiten gefunden werden können, wurden Halin-Graphen untersucht. Bei Halin-Graphen handelt es sich um minimal 3-zusammenhängende Graphen und für diese sind, ähnlich wie für serienparallele Graphen, einige Probleme bekannt, die in Polynomialzeit gelöst werden können.

Es wurde dann zunächst gezeigt, dass eine direkte Beziehung zwischen den in der Zerlegung verwendeten guten 3-Splits und Fans in Halin-Graphen besteht. Dieses Ergebnis war wesentlich für die spätere Entwicklung eines Algorithmus zur Bestimmung von Kreispackungen in Halin-Graphen. Zunächst konnten jedoch sukzessive Ergebnisse zu Kreispackungen spezieller Halin-Graphen gezeigt werden: Für Necklace-Halin-Graphen kann die Kreispackungszahl aufgrund der Grapheigenschaften abhängig von der Ordnung n und der Länge k des zugehörigen Weges P des Graphen bestimmt werden. Eine entsprechende maximale Kreispackung wurde ebenfalls angegeben. Für Caterpillar-Halin-Graphen konnte die Bestimmung der Kreispackungszahl auf die Bestimmung der Unabhängigkeitszahl des serienparallelen Dualgraphen zurückgeführt werden. So konnte gezeigt werden, dass die Kreispackungszahl sowie eine maximale Kreispackung für diese Graphklasse in linearer Zeit bestimmt werden kann. Zusätzlich konnte dann für solche Caterpillar-Halin-Graphen, deren Komponenten der Zerlegung isomorph zu geraden Wheels sind, die Kreispackungszahl auf die Kreispackungszahlen der Komponenten in der Zerlegung zurückgeführt werden. Da es sich hierbei um ein konstruktives Ergebnis handelte, konnte ein entsprechender Polynomialzeitalgorithmus zur Bestimmung einer maximalen Kreispackung formuliert werden. Dieses Verfahren wurde dann verwendet, um unter Ausnutzung von Eigenschaften der Facetten in Halin-Graphen einen Polynomialzeitalgorithmus zur Bestimmung von (möglicherweise maximalen) Kreispackungen in allgemeinen Halin-Graphen zu entwickeln. Für Halin-Graphen, deren Komponenten der Zerlegung isomorph zu geraden Wheels sind, konnte das Ergebnis ebensolcher Caterpillar-Halin-Graphen verallgemeinert werden, sodass auch für diese Graphen die Kreispackungszahl auf die Kreispackungszahlen der Komponenten in der Zerlegung zurückgeführt werden konnte. Es konnte gezeigt werden, dass der zuvor entwickelte Algorithmus für diese speziellen Halin-Graphen eine maximale Kreispackung bestimmt. Es

wird vermutet, dass der Algorithmus auch für allgemeine Halin-Graphen eine optimale Lösung des maximalen Kreispackungsproblems bestimmt.

In der weiteren Forschung wäre dann die Prüfung dieser Vermutung ein naheliegender nächster Schritt. Darüber hinaus wäre dem Aufbau dieser Arbeit folgend die Betrachtung von allgemeinen 3-zusammenhängenden Graphen und die Nutzung anderer dafür geeigneter Zerlegungen interessant. Beispielsweise könnte untersucht werden, ob die Zerlegung von Eppstein und Reed geeignet zur Bestimmung einer Kreispackungszahl für 3-zusammenhängende planare Graphen ist.

Literaturverzeichnis

[Aka14] Islam Akaria. „Packing Triangles in β-regular Tournaments“. Dissertation. University of Haifa, 2014 (siehe Seite 8).

[ALS91] Stefan Arnborg, Jens Lagergren und Detlef Seese. „Easy problems for tree-decomposable graphs“. In: *Journal of Algorithms* 12.2 (1991), Seiten 308–340 (siehe Seite 3).

[Arn85] Stefan Arnborg. „Efficient algorithms for combinatorial problems on graphs with bounded decomposability—a survey“. In: *BIT Numerical Mathematics* 25.1 (1985), Seiten 1–23 (siehe Seite 3).

[AY15] Islam Akaria und Raphael Yuster. „Packing edge-disjoint triangles in regular and almost regular tournaments“. In: *Discrete Mathematics* 338.2 (2015), Seiten 217–228 (siehe Seite 8).

[AZ11] Spyridon Antonakopoulos und Lisa Zhang. „Approximation algorithms for grooming in optical network design“. In: *Theoretical Computer Science* 412.29 (2011), Seiten 3738–3751 (siehe Seiten 20–22).

[Bal03] PAUL Balister. „Packing Digraphs with Directed Closed Trails“. In: *Combinatorics, Probability and Computing* 12.1 (2003), Seiten 1–15 (siehe Seite 26).

[Bar+06] Ziv Bar-Yossef, Yitzhak Birk, T. S. Jayram und Tomer Kol. „Index Coding with Side Information“. In: *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*. FOCS '06. USA: IEEE Computer Society, 2006, Seiten 197–206 (siehe Seiten 14, 15).

[BBT18] Stéphane Bessy, Marin Bougeret und Jocelyn Thiebaut. *(Arc-disjoint) cycle packing in tournament: classical and parameterized complexity*. 2018. arXiv: `1802.06669` (siehe Seite 8).

[Bes+19] Stéphane Bessy, Marin Bougeret, R. Krithika, Abhishek Sahu, Saket Saurabh, Jocelyn Thiebaut und Meirav Zehavi. „Packing Arc-Disjoint Cycles in Tournaments“. In: *44th International Symposium on Mathematical Foundations of Computer Science (MFCS 2019)*. Herausgegeben von Peter Rossmanith, Pinar Heggernes und Joost-Pieter Katoen. Band 138. Leibniz International Proceedings in Informatics (LIPIcs). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019, 27:1–27:14 (siehe Seite 8).

[BHJ20] Henning Bruhn, Matthias Heinlein und Felix Joos. „The Edge-Erdős-Pósa Property“. In: *Combinatorica* (2020), Seiten 1–27 (siehe Seite 7).

[BHP14] Darryn Bryant, Daniel Horsley und William Pettersson. „Cycle decompositions V: Complete graphs into cycles of arbitrary lengths“. In: *Proceedings of the London Mathematical Society* 108.5 (2014), Seiten 1153–1192 (siehe Seite 7).

[Bir+21] Péter Biró, Joris van de Klundert, David Manlove, William Pettersson, Tommy Andersson, Lisa Burnapp, Pavel Chromy, Pablo Delgado, Piotr Dworczak, Bernadette Haase, Aline Hemke, Rachel Johnson, Xenia Klimentova, Dirk Kuypers, Alessandro Nanni Costa, Bart Smeulders, Frits Spieksma, María O. Valentín und Ana Viana. „Modelling and optimisation in European Kidney Exchange Programmes“. In: *European Journal of Operational Research* 291.2 (2021), Seiten 447–456 (siehe Seite 7).

[BK98] Yitzhak Birk und T. Kol. „Informed-source coding-on-demand (ISCOD) over broadcast channels“. In: *1998 IEEE INFOCOM Proceedings*. Band 3. Jan. 1998, Seiten 1257–1264 (siehe Seite 13).

[BL11] Y. Berliner und M. Langberg. „Index coding with outerplanar side information“. In: *2011 IEEE International Symposium on Information Theory Proceedings*. 2011, Seiten 806–810 (siehe Seite 18).

[BMR09] Péter Biró, David Manlove und Romeo Rizzi. „Maximum weight cycle packing in directed graphs, with application to kidney exchange programs“. In: *Discrete Mathematics, Algorithms and Applications* 01.04 (2009), Seiten 499–517 (siehe Seite 7).

[BMS06] Anne Bergeron, Julia Mixtacki und Jens Stoye. „A Unifying View of Genome Rearrangements“. In: Band 4175. Sep. 2006, Seiten 163–173 (siehe Seiten 25, 26, 29, 32).

[Bod94] Hans L Bodlaender. „A tourist guide through treewidth“. In: *Acta cybernetica* 11.1-2 (1994), Seite 1 (siehe Seite 3).

[BPT05] R. Borie, R. Parker und C. Tovey. „Automatic generation of linear-time algorithms from predicate calculus descriptions of problems on recursively constructed graph families“. In: *Algorithmica* 7 (2005), Seiten 555–581 (siehe Seite 65).

[Bry+15] Darryn Bryant, Daniel Horsley, Barbara Maenhaut und Benjamin R. Smith. *Decompositions of complete multigraphs into cycles of varying lengths*. 2015. arXiv: `1508.00645` (siehe Seite 7).

[Cam17] Rosalind A. Cameron. *Cycle packings of the complete multigraph*. 2017. arXiv: `1701.05287` (siehe Seite 7).

[Cap97] Alberto Caprara. „Sorting by Reversals is Difficult“. In: *Proceedings of the First Annual International Conference on Computational Molecular Biology*. RECOMB '97. Association for Computing Machinery, 1997, Seiten 75–83 (siehe Seiten 3, 26).

[Car+20] Margarida Carvalho, Xenia Klimentova, Kristiaan Glorie, Ana Viana und Miguel Constantino. „Robust Models for the Kidney Exchange Problem“. In: *INFORMS Journal on Computing* (2020) (siehe Seite 7).

[CE80] William H. Cunningham und Jack Edmonds. „A Combinatorial Decomposition Theory“. In: *Canadian Journal of Mathematics* 32.3 (1980), Seiten 734–765 (siehe Seite 10).

[CGW93] C. R. Coullard, L. L. Gardner und D. K. Wagner. „Decomposition of 3-connected graphs“. In: *Combinatorica* 13.1 (1993), Seiten 7–30 (siehe Seiten 10, 66, 67, 72, 77, 79, 107).

[Cha+11] M. A. R. Chaudhry, Z. Asad, A. Sprintson und M. Langberg. „On the complementary Index Coding problem“. In: *2011 IEEE International Symposium on Information Theory Proceedings*. 2011, Seiten 244–248 (siehe Seiten vii, 15–18).

[Chi+13] M. Chimani, C. Gutwenger, M. Jünger, G. W. Klau, K. Klein und P. Mutzel. „The Open Graph Drawing Framework (OGDF)“. In: *Handbook of Graph Drawing and Visualization*. Herausgegeben von R. Tamassia. CRC Press, 2013. Kapitel 17, Seiten 543–570 (siehe Seiten 4, 59).

[Chi09] Markus Chimani. „Computing crossing numbers“. Dissertation. Fakultät für Informatik, TU Dortmund, 2009. URL: `http://hdl.handle.net/2003/25955` (siehe Seiten 48, 58).

[CM00] A. L. Chiu und E. H. Modiano. „Traffic grooming algorithms for reducing electronic multiplexing costs in WDM ring networks“. In: *Journal of Lightwave Technology* 18.1 (2000) (siehe Seite 21).

[CM87] Charles J. Colbourn und Rudolf Mathon, Herausgeber. Band 149. North-Holland Mathematics Studies. Elsevier, 1987 (siehe Seite 2).

[CNP83] G. Cornuéjols, D. Naddef und W.R. Pulleyblank. „Halin Graphs and the Travelling Salesman Problem“. In: *Mathematical Programming* 26.3 (Okt. 1983), Seiten 287–294 (siehe Seiten 10, 65, 80–82, 115).

[CNP85] G. Cornuéjols, D. Naddef und W.R. Pulleyblank. „The Travelling Salesman Problem in Graphs with 3-edge Cutsets“. In: *Journal of the ACM* 32.2 (Apr. 1985), Seiten 383–410 (siehe Seite 10).

[Con+13] Miguel Constantino, Xenia Klimentova, Ana Viana und Abdur Rais. „New insights on integer-programming models for the kidney exchange problem“. In: *European Journal of Operational Research* 231.1 (2013), Seiten 57–68 (siehe Seite 7).

[CPR03] A. Caprara, A. Panconesi und R. Rizzi. „Packing cycles in undirected graphs“. In: *Journal of Algorithms* 48.1 (2003), Seiten 239–256 (siehe Seiten 2, 9, 26, 43, 44).

[CR02] Alberto Caprara und Romeo Rizzi. „Packing triangles in bounded degree graphs“. In: *Information Processing Letters* 84.4 (2002), Seiten 175–180 (siehe Seite 9).

[CTW09] Z.-Z. Chen, R. Tanahashi und L. Wang. „An improved randomized approximation algorithm for maximum triangle packing“. In: *Discrete Applied Mathematics* 157.7 (2009), Seiten 1640–1646 (siehe Seite 9).

[DE63] G. Dirac und P. Erdős. „On the maximal number of independent circuits in a graph“. In: *Acta Mathematica Academiae Scientiarum Hungarica* 14.1-2 (1963), Seiten 79–94 (siehe Seite 1).

[Dic+16] John P. Dickerson, David F. Manlove, Benjamin Plaut, Tuomas Sandholm und James Trimble. „Position-Indexed Formulations for Kidney Exchange“. In: *Proceedings of the 2016 ACM Conference on Economics and Computation*. EC '16. New York, NY, USA: Association for Computing Machinery, 2016, Seiten 25–42. ISBN: 9781450339360 (siehe Seite 7).

[Dso16] Kimberly Sevin D'souza. „Excluding a Weakly 4-connected Minor“. In: (2016) (siehe Seiten 10, 11).

[DT89] G. Di Battista und R. Tamassia. „Incremental planarity testing“. In: *Foundations of Computer Science, 1989., 30th Annual Symposium on*. Okt. 1989, Seiten 436–441 (siehe Seiten 10, 44, 48).

[DT92] Dorit Dor und Michael Tarsi. „Graph Decomposition is NPC - a Complete Proof of Holyer's Conjecture“. In: *Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing*. STOC '92. Association for Computing Machinery, 1992, Seiten 252–263 (siehe Seite 2).

[ECS07] S. Y. El Rouayheb, M. A. R. Chaudhry und A. Sprintson. „On the Minimum Number of Transmissions in Single-Hop Wireless Coding Networks“. In: *2007 IEEE Information Theory Workshop*. 2007, Seiten 120–125 (siehe Seite 14).

[EG59] Paul Erdős und Tibor Gallai. „On maximal paths and circuits of graphs“. In: *Acta Mathematica Academiae Scientiarum Hungarica* 10.3-4 (1959), Seiten 337–356 (siehe Seite 1).

[EK05] Cesim Erten und Stephen Kobourov. „Simultaneous Embedding of a Planar Graph and Its Dual on the Grid“. In: *Theory of Computing Systems* 38.3 (Mai 2005), Seiten 313–327 (siehe Seite 90).

[EP62] P. Erdős und L. Pósa. „On the maximal number of disjoint circuits of a graph“. In: *Publicationes Mathematicae Debrecen* 9 (1962), Seiten 3–12 (siehe Seite 1).

[EP65] p. Erdös und L. Pósa. „On Independent Circuits Contained in a Graph“. In: *Canadian Journal of Mathematics* 17 (1965), Seiten 347–352 (siehe Seiten 2, 7).

[ER19] David Eppstein und Bruce Reed. „Finding maximal sets of laminar 3-separators in planar graphs in linear time“. In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2019, Seiten 589–605 (siehe Seiten 12, 127).

[FS11] Z. Friggstad und M. R. Salavatipour. „Approximability of Packing Disjoint Cycles“. In: *Algorithmica* 60.2 (2011), Seiten 395–400 (siehe Seite 9).

[FT93] Jean-Luc Fouquet und Henri Thuillier. „Decomposition of 3-connected cubic graphs“. In: *Discrete mathematics* 114.1-3 (1993), Seiten 181–198 (siehe Seite 10).

[Gar89] Leah Leslie Gardner. „Studies in 3-connected Graphs: A Decomposition Theory and a Decomposition-based Optimization Algorithm“. Dissertation. 1989 (siehe Seiten vii, 10, 12, 65, 67, 69, 77–79).

[GG95] L. L. Gardner und J. G. del Greco. „A Note on Y-Δ and Δ-Y Graphs“. In: *The Journal of Combinatorial Mathematics and Combinatorial Computing* 19 (1995), Seiten 259–272.

[GM01] C. Gutwenger und P. Mutzel. „A Linear Time Implementation of SPQR-Trees“. In: *Graph Drawing*. Herausgegeben von J. Marks. Band 1984. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2001, Seiten 77–90 (siehe Seite 58).

[Gro16] Martin Grohe. „Quasi-4-Connected Components“. In: (2016). arXiv: 1602.04505 (siehe Seite 11).

[Gus97] Dan Gusfield. „Models of Genome-Level Mutations“. In: *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press, 1997, Seiten 492–500 (siehe Seite 25).

[Gut10] C. Gutwenger. „Application of SPQR-trees in the planarization approach for drawing graphs“. In: (2010) (siehe Seiten vii, 42, 44, 49, 50).

[Har+10a] J. Harant, D. Rautenbach, P. Recht und F. Regen. „Packing edge-disjoint cycles in graphs and the cyclomatic number“. In: *Discrete Mathematics* 310.9 (2010). Cycles and Colourings 2008, Seiten 1456–1462 (siehe Seite 8).

[Har+10b] J. Harant, D. Rautenbach, P. Recht, I. Schiermeyer und E.-M. Sprengel. „Packing disjoint cycles over vertex cuts“. In: *Discrete Mathematics* 310.13–14 (2010), Seiten 1974–1978 (siehe Seite 8).

[Hei+20] Irene Heinrich, Till Heller, Eva Schmidt und Manuel Streicher. „2.5-connectivity: unique components, critical graphs, and applications“. In: *International Workshop on Graph-Theoretic Concepts in Computer Science*. Springer. 2020, Seiten 352–363 (siehe Seiten 2, 8, 9).

[Hor11] Daniel Horsley. „Maximum packings of the complete graph with uniform length cycles“. In: *Journal of Graph Theory* 68.1 (2011), Seiten 1–7 (siehe Seite 7).

[HT73a] J. E. Hopcroft und R. E. Tarjan. „Dividing a Graph into Triconnected Components.“ In: *SIAM J. Comput.* 2.3 (1973), Seiten 135–158 (siehe Seiten 8–10, 44, 49).

[HT73b] John Hopcroft und Robert Tarjan. „Algorithm 447: Efficient Algorithms for Graph Manipulation“. In: *Commun. ACM* 16.6 (Juni 1973), Seiten 372–378 (siehe Seiten 36, 38, 41).

[IR78] Alon Itai und Michael Rodeh. „Finding a minimum circuit in a graph“. In: *SIAM Journal on Computing* 7.4 (1978), Seiten 413–423 (siehe Seite 44).

[Jah09] Jürgen Jahns. *Photonik*. Berlin, Boston: De Gruyter, 2009. ISBN: 978-3-486-59384-6 (siehe Seite 19).

[JK20] Ajay Saju Jacob und R Krithika. „Packing Arc-Disjoint Cycles in Bipartite Tournaments“. In: *International Workshop on Algorithms and Computation*. Springer. 2020, Seiten 249–260 (siehe Seite 8).

[Kan+91] A. Kanevsky, R. Tamassia, G. Di Battista und J. Chen. „On-line maintenance of the four-connected components of a graph“. In: *[1991] Proceedings 32nd Annual Symposium of Foundations of Computer Science*. 1991, Seiten 793–801 (siehe Seiten 10, 11).

[Kar05] DV Karpov. „Blocks in k-connected graphs“. In: *Journal of Mathematical Sciences* 126.3 (2005), Seiten 1167–1181 (siehe Seite 11).

[Kar16] DV Karpov. „The Tree of Cuts and Minimal k-Connected Graphs“. In: *Journal of Mathematical Sciences* 212.6 (2016), Seiten 654–665 (siehe Seite 11).

[Kar20] DV Karpov. „On the Structure of a 3-Connected Graph. 2“. In: *Journal of Mathematical Sciences* 247.3 (2020), Seiten 406–437 (siehe Seiten 11, 12).

[KAV14] Xenia Klimentova, Filipe Alvelos und Ana Viana. „A New Branch-and-Price Approach for the Kidney Exchange Problem“. In: *Computational Science and Its Applications – ICCSA 2014*. Herausgegeben von Beniamino Murgante, Sanjay Misra, Ana Maria A. C. Rocha, Carmelo Torre, Jorge Gustavo Rocha, Maria Irene Falcão, David Taniar, Bernady O. Apduhan und Osvaldo Gervasi. Cham: Springer International Publishing, 2014, Seiten 237–252 (siehe Seite 7).

[Kaw+20] Ken-ichi Kawarabayashi, Stephan Kreutzer, O-joung Kwon und Qiqin Xie. *Half-integral Erdős-Pósa property of directed odd cycles*. 2020. arXiv: 2007.12257 (siehe Seite 7).

[Kir47] TP Kirkman. „On a problem in combinations, Camb. and Dublin Math“. In: *J* 2 (1847), Seiten 191–204 (siehe Seite 1).

[Kőn15] Dénes Kőnig. „Vonalrendszerek és determinánsok“. In: *Mathematikai és Természettudományi Értesítö* 33 (1915). [Deutsche Übersetzung des Titels: „Graphen und Determinanten“], Seiten 221–229 (siehe Seite 1).

[Kor94] N. M. Korneyenko. „Combinatorial algorithms on a class of graphs“. In: *Discrete Applied Mathematics* 54.2–3 (1994), Seiten 215–217 (siehe Seite 56).

[KP03] DV Karpov und AV Pastor. „On the structure of a k-connected graph“. In: *Journal of Mathematical Sciences* 113.4 (2003), Seiten 584–597 (siehe Seite 11).

[KP12] D. V. Karpov und A. V. Pastor. „The structure of a decomposition of a triconnected graph“. In: *Journal of Mathematical Sciences* 184.5 (2012), Seiten 601–628 (siehe Seiten 11, 12).

[Kri+07] M. Krivelevich, Z. Nutov, M. R. Salavatipour, J. Verstraete und R. Yuster. „Approximation algorithms and hardness results for cycle packing problems“. In: *ACM Transactions on Algorithms (TALG)* 3.4 (2007), Seite 48 (siehe Seiten 2, 9, 17, 43).

[Kri+18] R. Krithika, Abhishek Sahu, Saket Saurabh und Meirav Zehavi. *The Parameterized Complexity of Packing Arc-Disjoint Cycles in Tournaments*. 2018. arXiv: `1802.07090` (siehe Seite 8).

[Lin+19] Mugang Lin, Jianxin Wang, Qilong Feng und Bin Fu. „Randomized Parameterized Algorithms for the Kidney Exchange Problem“. In: *Algorithms* 12.2 (2019) (siehe Seite 7).

[LM20] Edward Lam und Vicky Mak-Hau. „Branch-and-cut-and-price for the cardinality-constrained multi-cycle problem in kidney exchange“. In: *Computers & Operations Research* 115 (2020) (siehe Seite 7).

[Lok+19] Daniel Lokshtanov, Amer E. Mouawad, Saket Saurabh und Meirav Zehavi. „Packing Cycles Faster Than Erdos–Posa“. In: *SIAM Journal on Discrete Mathematics* 33.3 (2019), Seiten 1194–1215 (siehe Seite 7).

[LS08] M. Langberg und A. Sprintson. „On the hardness of approximating the network coding capacity“. In: *2008 IEEE International Symposium on Information Theory*. 2008, Seiten 315–319 (siehe Seite 14).

[Luc92] Edouard Lucas. *Récréations Mathématiques, Tôme II*. 1892 (siehe Seite 1).

[Mac37] S. Mac Lane. „A structural characterization of planar combinatorial graphs“. In: *Duke Mathematical Journal* 3.3 (Sep. 1937), Seiten 460–472 (siehe Seite 44).

[NS18] Diego Nicodemos und Matěj Stehlík. „Packing and covering odd cycles in cubic plane graphs with small faces“. In: *European Journal of Combinatorics* 67 (2018), Seiten 208–221 (siehe Seite 8).

[NY04] Z. Nutov und R. Yuster. „Packing Directed Cycles Efficiently“. In: *Mathematical Foundations of Computer Science 2004*. Herausgegeben von J. Fiala, V. Koubek und J. Kratochvíl. Band 3153. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2004, Seiten 310–321 (siehe Seite 9).

[OR19] Christin Otto und Peter Recht. „Maximum cycle packing using SPR-trees“. In: *EJGTA* 7 (2019), Seiten 147–155 (siehe Seiten 18, 43).

[Ott14] Ch. Otto. „Maximum Cycle Packing Probleme - Heuristiken und exakte Lösungen für spezielle Graphen“. Masterarbeit. Technische Universität Dortmund, 2014 (siehe Seiten 2, 4, 8, 9, 43, 44).

[Pas18] AV Pastor. „On the Decomposition of a 3-Connected Graph into Cyclically 4-Edge-Connected Components“. In: *Journal of Mathematical Sciences* 232.1 (2018), Seiten 61–83 (siehe Seite 11).

[Ped14] João Pedro Pedroso. „Maximizing Expectation on Vertex-Disjoint Cycle Packing“. In: *Computational Science and Its Applications – ICCSA 2014*. Herausgegeben von Beniamino Murgante, Sanjay Misra, Ana Maria A. C. Rocha, Carmelo Torre, Jorge Gustavo Rocha, Maria Irene Falcão, David Taniar, Bernady O. Apduhan und Osvaldo Gervasi. Cham: Springer International Publishing, 2014, Seiten 32–46 (siehe Seite 7).

[Pet91] J. Petersen. „Die Theorie der regulären graphs“. In: *Acta Mathematica* 15.1 (1891), Seiten 193–220 (siehe Seite 1).

[PL86] Michael D Plummer und László Lovász, Herausgeber. Band 121. North-Holland Mathematics Studies. Elsevier, 1986 (siehe Seiten 1, 2).

[Rec+16] Peter Recht et al. „A dynamic programming approach for the max-min cycle packing problem in even graphs“. In: *Open Journal of Discrete Mathematics* 6.04 (2016), Seite 340 (siehe Seite 8).

[RR09] D. Rautenbach und F. Regen. „On Packing Shortest Cycles in Graphs“. In: *Inf. Process. Lett.* 109.14 (Juni 2009), Seiten 816–821 (siehe Seite 9).

[RS14] P. Recht und S. Stehling. „On maximum cycle packings in polyhedral graphs“. In: *Electronic journal of graph theory and applications* 2.1 (2014) (siehe Seite 8).

[RS91] Neil Robertson und Paul D Seymour. „Graph minors. X. Obstructions to tree-decomposition“. In: *Journal of Combinatorial Theory, Series B* 52.2 (1991), Seiten 153–190 (siehe Seite 11).

[Sch17] Olaf Schmidt. „Das genetische Material“. In: *Genetik und Molekularbiologie*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017, Seiten 1–12 (siehe Seite 25).

[Spr15] Eva-Maria Sprengel. „Maximale Kreispackungen für verallgemeinerte Petersen Graphen und die Bestimmung der Kreispackungszahl unter Verwendung von Knotenseparatoren“. Dissertation. Fakultät für Wirtschaftswissenschaften, TU Dortmund, 2015 (siehe Seiten 8, 26).

[SS17] Pijus Simonaitis und Krister M. Swenson. „Finding Local Genome Rearrangements“. In: *17th International Workshop on Algorithms in Bioinformatics (WABI 2017)*. Herausgegeben von Russell Schwartz und Knut Reinert. Band 88. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017, 24:1–24:13 (siehe Seiten 28, 31, 32, 34).

[SS18] Pijus Simonaitis und Krister M Swenson. „Finding local genome rearrangements“. In: *Algorithms for Molecular Biology* 13.1 (2018), Seiten 1–14 (siehe Seite 31).

[SS20] Abhishek Sahu und Saket Saurabh. „Kernelization of Arc Disjoint Cycle Packing in α-Bounded Digraphs“. In: *International Computer Science Symposium in Russia*. Springer. 2020, Seiten 367–378 (siehe Seite 8).

[SSB16] Krister Swenson, Pijus Simonaitis und Mathieu Blanchette. „Models and algorithms for genome rearrangement with positional constraints“. In: *Algorithms for Molecular Biology* 11 (Dez. 2016) (siehe Seite 32).

[Ste21] Stefan Stehling. „Maximum Cycle Packing - Obere Schranken in speziellen polyhedralen Graphen“. Dissertation. Fakultät für Wirtschaftswissenschaften, TU Dortmund, 2021 (siehe Seiten 8, 92, 97).

[SV05] M. R. Salavatipour und J. Verstraete. „Disjoint Cycles: Integrality Gap, Hardness, and Approximation“. In: *Integer Programming and Combinatorial Optimization*. Herausgegeben von M. Jünger und V. Kaibel. Band 3509. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2005, Seiten 51–65 (siehe Seite 17).

[Tar71] R. Tarjan. „Depth-first search and linear graph algorithms“. In: *12th Annual Symposium on Switching and Automata Theory (swat 1971)*. Okt. 1971, Seiten 114–121 (siehe Seite 41).

[Tol89] Ioannis G. Tollis. „On finding a minimum vertex cover of a series-parallel graph“. In: *Applied Mathematics Letters* 2.3 (1989), Seiten 305–309 (siehe Seite 92).

[Tur96] V. Turau. *Algorithmische Graphentheorie*. Addison-Wesley, 1996 (siehe Seite 41).

[Tut47] William T Tutte. „The factorization of linear graphs“. In: *Journal of the London Mathematical Society* 1.2 (1947), Seiten 107–111 (siehe Seite 1).

[Tut66] W. T. Tutte. *Connectivity in graphs*. Band 15. University of Toronto Press, 1966 (siehe Seiten 10, 44, 45, 47, 48).

[XW18] Mingyu Xiao und Xuanbei Wang. „Exact Algorithms and Complexity of Kidney Exchange.“ In: *IJCAI*. 2018, Seiten 555–561 (siehe Seite 7).

[Yus07] Raphael Yuster. „Combinatorial and computational aspects of graph packing and graph decomposition“. In: *Computer Science Review* 1.1 (2007), Seiten 12–26 (siehe Seite 1).

[Yus13] Raphael Yuster. „Packing triangles in regular tournaments“. In: *Journal of Graph Theory* 74.1 (2013), Seiten 58–66 (siehe Seite 8).

[YYD07] Lu Yunting, Li Yueping und Lou Dingjun. „An Algorithm to Find the Optimal Matching in Halin Graphs“. In: *IAENG International Journal of Computer Science* 34 (Nov. 2007) (siehe Seite 65).

[Zak08] Julia Zakotnik. „The double cut and join operation and its applications to genome rearrangements“. Dissertation. Universität Bielefeld, 2008 (siehe Seiten 26, 29–31).

[ZL97] Zhongfu Zhang und Linzhong Liu. „On the complete chromatic number of Halin graphs“. In: *Acta Mathematicae Applicatae Sinica* 13.1 (1997), Seiten 102–106 (siehe Seite 65).