Planarisierung von Graphen

Thomas Leichtle

Universität Ulm
Institut für theoretische Informatik
89069 Ulm, Germany
Thomas.Leichtle@uni-ulm.de

Zusammenfassung Im Folgenden werden einzelne Algorithmen zum Überprüfen der Planaritätseigenschaft von Graphen genannt und speziell auf den in Logspace liegenden Algorithmus von Datta & Prakriya eingegangen.

1 Einleitung

1.1 Geschichte

Kuratowski Bereits 1930 gab Kuratowski eine präzise Charakterisierung für planare Graphen. Er bewies, dass jeder nicht-planare Graph einen Teilgraphen, isomorph zu einem der folgenden zwei speziellen Graphen (siehe Abbildung 1) $(K_5, K_{3,3})$, enthält. Diesen entsprechenden Teilgraphen erhält man nach dem Entfernen aller Knoten vom Grad zwei. Ist ein Graph planar, so enthält er diesen nicht. Leider ist dieses Kriterium in der Praxis nur sehr schlecht zu verwenden, da ein direkter Test mindestens eine Laufzeit in $O(V^6)$ hätte [HT74].

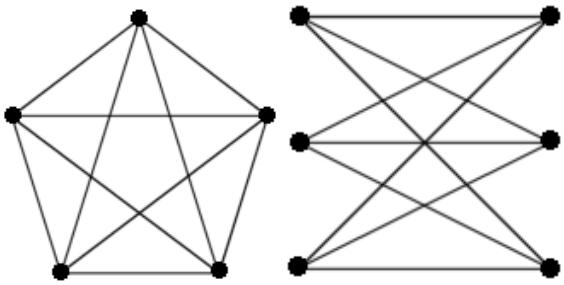


Abbildung 1. K_5 & $K_{3,3}$

Auslander & Parter; Goldstein Ein erster Algorithmus, welcher auf der Idee beruht, eine Repräsentation für eine planare Einbettung eines Graphen zu erzeugen, wurde 1961 von Auslander & Parter vorgeschlagen. Er entfernt Kreise

aus dem Graphen und zerlegt diesen somit in mehrere Teile, die dann rekursiv wieder zerlegt und in eine Ebene mit der Einbettung des ursprünglichen Graphens eingefügt werden. Ist es möglich den Graph komplett einzubetten, so ist er planar. Das Papier enthielt jedoch Fehler, denn es war möglich, dass eine Endlosschleife entstand. Diesen Fehler behob Goldstein 1963, der den Algorithmus zugleich iterativ definierte. Die Grundidee dieser Algorithmen scheint bisher der beste Ansatz zu sein, um die Planaritätseigenschaft zu überprüfen.

weitere Algorithmen 1966 präsentierten Lempel, Even und Cederbaum einen Algorithmus, der mit einem einzelnen Knoten begann und den Graphen dann Schritt für Schritt zusammensetzte. Für die Korrektheit musste die Reihenfolge jedoch speziell gewählt werden. 1969 veröffentlichte Shirey eine Möglichkeit mit einer Listenstruktur und einer Laufzeit von $O(V^3)$. Ein Algorithmus mit $O(V)$ Speicher und $O(V^2)$ Laufzeit wurde 1969 von Tarjan implementiert. Hopcroft und Tarjan entwickelten Goldsteins Variante noch weiter und kamen auf Grenzen von $O(VlogV)$ bzw. $O(V)$ Laufzeit.

1.2 Anwendungen

Die Planarität ist eine interessante Eigenschaft in der Graphentheorie, die zum einen direkte Anwendung findet und aus der zum anderen weitere Eigenschaften abgeleitet oder weitere Einschränkungen getroffen werden können.
Zur direkten Anwendung gehört vor allem das Gebiet der Elektrotechnik, speziell VLSI (Very-Large-Scale Integration), bei dem geprüft werden muss, dass sich keine Leiterbahnen schneiden. Aber auch das Planen des Verkehrsnetzes ohne Unterführungen/Brücken ist eine weitere direkte Anwendung.
Eigenschaften die abgeleitet werden können, sind z.B. die des Satzes von Euler, welcher noch erläutert wird.
Einschränkungen bzw. Vereinfachungen können u.a. beim Testen auf Isomorphie von Graphen [TW09], speziell bei Graphen zur Repräsentation chemischen Strukturen [HT74], getroffen werden, falls die Planaritätseigenschaft erfüllt ist.

1.3 Definitionen

Planar Ein Graph heißt planar, wenn er in eine Ebene eingebettet werden kann, d.h. dass man den Graphen so zeichnen kann, dass sich keine zwei Kanten überschneiden.

Logspace Es sei eine Mehrbandturingmaschine mit drei Bändern - einem nur lesbaren Eingabeband, einem nur beschreibbaren Ausgabeband und einem Arbeits- oder Speicherband, das sowohl beschrieben als auch gelesen werden kann - gegeben. Liefert nun eine solche Turingmaschine für jede beliebige Eingabe x eine korrektes Ergebnis und nutzt dabei nur die ersten $s(|x|)$ viele Stellen des Arbeitsbands, dann löst sie das Problem mit $s(.)$ viel Platzbedarf.
$SPACE(s(n))$ ist dann die Klasse der Entscheidungsprobleme, die mit maximal

$s(n)$ viel Platzbedarf gelöst werden kann, wobei n der Länge der Eingabe entspricht.
Logspace ist somit die Klasse der Entscheidungsprobleme, die mit maximal $O(log(n))$ Speicherplatz gelöst werden können.

Regionen Wird ein planarer Graph so gezeichnet, dass sich keine Kanten überschneiden, dann wird die Ebene von diesen Kanten in Regionen des Graphens zerlegt. Auch die äußerste Region zählt dazu, wie man in Abbildung 2 sehen kann:

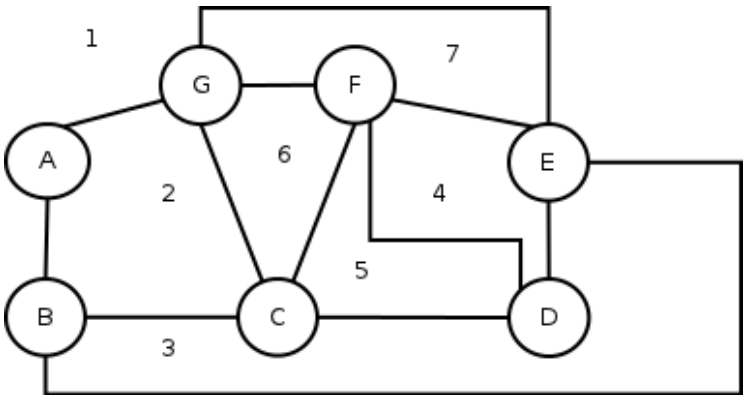


Abbildung 2. Ein Graph mit den Knoten A-G, 12 Kanten und den Regionen 1-7.

Satz von Euler In jedem zusammenhängenden, planaren Graphen gilt:
$Knotenzahl - Kantenzahl + Regionenzahl = 2$
Daraus kann man ableiten, dass für planare Graphen gilt:
$Kantenzahl < 3 * (Knotenzahl - 1)$
Im Beispiel Abb. $2 : 7 - 12 + 7 = 2$ und $12 < 3 * (7 - 1)$

K-Connected (k-fach verbunden) Ein Graph ist k-connected, falls der Grad jedes Knotens $\geq$ k ist, d.h. jeder Knoten hat mindestens k Kanten zu anderen Knoten. Dies kann auch auf einzelne Teile des Graphens angewendet werden, um k-fach verbundene Komponenten zu erhalten.
Für $k = 3$ spricht man von sogenannten triconnected oder 3-connected (3-fach verbundener) Komponenten. Diese können nach [DNTW09] in Logspace gefunden werden.

Bridges Für jede Zusammenhangskomponente $X \subseteq G \backslash C$ ist der induzierte Graph $G[X \cup A_x]$ eine Bridge des Kreises C. $A_x \subseteq C$ sind die zu X adjazenten

Knoten, die sogenannten Verbindungspunkte.
Ein **Konflikt** zwischen zwei Bridges B_1 und B_2 besteht in folgenden Fällen:

Fall 1: B_1 und B_2 haben drei gemeinsame Verbindungspunkte (Points of Attachment) bzgl. des Kreises C.

Fall 2: $a_1, b_1 \subset B_1$, $a_2, b_2 \subset B_2$ und a_i, b_i sind Verbindungspunkte bzgl. des Kreises C, sodass sie in der Reihenfolge a_1, a_2, b_1, b_2 entlang C auftreten.

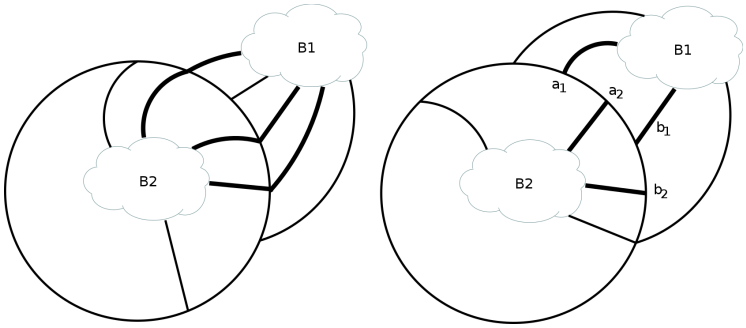


Abbildung 3. Links: Fall 1, Rechts: Fall 2. B_1, B_2 Bridges.
Die fettgedruckten Kanten verursachen jeweils den Konflikt.

Konfliktgraph & Bipartite Der Konfliktgraph zu einem Kreis eines Graphens wird derart konstruiert, dass die Bridges Knoten sind und zwei Knoten genau dann mit einer Kante verbunden werden, wenn die entsprechende Bridges in Konflikt miteinander stehen. Ein Beispiel ist in Abbildung 4 zu sehen.
Ein solcher Graph ist **bipartite**, falls seine Knotenmenge in zwei disjunkte Teilmengen zerlegbar ist, sodass innerhalb einer Menge keine Knoten sind.

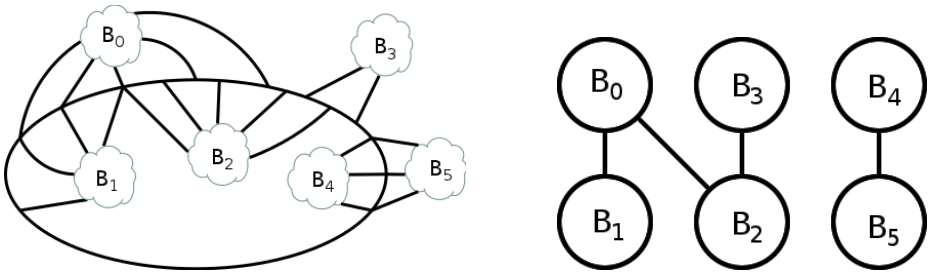


Abbildung 4. Links: Ein Beispielkreis mit dem Bridges B_0 bis B_5, Rechts: Der zugehörige Konfliktgraph. Er ist bipartite (oben/unten), jedoch nicht zusammenhängend.

Spannbaum & fundamentaler Zyklus Ein Spannbaum ist ein Teilgraph, der alle Knoten des zugehörigen zusammenhängenden Graphens enthält und die minimale Anzahl an Kanten, sodass er selbst ebenfalls zusammenhängend ist. Somit beträgt seine Kantenzahl = Knotenzahl des ursprünglichen Graph - 1. Er ist nicht eindeutig definiert.
Fügt man einem festen Spannbaum eines 2-fach verbundenen Graphens eine beliebige neue Kante e hinzu, so erhält man einen Zyklus C(e). Ein solcher heißt fundamentaler Zyklus von e. Es sei ein Graph mit V Knoten und E Kanten, dann folgt aufgrund der bereits erwähnten Tatsache, dass ein Spannbaum V-1 viele Kanten enthält, dass es E-V+1 fundamentale Zyklen gibt. Dies ist sehr einfach nachzuvollziehen, denn V-1 Kanten werden benötigt, dass der Graph zusammenhängend bleibt und für jede weitere Kante, kann ein Kreis gebildet werden. Man bezeichnet eine Region als fundamental, falls sie einem fundamentalen Zyklus entspricht.
Betrachtet man Abbildung 5 , so ist $I = C(e_1)$ der zur Kante e_1 gehörige fundamentale Zyklus, ebenso $II = C(e_2)$, $III = C(e_3)$ und I, II, III sind fundamentale Regionen.

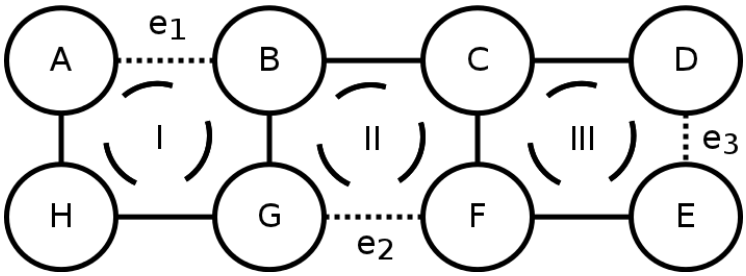


Abbildung 5. Der Beispielgraph besteht aus den Knoten A-H und allen gezeichneten Kanten. Ein zugehöriger Spannbaum T_1 besteht aus allen Knoten A-H und den durchgezogenen Kanten.

Kombinatorische Einbettung Eine kombinatorsiche Einbettung ist eine zyklische Reihenfolge indizenter Kanten. (Beispiel Abbildung 6). Ist eine kombinatorische Einbettung eines Graphens gegeben, so kann man damit die planare Einbettung zeichnen.

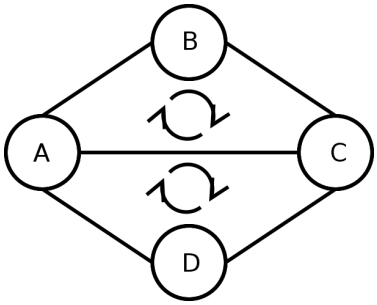


Abbildung 6. Beispielgraph

Eine kombinatorische Einbettung für diesen Graphen wäre:
$\rho = (\rho_A, \rho_B, \rho_C, \rho_D)$ mit
$\rho_A = \{(AB), (AC), (AD)\}$,
$\rho_B = \{(BC), (BA)\}$,
$\rho_C = \{(CD), (CA), (CB)\}$,
$\rho_D = \{(DA), (DC)\}$.
Hierfür wurde die Reihenfolge für die Kanten gemäß den Pfeilen im Graphen im Uhrzeigersinn gewählt.

2 Algorithmus von Datta & Prakriya

2.1 Algorithmus

Als Eingabe wird ein 3-fach verbundener Graph erwartet. Ausgabe ist eine planare Einbettung des Graphens, falls dieser planar ist. Andernfalls wird abgebrochen.

1. Finde eine Region. Erstelle dazu einen Spannbaum T und zu einer beliebigen Kante $e_0 \notin T$ eine zugehörige fundamentale Region über den entsprechenden fundamentalen Zyklus $C(e_0)$. Ist dies nicht möglich: beende.
2. Definiere diese als äußerste Region.
3. (a) Erstelle für jeden fundamentalen Zyklus C(e) bzgl. T den Konfliktgraph.
   (b) Ist einer dieser Graphen nicht bipartite, dann beende.
   (c) Wenn er bipartite ist, dann mit zwei Farben färben.
   (d) Sei $B_{e',e}$ die Bridge des Kreises $C(e)$, die e' enthält. Teste für alle Kanten $e' \notin T \wedge e' \neq e$: Erhält $B_{e',e}$ im Konfliktgraph von C(e) eine andere Farbe wie $B_{e_0,e}$, dann gilt $e' \prec e$ (d.h e' liegt „weiter innen“ als e).
4. Für jeden Fundamental Cycle $C(e)$:
   (a) Sei $P(e) = \{e' \in E(G) \backslash (E(T) \cup \{e\}) | e' \prec e \wedge \nexists \mathring{e} : e' \prec \mathring{e} \prec e\}$
   (b) Sei $F(e) = \bigoplus_{e' \in P(e) \cup \{e\}} C(e')$
5. Füge $C(e_0)$, also die äußere Region, den F's hinzu.
6. Teste für jede Kante $e \in E(G)$, ob sie in genau zwei $F(e')$'s liegt. Falls nein: beende.
7. Erstelle eine planare Einbettung für G und gib diese zurück. Dies funktioniert so: Jedes Triple von Knoten (u, v, w) ist ein Winkel, wenn (u, v), (v, w) aufeinanderfolgende Kanten einer Region sind. Zwei Winkel gelten als adjazent, wenn $v_1 = v_2$ und entweder $w_1 = u_2$ oder $u_1 = w_2$ ist. Der ungerichtete Graph mit den Winkeln als Knoten und Adjanzenzen als Kanten bildet eine Menge unverbundener Kreise. Das geordnete Durchlaufen der Kreise ergibt genau die kombinatorische Einbettung.

2.2 Erläuterungen

Schritt 1: Es ist möglich bei jedem planaren Graphen eine Region zu finden, denn jeder 2-fach verbundene planare Graph besitzt mindestens eine fundamentale Region bzgl. jeder seiner Spannbäume. Für Beweis siehe [DP11a] (Proposition 2).

Schritt 2: Die Kanten eines planaren Graphen können immer so umgezeichnet werden, dass jede beliebige fundamentale Region außen sein kann. Somit kann jede Region als unendliche, äußere betrachtet werden.

Schritt 3: Ein Graph ist planar, falls der Konfliktgraph jedes Kreises des Graphs bipartite ist. Siehe [Tut58]. Für jeden planaren Graphen G mit Zyklus C gilt: Der Konfliktgraph $H_C(G)$ ist bipartite und zusammenhängend.
Beweis: Nimmt man die Eigenschaft „im“ und „außerhalb“ des Kreises, so ist es ersichtlich, dass $H_C(G)$ bipartite sein muss. Nun zum Zusammenhang: Für jedes Paar $x, y \in C$ von Punkten gibt es zwei Pfade von x nach y entlang C. Diese seien $P_{x,y}$ und $P'_{x,y}$. Sei $\mathcal{B}$ eine Menge von Bridges, die eine Zusammenhangskomponente von $H_C(G)$ repräsentieren. Wähle eine Bridge $B' \notin \mathcal{B}$. Die Verbindungspunkte von B' zerlegen C in mehrere Segmente. Würden Verbindungspunkte von $\mathcal{B}$ in mehreren solchen Segmenten liegen, würde ein Konflikt bestehen und es würde gelten $B' \in \mathcal{B}$. Somit müssen alle Verbindungspunkte von $\mathcal{B}$ in einem solchen Segment liegen. Dann kann man jedoch die äußersten Verbindungspunkte von $\mathcal{B}$ in diesem Segment als u,v wählen, welche ein separierendes Paar darstellen. Da dies in einem 3-fach verbundenem Graphen nicht sein kann, muss eine Bridge $\tilde{B}$ existieren mit Verbindungspunkten in $P_{u,v}\backslash\{u,v\}$ und $P'_{u,v}\backslash\{u,v\}$, die somit in Konflikt steht und den Konfliktgraph verbindet.

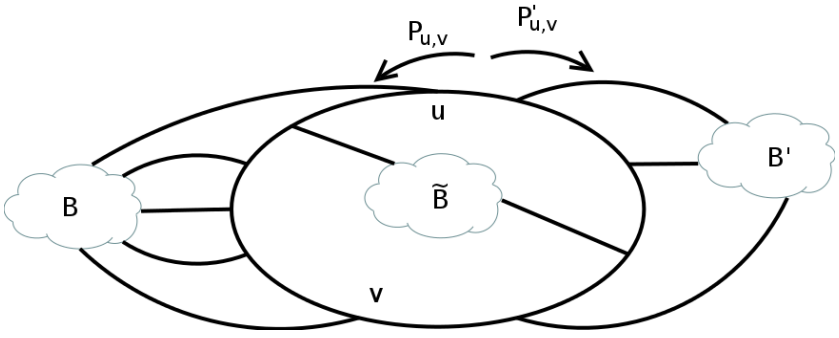


Abbildung 7. $\tilde{B}$ mit Verbindungspunkten in $P_{u,v}\backslash\{u,v\}$ und $P'_{u,v}\backslash\{u,v\}$.

Schritt 4: Hiermit werden aus den Zyklen die verschiedenen Regionen berechnet. Dabei sind $P(e)$ diejenige Kanten, die „direkt innerhalb“ des aktuellen Zyklus liegen und $F(e)$ die (über die symmetrische Differenz) daraus berechnete Region. Siehe 2.3 für Beispiel.

Schritt 5: Dieser Schritt wurde im ursprünglichen Algorithmus in [DP11a] zwar durchgeführt, jedoch wurde $F(e_0) = C(e_0)$ definiert. Diese sind jedoch unterschiedlich. D.h. es fehlte eine Region.

Schritt 6: Ein 3-fach verbundener Graph ist genau dann planar, wenn jede seiner Kanten in genau zwei nicht-separierenden Kreisen liegt [Die05].

Schritt 7: Siehe [DP11a] (Proposition 1) für Beweis und 2.3 für ein Beispiel.

2.3 Beispiel

Zum besseren Verständnis nun noch ein etwas komplexeres Beispiel.

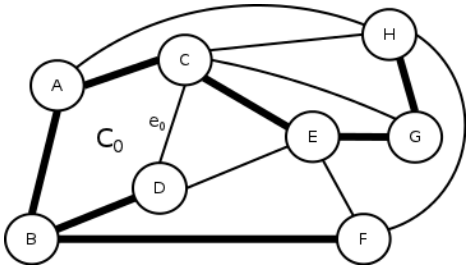


Abbildung 8. Der zu testende Beispielgraph G nach dem Erstellen eines Spannbaums T (fettgedruckte Kanten) und dem Auswählen einer beliebigen Kante $e_0 \in E(G \backslash T)$, sodass $C(e_0)$ eine fundamentale Region ist.

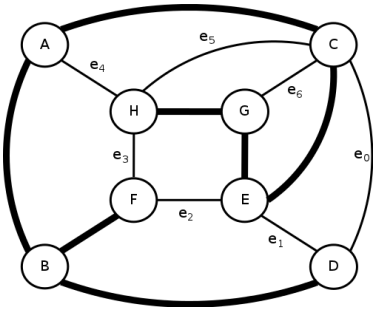


Abbildung 9. Der Graph G so umgezeichnet, dass $C(e_0)$ die äußere Region ist. Beliebige Beschriftung der restlichen Kanten eingefügt.

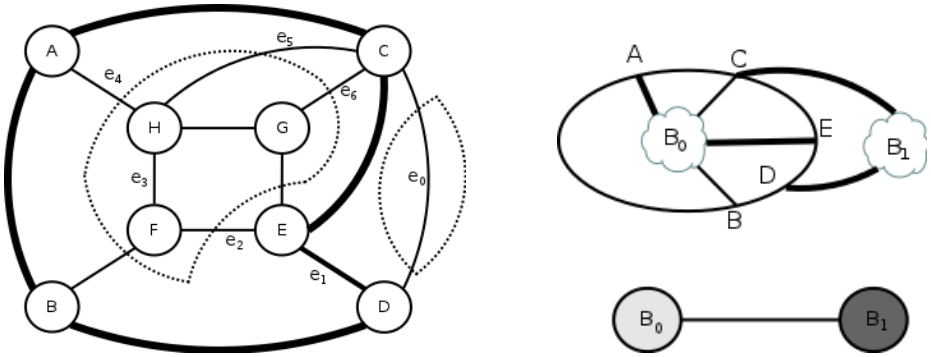


Abbildung 10. Links (fettgedruckt): $C(e_1)$. Zusammenhangskomponenten der Bridges sind gepunktet markiert. B_0 besteht aus den Knoten $\{F, G, H, A, B, C, E\}$ mit den Verbindungspunkten $\{A, B, C, E\}$, B_1 nur aus den Verbindungspunkten $\{C, D\}$.
Rechts: Zusammengefasste Darstellung der Bridges (Konfliktkanten fettgedruckt) mit dem zugehörigen 2-gefärbten Konfliktgraphen $H_{C_{e1}}(G)$.
B_{e_0,e_1} (diejenige Bridge bzgl. $C(e_1)$, die e_0 enthält) $= B1$. $B_{e_2,e_1} = B_{e_3,e_1} = B_{e_4,e_1} = B_{e_5,e_1} = B_{e_6,e_1} = B_0$. Da B_0 eine andere Farbe hat als B_1 folgt daraus, dass $e_i \prec e_1$ für $2 \leq i \leq 6$.

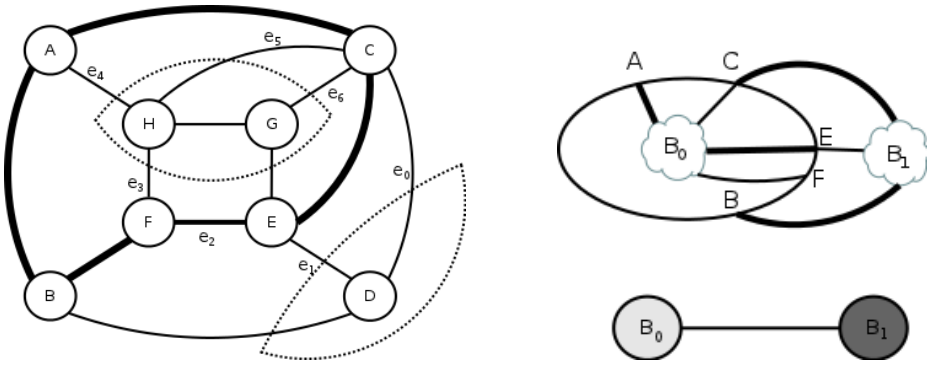
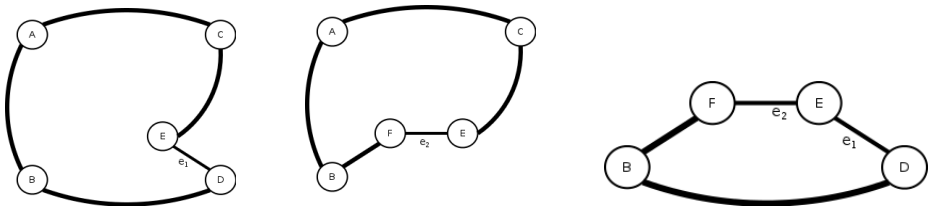


Abbildung 11. $C(e_2)$. $\{H, G, A, C, E, F\}$ mit den Verbindungspunkten $\{A, C, E, F\}$ gehören zu B_0, $\{D, B, C, E\}$ mit den Verbindungspunkten $\{B, C, E\}$ zu B_1.
$B_{e_0,e_2} = B_{e_1,e_2} = B_1$. $B_{e_3,e_2} = B_{e_4,e_2} = B_{e_5,e_2} = B_{e_6,e_2} = B_0$. Da B_0 andere Farbe hat als $B_1 \Rightarrow e_i \prec e_2$ für $3 \leq i \leq 6$.

Führt man den Algorithmus wie in Abb. 10 & 11 mit den restlichen Kanten fort, so erhält man folgendes Ergebnis: $e_6 \prec e_5 \prec e_4 \prec e_3 \prec e_2 \prec e_1 \prec e_0$

e	$P(e)$	$F(e)$
e_0	e_1	$C(e_0) \oplus C(e_1)$
e_1	e_2	$C(e_1) \oplus C(e_2)$
e_2	e_3	$C(e_2) \oplus C(e_3)$
e_3	e_4	$C(e_3) \oplus C(e_4)$
e_4	e_5	$C(e_4) \oplus C(e_5)$
e_5	e_6	$C(e_5) \oplus C(e_6)$
e_6	$\emptyset$	$C(e_6)$

Aus den vorher berechneten Relationen ergeben sich diese Werte für $P(e)$, bzw. $F(e)$. $P(e)$ ist dabei die Menge der Kanten, die direkt „innerhalb“ von e liegen, was bei diesem Beispiel jeweils maximal eine Kante ist - dies kann im Allgemeinen abweichen! $F(e)$ ist die symmetrische Differenz des Zyklus $C(e)$ mit den Zyklen, der zu den in $P(e)$ liegenden Kanten.



(a) $C(e_1)$ (b) $C(e_2)$ (c) $F(e_1) = C(e_1) \oplus C(e_2)$

Abbildung 12. Schritt 4b: hier an Hand des Beispiels für $F(e_1)$.

Betrachtet man $F(e_1)$, dann stellt man fest, dass dies genau einer Region entspricht. Dasselbe gilt für $F(e_0)$ und $F(e_2)$ bis $F(e_6)$.
Da wir bei diesem Beispiel bereits wissen, dass der Graph planar ist, könnte das Prüfen, ob jede Kante in genau zwei $F(e)'s$ liegt, entfallen. Zum Verständnis trotzdem eine kurze Aufzählung: $e_0 \in C(e_0), F(e_0)$; $e_1 \in F(e_0), F(e_1)$; $e_2 \in F(e_1), F(e_2)$; $e_3 \in F(e_2), F(e_3)$; $e_4 \in F(e_3), F(e_4)$; $e_5 \in F(e_4), F(e_5)$; $e_6 \in F(e_5), F(e_6)$.
In einem letzten Schritt wird nun noch die Einbettung über die Winkel erstellt:

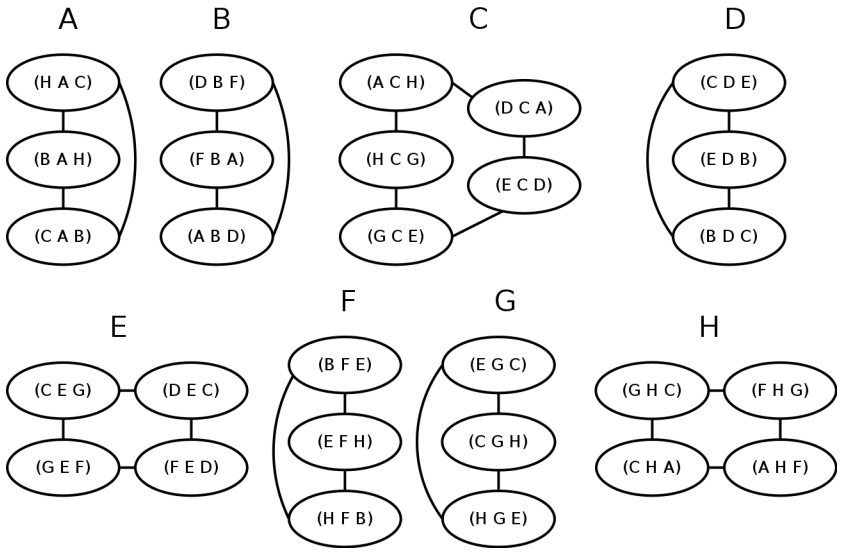
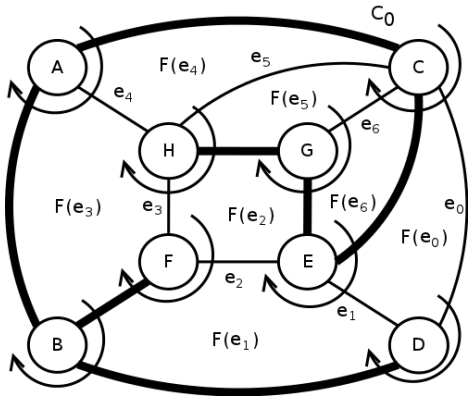


Abbildung 13. Der resultierende Graph der Winkel

Läuft man den Kreis für jeden Knoten ab und will die Reihenfolge der ausgehenden Kanten, so nimmt man jeweils die Kante zwischen den hinteren zwei Knoten jedes Triples. Es muss für jeden Knoten die „gleiche Reihenfolge“ eingehalten werden, z.B. so, dass der erste Knoten des aktuellen Triples dem letzten Knoten des Nachfolgertriples entspricht.

Eine kombinatorische Einbettung für den Graphen sieht dann wie folgt aus:
$\rho = (\rho_A, \rho_B, \rho_C, \rho_D, \rho_E, \rho_F, \rho_G)$
$\rho_A = \{(AC), (AH), (AB)\}$
$\rho_B = \{(BF), (BD), (BA)\}$
$\rho_C = \{(CH), (CA), (CD), (CE), (CG)\}$
$\rho_D = \{(DE), (DC), (DB)\}$
$\rho_E = \{(EG), (EC), (ED), (EF)\}$
$\rho_F = \{(FE), (FB), (FH)\}$
$\rho_G = \{(GC), (GE), (GH)\}$
$\rho_H = \{(HC), (HG), (HF), (HA)\}$



3 Schluss/Zusammenfassung

Referenzen, welche belegen, dass der vorgestellte Algorithmus in Logspace liegt, wurden in der gesamten Erläuterung absichtlich vernachlässigt. Hierzu in [DP11a]

oder [DP11b] nachschauen.
Zusammenfassend nochmals eine kurze Erläuterung des Vorgehens. Zuerst wurde ein Spannbaum erstellt, um eine fundamentale Region zu finden und die fundamentalen Zyklen berechnen zu können. Für diese wurden dann die Konfliktgraphen erstellt und auf Zusammenhang und Bipartitheit überprüft. Sollten die Bedingungen soweit erfüllt gewesen sein, wurde 2-gefärbt und eine Ordnung $\prec$ über den Kanten definiert und berechnet. Darüber konnten die Regionen berechnet werden. Nach einem weiteren Test, ob die Kanten in genau zwei Regionen liegen, konnte bei Planarität eine kombinatorische Einbettung über die Winkel berechnet werden.

Literatur

[Die05] R. Diestel. *Graph Theory*, volume 173 of *Graduate Texts in Mathematics*. Springer Verlag, 2005.

[DNTW09] Samir Datta, Prajakta Nimbhorkar, Thomas Thierauf, and Fabian Wagner. Graph isomorphism for $K_{3,3} - free$ and $K_5 - free$ graphs is in logspace. In Ravi Kannan and K Narayan Kumar, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2009)*, volume 4 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 145–156, Dagstuhl, Germany, 2009. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[DP11a] Samir Datta and Gautam Prakriya. Planarity testing revisited. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:9, 2011.

[DP11b] Samir Datta and Gautam Prakriya. Planarity testing revisited. In Mitsunori Ogihara and Jun Tarui, editors, *Theory and Applications of Models of Computation - 8th Annual Conference, TAMC 2011, Tokyo, Japan, May 23-25, 2011. Proceedings*, volume 6648 of *Lecture Notes in Computer Science*, pages 540–551. Springer, 2011.

[HT74] John Hopcroft and Robert Tarjan. Efficient planarity testing. *Journal of the ACM*, 21(4):549–568, October 1974.

[Tut58] W. T. Tutte. A homotopy theorem for matroids, i, ii. *Transactions of The American Mathematical Society*, 88:144–174, 1958.

[TW09] J. Torán and F. Wagner. The complexity of planar graph isomorphism. *Bulletin of the European Association for Theoretical Computer Science*, 97:60–, February 2009. Columns: Concurrency.