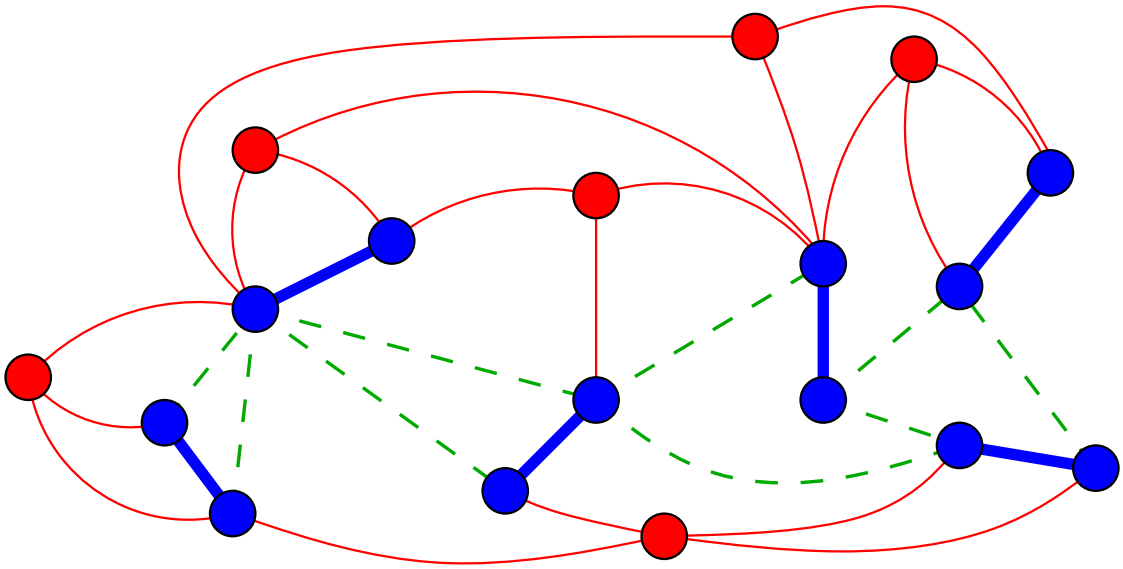
Matchings in planaren Graphen mit festem Minimalgrad

Studienarbeit am Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner
Fakultät für Informatik
Universität Karlsruhe (TH)

von

Robert Franke



Betreuer: Ignaz Rutter, Prof. Dr. Dorothea Wagner

August 2009

Erklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Karlsruhe, den 18. August 2009

Inhaltsverzeichnis

1 Einleitung

Ein *Matching* ist eine Kantenmenge mit paarweise nicht adjazenten Kanten. Fragestellungen, die sich mit Matchings befassen sind ausgiebig untersucht worden. Bereits Petersen [Pet91] beschäftigte sich damit: Er zeigte, dass in Graphen, in welchen jeder Knoten genau drei Nachbarn hat und zwischen zwei verschiedenen Knoten mindestens zwei disjunkte Wege existieren, immer ein *perfektes* Matching (also ein Matching, in dem jeder Knoten einen Partner hat) existiert. Wie die Berechnung eines perfekten Matchings in diesen Graphen effizient umgesetzt werden kann, zeigten Biedl et al. [BBDL01], indem sie einen Linearzeitalgorithmus dafür vorstellten.

Es wurde intensiv an der weiteren Charakterisierung von perfekten Matchings gearbeitet. Darunter fällt beispielsweise die Arbeit von Tutte [Tut47], der Bedingungen für die Existenz eines perfekten Matchings angab. Dénes Kőnig [Kőn16] zeigte, dass bipartite Graphen, in denen jeder Knoten genau k Nachbarn hat, ein perfektes Matching besitzen.

Das algorithmische Testen ob ein perfektes Matching existiert, bzw. dessen Berechnung wurde für einige Klassen von Graphen effizient umgesetzt. So kann beispielweise für planare bipartite Graphen mit n Knoten ein perfektes Matching in $\mathcal{O}(n \log^3 n)$ Zeit berechnet werden, vorausgesetzt es existiert [FR06] [MN95].

Neben der Existenz von perfekten Matchings ist natürlich die algorithmische Suche nach *maximalen* Matchings (d. h. Matchings mit maximaler Kardinalität) von großem Interesse. Das Problem, Maximale Matchings oder überhaupt möglichst große Matchings zu finden, hat vielerlei Anwendungen. Dies wird beispielsweise in dem Buch "Matching Theory" von Lovász and Plummer [LP86] verdeutlicht.

Für bipartite Graphen fanden Hopcraft und Karp [HK73] einen Algorithmus, der in $\mathcal{O}(\sqrt{n}m)$ (mit $n = |V|$ und $m = |E|$) Zeit ein maximales Matching findet. Für beliebige Graphen fanden Micali und Vazirani [MV80] einige Jahre später einen Algorithmus der ebenfalls in $\mathcal{O}(\sqrt{n}m)$ Zeit läuft. Dies ist der momentan asymptotisch schnellste Algorithmus für allgemeine Graphen, allerdings ist dieser Algorithmus um einiges komplizierter als der für bipartite Graphen, sodass in praktischen Anwendungen auf einfachere Algorithmen, die asymptotisch mehr Zeit benötigt, dafür aber einfacher zu implementieren ist, zurückgegriffen wird. So verwendet beispielsweise die Alogrithmenbibliothek LEDA den Algorithmus von Hopcraft und Karp für bipartitet Graphen, greift für allgemeine Graphen jedoch auf einen einfacheren Algorithmus zurück, der in $\mathcal{O}(nm\alpha(n, m))$ arbeitet [Alg]. Dabei bezeichnet $\alpha(n, m)$ die Inverse Ackermann-Funktion. Ebenfalls verwendet die Graphenbibliothek Boost [SLL] diesen einfacheren Algorithmus, der auf der wiederholten Suche nach erhöhenden Pfaden basiert [Tar83].

In jüngerer Zeit wurden für einige Klassen von Graphen asymptotisch schnelle Algorithmen vorgestellt, die auf Matrizenmultiplikation basieren. So gibt es Algorithmen für dichte Graphen [MS04] mit einer Laufzeit in $\mathcal{O}(n^\omega)$, für planare Graphen [MS06] in $\mathcal{O}(n^{\omega/2})$ sowie für Graphen ohne H-Minor [YZ07] mit einer Laufzeit in $\mathcal{O}(n^{3\omega/(\omega+3))}) \subseteq \mathcal{O}(n^{1,326})$. Dabei ist $\omega \leq 2,376$ der Exponennt in der asymptotischen Laufzeit des schnellsten bekannten Algorithmus' für Matrizenmultiplikation [CW87].

Eine weitere interessante Frage, die man in Bezug auf Matchings stellen kann, ist, ob man

anhand bestimmter Eigenschaften eines Graphen Aussagen über die Mindestgröße eines maximalen Matchings machen kann. Nishizeki und Baybars [NB79] haben für planare Graphen mit festem Minimalgrad und Knotenzusammenhang einige untere Schranken bewiesen. Diese wurden von Biedl et al. [BDD+04] weiter verfeinert und um zahlreiche weitere Schranken ergänzt. Die Beweise zeigen die Existenz von Matchings mit einer gewissen Mindestgröße, allerdings liefern sie kein Verfahren für die Konstruktion dieser Matchings.

Davon inspiriert fragten sich Rutter und Wolff [RW09] wie man diese Matchings finden kann und ob dies schneller möglich ist als die Bestimmung eines maximalen Matchings. Sie gaben für die meisten der von Biedel et al. aufgestellten Schranken effiziente Verfahren an, die dies bewerkstelligen.

Eine offene Frage, die verblieb, war, ob es auch für die Schranken aus [NB79] effiziente Algorithmen gibt. Diese Frage wird hier aufgegriffen und dient als Ausgangspunkt dieser Arbeit.

1.1 Die Ausgangslage

Uns interessiert die Mindestgröße von maximalen Matchings in planaren Graphen mit festem Minimalgrad δ.

Für $\delta \in \{1, 2\}$ findet man beliebig große Graphen, deren maximale Matchings allerdings nur ein oder zwei Kanten haben. Dazu kann man beispielsweise $K_{1,n}$ (also ein Baum, der nur aus einer Wurzel mit n Blättern besteht) betrachten. Für beliebiges n kann jedes Matching für $K_{1,n}$ höchstens eine Kante besitzen, da jede Kante zur Wurzel inzident ist.

Fügt man einen neuen Knoten hinzu und verbindet alle Blätter mit diesem neuen Knoten, so erhält man einen Graphen mit Minimalgrad 2 (nämlich $K_{2,n}$), dessen Matchings höchstens zwei Kanten enthalten können (die eine ist zur alten Wurzel inzident, die andere zum neuen Knoten).

Versucht man den gleichen Trick erneut anzuwenden um einen Graphen mit Minimalgrad 3 zu konstruieren, so verliert man die Planarität ($K_{3,n}$ enthält $K_{3,3}$ als Subgraph für $n \geq 3$). Diese Beobachtung legt nahe, dass es eine untere Schranke für die Mindestgröße eines maximalen Matchings in Abhängigkeit zur Größe des Graphen geben könnte.

In einem planaren Graphen gibt es immer einen Knoten mit Grad höchstens 5, somit bleibt nur noch zu untersuchen welche Schranken es für $\delta \in \{3, 4, 5\}$ gibt. Dies haben Takao Nishizeki und Ilker Baybars [NB79] getan. Ein Teil ihrer Resultate sind hier in Satz 1.1 angegeben.

Satz 1.1. *Sei $G = (V, E)$ ein planarer zusammenhängender Graph mit n Knoten und Minimalgrad δ. Für ein beliebiges maximales Matching M von G gilt:*

(a) $|M| \geq \frac{n+2}{3}$, *falls* $\delta = 3$ *und* $n \geq 10$

(b) $|M| \geq \frac{2n+3}{5}$, *falls* $\delta = 4$ *und* $n \geq 16$

(c) $|M| \geq \frac{5n+6}{11}$, *falls* $\delta = 5$ *und* $n \geq 34$

Takao Nishizeki und Ilker Baybars gaben noch weitere Schranken bei größerem Knotenzusammenhang an. In dieser Arbeit wird diese Unterscheidung nicht gemacht, deshalb sind für diese Arbeit nur die drei in Satz 1.1 angegebenen Schranken relevant.

Die in Satz 1.1 angegebenen Schranken wurden ebenfalls (bis auf die Konstanten) von Papadimitriou und Yannakakis [PY81] bewiesen. Dabei wenden sie ihre neu eingeführte Technik der Facetteninduktion an, die eine worst-case Analyse eines Matching-Algorithmus' auf der Basis von erhöhenden Wegen in planaren Graphen liefert. Dieses Vorgehen ist anschaulicher als der Beweis von Nishizeki und Baybars, die die Formel von Berge nutzten und die Anzahl der ungematchten Knoten abzuschätzen. Doch leider lieferen Papadimitriou und Yannakakis auch kein einfaches Verfahren um Matchings mit der bewiesenen Mindestgröße zu berechnen.

1.2 Zielsetzung

Der Beweis, den Takao Nishizeki und Ilker Baybars für die Schranken aus Satz 1.1 geführt haben, ist leider nicht konstruktiv. Es wird die Existenz eines Matchings mit einer gewissen Mindestgröße gezeigt, allerdings ohne dass daraus ein Verfahren abgeleitet werden kann, mit dem man solche Matchings findet.

Ein Ziel dieser Arbeit ist es, Verfahren zu finden, die für einen planaren zusammenhängenden Graphen mit bekanntem Minimalgrad ein Matching mit der aus Satz 1.1 vorgegebenen Mindestgröße finden. Der Beweis, dass das Verfahren das gewünschte Ergebnis liefert, ist dann gleichzeit ein konstruktiver Beweis für die Schranken aus Satz 1.1.

Ein weiteres Ziel ist es, möglichst effiziente Matchingalgorithmen zu finden, die für die gegebenen Graphen eine gewisse Mindestgröße der gefundenen Matchings garantieren und in kürzerer Laufzeit als Algorithmen für maximale Matchings arbeiten. Für planare Graphen ist es möglich ein maximales Matching in $\mathcal{O}(n^{1,188})$ Zeit zu finden [MS06]. Um dies zu unterbieten, wird eine asymptotische Laufzeit von $\mathcal{O}(n)$ für die zu entwickelnden Algorithmen angestrebt.

1.3 Überblick

In Abschnitt 2 werden die verwendeten Grundbegriffe und Notationen vorgestellt. Weiterhin wird eine implizite Färbung der Knoten und Kanten vorgestellt, die der Anschaulichkeit dienen soll und durchgehend beibehalten wird.

Im darauffolgenden Abschnitt 3 wird ein erster Algorithmus vorgestellt, der angewandt auf planare Graphen mit festem Minimalgrad etwas schwächere Schranken garantieren kann. Anhand einer Analyse einiger worst-case-Beispiele der berechneten Matchings werden wir einige strukturelle Forderungen an Matchings stellen die diese Schwächen verhindern sollen. Eine detailierte Darstellung und Untersuchung dieser Forderungen wird in Abschnitt 4 geschehen. Anschließend wird sich Abschnitt 5 mit der Frage befassen, wie man Matchings, die den aufgestellten Forderungen entsprechen, finden kann. Es wird ein Algorithmus vorgestellt, der für planare Graphen mit Minimalgrad 3 die angestrebte Schranke erreichen wird. Weiterhin wird gezeigt, dass der Algorithmus in Linearzeit realisierbar ist. Die Frage wie sich

das Verfahren für Minimalgrad 4 und 5 übertragen lässt wird ebenfalls behandelt und die dadurch entstehenden Probleme erläutert.

Abschließend wird Abschnitt 6 die vorgestellten Resultat zusammenfassend darstellen. Es werden Fragen vorgebracht, die noch offen bleiben, sowie einige Ansätze skizziert, die noch weiter verfolgt werden können und durch genauere Untersuchung noch möglicherweise interessante Einsichten liefern könnten.

2 Terminologie

In dieser Arbeit werden ausschließlich einfache, ungerichtete Graphen betrachtet (d.h. es gibt keine Schleifen oder Mehrfachkanten). Zudem wird immer angenommen, dass der Graph zusammenhängend ist. Dies ist keine Einschränkung, denn für unzusammenhängende Graphen kann das Suchen nach Matchings darauf zurück geführt werden, in den Zusammenhangskomponenten Matchings zu suchen.

Es folgt eine kurze Einführung der verwendeten Begriffe für Graphen und Matchings sowie die Festlegung einer Färbung der Knoten und Kanten, die durchgehend beibehalten wird.

2.1 Verwendete Begriffe und Notationen

Ein Matching heißt *maximal*, wenn es unter allen Matchings maximale Kardinalität besitzt. Ein Matching heißt *inklusionsmaximal*, wenn es nichtmehr durch das Hinzufügen von Kanten vergrößert werden kann. Ein Knoten der zu einer Matchingkante inzident ist, heißt *gematcht*. Ein Matching heißt *zusammenhängend*, wenn nach dem Entfernen der ungematchten Knoten der Graph zusammenhängend bleibt. Ein Matching wird *baumförmig* genannt, wenn das Entfernen der ungematchten Knoten ein Baum liefert.

Ein *erhöhender Pfad* ist ein einfacher Pfad, der abwechselnd Matchingkanten und nicht-Matchingkanten benutzt und dessen Start- und Endknoten ungematcht und verschieden sind. Die symmetrische Differenz eines Matchings und eines erhöhenden Pfades ergibt wieder ein Matching, dieses ist um genau eins größer als das ursprüngliche Matching. Nach Berge [Ber57] ist eine Matching genau dann maximal wenn keine erhöhenden Pfade existieren.

Der *Grad* eines Knotens x ist die Anzahl seiner Nachbarn und wird mit $d(x)$ bezeichnet. Für einen Graphen mit der Knotenmenge V und einer Teilmenge $W \subseteq V$ sei $d_W(x)$ die Anzahl der Nachbarn von x in W.

2.2 Induzierte Farben

Um eine einfache und klare Darstellung von gematchten und ungematchten Knoten zu ermöglichen, wird im Folgenden eine von einem Matching induzierte Färbung der Knoten und Kanten festgelegt.

Gematchte Knoten sind *blau* und ungematchte *rot*. Weiter sind Matchingkanten *blau*, Kanten die einen roten Endknoten besitzen *rot* und Kanten deren beider Endknoten blau sind, die aber nicht im Matching enthalten sind, sind *grün*.

Sowohl im Text, als auch in den Abbildungen wird diese Färbung verwendet. Zur besseren Unterscheidung sind auf Abbildung zusätzlich die grünen Kanten gestrichelt und die blauen Kanten dicker dargestellt.

Im Text werden die Farben synonym für gematcht und ungematcht verwendet, beispielsweise in Formulierungen wie: Ein Matching ist genau dann inklusionsmaximal, wenn jeder rote Knoten nur blaue Nachbarn hat.

3 Ein schneller und einfacher Algorithmus

In diesem Abschnitt werden wir einen einfachen Linearzeitalgorithmus vorstellen, für dessen gefundene Matchings wir mit Hilfe der Minimalgradbedingung gewisse Mindestgrößen garantieren können. Dazu wird zunächst ein inklusionsmaximales Matching berechnet und in einem zweiten Schritt das gefundene Matching weiter vergrößert. Diese Vergrößerung geschieht mittels erhöhender Pfade der Länge 3. Die Ausgabe, die der Algorithmus liefern soll ist ein inklusionsmaximales Matching ohne erhöhende Pfade der Länge 3.

Die genauere Betrachtung der Strukturen, die die gefundenen Matchings bilden, werden die Schwächen dieser Matchings zeigen. Stärkere Forderungen, die solche "ungünstigen" Matchings verhindern, werden mit darauffolgenden Abschnitt vorgestellt.

3.1 Der Algorithmus im Detail

Die erste Phase, die ein inklusionsmaximales Matching bestimmt, wird durch einen Greedy-Algorithmus realisiert (siehe Algorithmus 1): Es werden nacheinander Kanten ausgewählt und dem Matching hinzugefügt, die ausgewählte Kante sowie alle dazu inzidenten Kanten werden anschließend nicht mehr betrachtet.

Algorithmus 1 FINDE INKLUSIONSMAXIMALES MATCHING

Require: $G = (V, E)$ ist ein Graph
Ensure: M ist ein inklusionsmaximales Matching
1: $M \leftarrow \emptyset$
2: **while** $E \neq \emptyset$ **do**
3: Wähle beliebige Kante $\{x, y\} \in E$
4: Füge $\{x, y\}$ in M ein
5: Entferne x und y aus G
6: **end while**

Lemma 3.1. *Für einen beliebigen Graphen mit m Kanten findet der Algorithmus 1 in $\mathcal{O}(m)$ Zeit ein inklusionsmaximales Matching.*

Beweis. Das gefundene Matching ist offensichtlich inklusionsmaximal. Die Laufzeit kann durch eine Amortisierung über die Kanten abgeschätzt werden. □

In der zweiten Phase wird ein vorgegebenes Matching weiter verbessert, indem erhöhende Pfade der Länge 3 gesucht werden: Man betrachte eine Kante $\{x, y\}$ des eingegebenen Matchings. Hat nun x einen ungematchten Nachbarn x' und y einen ungematchten Nachbarn y' ($x' \neq y'$), so kann das Matching mittels Austausch der Kante $\{x, y\}$ durch die beiden Kanten $\{x, x'\}$ und $\{y, y'\}$ um eins verbessert werden (siehe Algorithmus 2).

Lemma 3.2. *Für einen beliebigen Graphen mit m Kanten und ein inklusionsmaximales Matching M liefert die Anwendung von Algorithmus 2 in $\mathcal{O}(m)$ Zeit ein inklusionsmaximales Matching M', das keinen erhöhenden Pfad der Länge 3 besitzt.*

Algorithmus 2 VERBESSERE INKLUSIONSMAXIMALES MATCHING

Require: $G = (V, E)$ ist ein Graph, $M \subseteq E$ ist ein inklusionsmaximales Matching
Ensure: M' ist ein Matching mit $|M'| \geq |M|$

```
1:  M' ← Kopie von M
2:  for all {x, y} ∈ M do
3:    N_u(x) ← Menge der roten Nachbarn von x
4:    N_u(y) ← Menge der roten Nachbarn von y
5:    if N_u(x) ≠ ∅ und N_u(y) ≠ ∅ und |N_u(x) ∪ N_u(y)| > 1 then
6:      if N_u(x) ⊆ N_u(y) then
7:        Wähle x' ∈ N_u(x) beliebig
8:      else
9:        Wähle x' ∈ N_u(x) − N_u(y) beliebig
10:     end if
11:     Wähle y' ∈ N_u(y) − {x'} beliebig
12:     Entferne {x, y} aus M'
13:     Füge {x, x'}, {y, y'} in M'
14:   end if
15: end for
```

Beweis. Es folgt eine Untersuchung ob nach der Ausführung noch ein erhöhender Pfad der Länge 3 vorhanden sein kann. Zunächst einmal stellt man fest, dass eine Kante $e \in M$ genau dann auf einem erhöhenden Pfad der Länge 3 liegt, wenn beide Endknoten von e verschiedene rote Nachbarn haben. Ist $N_u(v)$ die Menge der roten Nachbarn eines Knotens v, so entspricht dies genau den Bedingungen $N_u(x) \neq \emptyset$, $N_u(y) \neq \emptyset$ und $|N_u(x) \cup N_u(y)| > 1$, die in Zeile 5 des Algorithmus 2 abgefragt wird. Sind die Bedingungen erfüllt, so werden $x' \in N_u(x)$ und $y' \in N_u(y)$ mit $x' \neq y'$ ausgewählt. Somit ist $x'xyy'$ wirklich ein erhöhender Pfad der Länge 3, der dann zur Verbesserung des Matchings genutzt wird.

Wurden die Kanten $\{x', x\}$ und $\{y', y\}$ als Teil eines erhöhenden Weges $x'xyy'$ dem Matching hinzugefügt, so ist es nicht möglich dass eine der Kanten wiederum auf einem erhöhenden Pfad der Länge 3 liegt. Das liegt daran, dass x' und y' bevor sie blau wurden keine roten Nachbarn haben konnten, da $|M|$ sonst kein inklusionsmaximales Matching gewesen wäre. Also genügt es, jede Matchingkante aus M einmalig zu untersuchen um alle solche Pfade zu finden.

Widmen wir uns nun der Laufzeitanalyse. Um die von Algorithmus 2 benötigte Zeit abzuschätzen, nutzen wir eine amortisierte Analyse über die Kanten.

Die Schleife wird für jede Matchingkante M genau einmal aufgerufen. Der Aufwand, für die beiden Endknoten x und y der Kante die roten Nachbarn zu finden und die anschließenden Vergleiche und Operationen durchzuführen liegt in $\mathcal{O}(d(x) + d(y))$. Die Kosten können also in Teile konstanter Grösse c auf die zu x bzw. y inzidenten Kanten verteilt werden.

Wir teilen jeder Kante einen konstanten Kredit in der Höhe $2c$ an Aufwand zu. Der so verteilte Kredit ist ausreichend um alle Operationen zu bezahlen, denn für jede Kante wird pro Endknoten höchstens einmal Kredit in der Höhe c benutzt (Jeder Endknoten einer

Matchingkante von M wird genau einmal betrachtet). Damit ergibt sich eine Gesamtlaufzeit von $cm = \mathcal{O}(m)$ Zeit. □

Die beiden vorangegangen Lemmata zeigen, dass die Anwendung von Algorithmus 1 mit anschließender Anwendung von Algorithmus 2 auf einen Graphen mit m Kanten ein inklusionsmaximales Matching ohne erhöhende Pfade der Länge 3 in $\mathcal{O}(m)$ Zeit liefert. Im folgenden Abschnitt werden wir untersuchen, welche Schranken wir für solche Matchings in planaren Graphen mit festem Minimalgrad angeben können.

3.2 Mindestgröße des gefundenen Matchings für planare Graphen mit Minimalgrad δ

Um die Grösse des Matchings abzuschätzen, zählen wir die roten Knoten und schätzen das Verhältnis der Anzahl roter Knoten zur Anzahl blauer Knoten ab. Doch bevor wir damit anfangen müssen noch einige Vorbereitungen getroffen werden. Lemma 3.3 stellt eine wohlbekannte Aussage über planare bipartite Graphen vor, die später im Beweis von Satz 3.5 benutzt wird. Lemma 3.2 stellt sicher, dass die gefundenen Matchings die für Satz 3.5 benötigten Voraussetzungen erfüllen.

Lemma 3.3. *Ein bipartiter, planarer Graph mit n Knoten hat höchstens $2n - 4$ Kanten.*

Beweis. Dies ist ein bekanntes Resultat und kann durch doppeltes Abzählen der Kanten über die Facetten bewiesen werden (indem man verwendet, dass es keine Dreieckfacette gibt). Das Abzählen liefert $2m \geq 4f$ für die Anzahl der Kanten m und die Anzahl der Facetten f. Mit der Eulerformel $n - m + f = 2$ folgt die Behauptung. □

Betrachtet man Matchingkanten, die nichtmehr zur Vergrößerung des Matchings beitragen können, so fällt ins Auge, dass man diese Kanten in zwei Kategorien einteilen kann. Entweder einer der Endknoten einer Matchingkante hat keine rote Nachbarn oder beide haben nur einen gemeinsamen roten Nachbarn. Die folgende Definition präzisiert diese Unterteilung. In Abbildung 1 sind die Möglichkeiten einer Matchingkante beispielhaft dargestellt.

Definition 3.4. *Sei M ein Matching für den Graphen $G = (V, E)$.*

(i) Eine Matchingkante heißt offen*, wenn einer der Endknoten keinen roten Nachbarn besitzt. Eine Matchingkante heißt* besetzt *(durch r), wenn beide Endknoten einen gemeinsamen roten Nachbarn r besitzen und r jeweils der einzige rote Nachbar der Endknoten ist.*

(ii) Die Menge der offenen Kanten bezeichnen wir mit M_o und die der besetzten mit M_b.

(iii) Sei R die Menge der roten Knoten, dann bezeichne $R_b \subseteq R$, die Menge derjenigen roten Knoten, die eine Matchingkante besetzen. R_o bezeichne die Menge der roten Knoten, die keine Matchingkante besetzen.

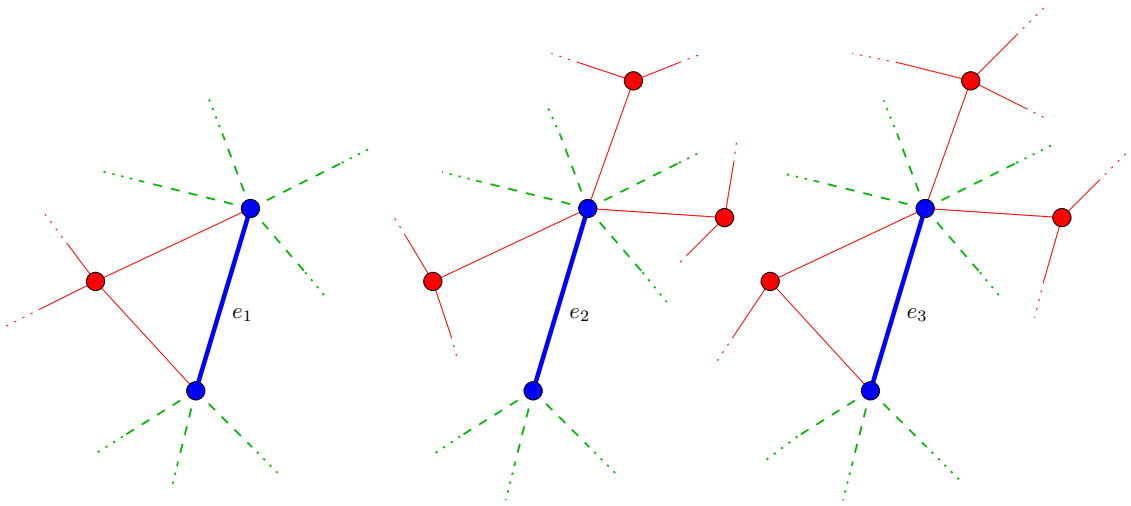


Abbildung 1: e_1 ist besetzt, e_2 ist offen und e_3 liegt auf einem erhöhenden Pfad der Länge 3

Nach Definition gilt $M_{\mathfrak{b}} \cap M_{\mathfrak{o}} = \emptyset$, $R_{\mathfrak{b}} \cap R_{\mathfrak{o}} = \emptyset$ und $R = R_{\mathfrak{b}} \cup R_{\mathfrak{o}}$. Eine beliebige Matchingkante ist nun entweder offen, besetzt oder liegt auf einem erhöhenden Pfad der Länge 3. Die Menge $R_{\mathfrak{b}}$ enthält genau diejenigen roten Knoten, die zu einem Endknoten einer besetzten Kante adjazent sind. Ist das Matching inklusionsmaximal, so enthält $R_{\mathfrak{o}}$ genau diejenigen roten Knoten, die ausschließlich zu Endknoten offener Kanten adjazent sind. Falls das Matching keinen erhöhenden Pfad der Länge 3 besitzt, gilt offensichtlich $M = M_{\mathfrak{b}} \cup M_{\mathfrak{o}}$ (jede Matchingkante ist entweder offen oder besetzt).

Satz 3.5. *Sei G ein planarer Graph mit Minimalgrad δ und M ein inklusionsmaximales Matching, das keinen erhöhenden Pfad der Länge 3 besitzt. Die Mengen R, $R_{\mathfrak{b}}$, $R_{\mathfrak{o}}$, $M_{\mathfrak{b}}$ und $M_{\mathfrak{o}}$ seien entsprechend der Definition 3.4 festgelegt. Es gilt:*

$$|R| \leq |M_{\mathfrak{b}}| + \frac{2|M_{\mathfrak{o}}| - 4}{(\delta - 2)}.$$

Beweis. Um die roten Knoten zu zählen, werden die Knoten aus $R_{\mathfrak{b}}$ und $R_{\mathfrak{o}}$ getrennt untersucht und abgeschätzt.

Wir beginnen damit, diejenigen rote Knoten zu zählen, die zu einem Endknoten einer Kante aus $M_{\mathfrak{b}}$ benachbart sind, also die aus $R_{\mathfrak{b}}$. Dies sind höchstens $|M_{\mathfrak{b}}|$ Stück, da die Endknoten einer Kante aus $M_{\mathfrak{b}}$ nur zu einem roten Knoten benachbart sind (sonst gäbe es einen erhöhenden Pfad der Länge 3).

Um die restlichen zu zählen ist es zweckmäßig, einen bipartiten Hilfsgraphen G' zu betrachten. Die Knotenmenge von G' setzt sich aus den Teilen $V_{\text{rot}} = R_{\mathfrak{o}}$ und $V_{\text{blau}} = M_{\mathfrak{o}}$ zusammen. Man beachte dass für V_{blau} die Kanten aus $M_{\mathfrak{o}}$ als Knoten interpretiert werden. Zwei Knoten $u \in V_{\text{rot}}$ und $v \in V_{\text{blau}}$ sind genau dann benachbart, wenn u zu einem Endknoten der Kante v in G benachbart ist. Betrachten wir nun weitere Eigenschaften von G'.

Der Graph G' ist planar und bipartit: Nach Konstruktion ist G' offensichtlich bipartit, da keine Kanten zwischen roten und blauen Knoten eingefügt wurden. Die Planarität lässt sich wie folgt einsehen: Betrachte eine planare Einbettung von G und entferne alle Knoten aus R_{b} und alle Endknoten von Kanten aus M_{b}. Kontraktiere weiterhin jede Kante aus M_{o} und entferne alle Kanten zwischen durch Kontraktion entstandenen Knoten und erhalte somit eine planare Einbettung von G' (insbesondere ist G' ein Minor von G).

Jeder Knoten aus V_{rot} hat Minimalgrad δ: Sei $x \in V_{\text{rot}}$, dann sind die Nachbarn von x in G Endknoten paarweise verschiedener Kanten aus M_{o} und somit bleibt jeder Nachbar auch in G' erhalten. Damit hat x in G' denselben Grad wie in G, d.h. mindestens Grad δ.

Sei m' die Anzahl an Kanten in G'. Aus Lemma 3.3 folgt nun $m' \leq 2(|R_{\mathsf{o}}| + |M_{\mathsf{o}}|) - 4$ (bipartit und planar). Andererseits folgt aus dem Minimalgrad der roten Knoten $m' \geq \delta|R_{\mathsf{o}}|$. Insgesamt ergibt sich die Ungleichung $\delta|R_{\mathsf{o}}| \leq 2(|R_{\mathsf{o}}| + |M_{\mathsf{o}}|) - 4$. Die folgende Rechnung schätzt $|R_{\mathsf{o}}|$ ab.

$$\begin{aligned}
\delta|R_{\mathsf{o}}| &\leq 2|R_{\mathsf{o}}| + 2|M_{\mathsf{o}}| - 4 \\
\delta|R_{\mathsf{o}}| - 2|R_{\mathsf{o}}| &\leq 2|M_{\mathsf{o}}| - 4 \\
|R_{\mathsf{o}}|(\delta - 2) &\leq 2|M_{\mathsf{o}}| - 4 \\
|R_{\mathsf{o}}| &\leq \frac{2|M_{\mathsf{o}}| - 4}{(\delta - 2)}
\end{aligned}$$

Insgesamt erhält man als obere Grenze für die Anzahl aller roten Knoten:

$$|R| = |R_{\mathsf{b}}| + |R_{\mathsf{o}}| \leq |M_{\mathsf{b}}| + \frac{2|M_{\mathsf{o}}| - 4}{(\delta - 2)}.$$

□

Im Falle von $\delta \in \{3, 4\}$ begünstigen die besetzten Kanten die Abschätzung. Dies ändert sich für $\delta = 5$, da hier das Verhältnis von offenen Kanten zu Knoten aus R_{o} erstmals besser wird als 1:1 (was dem Verhältnis der besetzten Kanten zu besetzenden Knoten entspricht). Die nächste Folgerung berücksichtigt dies und gibt Schranken für $\delta \in \{3, 4, 5\}$ an.

Folgerung 3.6. *Sei G ein planarer Graph mit n Knoten und Minimalgrad δ, weiter sei M ein inklusionsmaximales Matching, das keinen erhöhenden Pfad der Länge 3 besitzt. Es gilt:*

(i) $\frac{n+4}{4} \leq |M|$, *wenn* $\delta = 3$.

(ii) $\frac{n+2}{3} \leq |M|$, *wenn* $\delta = 4$.

(ii) $\frac{3n+4}{8} \leq |M|$, *wenn* $\delta = 5$ *und M keine besetzten Kanten enthält.*

Beweis. Es gilt $|R| = n - 2|M|$ (Es gibt $2|M|$ blaue Knoten und die übrigen sind rot). Mit Satz 3.5 ergibt sich

$$n - 2|M| \leq |M_{\mathsf{b}}| + \frac{2|M_{\mathsf{o}}| - 4}{(\delta - 2)}$$

Für $\delta = 3$ folgt

$$n - 2|M| \leq |M_{\mathsf{b}}| + 2|M_{\mathsf{o}}| - 4$$
$$n \leq 2|M| + 2(|M_{\mathsf{b}}| + |M_{\mathsf{o}}|) - 4$$
$$n \leq 4|M| - 4$$

und damit $\frac{n+4}{4} \leq |M|$, also (i). Analog folgt für $\delta = 4$

$$n - 2|M| \leq |M_{\mathsf{b}}| + |M_{\mathsf{o}}| - 2$$
$$n = 3|M| - 2$$

und damit $\frac{n+2}{3} \leq |M|$, also (ii). Gibt es keine besetzten Kanten, so gilt $M_{\mathsf{b}} = \emptyset$ und $M = M_{\mathsf{o}}$. Daraus folgt

$$n - 2|M| \leq \frac{2|M| - 4}{3}$$
$$n \leq \frac{8|M| - 4}{3}$$

und damit $\frac{3n+4}{8} \leq |M|$, also (iii). □

Es stellt sich die Frage wie groß das Matching vor der Ausführung von Algorithmus 2 ist, beziehungsweise wie groß die Verbesserung ist, die die Eliminierung der erhöhenden Pfade der Länge 3 mit sich bringt. Dazu untersuchen wir, wie groß beliebige inklusionsmaximale Matchings in planaren Graphen mit Minimalgrad $\delta \in \{3, 4, 5\}$ sind.

Ein inklusionsmaximales Matching muss mindestens halb so groß sein, wie ein maximales Matching. Daraus ergeben sich aus Satz 1.1 die unteren Schranken $\frac{n+2}{6}$ für $\delta = 3$, $\frac{2n+3}{10}$ für $\delta = 4$ und $\frac{5n+6}{22}$ für $\delta = 5$.

Nun erscheinen diese Schranken relativ schwach und man könnte hoffen, dass es noch bessere Schranken gibt. Diese Hoffnung wird durch Satz 3.7 und Folgerung 3.8 erfüllt.

Satz 3.7. *Sei G ein planarer Graph mit n Knoten und Minimalgrad δ und M ein inklusionsmaximales Matching. Es gilt:*

$$\frac{n(\delta - 2) + 4}{2\delta} \leq |M|$$

Beweis. Betrachte den Graphen G', der aus G entsteht, indem man alle blauen und grünen Kanten entfernt. G' enthält nur noch rote Kanten und somit ist es unmöglich, dass zwei blaue Knoten benachbart sind. Da M inklusionsmaximal ist, gibt es auch keine Kante zwischen zwei roten Knoten, also ist G' bipartit. Sei m' die Anzahl der Kanten von G'. Mit Lemma 3.3 folgt $m' \leq 2n - 4$. Außerdem gilt $\delta|R| \leq m'$, da jeder rote Knoten weiterhin mindestens Grad δ hat. Somit folgt

$$\delta|R| \leq 2n - 4$$
$$|R| = n - 2|M| \leq \frac{2n - 4}{\delta}$$

$$\frac{\delta n - 2n + 4}{\delta} \leq 2|M|$$
$$\frac{n(\delta - 2) + 4}{2\delta} \leq |M|$$

□

Durch Einsetzen konkreter Werte für δ erhält man folgende Aussage.

Folgerung 3.8. *Sei G ein planarer Graph mit n Knoten und Minimalgrad δ und M ein inklusionsmaximales Matching. Es gilt:*

(i) $\frac{n+4}{6} \leq |M|$, *wenn* $\delta = 3$.

(ii) $\frac{n+2}{4} \leq |M|$, *wenn* $\delta = 4$.

(iii) $\frac{3n+4}{10} \leq |M|$, *wenn* $\delta = 5$.

Der nächste Abschnitt wird einige Beispiele vorstellen, die zeigen dass die hier vorgestellten Schranken scharf sind.

3.3 Beispiele für Schwächen der gefundenen Matchings

Wir wollen nun untersuchen wie gut die bewiesenen Schranken sind und uns einige möglichst kleine Matchings anschauen, die inklusionsmaximal sind und keine erhöhenden Pfade der Länge 3 besitzen. Die Analyse einiger Beispiel wird uns auch zu zusätzlichen strukturellen Eigenschaften führen, die verhindern dass solche "kleinen" Matchings möglich sind.

Zunächst befassen wir uns mit Minimalgrad $\delta = 3$ und Matchings, die keine besetzte Kanten haben. Als Ausgangspunkt benutzen wir einen maximal planaren bipartiten Graphen (also einen Graphen, der nach dem Einfügen einer Kante nichtmehr bipartit oder nichtmehr planar ist). Zusätzlich soll eine der beiden Knotengruppen Minimalgrad 3 haben. Dazu definieren wir den Graphen $\Gamma_k := (B_k \cup R_k, E_k)$, wobei $B_k := \{b_0, \ldots, b_{2k-1}\}$ und $R_k := \{r_1^{(1)}, \ldots, r_{2k-2}^{(1)}, r_1^{(2)}, \ldots, r_{2k-2}^{(2)}\}$. Eine Kante $\{r_i^{(j)}, b_l\} \in E_k$ (für $j \in \{1, 2\}$, $i \in \{1, \ldots, 2k-2\}$ und $l \in \{0, \ldots, 2k-1\}$) existiert genau dann, wenn $l = 0$, $i = l$ oder $i = l + 1$. In Abbildung 2 ist Γ_3 zu sehen.

Jeder Knoten aus R_k hat Grad 3, somit besitzt Γ_k genau $m_{\Gamma_k} = 3|R_k| = 6(2k-2) = 12k-12$ Kanten. Die Anzahl der Knoten beträgt $n_{\Gamma_k} = |B_k| + |R_k| = 2k + 2(2k-2) = 6k-4$. Damit gilt $m_{\Gamma_k} = 2n_{\Gamma_k} - 4$. Daraus folgt mit Lemma 3.3, dass nach dem Hinzufügen einer Kante der resultierende Graph nichtmehr planar und bipartit sein kann.

Nun modifizieren wir Γ_k zu Γ'_k, indem wir für jedes $i \in \{0, \ldots, k-1\}$ noch eine zusätzliche Kante $\{b_{2i}, b_{2i+1}\}$ einfügen. Sei M'_k die Menge der neu eingefügten Kanten, dann ist M'_k ein inklusionsmaximales Matching in Γ'_k. Dies ist leicht zu sehen, denn jeder Knoten aus B_k wird gematcht und die roten Knoten bleiben die ungematchten, somit ist keine Kante zwischen zwei roten Knoten möglich. Allerdings gibt es noch erhöhende Pfade der Länge 3, wie z.B. $r_1^{(1)} b_0 b_1 r_1^{(2)}$. Abbildung 3 verdeutlicht dies. Es gilt:

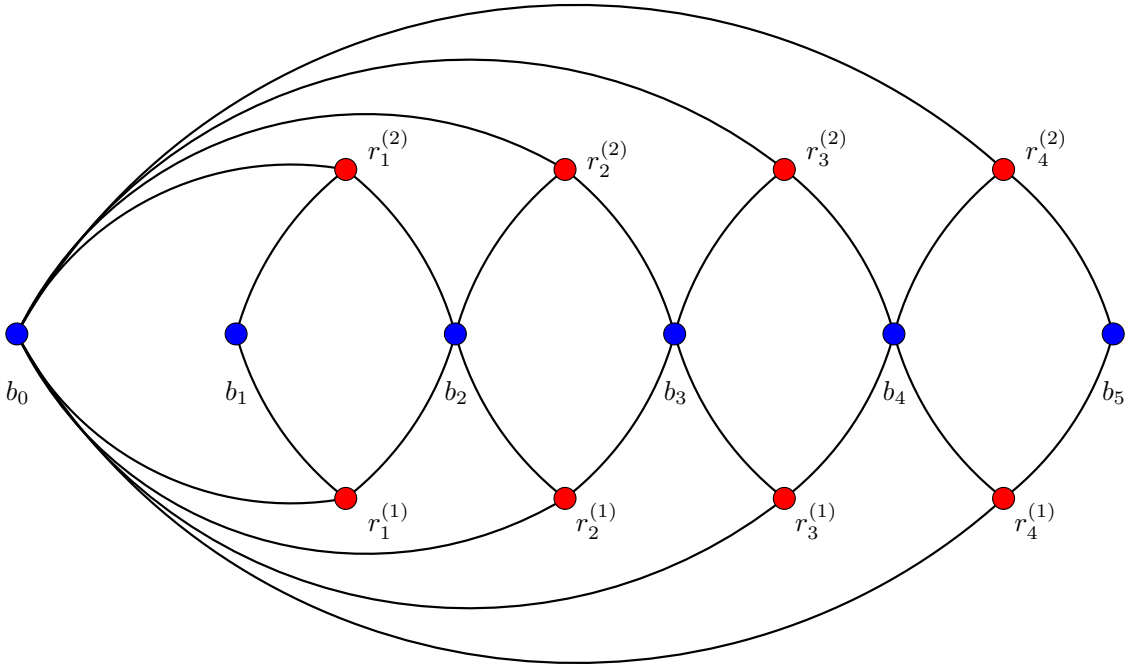


Abbildung 2: Der maximal planar bipartite Graph Γ_3

$$|M'_k| = k = \frac{6k-4+4}{6} = \frac{n_{\Gamma'_k}+4}{6}.$$

Dies ist die Gleichheit in der Ungleichung (i) aus Folgerung 3.8. Somit wissen wir, dass diese untere Schranke scharf ist und haben eine unendliche Familie von Graphen und Matchings gefunden die an die Grenze stößt.

Als nächstes wollen wir einen Graphen mit einen inklusionsmaximalen Matching ohne erhöhende Pfade der Länge 3 konstruieren. Dazu modifizieren wir abermals Γ_k. Zunächst fügen wir für jeden Knoten $b_i \in B_k$ einen neuen Knoten b'_i ein (dies soll der Matchingpartner zu b_i werden). Für jeden der neuen Knoten b'_i wird die Kante $\{b'_i, b_i\}$ eingefügt. Desweiteren fügen wir für jedes $i \in \{0, \ldots, k-1\}$ die drei Kanten $\{b'_{2i}, b'_{2i+1}\}$, $\{b'_{2i}, b_{2i+1}\}$ und $\{b'_{2i+1}, b_{2i}\}$ ein um Minimalgrad 3 zu gewährleisten.

Den so konstruierten Graphen nennen wir Γ''_k. Dass Γ''_k planar ist, zeigt Abbildung 4. Das passende Matching M''_k besteht aus den Kanten $\{b'_i, b_i\}$. Es ist inklusionsmaximal und enthält nur offene Kanten (kann also insbesondere keine erhöhende Pfade der Länge 3 ermöglichen). Die Anzahl der Knoten in Γ''_k hat sich um $2k$ auf $n_{\Gamma''_k} = 8k-4$ erhöht, damit gilt:

$$|M''_k| = 2k = \frac{8k-4+4}{4} = \frac{n_{\Gamma''_k}+4}{4}.$$

Somit haben wir auch für die Schranke (i) der Folgerung 3.6 eine Familie von Graphen und Matchings gefunden die die Gleichheit erfüllen.

Für $\delta = 4$ lassen sich auf analoge Weise maximal planar bipartite Graphen konstruieren,

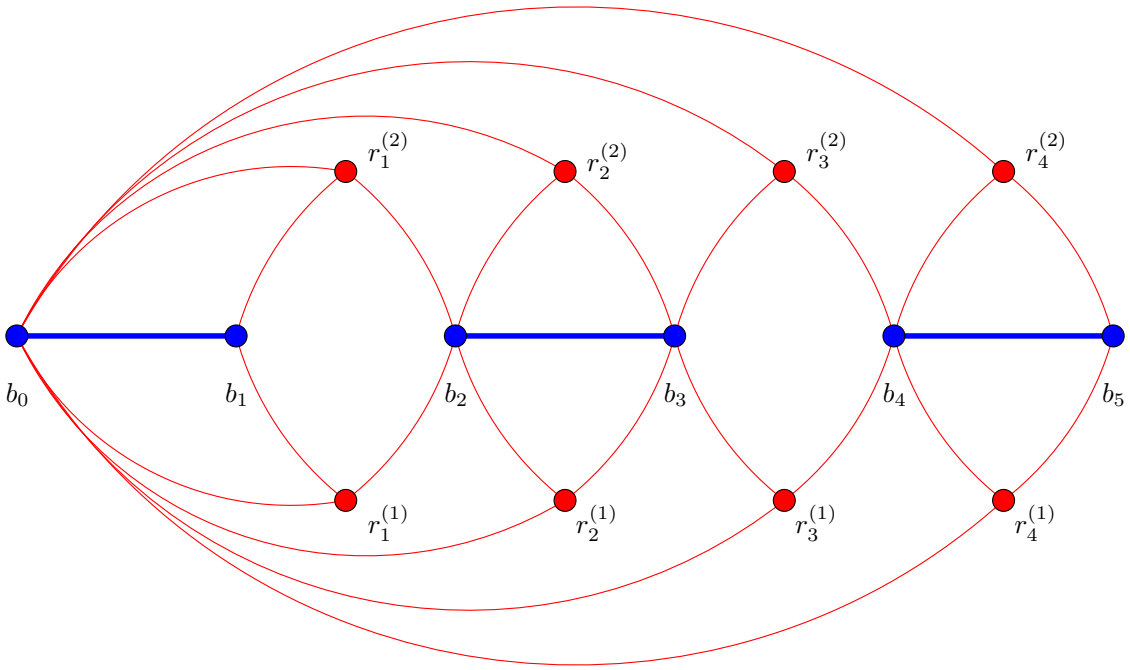


Abbildung 3: Γ_3' mit der durch M_3' induzierten Färbung

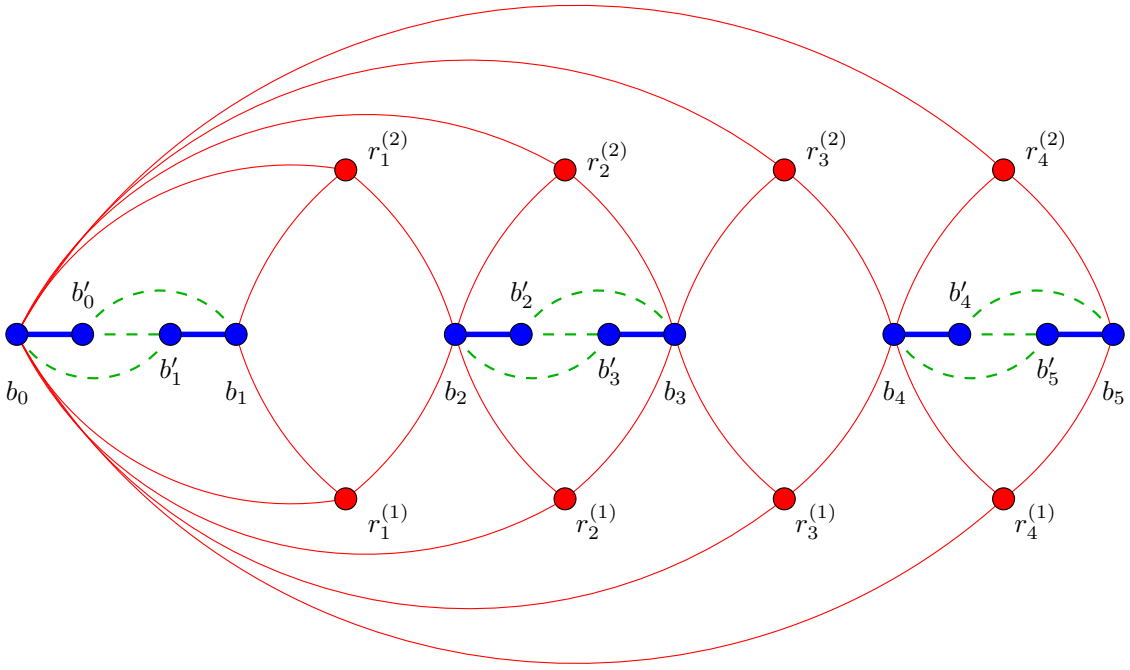


Abbildung 4: Γ_3'' mit der durch M_3'' induzierten Färbung

die sich dann wiederum zu Beispielen für Matchings, deren Grösse genau der in 3.6 und 3.8 gefundenen Schranken entsprechen, erweitern lassen.

Für $\delta = 5$ lassen sich ebenfalls analog Beispiele für die Schranke aus 3.6 finden. Allerdings geht dies nicht für die Schranke 3.8 für inklusionsmaximale Matchings. Das Problem ist, dass es nicht möglich ist für jeden blauen Knoten Minimalgrad 5 sicherzustellen. Hier bleibt offen, ob es vielleicht eine bessere Schranke für inklusionsmaximale Matchings in planaren Graphen mit Minimalgrad 5 gibt.

Beim Betrachten von Γ_3'' in Abbildung 4 fällt auf, dass sich die blauen Knoten in Komponenten aufteilen, wobei sich diejenigen blauen Knoten ohne rote Nachbarn "innen" befinden. Könnte man dafür sorgen, dass diese Komponenten größer werden (am besten sodass alle blauen Knoten in einer einzigen Komponente liegen), so wären weniger rote Knoten möglich. Dieser Ansatz wird im Abschnitt 4 genauer betrachtet und spezifiziert.

4 Bessere Matchings

Anhand der im letzten Abschnitt gewonnen Einsichten werden in diesem Abschnitt Forderungen an die Struktur eines Matchings stellen, die in einem Graphen mit festem Minimalgrad gewisse untere Schranken für die Grösse des Matchings garantieren. Dabei werden wir für Minimalgrad 3 und 4 die in Satz 1.1 vorgestellten Schranken erreichen. Für Minimalgrad 5 werden wir allerdings nur eine etwas schwächere Schranke erhalten.

4.1 Die benötigte Struktur

Das Matching sollte auf dem Graphen eine Struktur induzieren, die die blauen Knoten in einer Komponente zusammenhält. Dabei reicht es allerdings nicht zu fordern, dass das Matching zusammenhängend ist (d.h. der blaue Subgraph zusammenhängend ist). Das kann man beispielsweise an Γ_3'' (Abbildung 4) erkennen: Hier können noch die Kanten $\{b_1, b_2\}$ und $\{b_3, b_4\}$ planar eingefügt werden und das Matching wäre zusammenhängend, aber nicht größer.

Vielmehr sollten diejenigen blauen Knoten zusammenhängen, die keine roten Nachbarn haben. Das würde dazu führen dass die blaue Komponente eine Art blauen zusammenhängenden Kern hat, der nicht mit den roten Knoten verbunden ist. Am Rand der Komponente sind dann die blauen Knoten, die auch rote Nachbarn haben, aber nicht zum Zusammenhang beitragen.

Die Absicht hinter dieser Forderung ist folgende: Man stelle sich vor, man hat eine planare Einbettung vor sich und es gibt so eine blaue Komponente. Jetzt kann man eine einfache Kurve um die Komponente herum in die Ebene zeichnen, die durch jeden blauen Randpunkt verläuft aber keine Kante kreuzt (siehe Abbildung 5). Nun liegen alle roten Knoten außerhalb dieser Kurve und alle blauen innerhalb oder auf der Kurve selbst. Radiert man das Innere der Kurve aus, so bleibt ein bipartiter Graph zurück. Dieser kann jetzt nur halbsoviel rote Knoten wie ein maximal planarer bipartiter Graph mit derselben Anzahl an blauen Knoten haben. Dazu überlegt man sich, dass man das gesamte Äußere nach innen spiegeln kann. Somit wird die Anzahl der roten Knoten verdoppelt ohne die Anzahl der blauen Knoten zu erhöhen und das Resultat bleibt trotzdem planar und bipartit.

Um diese Struktur zu formalisieren, definieren wir eine neue Art von Matchingkante. Wir brauchen Matchingkanten, bei denen ein Endknoten ohne rote Nachbarn und der andere ohne blaue Nachbarn ist.

Definition 4.1. *Sei $G = (V, E)$ ein planarer Graph und $M \subseteq E$ ein Matching. Eine Kante $e \in M$ heißt* vollkommen offen*, wenn ein Endknoten nur blaue Nachbarn und der andere Endknoten außer seinem Matchingpartner nur rote Nachbarn hat. Ist jede Matchingkante vollkommen offen, so nennen wir das Matching ebenfalls vollkommen offen.*

Man kann sich leicht überlegen, dass wenn jede Matchingkante vollkommen offen ist und das Matching zusammenhängend ist, es genau so eine gewünschte Komponente geben muss.

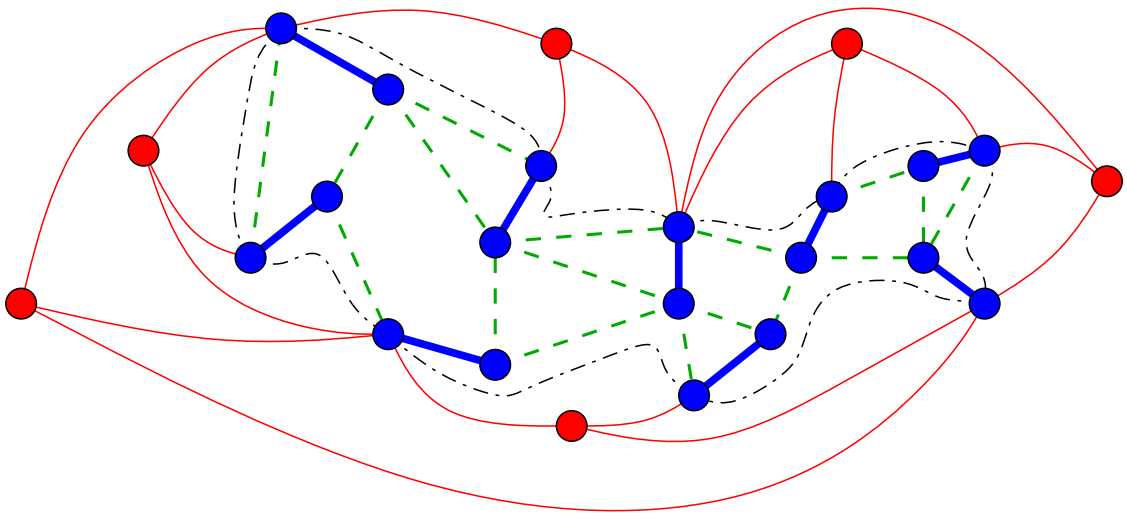


Abbildung 5: Ein Graph mit inklusionsmaximalen Matching ohne erhöhende Pfade der Länge 3, sodass der blaue Subgraph durch eine einfache geschlossene Kurve eingeschlossen werden kann ohne Kanten zu kreuzen.

4.2 Wie sich die neue Struktur auswirkt

Wir wollen nun genauer untersuchen wie sich die Forderung nach dieser zusätzlichen strukturellen Eigenschaft auf die Mindestgröße der Matchings auswirkt.

Satz 4.2. *Sei $G = (V, E)$ ein planarer Graph und $M \subseteq E$ ein zusammenhängendes inklusionsmaximales Matching, das ausschließlich vollkommen offene Kanten besitzt. Jeder rote Knoten habe mindestens Grad δ. Dann gilt:*

(i) $\frac{n+2}{3} \leq |M|$, *wenn* $\delta = 3$.

(ii) $\frac{2n+2}{5} \leq |M|$, *wenn* $\delta = 4$.

(iii) $\frac{3n+2}{7} \leq |M|$, *wenn* $\delta = 5$.

Beweis. Wir betrachten eine planare Einbettung $\mathcal{G}$ von G und führen einige Modifikationen durch um wieder einen planaren bipartiten Graphen zu erhalten. Dazu betrachten wir einen Spannbaum T des blauen Subgraphen (der ja zusammenhängend ist). In T ist jede Matchingkante enthalten, denn jede Matchingkante ist vollkommen offen und hat somit einen Endknoten der nur den anderen Endknoten als blauen Nachbarn hat. Dies zeigt auch dass jede Matchingkante zu einem Blatt inzident ist. Alle grünen Kanten, die nicht in T sind, werden aus $\mathcal{G}$ entfernt.

Nun beginnen wir eine einfache Kurve $\mathcal{K}$ um T herum zu zeichnen. Wir beginnen an einem beliebigen Blattknoten x, dessen Position als Start- und Endpunkt von $\mathcal{K}$ dient. Nun wird $\mathcal{K}$ entlang einer Traversierung des Baumes gezeichnet, bis wir wieder bei x angelangt sind. Dabei soll $\mathcal{K}$ immer knapp neben den blauen und grünen Kanten verlaufen und blaue

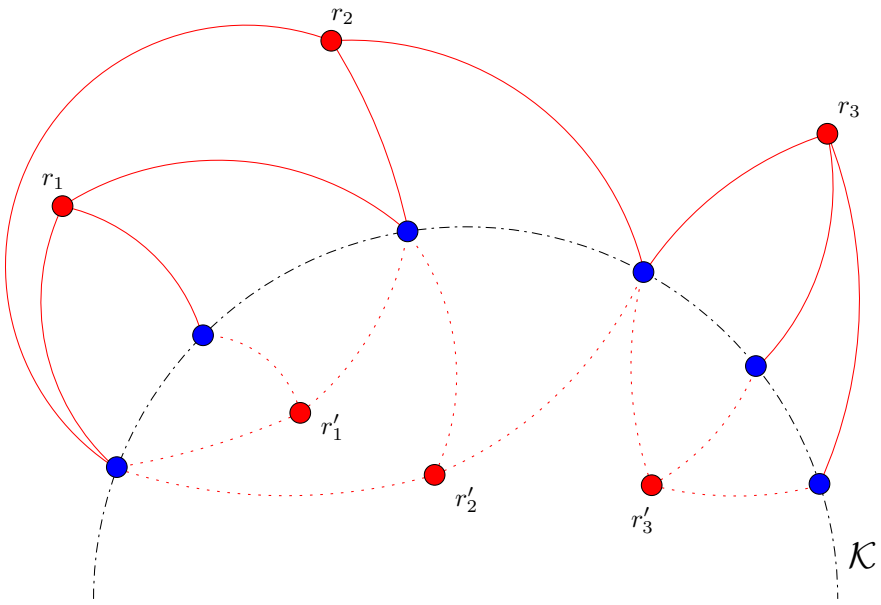


Abbildung 6: Die roten Knoten r_1,r_2 und r_3 werden dupliziert und im Inneren von $\mathcal{K}$ spiegelsymmetrisch kombinatorisch eingebettet

Knoten, die keine Blätter sind, nicht berühren. Die Blattknoten werden durchquert, und zwar so, dass keine rote Kante gekreuzt wird.

Durch dieses Konstruktionsverfahren kreuzt $\mathcal{K}$ keine Kante in $\mathcal{G}$. Die Anzahl der Knoten die auf $\mathcal{K}$ liegen ist genau $|M|$ (jede Matchingkante entspricht genau einem Blatt in T). Im Innern von $\mathcal{K}$ liegen nur blaue Knoten, die jetzt aus $\mathcal{G}$ entfernt werden. Außerdem verzerren wir die Einbettung so, dass $\mathcal{K}$ einen Kreis beschreibt und wir innen ein wenig Platz haben.

Sei R die Menge der roten Knoten (es sind immer noch alle vom Anfang erhalten). Jetzt wird jeder rote Knoten dupliziert und die Kopie im Innern von $\mathcal{K}$ eingezeichnet. Dies ist möglich ohne die Planarität zu verletzen (dies illustriert Abbildung 6).

Den so gewonnen planaren bipartiten Graphen nennen wir G' und hat offensichtlich $|M|$ blaue und $2|R|$ rote Knoten. Jeder rote Knoten hat Minimalgrad δ, somit hat G' mindestens $2\delta|R|$ Kanten. Andererseits sagt Lemma 3.3, dass G' höchstens $2(|M| + 2|R|) - 4$ Kanten haben kann. Damit folgt:

$$2\delta|R| \leq 2(|M| + 2|R|) - 4$$
$$(\delta - 2)|R| \leq |M| - 2$$
$$(\delta - 2)(n - 2|M|) \leq |M| - 2$$
$$(\delta - 2)n + 2 \leq |M| + 2|M|(\delta - 2) = |M|(1 + 2(\delta - 2))$$
$$\frac{(\delta - 2)n + 2}{(2\delta - 3)} \leq |M|$$

Setzt man für δ die Werte 3, 4 und 5 ein, so erhält man die behaupteten Schranken. □

Es stellt sich die Frage, wie man vollkommen offene Matchings finden kann. Um die Suche etwas leichter zu machen überlegen wir zunächst wie wir die Anforderungen an die Matchings etwas verringern können und trotzdem die gewünschte Struktur erhalten.

4.3 Die Forderung abschwächen

Die Forderung, dass alle Matchingkanten vollkommen offen sein sollen, ist ziemlich restriktiv. Deshalb wollen wir versuchen, eine etwas schwächere Eigenschaft zu formulieren, die aber auch ausreichend ist.

Betrachtet man eine vollkommen offene Matchingkante in einer planaren Einbettung, so stellt man fest, dass es nicht schlimm ist, wenn der Endknoten mit den roten Nachbarn auch noch blaue Nachbarn hat, solange die roten Kanten in einem Intervall "gebündelt" bleiben. Denn dann könnte man ebenfalls eine einfache Kurve einzeichnen, die die roten und die blauen Knoten trennt. Die nächste Definition formalisiert diese Eigenschaft.

Definition 4.3. *Sei $G = (V, E)$ ein planarer Graph mit einer planaren Einbettung $\mathcal{G}$. Für einen Knoten $v \in V$ mit einem Nachbar v_1 bezeichne $Z_{v_1}(v)$ die bzgl. $\mathcal{G}$ im Uhrzeigersinn geordnete Folge $(v_1, \ldots, v_{d(v)})$ der Nachbarn von v. Sei weiter $M \subseteq E$ ein Matching und $x \in V$ ein blauer Knoten mit Matchingpartner y.*

(i) $Z_M(x) := Z_y(x)$, *d. h. $Z_M(x)$ ist eine geordnete Folge der Nachbarn von x, beginnend mit seinem Matchingpartner.*

(ii) Falls es in der Folge $Z_M(x)$ zwischen zwei beliebigen roten Knoten keinen blauen gibt (d. h. die roten Knoten formen ein Intervall), so heißt x zyklisch rein *(bzgl. $\mathcal{G}$).*

(iii) Eine Matchingkante, bei der ein Endknoten keine rote Nachbarn hat und der andere Endknoten zyklisch rein ist, heißt zyklisch rein *(bzgl. $\mathcal{G}$).*

(iv) Ist jede Matchingkante zyklisch rein, so heißt das Matching zyklisch rein.

Folgerung 4.4. *Sei $G = (V, E)$ ein planarer Graph mit planarer Einbettung $\mathcal{G}$ und $M \subseteq E$ ein zusammenhängendes inklusionsmaximales Matching, das ausschließlich zyklisch reine Kanten besitzt. Jeder rote Knoten habe mindestens Grad δ. Dann gilt:*

(i) $\frac{n+2}{3} \leq |M|$, *wenn* $\delta = 3$.

(ii) $\frac{2n+2}{5} \leq |M|$, *wenn* $\delta = 4$.

(iii) $\frac{3n+2}{7} \leq |M|$, *wenn* $\delta = 5$.

Beweis. Wir modifizieren G zu einem planaren Graphen G', für den M ein zusammenhängendes inklusionsmaximales Matching ist, das nur aus vollkommen offenen Kanten besteht. Dann folgt die Behauptung aus Satz 4.2.

Für einen zyklisch reinen Knoten x und seinem Matchingpartner y, wobei die Kante $\{x, y\}$ nicht vollkommen offen ist, modifizieren wir G, indem wir die zu x inzidenten grünen

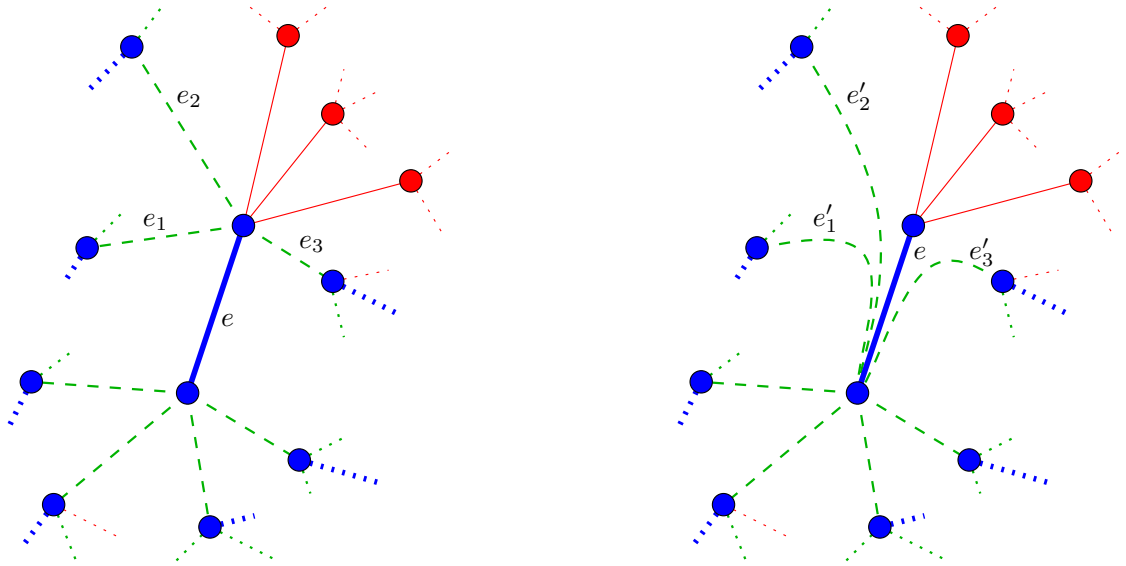


Abbildung 7: Eine zyklisch reine Kante e wird durch Umhängen der grünen Kanten e_1, e_2 und e_3 vollkommen offen

Kanten "umhängen". Eine grüne Kante $\{v, x\}$ wird entfernt und dafür die grüne Kante $\{v, y\}$ eingefügt (sofern diese noch nicht vorhanden ist). Da keine grüne Kante zwischen roten Kanten "eingesperrt" ist, ist dies möglich ohne die Planarität des Graphen zu verletzen (siehe Abbildung 7).

Durch diesen Kantenaustausch bleibt das Matching zusammenhängend, denn ein Weg, der die alte Kante benutzt hat, kann stattdessen auch die "umgehängte" Gegenstück und gegebenenfalls die Matchingkante benutzen. Auch die Inklusionsmaximalität bleibt erhalten: Ein roter Knoten hat nach der Modifikation immer noch dieselben (blauen) Nachbarn.

Wir wiederholen diese Modifikation solange, bis alle Matchingkanten vollkommen offen sind und erhalten dadurch den gesuchten Graph G', für den M ein Matching ist (Endknoten von Matchingkanten sind immer noch verschieden). Da G' die selbe Anzahl Knoten wie G hat, übertragen sich die Schranken für $|M|$ aus Satz 4.2. □

Um ein besseres Gefühl für die vorgestellte Struktur zu geben wird zur Veranschaulichung im folgenden Abschnitt ein Beispiel vorgestellt.

4.4 Ein Beispiel

Wie nehmen erneut den maximal planaren bipartiten Graphen Γ_k (siehe Abschnitt 3.3, Abbildung 2) als Ausgangspunkt und modifizieren ihn zu $\tilde{\Gamma}_k$ (siehe Abbilding 8): Dazu entfernen wir die roten Knoten $r_1^{(2)}, \ldots, r_{2k-2}^{(2)}$ und fügen die neuen blauen Knoten $b'_0, \ldots, b'_{2k-1}$ ein. Dann wird jedes b_i mit b'_i verbunden, diese neuen Kanten bilden das Matching $\tilde{M}_k$. Zusätzlich kommen noch die Kanten $\{b'_i, b'_{i+1}\}$ (für $i \in \{1, \ldots, 2k-2\}$) hinzu, um den Zusammenhang des blauen Subgraphen zu gewährleisten.

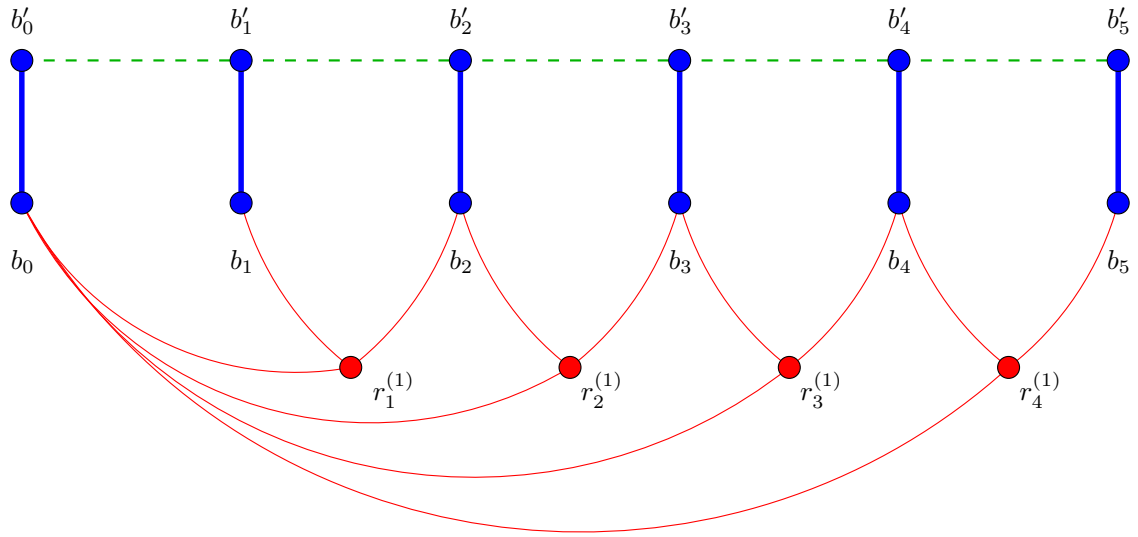


Abbildung 8: $\tilde{\Gamma}_3$ mit der durch $\tilde{M}_3$ induzierten Färbung

Der auf diese Weise erzeugte Graph $\tilde{\Gamma}_k$ hat $n_{\tilde{\Gamma}_k} = 6k - 2$ Knoten. Das Matching $\tilde{M}_k$ hat die Grösse $2k$. Damit gilt:

$$\frac{n_{\tilde{\Gamma}_k} + 2}{3} = \frac{6k - 2 + 2}{3} = 2k = |\tilde{M}_k|.$$

Dies ist die Gleichheit in der Ungleichung aus 4.2 (i).

Beim Betrachten von Abbildung 8 erkennt man wie die ursprüngliche Idee nach der Forderung einer blauen Komponente greift. Die b'_i dürfen keine roten Nachbarn haben und dadurch, dass sie zusammenhängen, versperren sie den Weg für rote Kanten. Würde eine der Kanten $\{b'_i, b'_{i+1}\}$ fehlen, so könnte man einen weiteren roten Knoten hinzufügen.

Für $\delta \in \{4, 5\}$ lassen sich vollkommen analog Graphen konstruieren, die zeigen, dass die Schranken (ii) und (iii) aus Satz 4.2 scharf sind.

Nach dieser ausführlichen Untersuchung von inklusionsmaximalen zusammenhängenden Matchings mit zyklisch reinen bzw. vollkommen offenen Kanten stellt sich die Frage, ob es solche Matchings immer gibt und wenn ja, wie man sie finden kann. Der nächste Abschnitt wird sich damit befassen, solche Matchings zu "beschaffen".

5 Die Algorithmische Umsetzung

In diesem Abschnitt entwickeln wir ein Verfahren, das uns Matchings mit der im letzten Abschnitt vorgestellten Struktur liefern soll. Dabei werden wir uns zuerst auf Minimalgrad 3 beschränken und später untersuchen in wie weit sich das Verfahren auf höheren Minimalgrad übertragen lässt. Wir werden beweisen, dass der Algorithmus die Schranke $(|V| + 2)/3$ aus Satz 1.1 garantiert. Weiter werden wir sehen, dass sich das Verfahren in Linearzeit umsetzen lässt.

5.1 Ein grober Überblick

Wir suchen ein Verfahren, welches uns ein zusammenhängendes inklusionsmaximales Matching mit ausschließlich zyklisch reinen Kanten liefert. Um dies zu erreichen soll das Matching nach und nach aufgebaut werden, und zwar so, dass zu jeder Zeit das bisher gefundene Matching zusammenhängend und jeder blaue Knoten zyklisch rein ist.

Aber genau darin liegt die Schwierigkeit, denn, ob man geeignete neue Matchingkanten finden kann, hängt von der Struktur des vorliegenden Graphen ab. Um dennoch immer neue Matchingkanten finden zu können, wird der Graph so modifiziert, dass die gewünschten Eigenschaften erhalten bleiben. Die Modifikationen sind allerdings unkritisch genug, sodass das gefundene Matching im modifizierten Graphen immer noch einem Matching im ursprünglichen Graphen entspricht. Die Modifikationen sind lediglich Hilfsmittel um Aussagen über die Mindestgröße des Matchings machen zu können.

Um das Matching aufzubauen, benutzen wir eine planare Einbettung des Graphen und wählen die erste Matchingkante beliebig. Die Vergrößerung des Matchings geschieht mittels erhöhender Pfade der Länge 3. Dadurch erhält man offene Kanten, die durch die Invariante der zyklischen Reinheit für blaue Knoten dann zyklisch rein sind. In Abschnitt 5.2 wird dieses Vorgehen im Detail beschrieben.

Es kann allerdings passieren, dass keine der gefundenen Matchingkanten auf einem erhöhenden Pfad der Länge 3 liegt, aber das Matching noch nicht inklusionsmaximal ist. In diesem Fall muss man eine geeignete Kante zwischen zwei roten Knoten finden, die es erlaubt das Matching weiter aufzubauen und die gewünschten Invarianten erhält. Wie man diese geeignete Kante findet wird in Abschnitt 5.3 dargestellt.

Zunächst befassen wir uns mit dem Fall, dass der Minimalgrad $\delta = 3$ ist. Die bereits erwähnten Modifikationen umfassen auch das Entfernen von Knoten, allerdings werden pro rotem Knoten zwei blaue entfernt. Bei Minimalgrad 3 wird dadurch die angestrebte Schranke erreicht, dieses Resultat wird in Abschnitt 5.4 vorgestellt. Anschließend wird ausgeführt, wie sich der Algorithmus in Linearzeit realisieren lässt, dies geschied in Abschnitt 5.5.

Wie sich das Verfahren auf höheren Minimalgrad übertragen lässt und welche Schwierigkeiten sich dann ergeben wird in Abschnitt 5.6 diskutiert.

5.2 Einen Teil des Graphen abarbeiten

Hier wird eine Prozedur beschrieben, die eine Kante zwischen zwei roten Knoten als Eingabe bekommt. Diese Kante wird dem Matching hinzugefügt und von ihr ausgehend wird das Matching mittels der Suche nach erhöhenden Pfaden der Länge 3 weiter ausgebaut.

Dazu verwendet man eine Warteschlange Q, die die noch zu untersuchenden Matchingkanten beinhalten soll. Am Anfang wird die Eingabekante in Q eingereiht. Solange die Warteschlange noch nicht leer ist nimmt man sich die nächste Kante $e = \{x, y\}$ herraus und überprüft, ob e auf einem erhöhenden Pfad der Länge 3 liegt. Nun gibt es drei Möglichkeiten für e: Entweder e liegt auf einem erhöhenden Pfad der Länge 3 oder e ist besetzt oder e ist offen (dann gibt es nichts zu tun).

Fall 1: Die Kante e liegt auf einem erhöhenden Pfad der Länge 3. Jetzt müssen die richtigen roten Nachbarn gefunden werden um die zyklische Reinheit der blauen Knoten zu erhalten. Dazu wählen wir in $Z_M(x)$ den ersten roten Nachbarn x' von x und in $Z_M(y)$ den ersten roten Nachbarn y' von y. Ist $x' = y'$ so gibt es in $Z_M(x)$ oder $Z_M(y)$ noch weitere rote Nachbarn (sonst wäre e besetzt). Gibt es in $Z_M(x)$ noch weitere rote Knoten, so setzen wir x' auf seinen Nachfolger, ansonsten setzen wir y' auf seinen Nachfolger in $Z_M(y)$.

Jetzt wird e in M durch die Kanten $\{x, x'\}$ und $\{y, y'\}$ ersetzt. Falls x' und y' noch andere blaue Nachbarn haben, werden die neu entstandenen grünen Kanten entfernt (Abbildung 9). Diese Kanten sind für den Zusammenhang nicht nötig und können die zyklische Reinheit gefährden. Das Entfernen von grünen Kanten ändert nur den Grad von blauen Knoten, nicht aber den von roten. Die neuen Matchingkanten $\{x, x'\}$ und $\{y, y'\}$ werden in die Warteschlange eingereiht, da sie möglicherweise auf einem erhöhenden Pfad der Länge 3 liegen und noch untersucht werden müssen.

Fall 2: Die Kante e ist besetzt durch den roten Knoten r. Hier unterscheiden wir wiederum zwei Fälle: Entweder r hat einen roten Nachbarn (Fall 2.1) oder r hat nurnoch blaue Nachbarn (Fall 2.2).

Fall 2.1: Der rote Knoten r hat rote Nachbarn. Wir gehen, beginnend bei x, die Nachbarn von r im Uhrzeigersinn durch und wählen den ersten roten Knoten r' aus. Die Kante $\{r, r'\}$ wird in M eingefügt (Abbildung 10) und in Q eingereiht. Anschließend entfernen wir alle grünen zu $\{r, r'\}$ inzidenten Kanten, außer $\{r, x\}$.

Fall 2.2: Der rote Knoten r hat keinen roten Nachbarn. Dieser Fall ist ein wenig unangenehm, da wir e ja offen machen wollen, dies aber durch r verhindert wird. Um dieser störenden Situation zu entgehen, löschen wir x, y und r.

Da dadurch der Zusammenhang des blauen Subgraphen verloren gehen kann, müssen wir noch sicherstellen dass dies nicht passiert. Dazu fügen wir grüne Kanten zwischen den blauen Nachbarn von x und y ein. Betrachten wir $\{x, y\}$ als einen Knoten (der durch Kontraktion von e entstehen würde) und $x_1, \ldots, x_k$ seien die im Uhrzeigersinn geordneten blauen Nachbarn

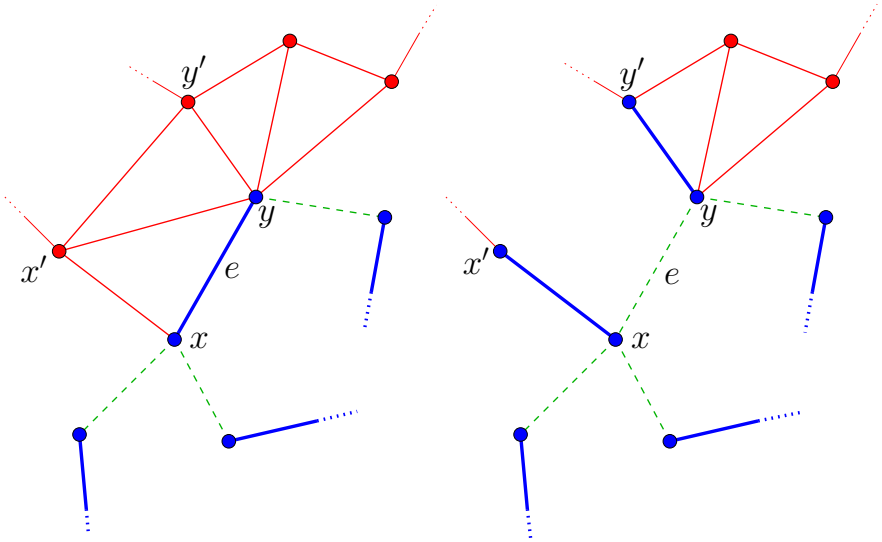


Abbildung 9: Fall 1: Die Kante e liegt auf einem erhöhenden Pfad der Länge 3 und wird durch $\{x, x'\}$ und $\{y, y'\}$ ersetzt.

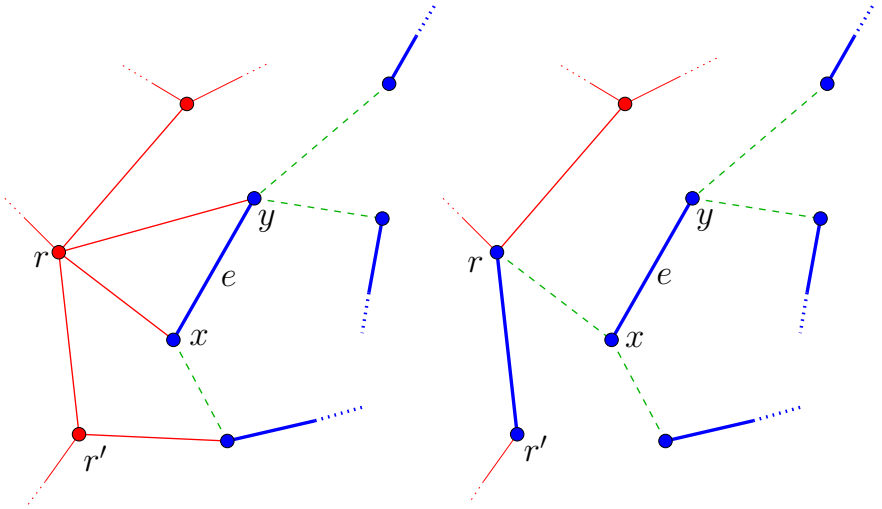


Abbildung 10: Fall 2.1: Ein roter Knoten r, der die Kante e besetzt wird mit einem anderen roten Knoten gematcht

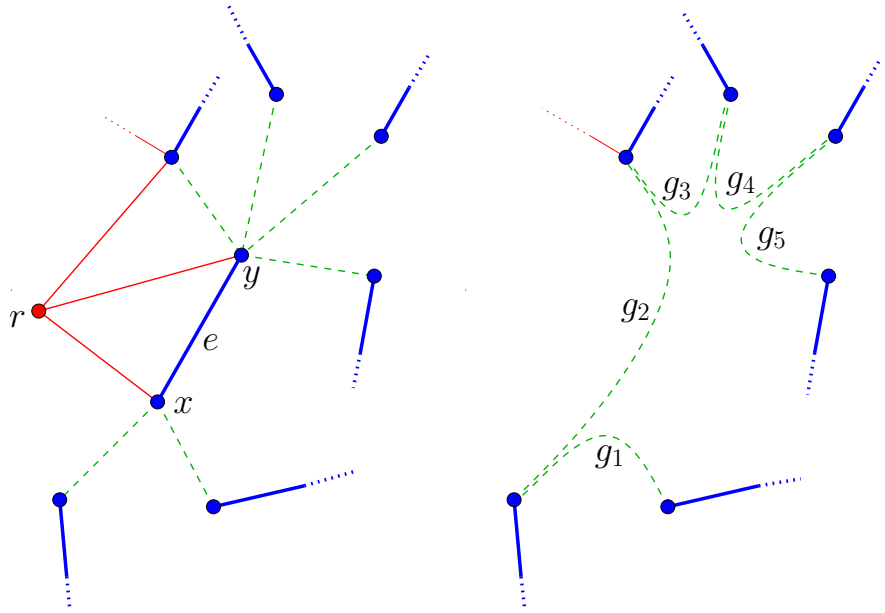


Abbildung 11: Fall 2.2: Der rote Knoten r besetzt die Kante e und hat keine rote Nachbarn. r, x und y werden entfernt.

von $\{x, y\}$. Dann werden für jedes $i \in \{1, \ldots, k-1\}$ die Knoten x_i und x_{i+1} verbunden. Abbildung 11 illustriert dieses vorgehen.

Ist die Warteschlange leer, so ist die Prozedur abgeschlossen. Algorithmus 3 fasst noch einmal das angegebene Vorgehen in Pseudocode zusammen.

Um zu beweisen, dass die angegebene Prozedur auch das liefert, was wir erwarten, stellen wir zunächst einige Schleifeninvarianten über die while-Schleife auf. Diese werden in Lemma 5.1 festgehalten. Anschließend werden die Invarianten benutzt um Aussagen über das Ergebnis der Prozedur selbst zu machen.

Lemma 5.1. *Wird Algorithmus 3 auf einen planar eingebetteten Graphen angewendet, dessen rote Knoten mindestens Grad 3 haben, so erhält die while-Schleife folgende Invarianten:*

(I1) Entfernt man alle roten Knoten, so ist der Rest ein Baum.

(I2) Jeder blaue Knoten ist zyklisch rein.

(I3) Der Grad von roten Knoten ändert sich nicht.

(I4) Der (modifizierte) Graph bleibt planar eingebettet.

(I5) Der (modifizierte) Graph enthält mindestens eine Matchingkante.

Beweis. Wir müssen zeigen, dass ein Schleifendurchlauf die Invarianten erhält. Das bedeutet wir gehen davon aus, dass direkt vor einem Schleifendurchlauf eine Invariante gegolten hat

Algorithmus 3 ERWEITERE MATCHING, AUSGEHEND VON e

Require: $G = (V, E)$ ist ein zusammenhängender planar eingebetteter Graph, M ist ein geeignetes Matching und e ist eine geeignete Kante zwischen zwei roten Knoten
Ensure: M wird vergrößert

```
1: Initialisiere leere Queue Q
2: Füge e in M ein und reihe e in Q ein
3: while Q ist nicht leer do
4:    Entferne die nächste Kante {x, y} aus Q
5:    if {x, y} liegt auf einem erhöhenden Pfad der Länge 3 (Fall 1) then
6:       Finde für x und y zyklisch geeignete rote Nachbarn x' und y'
7:       Ersetze {x, y} in M durch {x, x'} und {y, y'}
8:       Entferne alle grünen zu x' oder y' inzidenten Kanten
9:       Reihe {x, x'} und {y, y'} in Q ein
10:   else if {x, y} ist besetzt durch den roten Knoten r (Fall 2) then
11:      if r hat rote Nachbarn (Fall 2.1) then
12:         Finde einen zyklisch geeigneten roten Nachbarn r' von r
13:         Füge {r, r'} in M ein und reihe {r, r'} in Q ein
14:         Lösche alle grünen, zu {r, r'} inzidenten Kanten, außer {r, x}
15:      else if r hat nur blaue Nachbarn (Fall 2.2) then
16:         Lösche r, x und y
17:         Repariere den Zusammenhang durch das Einfügen grüner Kanten
18:      end if
19:   end if
20: end while
```

und zeigen dass sie direkt nach dem Durchlauf ebenfalls gilt. Dies müssen wir für jede der Invarianten I1, ..., I5 zeigen. Dazu untersuchen wir die Fälle 1, 2.1 und 2.2 getrennt.

Beginnen wir mit *Fall 1*, das heißt die Matchingkante $\{x, y\}$ liegt auf einem erhöhenden Pfad der Länge 3. Dann wechseln die beiden Knoten x' und y' ihre Farbe von rot nach blau, d. h. der blaue Subgraph wird vergrößert. Der einzige blaue Nachbar von x' (bzw. y') ist x (bzw. y), da Kanten zu anderen blauen Knoten entfernt werden. Damit ist der um x' und y' erweiterte blaue Subgraph wieder ein Baum, also gilt (I1).

Die neuen blauen Knoten x' und y' sind zyklisch rein, da sie jeweils nur einen einzigen blauen Nachbarn haben. Die Knoten x und y bleiben zyklisch rein, weil x' in $Z_M(x)$ (bzw. y' in $Z_M(y)$) neben einem blauen Knoten liegt und durch den Farbwechsel das rote Intervall nicht unterbrochen werden kann. Die anderen blauen Knoten v bleiben ebenfalls zyklisch rein, da sie höchstens Nachbarn verlieren (d. h. Knoten aus $Z_M(v)$ entfernt werden), aber keine Farbwechsel innerhalb $Z_M(x)$ stattfinden. Damit ist (I2) sichergestellt.

Es werden ausschließlich grüne Kanten entfernt, also keine zu einem roten Knoten inzidente Kante. Damit ändert sich der Grad eines nach Schleifenaufrufs noch roten Knotens nicht und Invariante (I3) gilt. Es werden nur Kanten aus der planaren Einbettung entfernt und keine neuen Kanten eingefügt, damit gilt Invariante (I4). Invariante (I5) gilt auch, da die neuen Matchingkanten $\{x, x'\}$ und $\{y, y'\}$ im Graphen enthalten sind.

Untersuchen wir nun *Fall 2.1*, in welchem eine Kante $\{r, r'\}$ dem Matching hinzugefügt wird, wobei r ein roter Knoten ist, der die blaue Kante $\{x, y\}$ besetzt. Außer $\{r, x\}$ werden alle Kanten zwischen blauen Knoten und r bzw. r' entfernt. Der blaue Baum wird um den Ast x, r, r' erweitert und das Resultat ist wiederum ein Baum, also ist Invariante (I1) erfüllt.

Der Knoten x bleibt zyklisch rein, da r in $Z_M(x)$ neben einem blauen Knoten liegt (das rote Intervall wird nicht unterbrochen). Der Knoten r' ist zyklisch rein, da er nur einen blauen Nachbarn hat, nämlich r. Der Knoten r ist zyklisch rein, da seine beiden blauen Nachbarn x und r' in $Z_M(r)$ direkt aufeinander folgen (r' wurde gerade so gewählt). Alle anderen blauen Knoten bleiben wie in Fall 1 zyklisch rein, damit gilt Invariante (I2). Die Invarianten (I3), (I4) und (I5) gelten mit der gleichen Begründung wie in Fall 1.

Nun bleibt noch *Fall 2.2* zu untersuchen. Der rote Knoten r besetzt die Matchingkante $\{x, y\}$ und hat nur blaue Nachbarn. Die Knoten r, x und y werden entfernt und es werden neue grüne Kanten eingefügt. Wir betrachten wieder $\{x, y\}$ als Knoten und $x_1, \ldots, x_k$ als die im Uhrzeigersinn geordneten blauen Nachbarn. Vor dem Entfernen von x und y waren beliebige x_i, x_j (ohne Einschränkung $i < j$) im blauen Baum durch einen eindeutigen Weg über x oder y (oder beide) verbunden. Nun gibt es wieder einen eindeutigen Weg von x_i nach x_j, nämlich $x_i, x_{i+1} \ldots, x_{j-1}, x_j$. Jeder blaue Weg im alten Graphen, der über x oder y verlief, hat somit einen eindeutigen Ersatz im modifizierten Graphen. Damit ist im modifizierten Graphen der blaue Subgraph wieder ein Baum und Invariante (I1) gilt.

Die neuen grünen Kanten werden so eingefügt, dass für jedes der x_i in $Z_M(x_i)$ der Knoten x (bzw. y) durch die beiden Knoten x_{i-1} und x_{i+1} (bzw. nur x_{i+1}, falls $i = 1$, oder nur x_{i-1}, falls $i = k$) ersetzt wird. Man beachte, dass in $Z_M(x_i)$ nicht sowohl x als auch y auftreten können, da es sonst vorher schon einen blauen Kreis der Länge 3 gegeben hätte. Damit bleibt jeder der x_i zyklisch rein, da das rote Intervall nicht unterbrochen wird. Somit gilt Invariante

(I2).

Wie in Abbildung 11 angedeutet, werden die Kanten $\{x_i, x_{i+1}\}$ dicht neben den gelöschten Kanten eingezeichnet. Waren x_i und x_{i+1} beide zu x (bzw. y) adjazent, so verläuft $\{x_i, x_{i+1}\}$ entlang $\{x_i, x\}$ und $\{x, x_{i+1}\}$ (bzw. $\{x_i, y\}$ und $\{y, x_{i+1}\}$). War x_i zu x und x_{i+1} zu y adjazent, so verläuft $\{x_i, x_{i+1}\}$ entlang der drei Kanten $\{x_i, x\}$, $\{x, y\}$ und $\{y, x_{i+1}\}$ (umgekehrt analog). Da die neuen Kanten planar eingebettet werden, ist Invariante (I4) erfüllt.

Alle der drei gelöschten Knoten x, y und r hatten nur blaue Nachbarn, der Grad der übrigen roten Knoten wird also nicht verringert. Die Enden aller eingefügten Kanten sind blau, damit bleibt der Grad der roten Knoten unverändert und Invariante (I3) ist gewährleistet.

Der rote Knoten r hatte mindestens Grad 3, also gibt es neben x und y noch einen dritten blauen Knoten z. Dann muss es noch eine weitere Matchingkante $\{z, z'\}$ im Graphen geben. Diese wird nicht gelöscht und somit bleibt Invariante (I5) erhalten.

Da in jedem Schleifendurchlauf entweder einer der drei untersuchten Fälle eintritt oder alles unverändert bleibt, haben wir insgesamt die Invarianten bewiesen. □

Das folgende Lemma 5.2 hält fest, dass die Ausführung von Algorithmus 3 das gewünschte Ergebnis liefert. In den Voraussetzungen und Aussagen wird schon berücksichtigt, dass der Algorithmus wiederholt ausgeführt werden kann. Deshalb kann es sein, dass das eingegebene Matching M auch Kanten enthält, die nicht im Eingabegraphen enthalten sind, da sie in einer früheren Ausführung schon entfernt wurden. Genau genommen ist M somit kein Matching für den eingegebenen Graphen G, sondern nur die Einschränkung von M auf G.

Lemma 5.2. *Sei $G = (V, E)$ planar eingebetteter, zusammenhängender Graph und M ein Matching (genauer: eine unabhängige Kantenmenge, die auch Kanten enthalten kann, die nicht in E liegen). Jeder Knoten, zu dem keine Kante aus M inzident ist, habe mindestens Grad 3. Weiter sei $e \in E - M$ eine Kante, mit der Eigenschaft, dass $M' := (E \cap M) \cup \{e\}$ ein baumförmiges Matching für G ist, in welchem jede Kante aus $(E \cap M)$ zyklisch rein ist. Weiter seien die beiden Endknoten von e zykisch rein.*

Liefert die Ausführung von Algorithmus 3 (mit G und e als Eingabeparameter) den modifizierten Graphen $\tilde{G} = (\tilde{V}, \tilde{E})$ (mit einer Einbettung) und die aus M entstandene Kantenmenge $\tilde{M}$, dann gilt:

(i) $\tilde{M} \cap \tilde{E}$ ist ein baumförmiges, zyklisch reines Matching für $\tilde{G}$.

(ii) $\tilde{M} \cap E$ ist ein Matching für G und $\tilde{M} - M \subseteq E$.

(iii) Jeder rote Knoten in $\tilde{G}$ (bzgl. $\tilde{M} \cap \tilde{E}$) hat mindestens Grad 3.

(iv) Die Einbettung von $\tilde{G}$ ist planar.

(v) $\tilde{M} \cap \tilde{E} \neq \emptyset$.

Beweis. Zu Beginn von Algorithmus 3 wird die Kante e in M und in eine leere Warteschlange eingefügt. Direkt vor dem ersten Schleifendurchlauf ist die aktuelle Kantenmenge somit $M' = M \cap \{e\}$ und der aktuelle Graph G.

Für G und M' gelten die Eigenschaften (I1), ..., (I5) aus Lemma 5.1 nach Voraussetzung und bleiben nach jedem Schleifendurchlauf erhalten. Induktiv folgt also, dass nach der Terminierung für $\tilde{M}$ und $\tilde{G}$ die Invarianten gelten. Damit folgen aus den Invarianten (I3), (I4) und (I5) sofort die Behauptungen (iii), (iv) und (v).

Überprüfen wir nun Eigenschaft (ii). Ist $M \cap E$ ein Matching dann ist auch $\tilde{M} \cap \tilde{E}$ ein Matching, da nur Kanten aufgenommen werden, die zu keiner blauen Kante inzident sind, bzw. falls sie zu einer blauen Kante e inzident sind, wird e dafür aus dem Matching entfernt. Es bleibt noch zu klären, ob jede Kante, die ins Matching aufgenommen wird, auch im ursprünglichen Graphen vorhanden war.

Dazu zwei Beobachtungen: Es werden ausschließlich rote Kanten aufgenommen (im Fall 1 bzw. 2.1 der while-Schleife) und es werden ausschließlich grüne Kanten eingefügt (im Fall 2.2 der while-Schleife). Da eine grüne Kante nichtmehr rot werden kann und nur rote Kanten aufgenommen werden, heißt das, dass keine der zusätzlich eingefügten Kanten ins Matching aufgenommen wird. Damit war jede hinzugefügte Matchingkante (also jede Kante in $\tilde{M} - M$) auch schon in E.

Nun bleibt noch Eigenschaft (i) zu zeigen. Invariante (I1) sagt uns, dass $\tilde{M} \cap \tilde{E}$ ein baumförmiges Matching in $\tilde{G}$ ist. Weiter wissen wir durch Invariante (I2), dass jeder blaue Knoten zyklisch rein ist. Um die Behauptung zu zeigen, müssen wir zusätzlich zeigen, dass jede Kante aus $\tilde{M} \cap \tilde{E}$ in $\tilde{G}$ einen Endknoten ohne rote Nachbarn besitzt.

Dazu betrachten wir eine neu hinzugekommene Matchingkante $e \in (\tilde{M} \cap \tilde{E}) - M$. Da e neu hinzugekommen ist, wurde sie auch irgendwann in die Warteschlange eingereiht. Als sie dann an der Reihe war, kann weder Fall 1 noch Fall 2.2 eingetreten sein, da e sonst aus dem Matching oder aus dem Graphen entfernt worden wäre. Ist der Fall 2.1 eingetreten, so war sie durch einen roten Knoten besetzt, der blau geworden ist. Dann haben nun beide Endknoten keine rote Nachbarn und e ist offen. Ist weder Fall 1 noch Fall 2 eingetreten, so war die Kante e schon zyklisch rein.

Eine "alte" Matchingkante $e \in M \cap \tilde{E}$ blieb von der Ausführung des Algorithmus unberührt und hat höchstens inzidente Kanten verloren. Damit ist e weiterhin zyklisch rein und Eigenschaft (i) gilt. □

In den Voraussetzungen von Lemma 5.2 wird an die Eingabekante e die Forderung gestellt, dass $M' := M \cup \{e\}$ ein zusammenhängendes Matching ist und jede Kante aus M' zyklisch rein ist. Zu Beginn, wenn $M = \emptyset$ und $M' = \{e\}$ ist, ist diese Forderung trivialerweise erfüllt. Es stellt sich die Frage, wie man so eine solche Kante finden kann, falls schon ein Teil des Graphen gematcht ist. Mit diesen Fall befasst sich der nächste Abschnitt.

5.3 Wo macht man weiter?

Zu Beginn, wenn alle Knoten noch rot sind, ist jede Kante geeignet, um als Ausgangspunkt für den Aufbau des Matchings zu dienen. Das in Abschnitt 5.2 vorgestellte Verfahren hinterlässt einen modifizierten Graphen der blaue Kanten enthält und in dem jede blaue Kante zyklisch rein ist (Lemma 5.2).

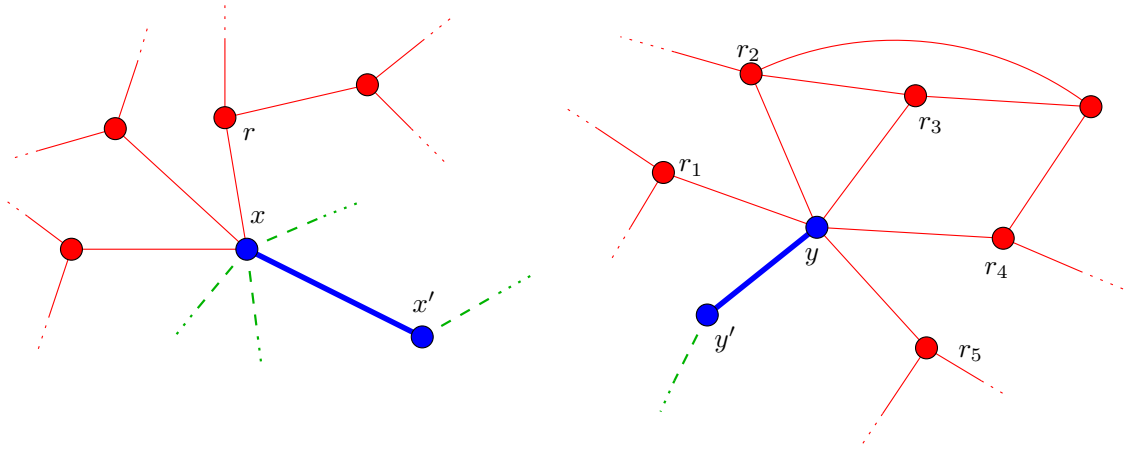


Abbildung 12: Der rote Knoten $r \in R_R$ führt zu einer geeigneten Kanten, um das Matching zu erweitern. Die beiden roten Nachbarn r_1 und r_5 von y liegen in R_B und sperren die Knoten $r_2, r_3, r_4 \in R_R$ aus. Bei y lässt sich keine geeignete Kante finden.

Allerdings können in dem neuen Graphen noch zwei rote Knoten benachbart sein (dann wäre das Matching noch nicht inklusionsmaximal). In dem Fall benutzen wir die in Abschnitt 5.2 vorgestellte Prozedur erneut. Das wird solange wiederholt bis am Ende ein Graph mit einen zusammenhängenden inklusionsmaximalen zyklisch reinem Matching übrig bleibt.

Das Problem ist, dass man nun nicht beliebig eine Kante mit zwei roten Enden auswählen kann, da möglicherweise die Auswahl dafür sorgt dass das Matching nicht zusammenhängend wird, oder dass eine blaue Kante entsteht die nicht zyklisch rein ist. Wie man an eine "geeignete" Kante kommt, wird nun genauer beschrieben.

Wir wollen die vorhandene Struktur, die die zyklisch reinen Kanten bilden, genauer untersuchen. Dazu unterteilen wir die roten Knoten in zwei Gruppen: Die Menge R_R enthalte diejenigen roten Knoten, die mindestens einen roten Nachbar haben und die Menge R_B die übrigen roten Knoten (also die, die nur blaue Nachbarn haben). Die Menge der blauen Knoten wird mit B bezeichnet und die Menge der roten Knoten mit R.

Solange die Menge R_R nicht leer ist, ist das Matching noch nicht inklusionsmaximal und wir sind noch nicht fertig. Die Knoten aus R_R liefern uns Kandidaten für Kanten, die wir an Algorithmus 3 übergeben können. Damit das Matching zusammenhängend und zyklisch rein bleibt, suchen wir einen blauen Knoten x und einen zu x adjazenten roten Knoten $r \in R_R$, sodass r in $Z_M(x)$ der erste oder der letzte rote Knoten ist. Dann liefert uns nämlich r eine geeignete Kante um erneut Algorithmus 3 auszuführen (siehe Abbildung 12), denn die zyklische Reinheit von x bleibt erhalten, da r ein roter "Randnachbar" von x ist.

Es stellt sich die Frage ob wir immer so einen blauen Knoten x finden können. Man könnte sich vorstellen, dass bei jedem blauen Knoten, die roten Nachbarn aus R_R durch zwei rote Nachbarn aus R_B "ausgesperrt" werden (siehe Abbildung 12). Das nachfolgende Lemma zeigt, dass es immer einen blauen Knoten gibt, der nur einen Nachbarn aus R_B hat, bei dem diese Situation also nicht auftreten kann.

Lemma 5.3. *Es sei $G = (V, E)$ ein planarer Graph mit einer durch ein zyklisch reines Matching induzierten Färbung, sodass der blaue Subgraph ein Baum ist. Gibt es Knoten in R_B (also rote Knoten, die nur blaue Nachbarn haben) und hat jeder Knoten aus R_B mindestens Grad 3, dann gibt es einen blauen Knoten $x \in B$ mit $d_{R_B}(x) = 1$.*

Beweis. Für den Beweis sind ausschließlich die Knoten aus R_B und die blauen Knoten relevant. Entfernen wir also die Knoten aus R_R und zeichnen (wie im Beweis von Satz 4.2) eine Kurve $\mathcal{K}$ um den blauen Subgraphen herum. Die blauen Kanten sind in unserer Situation zwar nur zyklisch rein, und nicht, wie in Beweis von Satz 4.2 vollkommen offen, aber wir können dennoch $\mathcal{K}$ mit den gewünschten Eigenschaften konstruieren.

Dies erreichen wir, indem wir bei einem blauen Knoten b mit $d_{R_B}(b) > 0$ beginnen und $\mathcal{K}$ entlang einer Traversierung des blauen Baumes zeichnen und zwar so, dass bei jedem blauen Knoten mit roten Nachbarn, den wir besuchen das "Bündel" der roten Kanten ausgesperrt wird (siehe auch Abbildung 13).

Die so konstruierte Kurve $\mathcal{K}$ kreuzt keine Kanten und auf ihr liegen ausschließlich blaue Knoten, die noch rote Nachbarn haben. Die Kurve trennt die blauen und die roten Knoten. Die äußere Facette liegt auf der roten Seite, da der blaue Subgraph ein Baum war und wir ihn eingekreist haben. Deshalb sprechen wir im Folgenden auch von äußerer und innerer Seite.

Auf der äußeren Seite induzieren die roten Knoten (von denen jeder mindestens mit drei blauen Knoten auf $\mathcal{K}$ verbunden ist) eine Klammerstruktur. Nun müssen wir uns nurnoch einen roten Knoten r herauspicken, dessen Klammer keine tiefer geschachtelte Klammer beinhaltet.

Seien $b_1, \ldots, b_k$ die Knoten auf $\mathcal{K}$, geordnet im Uhrzeigersinn (von innen gesehen). Dabei soll b_1 an der äußeren Facette liegen (damit für die Nachbarn eines roten Knoten, diejenigen mit den kleinsten und größten Index den Klammeranfang und das Klammerende markieren). Es muss Knoten auf $\mathcal{K}$ geben, die an der äußeren Facette liegen, da beispielsweise der Klammeranfang eines rote Knotens, der an der äußeren Facette liegt, auch an der äußeren Facette liegen muss.

Nehmen wir nun die Knoten b_i und b_j am Klammeranfang und Klammerende von r. Da r mindestens Grad 3 hat, gibt es noch einen weiteren blauen Knoten b_x mit $i < x < j$. Dieser Knoten b_x kann keine anderen rote Nachbarn haben, da r ja eine innere Klammer war. Damit gilt $d_{R_B}(b_x) = 1$. □

Somit können wir einen blauen Knoten x mit $d_{R_B}(x) = 1$ finden. Wir unterscheiden zwischen dem Fall, dass x noch weitere rote Nachbarn hat (die dann aus R_R sein müssen), und dem Fall, dass x nur einen einzigen roten Nachbar hat (der in R_B ist).

Fall 1: Hat x nun noch rote Nachbarn aus R_R, so ist der erste oder der letzte rote Knoten in $Z_M(x)$ aus R_R. Sei $r \in R_R$ ein solcher Knoten. Wir wählen einen roten Nachbarn r' von r, und zwar so, dass r' der erste rote Knoten in $Z_x(r)$ ist. Die Kante $\{r, r'\}$ ist dazu geeignet, das Matching mittels Algorithmus 3 weiter zu vergrößern.

Wir müssen noch sicherstellen, dass keine störenden grünen Kanten entstehen, die die zyklische Reinheit von anderen blauen Knoten gefährden. Dazu entfernen wir alle zu $\{r, r'\}$

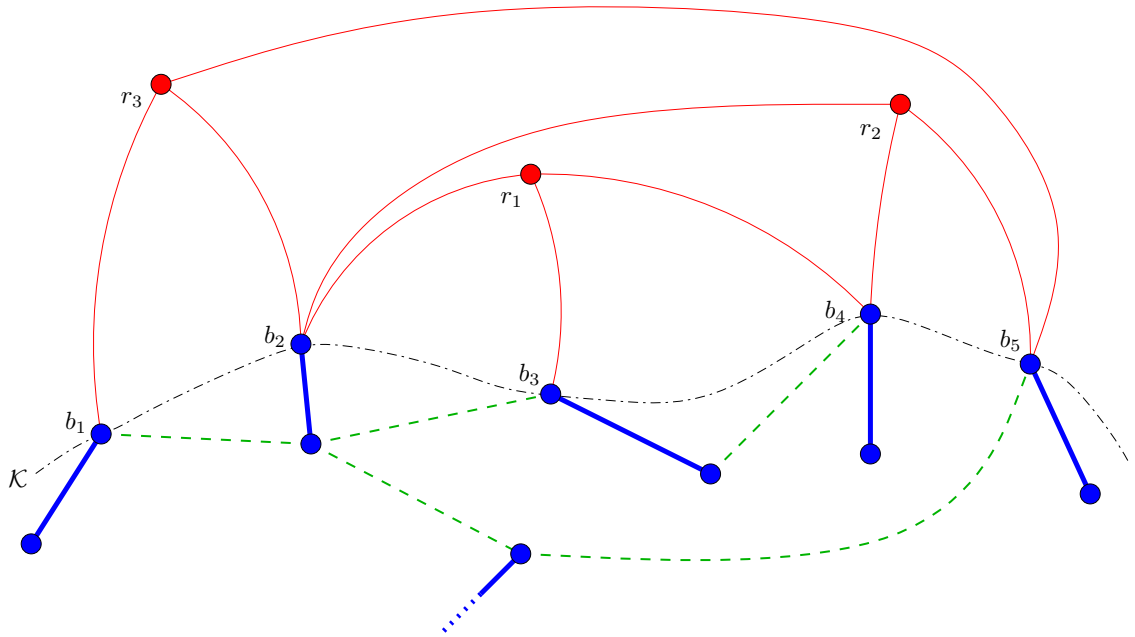


Abbildung 13: Die von r_1 erzeugte Klammer enthält keine weitere Klammer, deshalb gibt es einen blauen Nachbarn (hier b_3) von r_1, der keinen anderen roten Nachbarn hat.

inzidenten Kanten mit einem blauen Ende außer $\{x, r\}$. Die Kante $\{x, r\}$ sorgt dafür dass das Matching zusammenhängend bleibt. Nach dem Entfernen rufen wir "ERWEITERE MATCHING, AUSGEHEND VON $\{r, r'\}$" auf und untersuchen später erneut ob R_R dadurch leer geworden ist.

Fall 2: Sei r der rote Nachbar von x. Wir können x, r und den Matchingpartner von x aus dem Graphen entfernen, um dadurch bei den anderen blauen Nachbarn von r vielleicht einen roten Nachbarn aus R_R "freizulegen". Wie im Fall 2.2 des Algorithmus 3 müssen wir sicherstellen, dass der blaue Subgraph ein Baum bleibt. Dazu fügen wir auf dieselbe Weise wie dort neue grüne Kanten ein.

Ist R_B noch nicht leer, so wird erneut ein blauer Knoten x mit $d_{R_B}(x) = 1$ ausgewählt und wiederum geschaut ob x noch weitere rote Nachbarn hat.

Tritt irgendwann der Fall ein, dass R_B leer ist, aber R_R noch nicht, so lässt sich ein blauer Knoten x finden, der rote Nachbarn hat (sonst wäre der Graph nicht zusammenhängend). Jetzt kann man mit x wie im Fall 1 verfahren und als r den ersten roten Nachbarn in $Z_M(x)$ wählen.

Wir wiederholen die beschriebenen Schritte solange, bis R_R leer geworden ist. Für den Graphen G', der dann übrig bleibt, haben wir ein inklusionsmaximales Matching gefunden. Zusätzlich ist jede blaue Kante zyklisch rein. Die Erkenntnisse aus Abschnitt 4 sagen uns nun, dass für den in G' liegenden Teil des gefundenen Matchings die gewünschte Mindestgröße vorhanden ist.

Das gesamte Verfahren ist als Algorithmus 4 zusammengefasst. Algorithmus 3 wird als Prozedur "ERWEITERE MATCHING, AUSGEHEND VON e" verwendet.

Algorithmus 4 MATCHE DEN GRAPHEN

Require: $G = (V, E)$ ist ein planar eingebetteter, zusammenhängender Graph mit Minimalgrad 3
Ensure: M ist ein Matching mit $|M| \geq \frac{|V|+2}{3}$

$M \leftarrow \emptyset$
Wähle beliebige Kante $e \in E$
ERWEITERE MATCHING, AUSGEHEND VON e
while R_R ist nicht leer **do**
  if R_B ist leer **then**
    Wähle blauen Knoten x mit $d_R(x) > 0$
    Finde ersten roten Knoten r in $Z_M(x)$
    Finde ersten roten Knoten r' in $Z_x(r)$
    Entferne alle zu $\{r, r'\}$ inzidenten Kanten mit blauem Endknoten, außer $\{x, r\}$
    ERWEITERE MATCHING, AUSGEHEND VON $\{r, r'\}$
  else
    Wähle einen blauen Knoten x mit $d_{R_B}(x) = 1$
    if $d_{R_R}(x) > 0$ **then**
      Finde ersten roten Knoten r_1 in $Z_M(x)$
      Finde letzten roten Knoten r_2 in $Z_M(x)$
      Wähle $r \in \{r_1, r_2\} \cap R_R$
      Finde ersten roten Knoten r' in $Z_x(r)$
      Entferne alle zu $\{r, r'\}$ inzidenten Kanten mit blauem Endknoten, außer $\{x, r\}$
      ERWEITERE MATCHING, AUSGEHEND VON $\{r, r'\}$
    else
      Entferne x, den roten Nachbarn von x und den Matchingpartner von x
      Repariere den Zusammenhang durch das Einfügen von grünen Kanten
    end if
  end if
end while

Wir wollen noch zeigen, dass nach der Ausführung von Algorithmus 4 das gefundene Matching eingeschränkt auf den modifizierten Graphen die in Abschnitt 4 beschriebene Struktur besitzt. Im wesentlichen werden hier die Ergebnisse aus Lemma 5.2 für die verwendete Unterprozedur auf das gesamte Verfahren übertragen und um eine zusätzliche Eigenschaft, nämlich die Inklusionsmaximalität, erweitert.

Lemma 5.4. *Sei $G = (V, E)$ ein planar eingebetteter Graph mit Minimalgrad 3. Die Ausführung von Algorithmus 4 liefere die Kantenmenge M und den Graphen $\tilde{G} = (\tilde{V}, \tilde{E})$. Dann gilt:*

(i) $M \cap \tilde{E}$ ist ein inklusionsmaximales, baumförmiges Matching für $\tilde{G}$.

(ii) *M ist ein Matching für G.*

(iii) *Jede blaue Kante in $\tilde{G}$ ist zyklisch rein.*

(iv) *Jeder rote Knoten in $\tilde{G}$ hat mindestens Grad 3.*

(v) *$M \cap \tilde{E}$ ist nicht leer.*

Beweis. Zu Beginn wird M mit einer beliebigen Kante initialisiert und der restliche Graph ist rot. Damit gelten die Voraussetzungen aus Lemma 5.2 trivialerweise. Nach dem ersten Aufruf von Algorithmus 3 haben wir somit im modifizierten Graphen ein baumförmiges Matching, das ausschließlich zyklisch reine Kanten besitzt. Jeder rote Knoten hat mindestens Grad 3, der Graph enthält noch blaue Kanten und M bleibt ein Matching für G.

Nun zeigen wir, dass die while-Schleife in Zeile 4 die genannten Eigenschaften erhält. Damit erhalten wir induktiv, dass sich die Eigenschaften auf $\tilde{G}$ übertragen. Betrachten wir also den Fall dass R_R nicht leer ist und zeigen, dass nach dem Schleifendurchlauf die Eigenschaften immer noch gelten.

Ist R_B leer, so wird eine Kante $\{r, r'\}$ ausgewählt (r und r' sind beide rot), wobei r zu einem blauen Knoten x adjazent ist. Nimmt man nun die Kante $\{r, r'\}$ ins Matching auf, so bleibt der blaue Subgraph ein Baum, da außer der Kante $\{x, r\}$ alle Verbindungen zu anderen blauen Knoten entfernt wurden. Desweiteren wurden r und r' gerade so gewählt, dass x, r und r' zyklisch rein werden. Damit sagt uns Lemma 5.2 dass nach der Ausführung von "ERWEITERE MATCHING, AUSGEHEND VON $\{r, r'\}$" die Behauptung gilt.

Ist R_B nicht leer, so wissen wir durch Lemma 5.3, dass es einen blauen Knoten x mit $d_{R_B}(x) = 1$ gibt. Hat x noch rote Nachbarn aus R_R (Fall 1), so finden wir wieder eine geeignete Kante $\{r, r'\}$, die die Voraussetzungen von Lemma 5.2 erfüllt und zeigen damit die Behauptung.

Hat andererseits x keine weiteren roten Nachbarn, so wird x samt seines roten Nachbars $\tilde{r}$ und seines Matchingpartners entfernt. Der Zusammenhang wird so repariert, dass der blaue Subgraph ein Baum bleibt (analog wie im Beweis von Lemma 5.1). Die anderen Matchingkanten bleiben zyklisch rein und der Graph enthält noch blaue Kanten, da $\tilde{r}$ zu mindestens zwei weiteren blauen Knoten benachbart war, die nicht gelöscht wurden. Der Grad der anderen roten Knoten wurde nicht geändert, da keiner der gelöschten Knoten zu einem von ihnen adjazent war. Da sich M nicht ändert, bleibt M auch ein Matching für den ursprünglichen Graphen G.

Terminiert der Algorithmus, so ist R_R leer und das Matching somit inklusionsmaximal. Die Induktion sagt uns zusätzlich, dass jede Matchingkante zyklisch rein ist, das Matching nicht leer ist, jeder rote Knoten mindestens Grad 3 hat und M ein Matching für den ursprünglichen Graphen G ist. Somit haben wir alle Eigenschaften nachgewiesen. □

Im nächsten Abschnitt werden die bereits vorgestellten Teilergnisse benutzt um eine untere Schranke für die gefundenen Matchings zu zeigen.

5.4 Die Schranke für Minimalgrad 3

Es ist an der Zeit, die einzelnen Stücke zusammenzusetzen und die Mindestgröße der mit Hilfe von Algorithmus 4 gefundenen Matchings zu beweisen. Ein großer Teil der Arbeit wurde schon durch die vorangegangenen Lemmas und den Ergebnissen aus Abschnitt 4 vollbracht. Der nachfolgende Satz wird darauf aufbauend für die gefundenen Matchings die Schranke $(|V|+2)/3$ aus Satz 1.1 zeigen.

Satz 5.5. *Für einen zusammenhängenden, planar eingebetteten Graphen $G = (V, E)$ mit Minimalgrad 3 liefert die Ausführung von Algorithmus 4 ein Matching M mit $|M| \geq \frac{|V|+2}{3}$.*

Beweis. Sei $\tilde{G} = (\tilde{V}, \tilde{E})$ der modifizierte Graph, der durch die Ausführung des Verfahrens entstanden ist. Wir wissen bereits durch Lemma 5.4, dass das Matching $M \cap \tilde{E}$ für den Graphen G die für Folgerung 4.4 benötigte Struktur hat. Damit folgt sofort:

$$\frac{|\tilde{V}|+2}{3} \leq |M \cap \tilde{E}|.$$

Wir wissen auch (ebenfalls aus Lemma 5.4), dass M ein Matching für den ursprünglichen Graphen G ist. Jetzt müssen wir nurnoch die Mindestgröße des auf $\tilde{G}$ eingeschränkten Matchings auf das gesamte Matching M übertragen. Hier hilft uns nun die Beobachtung, dass für jeden roten Knoten, der entfernt wurde, zwei blaue Knoten einer Matchingkante entfernt wurden.

Sei s die Anzahl der während der Ausführung gelöschten roten Knoten. Dann ist $|\tilde{V}| = |V| - 3s$ und $|M \cap \tilde{E}| = |M| - s$. Damit ergibt sich:

$$\frac{|V|-3s+2}{3} \leq |M| - s.$$

Addiert man auf beiden Seiten s, so erhält man das gewünschte Ergebnis:

$$\frac{|V|+2}{3} \leq |M|.$$

□

Dies liefert einen konstruktiven Beweis für die erste Schranke aus Satz 1.1. Ob wir diesen Erfolg auch auf Minimalgrad 4 und 5 übertragen können werden wir später diskutieren. Doch zunächst klären wir die Frage, wie sich das angegebene Verfahren effizient umsetzen lässt.

5.5 Die Umsetzung in Linearzeit

Trotzdem das Verfahren recht kompliziert anmutet, lässt es sich glücklicherweise in Linearzeit realisieren. Zunächst wird erläutert welche Datenstrukturen benutzt werden, um alle nötigen Entscheidungen und Knotenwahlen in konstanter Zeit bewerkstelligen zu können.

Für jeden Knoten verwalten wir, ob er der Menge B (der blauen Knoten), der Menge R_R (der roten Knoten mit rotem Nachbar) oder der Menge R_B (den roten Knoten ohne rote

Nachbarn) angehört. Weiter verwalten wir jede dieser Mengen in einer verketteten Liste, so sind Einfügen und Entfernen in konstanter Zeit möglich. Außerdem kann man so konstanter Zeit testen ob eine der Mengen leer ist.

Desweiteren wird für jeden Knoten x die Anzahl der Nachbarn aus R_R (also $d_{R_R}(x)$) und die Anzahl der Nachbarn aus R_B (also $d_{R_B}(x)$) gespeichert. Um diese Daten aktuell zu halten, benachrichtigt ein Knoten, der seine Menge wechselt oder gelöscht wird, alle seine Nachbarn. Für blaue Knoten b mit $d_{R_B}(b) = 1$ und für blaue Knoten b' mit $d_{R_R}(b') > 0$ wird jeweils eine eigene verkettete Liste verwaltet.

Dass das Verwalten dann amortisiert nur linear viel Zeit benötigt, erkennt man daran, dass ein Knoten höchstens drei mal seine Nachbarn benachrichtigen muss. Zu Beginn sind alle Knoten in R_R. Ein Knoten $r \in R_R$ wechselt irgendwann nach R_B (falls $d_{R_R}(r)$ und $d_{R_B}(r)$ beide 0 werden) oder nach B (falls r gematcht wird). Ein Knoten aus R_B wechselt nach B, wird gelöscht oder bleibt in R_B bis zur Terminierung. Ein blauer Knoten wird entweder gelöscht oder bleibt blau bis zum Ende.

Für jeden blauen Knoten b verwalten wir noch drei Zeiger, einen auf den ersten roten Nachbarn in $Z_M(b)$ (dieser heiße $r_1(b)$), einen auf den letzten (dieser heiße $r_2(b)$) und einen auf seinen Matchingpartner (dieser heiße $b_M(b)$).

Dass das Verwalten des Matchingpartners in linearer Zeit möglich ist, ist leicht einzusehen, denn wenn sich einmal der Matchingpartner ändert, so geht dies in konstanter Zeit und insgesamt kann es nur linear viele Änderungen geben, da ein ehemaliger Matchingpartner von b nicht nocheinmal als Matchingpartner in Frage kommt. Eine blaue Kante kann grün werden, aber eine grüne Kante wird nie mehr blau werden.

Das Verwalten der beiden anderen Zeiger ist amortisiert ebenfalls in linearer Zeit möglich. Dazu betrachten wir exemplarisch den Zeiger r_1 auf den ersten roten Knoten in $Z_M(b)$. Wir verteilen an jede Kante einen gewissen konstanten Kredit an Zeit. Wird der Knoten b blau, so ermitteln wir r_1 indem wir ausgehend vom Matchingpartner von b im Uhrzeigersinn die Nachbarn besuchen und stoppen wenn wir beim ersten roten Knoten angelangt sind. Dafür verbrauchen wir unterwegs jeweils die Hälfte des Kredits jeder Kante die wir benutzen. Ändert r_1 seine Farbe oder wird gelöscht, so müssen wir weiter im Uhrzeigersinn weitergehen um den nächsten roten Knoten zu bekommen. Da aber ein blauer Knoten nichtmehr rot werden kann, gehen wir immer nur in eine Richtung und benutzen jeweils nur einmal den Kredit. Die andere Hälfte des Kredits einer Kante kann gegebenenfalls für den anderen Endknoten verwendet werden.

Damit haben wir alle Datenstrukturen zusammen, die wir für die Realisierung in linearer Zeit brauchen. Für die weitere Analyse verteilen wir noch zwei weitere Kredite. Jeder Knoten v des Graphen erhält einen *Such-Kredit* und einen *Lösch-Kredit* jeweils in $\mathcal{O}(d(v))$. Die Summe der verteilten Kredite liegt damit in $\mathcal{O}(|E|)$. Falls wir später einen der Kredite benutzen werden, werden wir sicherstellen dass er in Zukunft nichtmehr benutzt wird.

Widmen wir uns nun der Laufzeitanalyse der Algorithmen. Dazu betrachten wir zuerst Algorithmus 3 getrennt.

Lemma 5.6. *Ist M die Kantenmenge vor der Ausführung von Algorithmus 3 und $\tilde{M}$ die Kantenmenge dannach, so sei $\Delta = |\tilde{M}| - |M|$. Die Ausführung des Algorithmus' benötigte*

$\mathcal{O}(\Delta)$ *Zeit.*

Beweis. Wir werden zunächst zeigen, dass die Anzahl der Schleifendurchläufe der while-Schleife in Zeile 3 von Algorithmus 3 in $\mathcal{O}(\Delta)$ liegt. Dazu betrachten wir einen konkreten Durchlauf mit der Kante e. Die Anzahl der zukünftigen Schleifendurchläufe erhöht sich maximal um zwei (im Fall 1). Dann erhöht sich aber auch die Grösse von M um eins. Wenn sich im Fall 2.1 die Anzahl der zukünftigen Schleifendurchläufe um eins erhöht, erhöht sich $|M|$ ebenfalls um eins. Damit kann es höchstens 2Δ Schleifendurchläufe geben.

Wir müssen noch prüfen wie lange ein einzelner Schleifendurchlauf dauert. Unsere zuvor definierten Datenstrukturen helfen nun, einen Durchlauf auf konstante Zeit zu amortisieren. Betrachten wir wieder einen Durchlauf mit der Kante $e = \{x, y\}$. Wir müssen prüfen, ob e auf einem erhöhenden Pfad der Länge 3 liegt.

Gilt $d_R(x) > 0$ und $d_R(y) > 0$, so tritt entweder Fall 1 oder Fall 2 ein. Ist einer der beiden Grade Null, so machen wir nichts. Die Entscheidung ist in konstanter Zeit möglich.

Wissen wir, dass Fall 1 oder 2 eintritt, so tritt Fall 2 genau dann ein, wenn $r_1(x) = r_2(x) = r_1(y) = r_2(y)$. Dies zu prüfen ist ebenfalls in konstanter Zeit möglich.

Sind wir im Fall 2, so tritt Fall 2.1 genau dann ein, wenn $d_R(r_1(x)) > 0$. Auch das kann in konstanter Zeit geprüft werden.

Wir können also schon schnell genug entscheiden welcher Fall eintritt. Jetzt müssen wir noch prüfen, dass die Aktionen in den Fällen auch schnell genug durchführbar sind.

Im Fall 1, müssen wir unter der vier Kombinationen die durch $r_i(x)$ und $r_j(y)$ möglich sind, eine auswählen, für die $r_i(x) \neq r_j(y)$ gilt, das geht in konstanter Zeit. Das Aktualisieren der Datenstrukturen ist dann auch in konstanter Zeit möglich.

Im Fall 2.1 müssen wir für $d_R(r_1(x))$ einen geeigneten roten Nachbarn finden, dazu benutzen wir den Such-Kredit von $r_1(x)$. Das anschließende Ändern der Datenstrukturen ist wieder unproblematisch. Der Knoten $r_1(x)$ wird danach blau sein und Such-Kredite von blauen Knoten werden nicht verwendet.

Im Fall 2.2 werden $r_1(x), x$ und y gelöscht, dazu verwenden wir deren Lösch-Kredite. Offensichtlich kann ein Lösch-Kredit nur einmal benutzt werden.

Damit haben wir alle Fälle abgehakt und die Behauptung gezeigt. □

Da die Kantenmenge M nur in der durch Algorithmus 3 beschriebenen Struktur geändert wird, summieren sich die Laufzeiten der Prozeduraufrufe auf einen Wert in $\mathcal{O}(|M|)$. Was noch zu untersuchen bleibt ist, ob die Entscheidungen und Aktionen in Algorithmus 4 amortisiert auch in Linearzeit ablaufen. Dies wird im Beweis des nächsten Lemmas getan.

Lemma 5.7. *Die Ausführung von Algorithmus 4 auf einem planaren zusammenhängenden Graphen mit n Knoten und Minimalgrad 3 geschieht in $\mathcal{O}(n)$ Zeit.*

Beweis. Wir bauen auf dem Resultat aus Lemma 5.6 auf und ignorieren die Prozeduraufrufe. Wir zeigen nur noch, dass die übrigen Aktionen und Entscheidungen insgesamt in Linearzeit ablaufen. Zunächst stellen wir fest, dass in jedem Schleifendurchlauf der while-Schleife in Zeile 4 die Anzahl der roten Knoten im Graphen kleiner wird. Damit ist die Anzahl der Durchläufe beschränkt durch n.

Betrachten wir nun einen Durchlauf. Die Entscheidung, ob R_B leer ist oder nicht, ist dank der Datenstrukturen in konstanter Zeit möglich ist. Ist R_B leer, so finden wir ebenfalls mittels unserer Datenstrukturen, einen blauen Knoten x mit $d_R(x) > 0$ und dann suchen wir einen roten Nachbarn von $r_1(x)$. Dazu benutzen wir den Such-Kredit von $r_1(x)$. Anschließend wird $r_1(x)$ blau sein und wir werden den Kredit nichtmehr benutzen.

Schauen wir uns den Fall an, in dem R_B nicht leer ist. Wir haben eine Liste mit blauen Knoten, die genau einen Nachbarn aus R_B haben. Aus dieser Liste nehmen wir den ersten Knoten x. Das Prüfen ob $d_{R_R}(x) > 0$ gilt, ist ebenfalls in konstanter Zeit möglich.

Ist dies der Fall, so können wir für $r_1(x)$ und $r_2(x)$ testen, welche von ihnen in R_R liegen. Falls mindestens einer der Beiden in R_R liegt, so suchen wir für denjenigen einen geeigneten Matchingpartner und benutzen dazu seinen Lösch-Kredit. Andernfalls gilt $r_1(x) = r_2(x) \in R_B$, dann werden x, $r_1(x)$ und $b_M(x)$ entfernt und verbrauchen ihren Lösch-Kredit.

Somit haben wir alle Fälle betrachtet und festgestellt, dass für die Entscheidungen und Operationen aus Algorithmus 4 insgesamt $\mathcal{O}(|E|)$ Zeit und für alle Aufrufe von Algorithmus 3 insgesamt $\mathcal{O}(|M|)$ Zeit benötigt wird. Da der Graph planar ist benötigt das gesamte Verfahren insgesamt $\mathcal{O}(n)$ Zeit. □

Durch Lemma 5.7 und Satz 5.5 können wir nun folgendes Endergebnis festhalten:

Satz 5.8. *Für einen planaren Graphen mit n Knoten und Minimalgrad 3 berechnet Algorithmus 4 ein Matching, dass mindestens* $(n+2)/3$ *Kanten enthält, in* $\mathcal{O}(n)$ *Zeit.*

Da wir durch das vorgestellte Verfahren die angestrebte Schranke für Minimalgrad 3 erreichen konnten, wäre zu hoffen, dass es sich auch auf Minimalgrad 4 und 5 übertragen lässt. Dieser Versuch wird im nächsten Abschnitt unternommen.

5.6 Das Verfahren für Minimalgrad 4 und 5 abändern

Wir werden versuchen, das Verfahren für Minimalgrad 4 und 5 anzupassen, um für diese Fälle bessere Schranken zu erreichen. Allerdings wird uns dies nur bedingt gelingen.

Wendet man das Verfahren auf einen Graphen mit Minimalgrad $\delta > 3$ an, so wissen wir schon, dass im resultierenden Graphen die Schranken aus Folgerung 4.4 gelten. Allerdings wurden ja unter Umständen noch Knoten entfernt und zwar im Verhältnis zwei Blaue für einen Roten. Für Minimalgrad 3 war dieses Verhältnis genau ausreichend, für größeren Minimalgrad ist es zu schwach. Bei Minimalgrad 4 müsste man vier Blaue für einen Roten löschen und bei Minimalgrad 5 müsste man sogar sechs Blaue für einen Roten eliminieren.

Wir müssen also versuchen, das Verfahren abzuändern, sodass das Verhältnis wieder passt. Es gibt zwei Stelle innerhalb des Algorithmus, an denen Knoten gelöscht werden. Einmal im Algorithmus 3, wenn man auf eine besetzte Kante stößt und einmal im Algorithmus 4, wenn wir bei einem blauen Knoten sind, der nur einen einzigen roten Nachbar hat und dieser in R_B liegt.

Der letztere Fall basiert darauf, dass falls R_B nicht leer ist, immer ein blauer Knoten x mit $d_{R_B}(x) = 1$ ausgewählt werden kann. Dies ist das Resultat aus Lemma 5.3. Dieses Lemma kann man für Minimalgrad 4 und 5 anpassen.

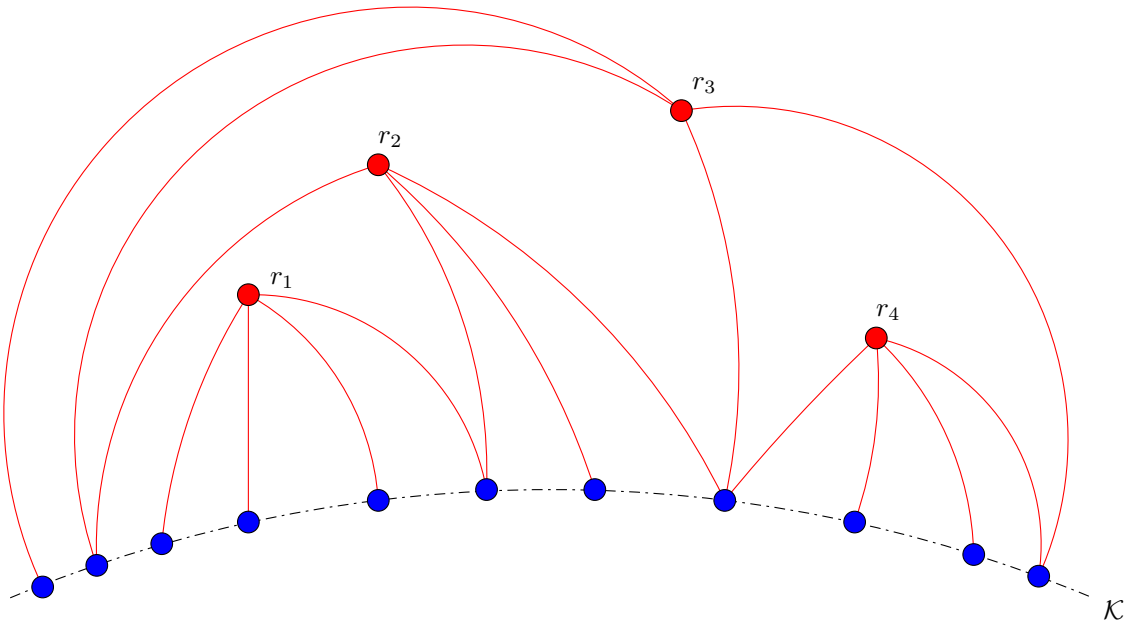


Abbildung 14: Die Knoten r_1 und r_4 induzieren "innere" Klammern und haben jeweils zwei blaue Nachbarn die keine anderen roten Nachbarn haben.

Lemma 5.9. *Es sei $G = (V, E)$ ein planarer Graph mit einer durch ein zyklisch reines Matching induzierten Färbung, sodass der blaue Subgraph ein Baum ist. Gibt es Knoten in R_B und hat jeder Knoten aus R_B mindestens Grad δ, dann gibt es einen roten Knoten $r \in R_B$ mit $\delta - 2$ blauen Nachbarn $b_1, \ldots, b_{\delta-2}$, sodass $d_{R_B}(b_i) = 1$ für $i = 1, \ldots, \delta - 2$.*

Beweis. Wir betrachten ausschließlich die Knoten aus R_B und B. Es sei wieder die Kurve $\mathcal{K}$ wie im Beweis von Lemma 5.3. Dann induzieren die roten Knoten wieder eine Klammerstruktur.

Wir betrachten einen roten Knoten r, der eine "innere" Klammer induziert. Nun muss r mindestens $\delta - 2$ blaue Nachbarn haben, die keine anderen roten Nachbarn haben (siehe Abbildung 14). Dies ist die gleiche Argumentation wie im Beweis von Lemma 5.3. □

Dieses Lemma erlaubt uns das Verfahren so abzuändern, dass man im Fall, dass R_B nicht leer ist einen roten Knoten $r \in R_B$ auswählt, der $\delta - 2$ blaue Nachbarn $b_1, \ldots, b_{\delta-2}$ mit $d_{R_B}(b_i) = 1$ besitzt. Hat jetzt einer dieser blauen Knoten noch einen roten Nachbarn aus R_R, so kann man eine geeignete Kante finden um das Matching mithilfe der Unterprozedur zu vergrößern. Andernfalls löscht man $b_1, \ldots, b_{\delta-2}$, deren Matchingpartner und r. Damit hat man an dieser Stelle das notwendige Verhältnis eingehalten.

Die andere Stelle, an der Knoten gelöscht werden, ist nicht so leicht anzupassen. Bis jetzt existiert dazu noch keine befriedigende Lösung. Man kann jedoch versuchen das Auftreten dieses Falls mit Heuristiken zu minimieren.

Angenommen wir haben eine Matchingkante $\{x, y\}$, die durch den roten Knoten $r \in R_B$ besetzt ist. Bevor wir anfangen zu Löschen, können wir erstmal noch schauen ob r vielleicht

noch vermöge einer seiner anderen Nachbarn auf einem erhöhenden Pfad der Länge 3 liegt. Ist dies der Fall, so kann r gematcht werden und wir brauchen nicht zu Löschen.

Falls der Knoten r einen Grad echt größer als δ hat, könnte man einfach die Kante $\{r, x\}$ entfernen und macht dadurch die Matchingkante $\{x, y\}$ zyklisch rein. Wieder spart man sich einen Löschvorgang.

Diese Vorgehen kann man in eine Vorberechnung umwandeln, denn x, y und r bilden ein Dreieck. Man könnte versuchen im Vorfeld die Anzahl der Kreise der Länge 3 zu reduzieren. Dazu kann man, falls ein in einem Dreieck zwei Knoten mit Grad echt größer δ existieren, die Kante zwischen den beiden entfernen und damit die Anzahl der Dreiecke verringern. Es ist allerdings nicht sicher ob diese Vorberechnung zu besseren Ergebnissen führt als die Beseitigung der Dreiecke on-the-fly.

Für $\delta = 5$ kann eine etwas schwächere Schranke garantiert werden, indem man für jeden gelöschten roten Knoten mindestens vier blaue Knoten löscht. Wie man dies bewerkstelligt, wird im Folgenden skizziert. Hat man eine Matchingkante e, die durch den roten Knoten $r \in R_B$ besetzt ist, stellt man zunächst sicher das r auf keinem erhöhenden Pfad der Länge 3 liegt (ansonsten vergrößert man das Matching). Falls r noch weitere Matchingkanten besetzt, so löschen wir einfach zwei besetzte Matchingkanten und haben sichergestellt das vier blaue Knoten gelöscht wurden. Im Folgenden gehen wir daher davon aus, dass e die einzige Kante ist, die von e bestzt wird.

Nun induziert r eine Klammer, die am blauen Subgraphen anliegt. Dann muss es drei blaue Nachbarn b_1, b_2 und b_3 von r geben, die im "Inneren" dieser Klammer liegen (da r mindestens fünf Nachbarn hat und nur zwei davon liegen auf dem Rand der Klammer). Der Algorithmus entfernt die Endknoten von e sowie den roten Knoten r. Dann bleibt allerdings mindestens einer der blauen Knoten b_1, b_2, b_3 übrig. Sei dies ohne Einschränkung b_1. Im weiteren Verlauf des Algorithmus wird b_1 weiterhin erhalten bleiben, dies liegt daran, dass falls b_1 einen Nachbar $r' \in R_R$ hat, die von r' induzierte Klammer b_1 nicht einschließen kann. Würde man nämlich r wieder einfügen, so würden sich die induzierten Klammern von r und r' kreuzen, was einen Widerspruch zur Planarität bedeuten würde. Deshalb wird b_1 nicht gelöscht werden, denn im Laufe des Algorithmus wird ein blauer Knoten x nur dann gelöscht, wenn seine Matchingkante besetzt ist oder wenn x oder sein Matchingpartner ein Nachbar eines roten Knotens ist und von dessen Klammer eingeschlossen ist. Damit wissen wir, dass b_1 irgendwann alle roten Nachbarn verlieren wird. Wenn dies geschied wird b_1 und sein Matchingpartner gelöscht und dadurch sichergestellt dass für r vier blaue Knoten entfernt wurden.

Dies liefert uns die Abschätzung $|R| \leq 4 \cdot |B|$ für das berechnete Matching M, was zu einer Verbesserung der Schranke führt:

$$|M| \geq \frac{2n}{5}.$$

6 Zusammenfassung und Ausblick

In diesem letzten Abschnitt werden wir Bilanz ziehen und die erzielten Resultate noch einmal kurz zusammenfassen. Wir werden überprüfen inwieweit die anfangs gesteckten Ziele erreicht wurden. Zum Abschluss wird ein Ausblick auf weiter zu untersuchende Fragen gegeben.

6.1 Zusammenfassung

Angefangen haben wir mit dem Entwurf eines einfachen Linearzeitalgorithmus, der uns ein inklusionsmaximales Matching ohne erhöhende Pfade der Länge 3 liefert. Wir haben gezeigt, dass Matchings mit dieser Struktur in planaren Graphen mit Minimalgrad δ gewisse Mindestgrößen besitzen. Für $\delta = 3$ wurde die untere Schranke $(n + 4)/4$ für die Grüße der Matchings erreicht. Die besetzten Kanten sorgten dafür, dass für $\delta = 4, 5$ die Schranke $n/3$ nicht ohne weiteres überwindet werden konnte. Anhand einer Familie von Graphen haben wir weiterhin gezeigt, dass diese unteren Schranken für die Größe der Matchings mit diesen Eigenschaften für $\delta = 3, 4$ scharf sind.

Die dadurch erhaltenen unteren Schranken für die Größe maximaler Matchings in solchen Graphen waren etwas Schwächer als die von Takao Nishizeki und Ilker Baybars bewiesenen. Wir haben deshalb stärkere Voraussetzungen an die Struktur der Matchings gestellt, indem wir zusätzlich forderten dass die Matchings zusammenhängend und zyklisch rein sein sollen. Mit Hilfe dieser weiteren Eigenschaften ist es gelungen stärkere Mindestgrößen zu beweisen.

Für $\delta = 3$ haben wir genau die Schranke aus [NB79] getroffen und für $\delta = 4$ kamen wir bis auf eine Konstante (die kleiner eins ist) heran. Allerdings fanden wir für $\delta = 5$ nur die Schranke $\frac{3n+2}{7}$, welche verglichen mit der Schranke $\frac{5n+6}{11}$ aus [NB79] noch etwas zu schwach ist.

Anschließend haben wir uns daran gemacht, ein Verfahren zu entwickeln, das uns Matchings mit den geforderten Eigenschaften beschaffen sollte. Das Resultat war ein Linearzeitalgorithmus, der für $\delta = 3$ ein Matching mit Mindestgröße $\frac{n+2}{3}$ liefert. Bei Minimalgrad $\delta = 4, 5$ kann das Auftreten der Situation, dass eine Matchingkante durch einen roten Knoten besetzt ist negativ ins Gewicht fallen, weswegen wir dafür keine besseren Schranken garantieren konnten. Wir haben einige heuristische Ansätze vorgestellt, die das Auftreten solcher Situationen verringern sollen.

6.2 Ausblick

Eine offene Frage ist, ob wir eine Lösung für das Problem der besetzten Kanten finden können. Würde dies gelingen, so hätten wir ein Verfahren das für $\delta = 4$ ein Matching liefert, dessen Kardinalität mindestens $\frac{2n+2}{5}$ beträgt. Dann könnte man für das Ergebnismatching einen erhöhenden Pfad suchen (was in $\mathcal{O}(n)$ Zeit möglich ist) und dadurch die Schranke $\frac{2n+3}{5}$ aus [NB79] garantieren.

Für $\delta = 5$ würde allerdings nur die Mindestgröße $\frac{3n+2}{7}$ heraus springen, was verglichen mit der unteren Schranke $\frac{5n+6}{11}$ von Nishizeki und Baybars noch zu schwach ist. Hier bleibt die

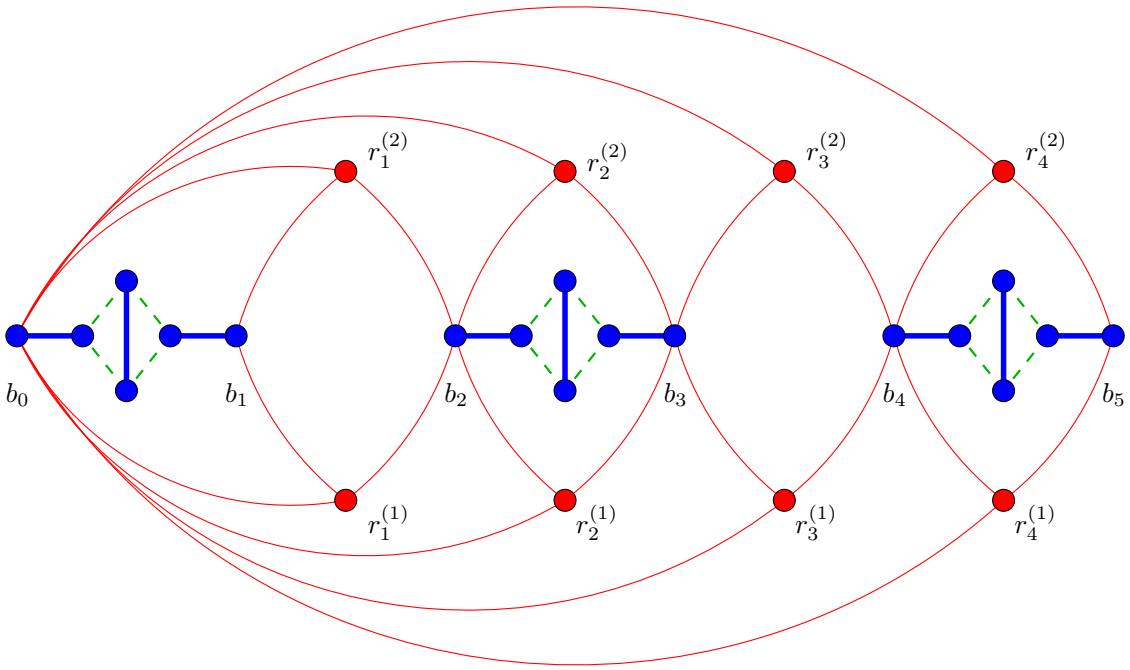


Abbildung 15: Ein Graph mit Minimalgrad 3 und einem inklusionsmaximalen Matching ohne erhöhende Pfade der Länge 3 oder 5

Frage offen, welche zusätzlichen strukturellen Eigenschaften für den Fall von Minimalgrad 5 verwendet werden können.

Ein weiterer Ansatz, der verfolgt werden könnte ist die Untersuchung von Matchings ohne erhöhende Pfade der Länge 5. Man kann vermutlich relativ einfach einen Algorithmus entwerfen, der ein inklusionsmaximales Matching ohne erhöhende Pfade der Länge 5 liefert.

In Abbildung 15 ist ein Graph mit Minimalgrad 3 und einem Matching ohne Pfade der Länge 3 oder 5 zu sehen. Dieser Graph lässt sich noch weiter auf natürliche Weise vergrößern, wodurch wir eine Familie von Graphen und Matchings erhalten. Dabei fällt auf, dass wenn wir drei zusätzliche Matchingkanten hinzufügen, wir noch vier weitere rote Knoten einfügen können. Daran erkennt man schon dass diese Familie die untere Schranke $\frac{n+2}{3}$ nicht einhalten kann. Genauere Analyse dieser Graphen ergibt eine Matchinggröße $|M| = \frac{3n+6}{10}$. Das legt die Vermutung nahe, dass es eine untere Schranke für inklusionsmaximale Matchings ohne erhöhende Pfade der Länge 3 oder 5 bei planaren Graphen mit Minimalgrad 3 in dieser Größenordnung gibt.

Es ist nicht klar, ob man einen einfachen Linearzeitalgorithmus finden kann, der ein inklusionsmaximales Matching ohne erhöhende Pfade bis zur Länge 7 oder allgemeiner bis zur Länge k (für festes k) liefert. Würde man so einen Algorithmus finden, so wäre es interessant eine allgemeine Abschätzung (in Abhängigkeit von k) der Größe dieser Matchings in planaren Graphen mit Minimalgrad δ durchzuführen. Vielleicht könnte man auch auf diese Weise an die stärkeren Schranken für $\delta = 4, 5$ herankommen.

Eine weitere offene Frage ist, wie sich die hier vorgestellten Algorithmen in einer prak-

tischen Analyse verhalten. Die Hoffnung ist, dass die garantierten Schranken übertroffen werden und der Abstand zur Größe von maximalen Matchings meistens gering sein wird.

Literatur

[Alg] Algorithmic Solutions: *The LEDA User Manual.* `http://www.algorithmic-solutions.info/leda_manual/`.

[BBDL01] Biedl, Therese, Prosenjit Bose, Eric Demaine und Anna Lubiw: *Efficient algorithms for Petersen's theorem.* In: *J. Algorithms*, Band 38, Seiten 110–134. 2001.

[BDD+04] Biedl, Therese, Erik D. Demaine, Christian A. Duncan, Rudolf Fleischer und Stephen G. Kobournov: *Tight bounds on maximal and maximum matchings.* In: *Discrete Mathemactics*, Band 285, Seiten 7–15. Elsevier B.V., 2004.

[Ber57] Berge, Claude: *Two theorems in graph theory.* Proceedings of the National Academy of Sciences, 43:842–844, 1957.

[CW87] Coppersmith, Don und Shmuel Winograd: *Matrix multiplication via arithmetic progressions.* In: *STOC '87: In Proc. 19th Annu. ACM Conf. Theory Comput.*, Seiten 1–6, 1987.

[FR06] Fakcharoenphol, Jittat und Satish Rao: *Planar graphs, negative weight, shortest paths, and near linear time.* In: *J. Comput. System Sci.*, Band 72, Seiten 868–889. 2006.

[HK73] Hopcraft, John E. und Richard M. Karp: *An $n^{3/2}$ algorithm for maximum matchings in bipartite graphs.* In: *SIAM J. Comput.*, Band 2, Seiten 225–231. 1973.

[Kőn16] Kőnig, Dénes: *Über Graphen und ihre Anwendung auf Determinantentheorie und Mengenlehre.* In: *Math. Ann.*, Band 77, Seiten 453–465. 1916.

[LP86] Lovász, László und Michael D. Plummer: *Matching Theory.* North Holland, Amsterdam, 1986.

[MN95] Miller, Gary L. und Joseph Naor: *Flow in planar graphs with multiple sources and sinks.* In: *SIAM J. Comput.*, Band 24, Seiten 1002–1017. 1995.

[MS04] Mucha, Marcin und Piotr Sankowski: *Maximum matchings via Gaussian elimination.* In: *FOCS '04: Proc. 45th Annu. IEEE Sympos. Foundat. Comput. Sci.*, Seiten 248–255, 2004.

[MS06] Mucha, Marcin und Piotr Sankowski: *Maximum matchings in planar graphs via Gaussian elimination.* In: *Algorithmica*, Band 45, Seiten 3–20, 2006.

[MV80] Micali, Silvio und Vijay V. Vazirani: *An $O(\sqrt{|V| \cdot |E|})$ algorithm for finding maximum matchings in general graphs.* In: *Proc. 21st Annu. IEEE Sympos. Found. Comput. Sci.*, Seiten 17–27. 1980.

[NB79] Nishizeki, Takao und Ilker Baybars: *Lower bounds on the cardinality of the maximum matchings of planar graphs.* In: *Discrete Mathemactics*, Band 28, Seiten 255–267. North-Holland Publishing Company, 1979.

[Pet91] Petersen, Julius: *Die Theorie der regulären Graphs.* In: *Acta Mathematica*, Band 15, Seiten 193–220. 1891.

[PY81] Papadimitriou, Christos H. und Mihalis Yannakakis: *Worst-case ratios for planar graphs and the method of induction on faces.* In: *FOCS '81: Proceedings of the 22nd Annual Symposium on Foundations of Computer Science*, Seiten 358–363, 1981.

[RW08] Rutter, Ignaz und Alexander Wolff: *Computing large matchings fast.* In: *SODA '08: Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, Seiten 183–192, Philadelphia, PA, USA, 2008. Society for Industrial and Applied Mathematics.

[RW09] Rutter, Ignaz und Alexander Wolff: *Computing large matchings fast.* In: *Transactions on Algorithms*, 2009.

[SLL] Siek, Jeremiy, Lie-Quan Lee und Andrew Lumsdaine: *The Boost Graph Library documentation.* `www.boost.org/libs/graph`.

[Tar83] Tarjan, Robert E.: *Data structures and network algorithms.* SIAM, Philadelphia, 1983.

[Tut47] Tutte, William: *The factorization of linear graphs.* In: *J. Lond. Math. Soc.*, Band 22, Seiten 107–111. 1947.

[YZ07] Yuster, Raphael und Uri Zwick: *Maximum matching in graphs with an exluded minor.* In: *SODA '07: In Proc. 18th Annu. ACM-SIAM Sympos. Discrete Algorithms*, Seiten 108–117, 2007.